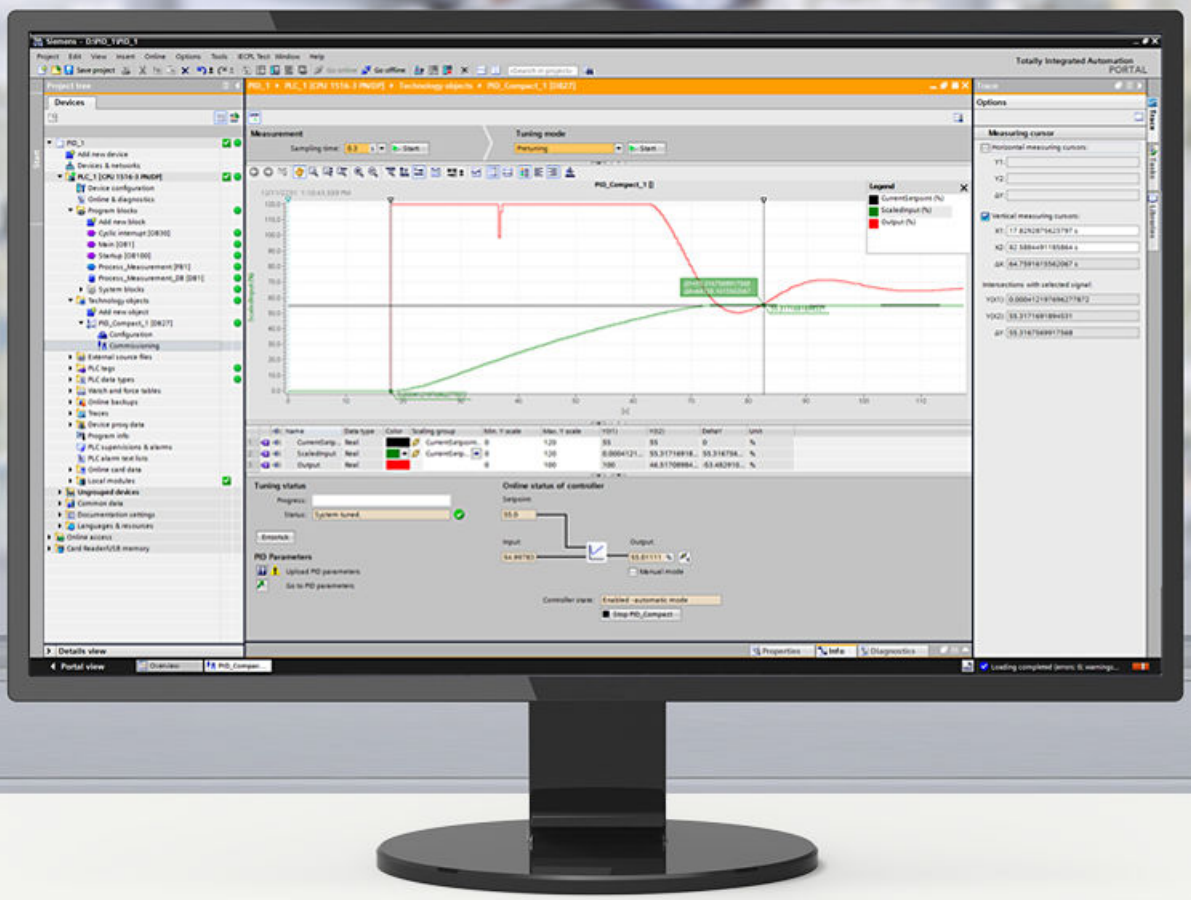
SIEMENS



Édition 11/2023

DESCRIPTION FONCTIONNELLE

SIMATIC

S7-1200, S7-1500

Régulation PID

support.industry.siemens.com

SIEMENS

SIMATIC

S7-1200, S7-1500 Régulation PID

Description fonctionnelle

11/2023
A5E35300228-AG

Mentions légales

Signalétique d'avertissement

Ce manuel donne des consignes que vous devez respecter pour votre propre sécurité et pour éviter des dommages matériels. Les avertissements servant à votre sécurité personnelle sont accompagnés d'un triangle de danger, les avertissements concernant uniquement des dommages matériels sont dépourvus de ce triangle. Les avertissements sont représentés ci-après par ordre décroissant de niveau de risque.

⚠ DANGER

signifie que la non-application des mesures de sécurité appropriées **entraîne** la mort ou des blessures graves.

⚠ ATTENTION

signifie que la non-application des mesures de sécurité appropriées **peut entraîner** la mort ou des blessures graves.

⚠ PRUDENCE

signifie que la non-application des mesures de sécurité appropriées peut entraîner des blessures légères.

IMPORTANT

signifie que la non-application des mesures de sécurité appropriées peut entraîner un dommage matériel.

En présence de plusieurs niveaux de risque, c'est toujours l'avertissement correspondant au niveau le plus élevé qui est reproduit. Si un avertissement avec triangle de danger prévient des risques de dommages corporels, le même avertissement peut aussi contenir un avis de mise en garde contre des dommages matériels.

Personnes qualifiées

L'appareil/le système décrit dans cette documentation ne doit être manipulé que par du **personnel qualifié** pour chaque tâche spécifique. La documentation relative à cette tâche doit être observée, en particulier les consignes de sécurité et avertissements. Les personnes qualifiées sont, en raison de leur formation et de leur expérience, en mesure de reconnaître les risques liés au maniement de ce produit / système et de les éviter.

Utilisation des produits Siemens conforme à leur destination

Tenez compte des points suivants:

⚠ ATTENTION

Les produits Siemens ne doivent être utilisés que pour les cas d'application prévus dans le catalogue et dans la documentation technique correspondante. S'ils sont utilisés en liaison avec des produits et composants d'autres marques, ceux-ci doivent être recommandés ou agréés par Siemens. Le fonctionnement correct et sûr des produits suppose un transport, un entreposage, une mise en place, un montage, une mise en service, une utilisation et une maintenance dans les règles de l'art. Il faut respecter les conditions d'environnement admissibles ainsi que les indications dans les documentations afférentes.

Marques de fabrique

Toutes les désignations repérées par ® sont des marques déposées de Siemens AG. Les autres désignations dans ce document peuvent être des marques dont l'utilisation par des tiers à leurs propres fins peut enfreindre les droits de leurs propriétaires respectifs.

Exclusion de responsabilité

Nous avons vérifié la conformité du contenu du présent document avec le matériel et le logiciel qui y sont décrits. Ne pouvant toutefois exclure toute divergence, nous ne pouvons pas nous porter garants de la conformité intégrale. Si l'usage de ce manuel devait révéler des erreurs, nous en tiendrons compte et apporterons les corrections nécessaires dès la prochaine édition.

Siemens AG
Digital Industries
Postfach 48 48
90026 NÜRNBERG
ALLEMAGNE

A5E35300228-AG
Ⓟ 10/2023 Sous réserve de modifications

Sommaire

Introduction

1

1.1 Objet, conventions et informations complémentaires

Objet de cette documentation

La présente documentation a pour but de vous aider lors de la configuration et la programmation de tâches de régulation avec les systèmes d'automatisation S7-1200 et S7-1500.

Connaissances de base requises

Pour bien exploiter les informations contenues dans cette documentation, les connaissances suivantes sont nécessaires :

- Connaissances générales en technique d'automatisation
- Connaissances de l'automate programmable industriel SIMATIC
- connaissances sur l'utilisation de STEP 7 (TIA Portal)

Domaine de validité de la documentation

Cette documentation s'applique en cas d'utilisation de régulateurs logiciels sur les CPU des systèmes d'automatisation S7-1200 et S7-1500 en jonction avec STEP 7 (TIA Portal). En cas d'utilisation de S7-300 et S7-400 avec STEP 7 (TIA Portal), d'autres régulateurs logiciels sont disponibles. Ils ne font pas l'objet de cette documentation. Le chapitre Présentation des régulateurs de logiciel (Page 43) fournit une vue d'ensemble de tous les régulateurs logiciels dans STEP 7 (TIA Portal) et de leurs possibilités de mise en œuvre.

Conventions

Tenez compte des remarques identifiées de la façon suivante :

REMARQUE

Un nota contient des informations importantes sur le produit décrit dans la documentation, sur la manipulation du produit ou sur la partie de la documentation qu'il faut particulièrement mettre en relief.

Industry Mall

L'Industry Mall est le catalogue et le système de commande de Siemens AG pour les solutions d'automatisation et d'entraînement basées sur Totally Integrated Automation (TIA) et Totally Integrated Power (TIP).
Les catalogues pour tous les produits d'automatisation et d'entraînement sont disponibles sur Internet (https://mall.industry.siemens.com).

Voir aussi

Page thématique "SIMATIC Technology - PID Control : Vue d'ensemble et liens importants"
https://support.industry.siemens.com/cs/ww/fr/view/109751051
(https://support.industry.siemens.com/cs/ww/fr/view/109751051)

1.2 Guide de la documentation Description fonctionnelle

1.2.1 Classes d'information Descriptions fonctionnelles

La documentation pour le système d'automatisation SIMATIC S7-1500, pour les CPU 1513/1516pro-2 PN, SIMATIC Drive Controller basées sur SIMATIC S7-1500 et les systèmes de périphérie décentralisée SIMATIC ET 200MP, ET 200SP, ET 200AL et ET 200eco PN se compose de trois parties.
Cette subdivision vous permet d'accéder de manière ciblée aux contenus souhaités.
Vous pouvez télécharger gratuitement la documentation sur Internet (https://support.industry.siemens.com/cs/ww/fr/view/109742705).

Informations de base

Le manuel système et le guide de mise en route (Getting Started) décrivent en détail la configuration, le montage, le câblage et la mise en service des systèmes SIMATIC S7-1500, SIMATIC Drive Controller, ET 200MP, ET 200SP, ET 200AL et ET 200eco PN. Vous utilisez pour les CPU 1513/1516pro-2 PN les instructions de service correspondantes.
L'aide en ligne de STEP 7 vous assiste dans la configuration et la programmation.
Exemples :

- Getting Started S7-1500
- Manuels système
- Instructions de service ET 200pro et CPU 1516pro-2 PN
- Aide en ligne TIA Portal

Informations sur les appareils

Les manuels contiennent une description compacte des informations spécifiques aux modules, telles que les propriétés, les schémas de raccordement, les courbes caractéristiques, les caractéristiques techniques.
Exemples :

- Manuels sur les CPU
- Manuels sur les modules d'interface
- Manuels sur les modules TOR
- Manuels sur les modules analogiques
- Manuels sur les modules de communication
- Manuels sur les modules technologiques
- Manuels sur les modules d'alimentation
- Manuels sur les BaseUnits

Informations globales

Vous trouverez dans les descriptions fonctionnelles des descriptions détaillées sur des thèmes transversaux relatifs au SIMATIC Drive Controller et au système d'automatisation S7-1500.
Exemples :

- Description fonctionnelle Diagnostic
- Description fonctionnelle Communication
- Descriptions fonctionnelles Motion Control
- Description fonctionnelle Serveur web
- Description fonctionnelle Temps de cycle et de réaction
- Description fonctionnelle PROFINET
- Description fonctionnelle PROFIBUS

Information produit

Les modifications et compléments apportés aux manuels sont documentés dans une information produit. Les informations qu'elle contient prévalent sur celles du manuel de l'appareil et du manuel système.
Vous trouverez les informations produit les plus récentes sur Internet :

- S7-1500/ET 200MP (https://support.industry.siemens.com/cs/fr/fr/view/68052815)
- SIMATIC Drive Controller (https://support.industry.siemens.com/cs/de/fr/view/109772684/fr)
- Motion Control (https://support.industry.siemens.com/cs/de/fr/view/109794046/fr)
- ET 200SP (https://support.industry.siemens.com/cs/fr/fr/view/73021864)
- ET 200eco PN (https://support.industry.siemens.com/cs/ww/fr/view/109765611)

Collections de manuels

Les collections de manuels contiennent dans un fichier la documentation complète relative aux systèmes correspondants.
Vous trouverez les collections de manuels sur Internet.

- S7-1500/ET 200MP/SIMATIC Drive Controller (https://support.industry.siemens.com/cs/ww/fr/view/86140384)
- ET 200SP (https://support.industry.siemens.com/cs/ww/fr/view/84133942)
- ET 200AL (https://support.industry.siemens.com/cs/ww/fr/view/95242965)
- ET 200eco PN (https://support.industry.siemens.com/cs/ww/fr/view/109781058)

1.2.2 Outils de base

Outils

Les outils décrits ci-après vous assistent lors de toutes les étapes, de la planification à l'analyse en passant par la mise en service de votre installation.

TIA Selection Tool

Le TIA Selection Tool vous assiste dans la sélection, la configuration et la commande d'appareils pour Totally Integrated Automation (TIA).
En tant que successeur du SIMATIC Selection Tool, TIA Selection Tool regroupe dans un outil tous les configurateurs déjà connus pour l'automatisation.
TIA Selection Tool vous permet de générer une liste de commande complète à partir de votre sélection ou configuration de produits.
Vous trouverez TIA Selection Tool sur Internet.
(https://support.industry.siemens.com/cs/ww/fr/view/109767888/en)

SIMATIC Automation Tool

SIMATIC Automation Tool vous permet d'exécuter des opérations de masse pour des tâches de mise en service et de maintenance sur différentes stations SIMATIC S7 indépendamment de TIA Portal.
SIMATIC Automation Tool propose de nombreuses fonctions :

- Scan d'un réseau procédé PROFINET/Ethernet et identification de toutes les CPU connectées
- Affectation d'adresses (IP, sous-réseau, Gateway) et d'un nom d'appareil (périphérique PROFINET) à une CPU
- Transmission de la date et de l'heure de PG/PC convertie en heure UTC sur le module
- Chargement du programme sur la CPU
- Commutation du mode de fonctionnement RUN/STOP
- Localisation de la CPU par clignotement de LED
- Lecture des informations d'erreur de la CPU
- Lecture du tampon de diagnostic de la CPU
- Réinitialisation aux réglages usine
- Mise à jour du firmware de la CPU et des modules raccordés

Vous trouverez l'outil SIMATIC Automation Tool sur Internet.
(https://support.industry.siemens.com/cs/ww/fr/view/98161300)

PRONETA

SIEMENS PRONETA (analyse de réseau PROFINET) est un outil de mise en service et de diagnostic pour des réseaux PROFINET. PRONETA Basic dispose de 2 fonctions principales :

- Dans l'analyse réseau, vous obtenez une vue d'ensemble de la topologie PROFINET. Comparez une structure réelle avec une installation de référence ou effectuez des modifications de paramètres simples, p. ex. du nom et de l'adresse IP des appareils.
- Le "Test I/O" vous permet d'effectuer un test simple et rapide du câblage et de la configuration des modules d'une installation, avec une documentation des résultats du test.

Vous trouverez SIEMENS PRONETA Basic sur Internet :
(https://support.industry.siemens.com/cs/ww/fr/view/67460624)

En tant que produit sous licence, SIEMENS PRONETA Professional vous offre des fonctions supplémentaires. Il vous permet une gestion des actifs simple dans des réseaux PROFINET et assiste les exploitants d'installations d'automatisation dans l'acquisition automatisée de données des composants utilisés par une multitude de fonctions :

- L'interface utilisateur (API) offre un point d'accès dans la cellule d'automatisation afin d'automatiser les fonctions de scan via MQTT ou via une ligne de commande.
- Le diagnostic PROFIenergy permet, pour les appareils prenant en charge PROFIenergy, de détecter très rapidement, et modifier si nécessaire, le mode pause actuel ou la disponibilité.
- L'assistant d'enregistrement aide les développeurs PROFINET à lire et à écrire rapidement et simplement des enregistrements PROFINET acycliques, et ce, sans automate ni ingénierie.

Vous trouverez SIEMENS PRONETA Professional sur Internet.
(https://www.siemens.com/proneta-professional)

SINETPLAN

SINETPLAN, le Siemens Network Planner, vous assiste lorsque vous planifiez des systèmes et des réseaux d'automatisation sur la base de PROFINET. Dès la phase de planification, cet outil facilite le dimensionnement professionnel et prévisionnel de votre installation PROFINET. SINETPLAN vous aide en outre à optimiser le réseau, à en exploiter au mieux les ressources et à prévoir les réserves nécessaires. Vous évitez ainsi, dès la planification, des problèmes lors de la mise en service ou des défaillances en mode de production. Cela augmente la disponibilité de la production et contribue à l'amélioration de la sécurité de fonctionnement.

Les avantages en bref

- Optimisation du réseau grâce au calcul de la charge du réseau pour chaque port
- Disponibilité accrue en production par une analyse en ligne et une vérification des installations existantes
- Transparence avant la mise en service grâce à l'importation et à la simulation de projets STEP 7 existants
- Efficience grâce à la pérennité garantie des investissements et exploitation optimale des ressources

Vous trouverez SINETPLAN sur Internet.
(https://new.siemens.com/global/en/products/automation/industrial-communication/profinet/sinetplan.html)

1.2.3 Documentation technique de SIMATIC

Des documents SIMATIC supplémentaires complètent vos informations. Vous trouverez ces documents et leur utilisation via les liens et codes QR suivants.
Industry Online Support complète les possibilités pour obtenir des informations sur tous les sujets. Et les exemples d'applications vous assistent dans la résolution de vos tâches d'automatisation.

Vue d'ensemble de la documentation technique de SIMATIC

Vous trouverez ici une vue d'ensemble de la documentation relative à SIMATIC disponible dans Siemens Industry Online Support :

Industry Online Support International
(https://support.industry.siemens.com/cs/ww/fr/view/109742705)

Nous vous montrons dans une courte vidéo comment trouver la vue d'ensemble directement dans Siemens Industry Online Support et comment utiliser Siemens Industry Online Support sur votre terminal mobile :

Accès rapide à la documentation technique de produits d'automatisation par le biais d'une vidéo (https://support.industry.siemens.com/cs/fr/fr/view/109780491)

Vidéo YouTube : Siemens Automation Products - Technical Documentation at a Glance (https://youtu.be/TwLSxxRQQsA)

Conservation de la documentation

Conservez la documentation pour une utilisation ultérieure.
Pour la documentation jointe numériquement :

1. Après avoir reçu le produit, téléchargez la documentation associée au plus tard avant le premier montage/la première mise en service. Utilisez les options de téléchargement suivantes :
   - Industry Online Support International : (https://support.industry.siemens.com)
     La documentation est attribuée au produit via le numéro d'article. Vous trouverez le numéro d'article sur le produit et sur l'étiquette de l'emballage. Les produits dotés de nouvelles fonctionnalités non compatibles recevront un nouveau numéro d'article et une nouvelle documentation.
   - Lien d'identification :
     Si votre produit est marqué d'un ID Link, vous le reconnaîtrez avec code QR doté d'un cadre et d'un coin de cadre noir en bas à droite. L'ID Link vous dirige vers la plaque signalétique numérique de votre produit. Scannez le QR-Code figurant sur le produit ou sur l'étiquette d'emballage avec une caméra de smartphone, un scanner de code-barres ou une appli de lecture. Appelez l'ID Link.
2. Conservez cette version de la documentation.

Mise à jour de la documentation

La documentation du produit est mise à jour sous forme numérique. Des nouvelles caractéristiques sont mises à disposition dans une version mise à jour, en particulier en cas d'extension des fonctions.

1. Téléchargez la version actuelle comme décrit ci-dessus via l'Industry Online Support ou le lien ID.
2. Conservez également cette version de la documentation.

mySupport

"mySupport" vous permet de tirer au mieux profit de votre Industry Online Support.

Enregistrement	Pour bénéficier de toutes les fonctions de mySupport, vous devez vous enregistrer une fois. Après l'enregistrement, vous avez la possibilité de créer des filtres, des favoris et des onglets dans votre espace de travail personnel.
Demandes de support	En cas de demande d'assistance, vos coordonnées sont déjà renseignées et vous pouvez consulter à tout moment l'état d'avancement de vos demandes.
Documentation	Vous constituez votre bibliothèque personnelle dans la zone Documentation.
Favoris	Le bouton "Ajouter aux favoris mySupport" vous permet de marquer des contenus particulièrement intéressants ou fréquemment requis. Vous trouverez sous le point "Favoris" une liste de vos entrées marquées.
Dernières contributions consultées	Vous trouverez sous "Dernières contributions consultées" les dernières pages que vous avez appelées dans mySupport.

Données CAx	La zone Données CAx vous donne accès aux données de produit actuelles pour votre système CAx ou CAe. Quelques clics suffisent pour configurer votre pack à télécharger personnel : • Des images de produit, des plans cotés 2D, des modèles 3D, des schémas de connexion des appareils, des fichiers macro EPLAN • Des manuels, des caractéristiques, des instructions de service, des certificats et • Données de base des produits

Vous trouverez "mySupport" sur Internet. (https://support.industry.siemens.com/My/ww/fr)

Exemples d'application

Les exemples d'application mettent à votre disposition différents outils et exemples pour la résolution de vos tâches d'automatisation. Les solutions sont représentées en interaction avec plusieurs composants dans le système - sans se focaliser sur des produits individuels.
Vous trouverez les exemples d'application sur Internet.
(https://support.industry.siemens.com/cs/ww/fr/ps/ae)

Consignes de sécurité 2

2.1 Notes relatives à la cybersécurité

Siemens commercialise des produits et solutions comprenant des fonctions de cybersécurité industrielle qui contribuent à une exploitation sûre des installations, systèmes, machines et réseaux.

Pour protéger les installations, systèmes, machines et réseaux des cybermenaces, il est recommandé de mettre en œuvre un concept de cybersécurité industrielle intégré (et continuellement mis à jour) qui corresponde à l'état actuel de la technique. Les produits et solutions de Siemens constituent une partie de ce concept.

Il incombe aux clients d'empêcher tout accès non autorisé à ses installations, systèmes, machines et réseaux. Ces systèmes, machines et composants doivent uniquement être connectés au réseau d'entreprise ou à Internet si et dans la mesure où cela est nécessaire et seulement si des mesures de protection adéquates (ex : pare-feu et/ou segmentation du réseau) ont été prises.

Pour plus d'informations sur les mesures de cybersécurité industrielle possibles, voir sous (https://www.siemens.com/global/en/products/automation/topic-areas/industrial-cybersecurity.html).

Les produits et solutions Siemens font l'objet de développements continus pour être encore plus sûrs. Siemens recommande vivement d'effectuer des mises à jour dès que celles-ci sont disponibles et d'utiliser la dernière version des produits. L'utilisation de versions qui ne sont plus prises en charge et la non-application des dernières mises à jour peut augmenter le risque de cybermenaces pour nos clients.

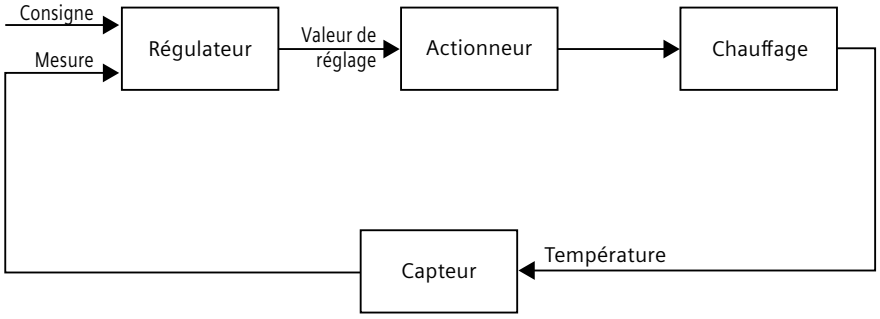
Pour être informé sur les mises à jour produit dès leur sortie, s'abonner au flux RSS Siemens Industrial Cybersecurity sur (https://new.siemens.com/global/en/products/services/cert.html) :

Bases de régulation 3

3.1 Boucles de régulation et actionneurs

Boucle de régulation

Un exemple simple de boucle de régulation est la régulation de la température ambiante par un chauffage. La température ambiante est mesurée à l'aide d'un capteur et conduite à un régulateur. Celui-ci compare la température ambiante actuelle à une consigne et calcule une valeur de réglage pour l'activation du chauffage.



Un régulateur PID correctement réglé atteint le plus vite possible la consigne et la maintient à une valeur constante. Après une modification de la valeur de réglage, la mesure ne change souvent qu'avec un certain retard. Ce comportement doit être compensé grâce au régulateur.

Actionneurs

L'actionneur fait partie de la boucle de régulation et est influencé par le régulateur. Le flux de masse ou énergétique se voit donc modifié
Le tableau suivant donne un aperçu sur l'utilisation des actionneurs.

Utilisation pour ...	Actionneur
Flux de masse sous forme liquide ou gazeuse	Vanne, volet, poussoir
Flux de masse solide, matières en vrac	Vanne de vidange, convoyeur à bande, distributeur vibrant
Flux énergétique électrique	Contact de commutation, électrovanne, relais, thyristor
	Résistance de réglage, transformateur réglable, transistor

On distingue les actionneurs suivants :

- Actionneurs proportionnels avec signal de réglage constant
  Les degrés d'ouverture, les écarts angulaires ou les positions sont pris proportionnellement à la valeur de réglage. La valeur de réglage a un effet analogue sur le processus au sein de la plage de réglage.
  Des entraînements pneumatiques à ressort font partie des actionneurs de ce groupe, mais également des entraînements moteur avec signalisation de position pour lesquels un système réglé de position est créé.

La valeur de réglage est générée par un régulateur continu, par ex. PID_Compact.

- Actionneurs proportionnels avec signal modulé en largeur d'impulsion
  Avec ces actionneurs, une impulsion de longueur proportionnelle à la valeur de réglage est fournie suivant le temps d'échantillonnage. L'actionneur, une résistance de chauffage ou une unité de refroidissement par exemple, est activé plus ou moins longtemps de manière synchrone selon la valeur de réglage.
  Le signal de réglage peut prendre de manière unipolaire les états "ON" et "OFF" ou représenter de manière bipolaire les valeurs "Ouvert/Fermé", "Avant/Arrière", "Accélérer/Freiner".
  La valeur de réglage est générée par un régulateur à deux échelons, par ex. PID_Compact avec modulation de largeur d'impulsions.
- Actionneurs à effet intégral avec signal de réglage à 3 échelons
  Les actionneurs sont fréquemment utilisés par des moteurs pour lesquels le facteur de marche est proportionnel à la course de réglage de l'organe d'inductance. Les vannes, volets, poussoirs ont font par ex. partie. Malgré les différentes formes, les actionneurs possèdent un point commun : ils correspondent à l'effet d'une action I sur l'entrée du système.
  La valeur de réglage est générée par un régulateur pas à pas, par ex. PID_3Step.

3.2 Systèmes réglés

Les propriétés d'un système réglé sont déterminées par des critères spécifiques au procédé/à la machine concerné(e) et ne sont quasiment pas influençables. En vue d'un bon résultat de régulation, il s'impose donc de sélectionner le type de régulateur le mieux adapté à ce type de système réglé et d'adapter ce régulateur au comportement du système dans le temps. De bonnes connaissances du type et des données caractéristiques du système réglé sont donc indispensables pour la configuration de l'action P, I et D.

Types de systèmes réglés

Les systèmes réglés sont classés suivant leur réaction à une variation brusque de la valeur de réglage.
On distingue les types de systèmes réglés suivants :

- Systèmes réglés avec compensation
  - Systèmes réglés P
  - Systèmes réglés PT1
  - Systèmes réglés PT2
- Systèmes réglés sans compensation
- Systèmes réglés avec ou sans temps mort

Systèmes réglés avec compensation

Systèmes réglés P

Dans les systèmes réglés proportionnels, la mesure suit la valeur de réglage presque sans retard. Le rapport entre mesure et valeur de réglage est indiqué par le coefficient proportionnel Gain du système réglé.

Exemples :

- Vanne dans un système de tuyaux
- Diviseur de tension
- Rapport de réduction dans les systèmes hydrauliques

Systèmes réglés PT1

Dans un système réglé PT1, la mesure change d'abord proportionnellement à la modification de la valeur de réglage. Avec le temps, la mesure change plus lentement, donc de manière décalée, jusqu'à atteindre une valeur finale.

Exemples :

- Système d'amortisseurs à ressort
- Chargement de circuits RC
- Réservoir d'eau chauffé à la vapeur.

Les mêmes constantes temporelles s'appliquent fréquemment pour les opérations de chauffage ou de refroidissement ou les courbes de chargement/déchargement. Si les constantes de temps divergent, la régulation devient nettement plus difficile.

Systèmes réglés PT2

Dans un système réglé PT2, la mesure ne change tout d'abord pas lors d'un saut de la valeur de réglage, elle augmente ensuite selon une pente croissante et tend enfin vers la consigne selon une pente décroissante. Le système réglé affiche un comportement de transmission proportionnel avec une temporisation de second rang.

Exemples :

- Régulation de pression
- Débit
- Régulation de la température

Systèmes réglés sans compensation

Systèmes réglés sans compensation affichent un comportement intégral. La mesure vise une valeur infinie.

Exemple :

- Ecoulement d'un liquide dans un réservoir

Systèmes réglés avec temps mort

Un temps mort est toujours le temps de transport ou d'exécution jusqu'à la mesure de la modification sur l'entrée/la sortie du système.

La mesure est temporisée par le temps mort pour les systèmes réglés avec temps mort.

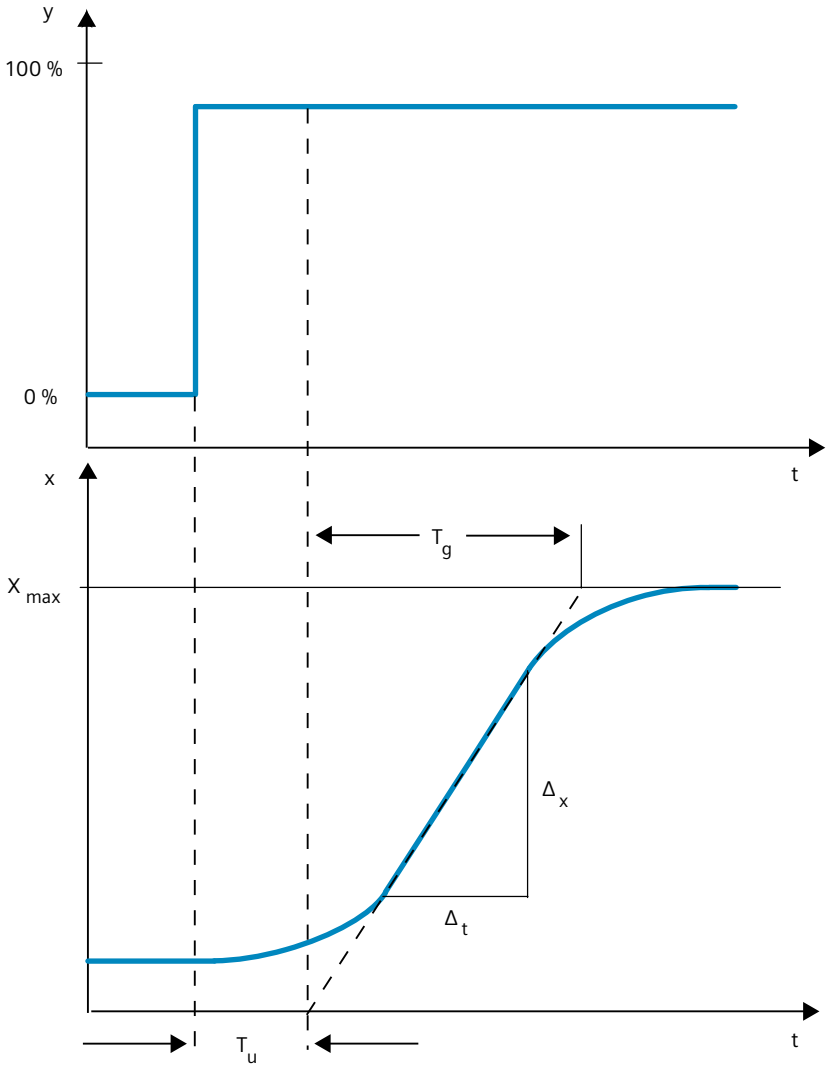
Exemple :

- Convoyeur à bande

3.3 Caractéristiques du système réglé

Détermination de la réponse temporelle à partir de la réponse indicielle

La réponse temporelle du système réglé peut être déterminée par la courbe de la mesure x en fonction du temps après un changement brusque de la valeur de réglage. La plupart des systèmes réglés sont ceux avec compensation.



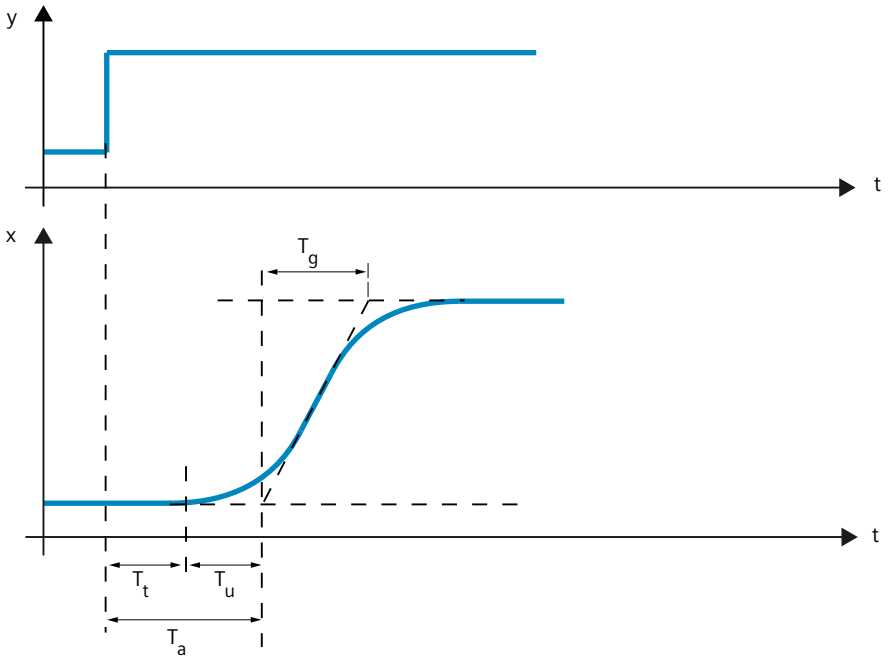
La réponse temporelle peut être définie de manière approximative par les grandeurs temps de retard T_u, temps de compensation T_g et valeur maximale X_{max}. Les grandeurs sont déterminées par la création de tangentes à la valeur maximale et au point d'inflexion de la réponse indicielle. Dans de nombreux cas, l'emploi de la fonction de transition jusqu'à la valeur maximale est impossible, car la mesure ne doit pas dépasser certaines valeurs. On utilise donc la vitesse de montée v_{max} pour identifier le système réglé ($v_{max} = \Delta_x/\Delta_t$).

Le rapport T_u/T_g ou $T_u \times v_{max}/X_{max}$ permet d'évaluer l'aptitude à la régulation du système réglé. On a :

Type de système	T_u / T_g	Aptitude à la régulation du système réglé
I	< 0,1	régulation facile
II	0,1 à 0,3	régulation juste possible
III	> 0,3	régulation difficile

Influence du temps mort sur l'aptitude à la régulation du système réglé

Un système réglé avec temps mort et compensation réagit de la manière suivante à un saut de la valeur de réglage.



T_t	Temps mort
T_u	Temps de retard
T_g	Temps de compensation
y	Valeur de réglage
x	Mesure

L'aptitude à la régulation du système réglé avec temps mort et compensation est déterminé par le rapport entre T_t et T_g. T_t doit être petit par rapport à T_g. Règle à appliquer :
$T_t/T_g \leq 1$

Vitesse de réaction de systèmes réglés

Les systèmes réglés peuvent être jugés selon ces valeurs :
$T_u < 0,5$ min, $T_g < 5$ min = système réglé rapide
$T_u > 0,5$ min, $T_g > 5$ min = système réglé lent

Paramètres de certains systèmes réglés

Grandeur physique	Système réglé	Temps de retard T_u	Temps de compensation T_g	Vitesse de montée v_{max}
Température	petit four électrique	0,5 à 1 min	5 à 15 min	jusqu'à 60 K/min
	grand four électrique de recuit	1 à 5 min	10 à 20 min	jusqu'à 20 K/min
	grand four à gaz de recuit	0,2 à 5 min	3 à 60 min	1 à 30 K/min
	colonne de distillation	1 à 7 min	40 à 60 min	0,1 à 0,5 °C/s
	autoclave (2,5 m^3)	0,5 à 0,7 min	10 à 20 min	pas de mention
	autoclaves haute pression	12 à 15 min	200 à 300 min	pas de mention
	surchauffeur de vapeur	30 s à 2,5 min	1 à 4 min	2 °C/s
	machines de moulage par injection	0,5 à 3 min	3 à 30 min	5 à 20 K/min
	extrudeuse	1 à 6 min	5 à 60 min	
	machines d'emballage	0,5 à 4 min	3 à 40 min	2 à 35 K/min
	chauffage de locaux	1 à 5 min	10 à 60 min	1 °C/min
Débit	conduite de gaz	0 à 5 s	0,2 à 10 s	sans objet
	conduite de liquide	aucun	aucun	
Pression	conduite de gaz	aucun	0,1 s	sans objet
	chaudière à tambour, à foyer gaz ou fioul	aucun	150 s	sans objet
	chaudière à tambour avec concasseurs à marteaux fixes	1 à 2 min	2 à 5 min	sans objet
Niveau du réservoir	chaudière à tambour	0,6 à 1 min	pas de mention	0,1 à 0,3 cm/s
Vitesse de rotation	petite motorisation électrique	aucun	0,2 à 10 s	sans objet
	grande motorisation électrique	aucun	5 à 40 s	sans objet
	turbine à vapeur	aucun	pas de mention	50 min^{-1}
Tension électrique	petits générateurs	aucun	1 à 5 s	sans objet
	grands générateurs	aucun	5 à 10 s	sans objet

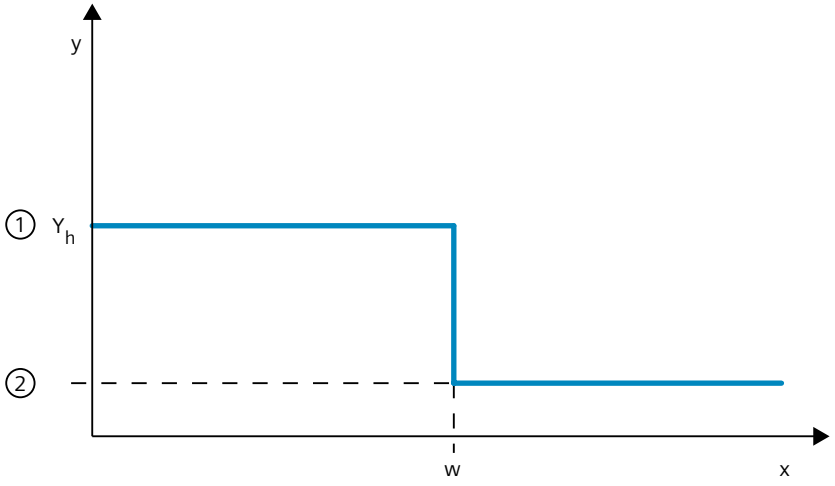
3.4 Régulateur à impulsions

Régulateur à deux échelons sans chaîne de réaction

Les régulateurs à deux échelons possèdent les états "MARCHE" et "ARRÊT" comme fonction de commutation. Cela correspond à une puissance de 100 % ou de 0 %. Ce comportement provoque une oscillation entretenue de la mesure x autour de la valeur de consigne w.

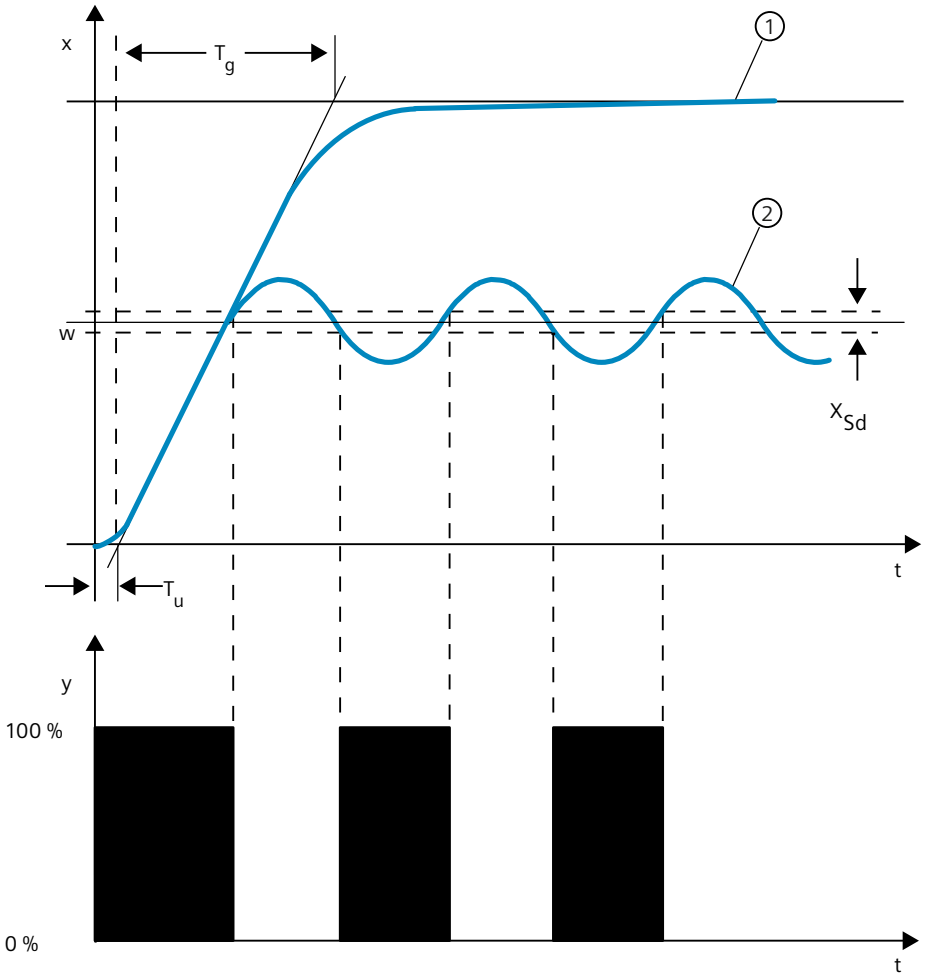
L'amplitude et la durée d'oscillation croissent avec le rapport entre le temps de retard T_u et le temps de compensation T_g du système réglé. Ces régulateurs sont utilisés essentiellement pour des régulations de température simples (par exemple pour des fours à chauffage électrique direct) ou bien comme indicateur de valeur limite.

Le graphique ci-dessous illustre la courbe caractéristique d'un régulateur à deux échelons.



① Activé
② Désactivé
Y_h Plage de réglage
w Consigne

Le graphique ci-dessous illustre la fonction régulatrice d'un régulateur à deux échelons.



①	Fonction de transition sans régulateur
②	Fonction de transition avec régulateur à deux échelons
T_u	Temps de retard
T_g	Temps de compensation
X_{Sd}	Différence de commutation

Régulateur à deux échelons avec chaîne de réaction

Le comportement de régulateurs à deux échelons pour des systèmes réglés à temps de retard plus grands, par exemple des fours dont l'enceinte est séparée du chauffage, peut être amélioré par des chaînes de réaction électroniques.

La chaîne de réaction permet d'augmenter la fréquence de commutation du régulateur et donc de réduire l'amplitude de la mesure. De plus, en mode dynamique, les résultats de la régulation seront nettement améliorés. La limite de la fréquence de commutation est définie par le niveau de sortie. Pour des actionneurs mécaniques, tels des relais et contacteurs, elle ne doit pas dépasser 1 à 5 commutations par minute. Dans le cas de sorties de tension et de courant binaires suivies d'actionneurs à thyristor ou à triac, il est possible de choisir des fréquences de commutation élevées se situant bien au-delà de la fréquence limite du système réglé.

Etant donné que les impulsions de commutation ne peuvent plus être détectées à la sortie du système réglé, on obtient des résultats comparables à ceux de régulateurs à action continue.

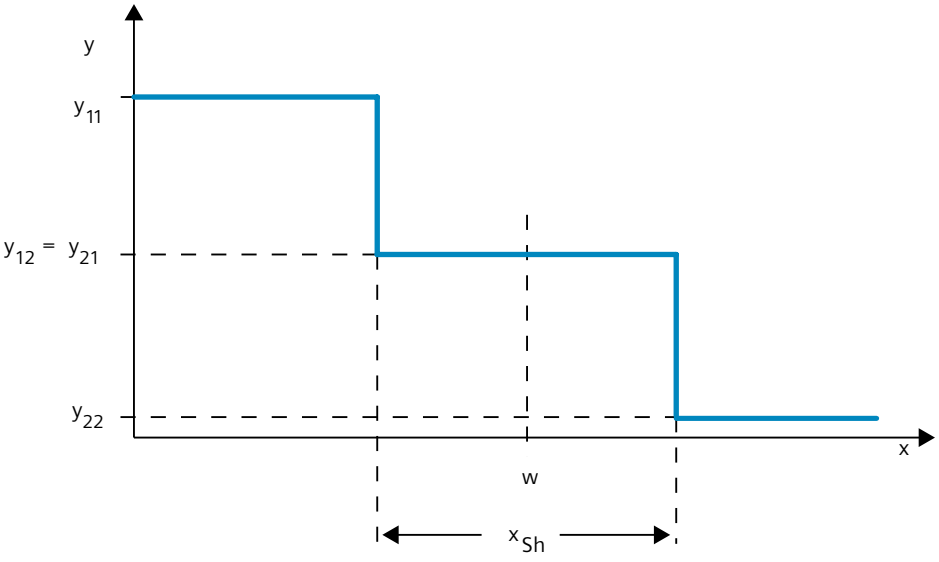
La valeur de réglage est générée par modulation de largeur d'impulsions de la valeur de réglage d'un régulateur continu.

Les régulateurs à deux échelons à chaîne de réaction sont utilisés pour la régulation de température dans des fours, sur des machines de transformation dans les industries des matières plastiques, textile, du papier, du caoutchouc et agroalimentaire ainsi que pour les appareillages de chauffage et de refroidissement.

Régulateur à trois échelons

Les régulateurs à trois échelons sont utilisés pour le chauffage/refroidissement. Ils ont comme sortie deux points de commutation. Les résultats de régulation peuvent être optimisés par des structures de réactions électroniques. Les domaines d'application de ces régulateurs sont les chambres chaudes, froides, climatiques et chauffages d'outillages pour machines de transformation de matières plastiques.

Le graphique ci-dessous illustre la courbe caractéristique d'un régulateur à trois échelons.

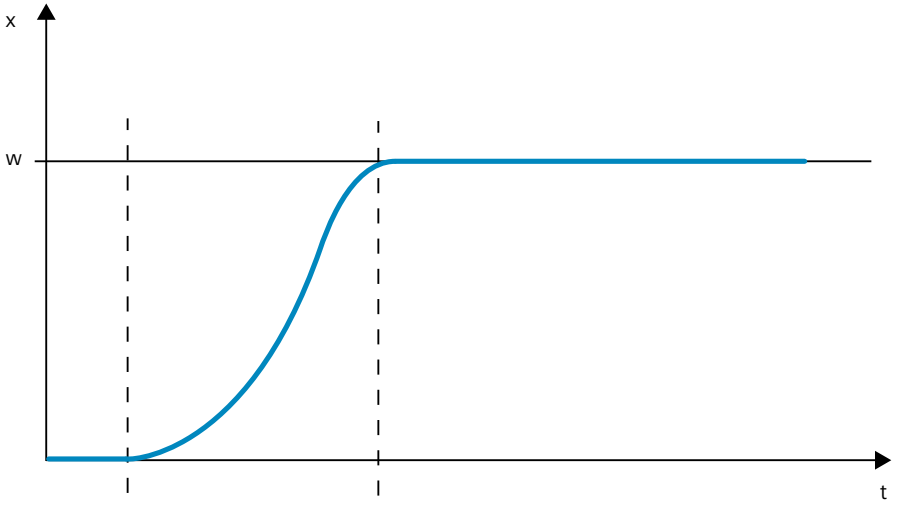


y	Valeur de réglage, par ex. : y11 = 100 % chauffage y12 = 0 % chauffage y21 = 0 % refroidissement y22 = 100 % refroidissement
x	Grandeur physique de la mesure, par exemple température en °C
w	Consigne
x_{Sh}	écart entre point de commutation 1 et point de commutation 2

3.5 Comportement de perturbation et de référence

Comportement de référence

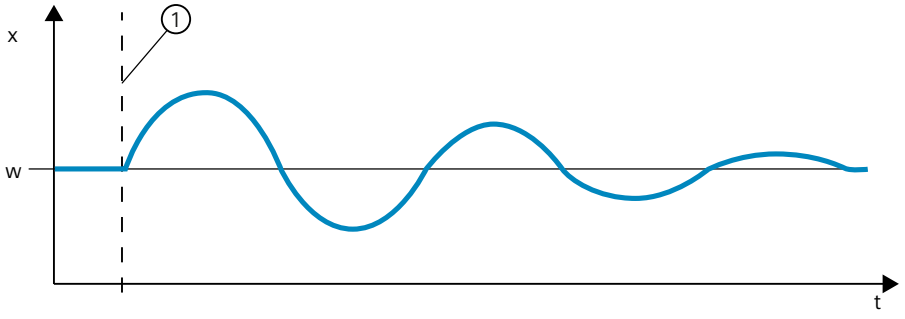
La mesure doit suivre le plus rapidement possible la modification de la consigne. Plus le temps durant lequel la nouvelle consigne sera atteinte est court, plus le comportement de référence sera efficace et moins importante sera l'oscillation de la mesure.



x Mesure
w Consigne

Comportement de perturbation

La consigne est influencée par des grandeurs perturbatrices. Le régulateur doit donc éliminer le plus vite possible le signal d'écart survenu par la suite. Plus le temps durant lequel la consigne sera atteinte est court, plus le comportement de perturbation sera efficace et moins importante sera l'oscillation de la mesure.



x Mesure
w Consigne
① Point d'action d'une grandeur perturbatrice

Une grandeur perturbatrice est compensée par un régulateur à l'action I. La qualité de la régulation n'est pas diminuée par une grandeur perturbatrice à effet constant, étant donné que le signal d'écart est relativement contant. Une grandeur perturbatrice dynamique

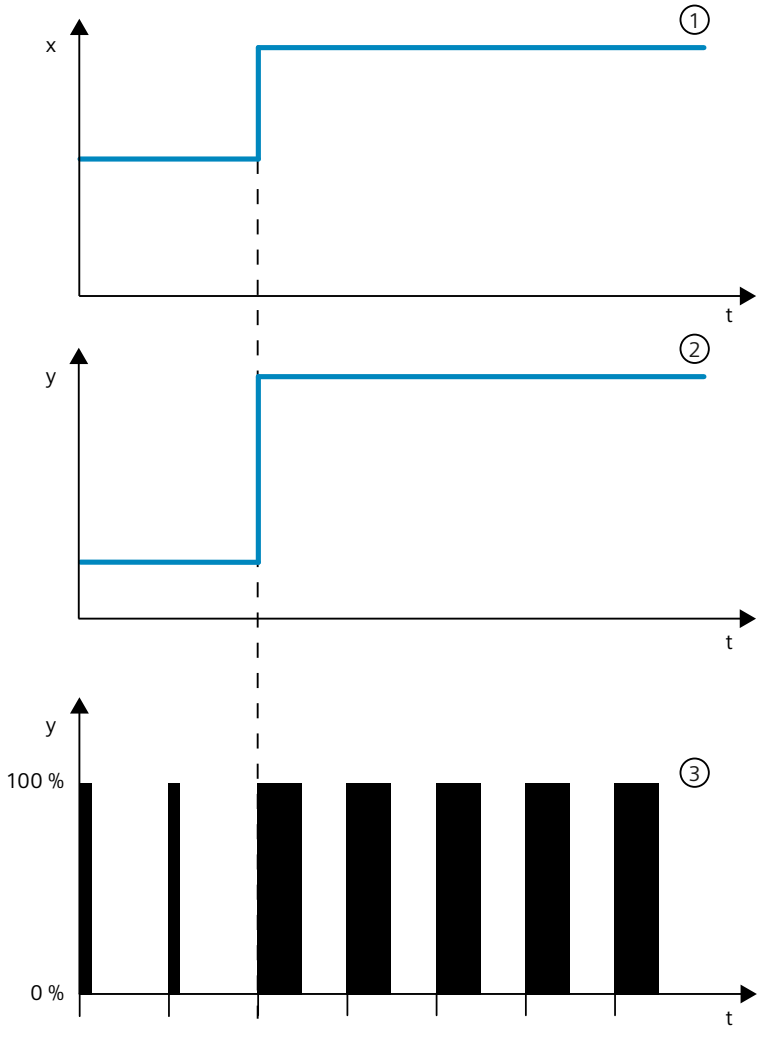
influence plus la qualité de la régulation étant donné que le signal d'écart n'est jamais le même. Seule l'action I à croissance lente permet de réduire le signal d'écart.
Une grandeur perturbatrice mesurable peut entrer en compte dans le système réglé. La réaction du régulateur peut donc être nettement accélérée.

3.6 Comportement de régulation en fonction des structures de réaction

Comportements des régulateurs

Plus le régulateur est adapté au comportement temporel du système réglé, plus il sera précis lors du réglage de la consigne et réagira de manière optimale aux grandeurs perturbatrices
Le circuit de réaction peut avoir un comportement proportionnel (P), proportionnel-différentiel (PD), proportionnel- intégral (PI) ou proportionnel-intégral-différentiel (PID).
Si une fonction transitoire est émise pour le signal d'écart, différentes réponses indicielles sont générées selon le régulateur.

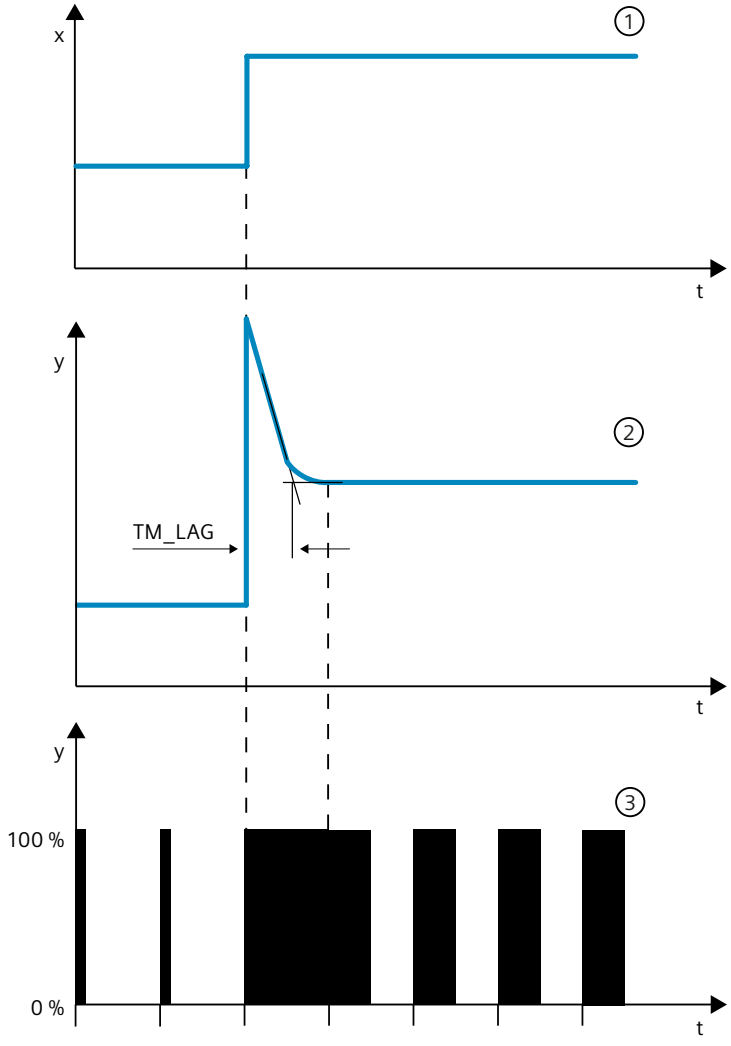
Réponse indicielle d'un régulateur PID



① Signal d'écart
② Valeur de réglage d'un régulateur continu.
③ Valeur de réglage d'un régulateur à impulsions

Formule pour régulateur P
La valeur de réglage et le signal d'écart sont directement proportionnels, à savoir :
Valeur de réglage = coefficient proportionnel × signal d'écart
y = GAIN × x

Réponse indicielle d'un régulateur PD



①	Signal d'écart
②	Valeur de réglage d'un régulateur continu.
③	Valeur de réglage d'un régulateur à impulsions
TM_LAG	Temps de retard de l'action D

Formule pour régulateur PD

Pour la réponse indicielle du régulateur PD dans la plage de temps, on a :

$$y = GAIN \cdot X_W \cdot \left(1 + \frac{TD}{TM\_LAG} \cdot e^{\frac{-t}{TM\_LAG}} \right)$$

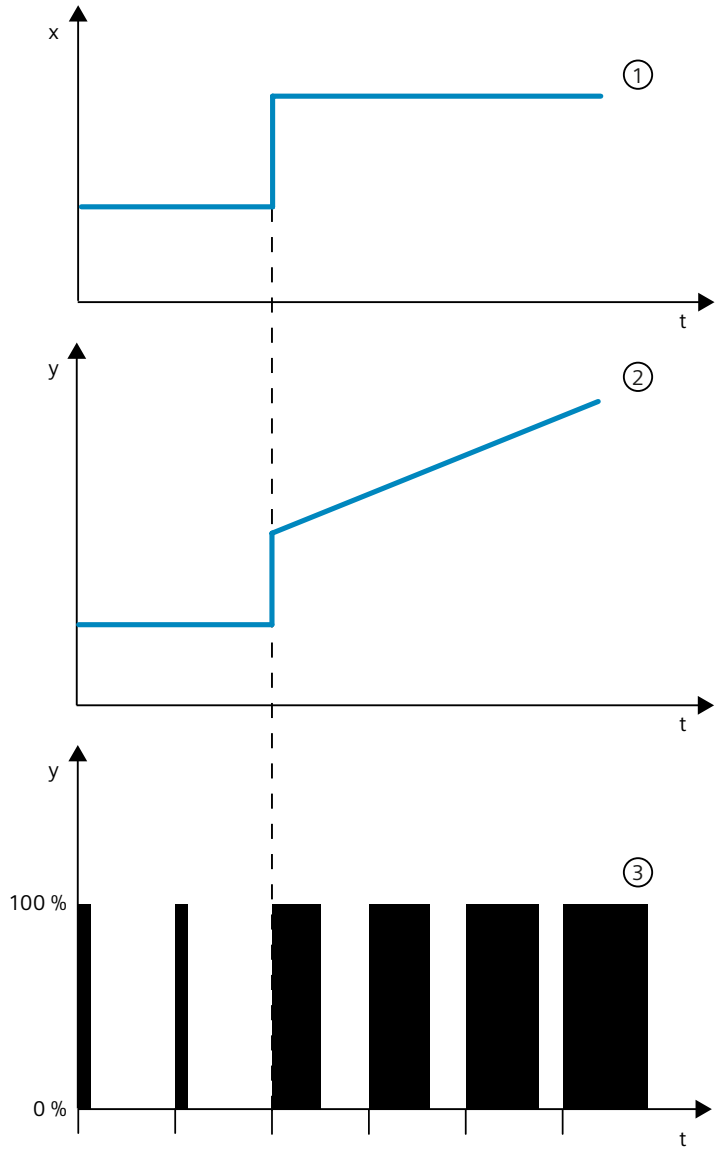
t = depuis le saut du signal d'écart

L'action D génère une valeur de réglage en fonction de la vitesse à laquelle la mesure change. Une action D intégrale ne convient pas à la régulation, puisque seule une modification de la mesure provoque une modification de la valeur de réglage. Quand la mesure reste constante, la valeur de réglage ne change plus.

Combinée avec une action P, l'action D améliore le comportement de perturbation Les perturbations ne sont pas complètement éliminées. Le bon comportement dynamique est un avantage. Lors du démarrage et de la modification de consigne, on atteint une transition sans oscillations et bien amortie.

Un régulateur à action D n'est pas indiqué lorsqu'un système réglé a des grandeurs de mesure pulsatoires, par exemple pour des régulations de pression ou de débit.

Réponse indicielle d'un régulateur PI



① Signal d'écart

② Valeur de réglage d'un régulateur continu.

③ Valeur de réglage d'un régulateur à impulsions

Une action I dans le régulateur totalise le signal d'écart dans le temps. Le régulateur ajustera donc tant que le signal d'écart n'est pas rattrapé. Un signal d'écart permanent survient en cas de régulateur à action purement P. Il peut être corrigé par une action I dans le régulateur.

Dans la pratique, une combinaison des actions P, I et D est idéale en fonction des exigences imposées au comportement de régulation. La réponse temporelle des actions individuelles peut être décrite par les paramètres du régulateur suivants : coefficient d'action proportionnelle GAIN, temps de dosage d'intégration TI (action I) et temps d'anticipation TD (action D).

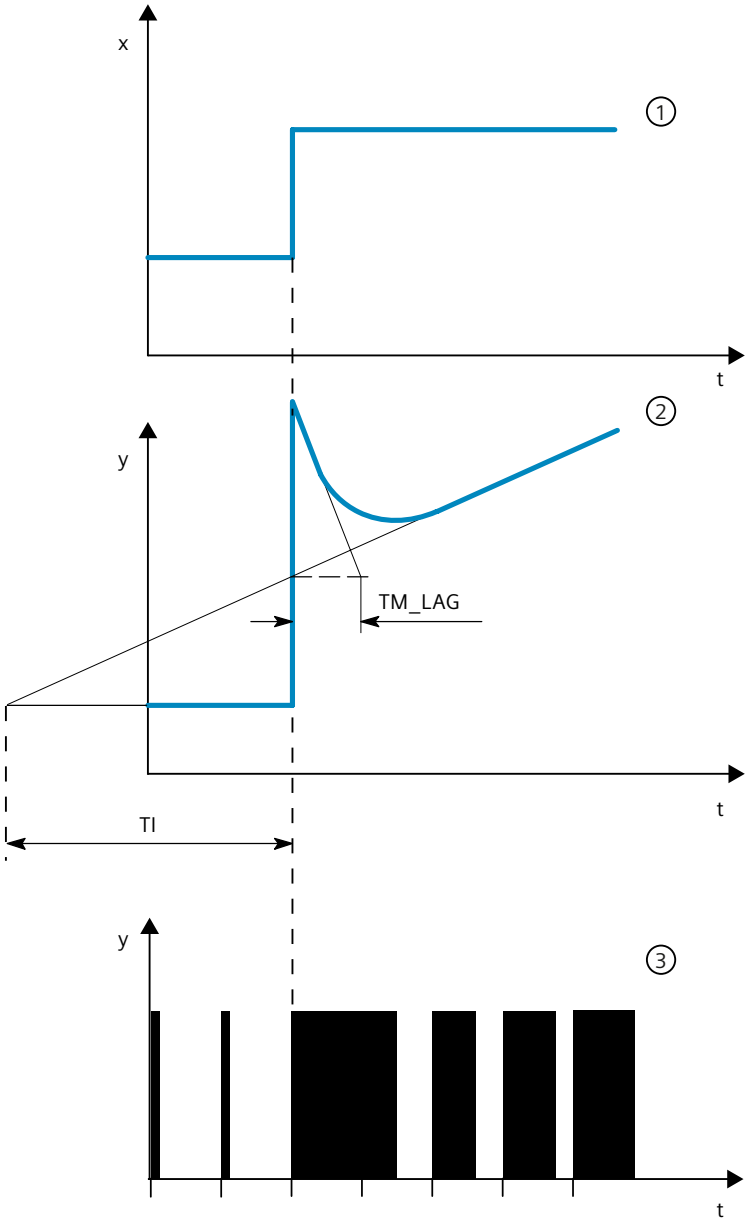
Formule pour régulateur PI

Pour la réponse indicielle du régulateur PI dans la plage de temps, on a :

$$y = GAIN \cdot X_W \cdot \left(1 + \frac{1}{TI \cdot t} \right)$$

t = depuis le saut du signal d'écart

Réponse indicielle d'un régulateur PID



①	Signal d'écart
②	Valeur de réglage d'un régulateur continu.
③	Valeur de réglage d'un régulateur à impulsions
TM_LAG	Temps de retard de l'action D
T_i	Temps d'intégration

Formule pour régulateur PID

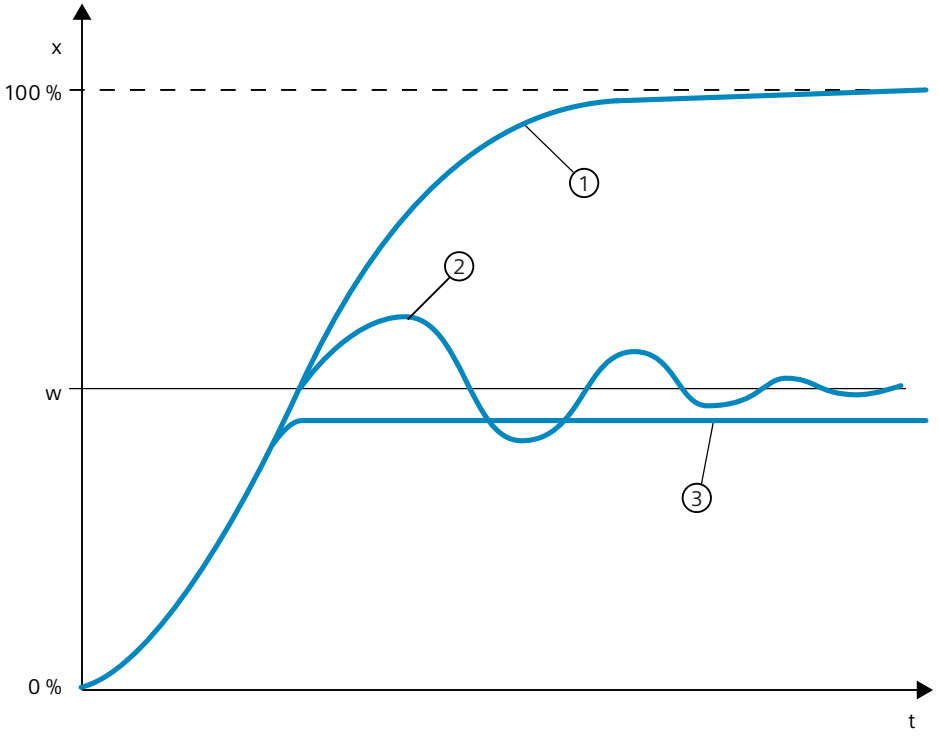
Pour la réponse indicielle du régulateur PID dans la plage de temps, on a :

$$y = GAIN \cdot X_w \cdot \left(1 + \frac{1}{TI \cdot t} + \frac{TD}{TM\_LAG} \cdot e^{\frac{-t}{TM\_LAG}} \right)$$

t = depuis le saut du signal d'écart

Comportement d'un système réglé pour différentes structures de régulateur

La plupart des régulations utilisées en génie des procédés peuvent être maîtrisées avec un régulateur à comportement PI. Dans le cas de systèmes réglés à grand temps de retard, par exemple les régulations de température, le résultat peut être amélioré grâce à un régulateur à comportement PID.



①	Aucun régulateur
②	Régulateur PID
③	Régulateur PD
w	Consigne
x	Mesure

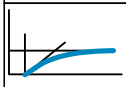
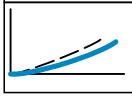
Les régulateurs à comportement PI et PID présente l'avantage qu'après le régime transitoire, la mesure ne présente pas de déviation par rapport à la consigne. La mesure oscille au démarrage via la consigne.

3.7 Sélection de la structure du régulateur pour un système réglé donné

Sélection des structures appropriées de régulateurs

Vous n'obtenez un résultat de régulation optimal que si vous sélectionnez une structure de régulateur adaptée au système réglé et pouvant être ajusté dans le cadre de certaines limites au système réglé.

Le tableau suivant indique quelle structure de régulateur convient à quel système réglé.

Système réglé		Structure de régulateur			
		P	**PD**	**PI**	**PID**
	Seulement avec temps mort	non approprié	non approprié	approprié	non approprié
	PT1 avec temps mort	non approprié	non approprié	bien approprié	bien approprié
	PT 2 avec temps mort	non approprié	approprié sous conditions	bien approprié	bien approprié
	d'ordre supérieur	non approprié	non approprié	approprié sous conditions	bien approprié
	sans compensation	bien approprié	bien approprié	bien approprié	bien approprié

Le tableau suivant vous donne un aperçu sur les structures de régulateur adaptées aux grandeurs physiques.

Grandeur physique	Structure de régulateur			
	P	**PD**	**PI**	**PID**
	Signal d'écart résiduel		**Sans signal d'écart résiduel**	
Température	pour besoins peu exigeants et pour systèmes P avec $T_u/Tg < 0,1$	bien approprié	types de régulateurs les mieux adaptés aux besoins exigeants (à l'exception des régulateurs spéciaux adaptés au cas par cas)	
Pression	adapté si pas de temps de retard notable	non approprié	types de régulateurs les mieux adaptés aux besoins exigeants (à l'exception des régulateurs spéciaux adaptés au cas par cas)	
Débit	non approprié car zone GAIN nécessaire la plupart du temps trop grande	non approprié	utilisable, mais régulateur I seul souvent mieux	à peine nécessaire

3.8 Réglage des paramètres PID

Formule essentielle pour le réglage des paramètres

Structure de régulateur	Réglages
P	GAIN ≈ $v_{max} \times T_u$ [°C]
PI	GAIN ≈ 1,2 × $v_{max} \times T_u$ [°C] TI ≈ 4 × T_u [min]
PD	GAIN ≈ 0,83 × $v_{max} \times T_u$ [°C] TD ≈ 0,25 × $v_{max} \times T_u$ [min] TM_LAG ≈ 0,5 × TD[min]
PID	GAIN ≈ 0,83 × $v_{max} \times T_u$ [°C] TI ≈ 2 × T_u [min] TD ≈ 0,4 × T_u [min] TM_LAG ≈ 0,5 × TD[min]
PD/PID	GAIN ≈ 0,4 × $v_{max} \times T_u$ [°C] TI ≈ 2 × T_u [min] TD ≈ 0,4 × T_u [min] TM_LAG ≈ 0,5 × TD[min]

Au lieu de $v_{max} = \Delta_x / \Delta_t$ il est possible d'utiliser X_{max} / T_g.

Sur les régulateurs à structure PID, le réglage du temps de dosage d'intégration et du temps de dérivation est le plus souvent combiné.

Le rapport TI/TD se situe entre 4 et 5 et est optimal pour la plupart des systèmes réglés.

Le non-respect du temps dérivation TD est non critique pour les régulateurs PD.

Sur les régulateurs PI ou PID, des oscillations de régulation se produisent lorsque le temps d'action par intégration TI est choisi de plus de la moitié trop petit.

Un temps de dosage d'intégration trop élevé ralentit l'élimination des perturbations. On ne peut pas s'attendre à ce que les boucles de régulation fonctionnent "de manière optimale" après les premiers réglages de paramètres. L'expérience montre qu'un réajustement est toujours nécessaire lorsqu'on est en présence d'un système réglé "difficilement régulable" avec $T_u / T_g > 0,3$.

Configuration d'un régulateur logiciel

4

4.1 Présentation des régulateurs de logiciel

Pour configurer un régulateur de logiciel, il vous faut une instruction avec l'algorithme de réglage et un objet technologique. L'objet technologique d'un régulateur de logiciel correspond au DB d'instance de l'instruction. La configuration du régulateur est enregistrée dans l'objet technologique. À la différence des DB d'instance d'autres instructions, les objets technologiques ne sont pas classés parmi les ressources du programme, mais sous CPU > Objets technologiques.

Objets technologiques et instructions

CPU	Bibliothèque	Instruction	Objet technologique	Description
S7-1200	Compact PID	PID_Compact V1.x	PID_Compact V1.x	Régulateur PID universel avec optimisation intégrée
S7-1200		PID_3Step V1.x	PID_3Step V1.x	Régulateur PID pour vannes avec optimisation intégrée
S7-1500 S7-1200 V4.x		PID_Compact V2.x	PID_Compact V2.x	Régulateur PID universel avec optimisation intégrée
S7-1500 S7-1200 V4.x		PID_3Step V2.x	PID_3Step V2.x	Régulateur PID pour vannes avec optimisation intégrée
S7-1500 ≥ V1.7 S7-1200 ≥ V4.1		PID_Temp V1.x	PID_Temp V1.x	Régulateur de température PID universel avec optimisation intégrée
S7-1500 ≥ V3.1		PID_Compact V3.x	PID_Compact V3.x	Régulateur PID universel avec optimisation intégrée
S7-1500/300/400	Fonctions de base PID	CONT_C	CONT_C	Régulateur à action continue
S7-1500/300/400		CONT_S	CONT_S	Régulateur pas-à-pas pour actionneurs intégrateurs
S7-1500/300/400		PULSEGEN	-	Générateur d'impulsions pour actionneur à action proportionnelle
S7-1500/300/400		TCONT_CP	TCONT_CP	Régulateur de température continu avec générateur d'impulsions
S7-1500/300/400		TCONT_S	TCONT_S	Régulateur de température pour actionneurs intégrateurs
S7-300/400	PID Self Tuner	TUN_EC	TUN_EC	Optimisation d'un régulateur continu
S7-300/400		TUN_ES	TUN_ES	Optimisation d'un régulateur pas-à-pas
S7-300/400	Standard PID Control (progiciel optionnel PID Professional)	PID_CP	PID_CP	Régulateur continu avec générateur d'impulsions
S7-300/400		PID_ES	PID_ES	Régulateur pas-à-pas pour actionneurs intégrateurs
S7-300/400		LP_SCHED	-	Distribution des appels de régulateur

CPU	Bibliothèque	Instruction	Objet technologique	Description
S7-300/400	Modular PID Control (progiciel optionnel PID Professional)	A_DEAD_B	-	Filtrage des signaux perturbateurs provenant d'un signal d'écart
S7-300/400		CRP_IN	-	Mise à l'échelle du signal d'entrée analogique
S7-300/400		CRP_OUT	-	Mise à l'échelle du signal de sortie analogique
S7-300/400		DEAD_T	-	Emission retardée du signal d'entrée
S7-300/400		DEADBAND	-	Suppression des légères fluctuations de la mesure
S7-300/400		DIF	-	Différenciation du signal d'entrée par le temps
S7-300/400		ERR_MON		Surveillance du signal d'écart
S7-300/400		INTEG	-	Intégration du signal d'entrée par le temps
S7-300/400		LAG1ST	-	Élément de retard de 1er ordre
S7-300/400		LAG2ND	-	Élément de retard de 2e ordre
S7-300/400		LIMALARM	-	Signalement des valeurs limites
S7-300/400		LIMITER	-	Limitation de la valeur de réglage
S7-300/400		LMNGEN_C	-	Calcul de la valeur de réglage pour le régulateur continu
S7-300/400		LMNGEN_S	-	Calcul de la valeur de réglage pour le régulateur pas-à-pas
S7-300/400		NONLIN	-	Linéarisation du signal de capteur
S7-300/400		NORM	-	Normalisation physique de la mesure
S7-300/400		OVERRIDE	-	Commutation des valeurs de réglage de 2 régulateurs PID sur 1 actionneur
S7-300/400		PARA_CTL	-	Commutation des jeux de paramètres
S7-300/400		PID	-	Algorithme PID
S7-300/400		PUSLEGEN_M	-	Génération d'impulsions pour actionneur à action proportionnelle
S7-300/400		RMP_SOAK	-	Spécification des consignes selon le programme
S7-300/400		ROC_LIM	-	Limitation de la vitesse de variation
S7-300/400		SCALE_M	-	Mise à l'échelle de la mesure
S7-300/400		SP_GEN	-	Spécification manuelle de la consigne
S7-300/400		SPLT_RAN	-	Division des plages de la valeur de réglage
S7-300/400		SWITCH	-	Commutation des valeurs analogiques
S7-300/400		LP_SCHED_M	-	Distribution des appels de régulateur

4.2 Étapes de la configuration d'un régulateur logiciel

Configurez tous les régulateurs logiciels selon le même schéma :

Etape	Description
1	Ajouter l'objet technologique (Page 45)
2	Configurer l'objet technologique (Page 46)
3	Appeler l'instruction dans le programme utilisateur (Page 47)
4	Charger un objet technologique dans l'appareil (Page 48)
5	Mise en service du régulateur logiciel (Page 49)
6	Enregistrer les paramètres PID optimisés dans le projet (Page 49)
7	Comparer valeurs (Page 53)
8	Afficher les instances d'un objet technologique (Page 75)

4.3 Ajouter des objets technologiques

Ajouter un objet technologique dans le navigateur de projet

Lorsque vous ajoutez un objet technologique, un DB d'instance est généré pour l'instruction de cet objet technologique. Ce DB d'instance contient la configuration de l'objet technologique.

Condition

Un projet contenant une CPU est créé.

Marche à suivre

Pour ajouter un objet technologique, procédez de la manière suivante :

1. Dans le navigateur du projet, ouvrez le dossier de la CPU.
2. Ouvrez le dossier "Objets technologiques".
3. Effectuez un double clic sur "Ajouter nouvel objet".
   La boîte de dialogue "Ajouter nouvel objet" s'ouvre.
4. Cliquez sur le bouton "Régulateur PID".
   Tous les régulateurs PID disponibles pour cette CPU s'affichent.
5. Sélectionnez l'instruction pour l'objet technologique, p. ex. PID_Compact.
6. Entrez un nom spécifique à l'objet technologique dans le champ de saisie "Nom".
7. Si vous souhaitez modifier le numéro proposé pour le bloc de données du DB d'instance, sélectionnez l'option "Manuel".
8. Si vous souhaitez fournir vos propres informations sur l'objet technologique, cliquez sur "Informations complémentaires".
9. Confirmez en cliquant sur "OK".

Résultat

Le nouvel objet technologique est généré et rangé dans le dossier "Objets technologiques" dans le navigateur du projet. L'objet technologique est utilisé lorsque l'instruction dudit objet technologique est appelée dans un OB d'alarme cyclique.

REMARQUE

Vous pouvez cocher la case "Ajouter et ouvrir" dans le bas de la boîte de dialogue. Ainsi, dès qu'il aura été ajouté, la configuration de l'objet technologique s'ouvrira.

4.4 Configurer les objets technologiques

Vous pouvez configurer de deux manières les propriétés d'un objet technologique sur une CPU S7-1200.

- Dans la fenêtre d'inspection de l'éditeur de programmation
- Dans l'éditeur de configuration

Vous pouvez uniquement configurer les propriétés d'un objet technologique sur une CPU S7-300/400 dans l'éditeur de configuration.

Fenêtre d'inspection de l'éditeur de programmation

Dans la fenêtre d'inspection de l'éditeur de programmation, vous pouvez uniquement configurer les paramètres nécessaires au fonctionnement.

En mode en ligne, les valeurs hors ligne des paramètres sont affichées. Vous pouvez uniquement modifier les valeurs en ligne dans la fenêtre Mise en service.

Pour ouvrir la fenêtre d'inspection d'un objet technologique, procédez de la manière suivante :

1. Dans le navigateur du projet, ouvrez le dossier "Blocs de programme".
2. Double-cliquez sur le bloc (OB d'alarme cyclique) dans lequel vous appelez l'instruction du régulateur logiciel.
   Ce bloc s'ouvre dans la zone de travail.
3. Cliquez sur l'instruction du régulateur logiciel.
4. Dans la fenêtre d'inspection, sélectionnez successivement les onglets "Propriétés" et "Configuration".

Fenêtre de configuration

Chaque objet technologique possède une fenêtre de configuration spécifique dans laquelle vous pouvez configurer toutes les propriétés.

Pour ouvrir la fenêtre de configuration d'un objet technologique, procédez de la manière suivante :

1. Dans le navigateur du projet, ouvrez le dossier "Objets technologiques".
2. Dans le navigateur du projet, ouvrez l'objet technologique.
3. Effectuez un double clic sur l'objet "Configuration".

Symboles

Dans la navigation de zone de la fenêtre de configuration et d'inspection, des symboles fournissent des détails supplémentaires sur l'intégralité de la configuration :

	**La configuration contient des valeurs par défaut et elle est complète**. La configuration contient exclusivement des valeurs par défaut. Avec ces valeurs par défaut, il est possible d'utiliser l'objet technologique sans autre modification.
	**La configuration contient des valeurs définies par l'utilisateur ou des valeurs modifiées automatiquement et elle est complète**. Tous les champs de saisie de la configuration contiennent des valeurs valides et une valeur par défaut au moins a été modifiée.
	**La configuration est incomplète ou erronée**. Un champ de saisie ou une liste déroulante au moins ne contient pas de valeur ou contient une valeur incorrecte. Le champ ou la liste en question sont marqués en rouge. Un clic affiche la cause de l'erreur dans la liste déroulante des messages d'erreur.

Les propriétés d'un objet technologique sont décrites en détail au chapitre relatif à l'objet technologique.

4.5 Appeler l'instruction dans le programme utilisateur

L'instruction du régulateur logiciel doit être appelée dans un OB d'alarme cyclique. La période d'échantillonnage du régulateur logiciel est déterminé par l'intervalle de temps entre les appels de l'OB d'alarme cyclique.

Condition

L'OB d'alarme cyclique est créé et le temps de cycle de l'OB d'alarme cyclique est correctement configuré.

Marche à suivre

Pour appeler l'instruction dans le programme utilisateur, procédez de la manière suivante :

1. Dans le navigateur du projet, ouvrez le dossier de la CPU.
2. Ouvrez le dossier "Blocs de programme".
3. Effectuez un double clic sur l'OB d'alarme cyclique.
   Le bloc s'ouvre dans la zone de travail.
4. Ouvrez le groupe "Technologie" et le dossier "PID Control" dans la fenêtre "Instructions".
   Le dossier contient toutes les instructions des régulateurs logiciels pouvant être configurés sur la CPU.
5. Sélectionnez une instruction et faites-la glisser dans votre OB d'alarme cyclique.
   La boîte de dialogue "Options d'appel" s'ouvre.
6. Dans la liste "Nom", sélectionnez un objet technologique ou entrez le nom d'un nouvel objet technologique.

Résultat

Si l'objet technologique n'existe pas encore, il est ajouté. L'instruction est ajoutée dans l'OB d'alarme cyclique. L'objet technologique est affecté à cet appel de l'instruction.

4.6 Charger des objets technologiques dans l'appareil

Une configuration de l'objet technologique, qu'elle soit nouvelle ou modifiée, doit être chargée dans la CPU pour le mode de fonctionnement en ligne. Lors du chargement de données rémanentes, les particularités suivantes s'appliquent :

- **Logiciel (seulement les modifications)**
  - S7-1200, S7-1500 :
    Les données rémanentes sont conservées.
  - S7-300/400 :
    Les données rémanentes sont immédiatement actualisées. La CPU ne se met pas à l'arrêt.
- **Charger le programme API dans l'appareil et le réinitialiser**
  - S7-1200, S7-1500 :
    Les données rémanentes sont actualisées lors du prochain passage de STOP à RUN. Le programme API ne peut être chargé qu'en intégralité.
  - S7-300/400 :
    Les données rémanentes sont actualisées lors du prochain passage de STOP à RUN.

Chargement des données rémanentes sur une CPU S7-1200 ou S7-1500

REMARQUE

Le chargement et la réinitialisation du programme API alors que l'installation est en fonctionnement peut entraîner des dommages matériels et corporels graves en cas de dysfonctionnement ou d'erreur de programme !

Assurez-vous qu'aucun état dangereux ne puisse se produire avant de charger et réinitialiser le programme API !

Pour charger des données rémanentes, procédez comme suit :

1. Sélectionnez l'entrée de la CPU dans la navigation du projet.
2. Dans le menu "En ligne", sélectionnez la commande "Charger le programme dans l'appareil et le réinitialiser".
   - Si vous n'aviez établi aucune liaison en ligne jusqu'à cet instant, la boîte de dialogue "Chargement élargi" s'ouvre. Définissez dans ce cas tous les paramètres nécessaires pour la liaison et cliquez sur "Charger".
   - Une fois que la liaison en ligne est définie, les données du projet sont compilées si nécessaire, et la boîte de dialogue "Aperçu chargement" s'ouvre. Cette boîte de dialogue affiche des messages et vous propose des actions requises pour le chargement.
3. Vérifiez les messages.
   Dès que le chargement est possible, le bouton "Charger" devient actif.
4. Cliquez sur "Charger".
   Le programme API est complètement chargé et la boîte de dialogue "Résultats chargement" s'ouvre. Cette boîte de dialogue affiche l'état et les actions après l'opération de chargement.
5. Si les modules doivent être redémarrés juste après le chargement, cochez la case "Démarrer tout".
6. Fermez la boîte de dialogue "Résultats chargement" en cliquant sur "Terminer".

Résultat

Le programme API est complètement chargé dans l'appareil. Les blocs qui existent seulement en ligne dans l'appareil, sont supprimés. Le chargement de tous les blocs concernés et la suppression des blocs inutiles dans l'appareil permettent d'éviter des incohérences entres les blocs dans le programme utilisateur.
Vous pouvez savoir si l'opération de chargement a réussi grâce aux messages figurant dans la fenêtre d'inspection "Info > Général".

4.7 Mise en service du régulateur logiciel

Marche à suivre

Pour ouvrir la zone de travail "Mise en service" d'un objet technologique, procédez de la manière suivante :

1. Dans le navigateur du projet, ouvrez le dossier "Objets technologiques".
2. Dans le navigateur du projet, ouvrez l'objet technologique.
3. Double-cliquez sur l'objet "Mise en service".

Les fonctions de mise en service sont décrites pour chaque régulateur, de manière spécifique.

4.8 Enregistrer les paramètres PID optimisés dans le projet

Le régulateur logiciel est optimisé dans la CPU. Les valeurs du DB d'instance de la CPU ne correspondent plus à celles du projet.
Pour actualiser les paramètres PID du projet avec les paramètres PID optimisés de la CPU, procédez de la manière suivante :

Condition

- Une connexion en ligne est établie avec la CPU et celle-ci se trouve à l'état de fonctionnement "RUN".
- Les fonctions de la fenêtre de mise en service sont validées par le bouton "Démarrage".

Marche à suivre

1. Dans le navigateur du projet, ouvrez le dossier de la CPU.
2. Ouvrez le dossier "Objets technologiques".
3. Ouvrez un objet technologique.
4. Effectuez un double clic sur "Mise en service".
5. Cliquez sur l'icône "Charger les paramètres PID".
6. Enregistrez le projet.

Résultat

Les paramètres PID actifs sont enregistrés dans les données de projet. Lors du chargement suivant des données de projet dans la CPU, les paramètres optimisés sont utilisés.

4.9 Travailler avec des objets de multi-instance

Si un bloc fonctionnel (FB) appelle un autre FB, ses données d`instance peuvent également être enregistrées dans le DB d'instance du FB appelant. Ce type d'appel de bloc est appelé multi-instance. Les régulateurs logiciels PID prennent en charge ce type d'appel et peuvent être utilisés comme multi-instance.

Avantages

L'utilisation de multi-instances présente les avantages suivants :

- Bonne possibilité de configuration
- Nombre restreint de DB d'instance
- FB configurés individuellement comme modèle de régulateur logiciel que vous pouvez instancier aussi souvent que souhaité

Restrictions

Lors de l'utilisation de multi-instances pour les régulateurs logiciels PID, les restrictions suivantes s'appliquent par rapport à l'utilisation en tant qu'instance unique :

- Pas de prise en charge de Openness pour les objets de multi-instance PID
- Pas de comparaison d'objets de multi-instance PID dans un éditeur de comparaison. La comparaison est possible uniquement via le bloc qui contient les objets de multi-instance.
- Pas de fenêtre d'inspection spécifique aux objets technologiques de l'éditeur de programmation pour l'appel des instructions PID_Compact, PID_3Step et PID_Temp

Configuration d'objets de multi-instance

La configuration et la mise en service d'objets de multi-instance PID ne s'ouvrent pas via le dossier "Objets technologiques" dans la navigation du projet, comme cela est le cas pour les objets d'instance unique. Pour les objets de multi-instance, vous trouverez la configuration et la mise en service dans la vue détaillée sous l'onglet "Objets technologiques".

Pour ouvrir la configuration d'objets de multi-instance, procédez de la manière suivante :

1. Dans le navigateur du projet, ouvrez le FB ou le DB d'instance avec l'objet de multi-instance.
2. Dans la vue détaillée, cliquez sur l'onglet " Objets technologiques".
3. Accédez à l'objet de multi-instance souhaité.
4. Ouvrez la configuration de l'objet de multi-instance.

REMARQUE

L'éditeur de configuration n'offre pas de fonction en ligne pour les objets de multi-instance dans un FB. Cette restriction n'existe pas pour les DB d'instance.

Mise en service d'objets de multi-instance

Pour ouvrir la mise en service d'objets de multi-instance, procédez de la manière suivante :

1. Dans le navigateur du projet, ouvrez le DB d'instance avec l'objet de multi-instance.
2. Dans la vue détaillée, cliquez sur l'onglet " Objets technologiques".
3. Accédez à l'objet de multi-instance souhaité.
4. Ouvrez la mise en service de l'objet de multi-instance.

Cette fonction n'existe pas pour les objets de multi-instance PID dans les FB.

REMARQUE

S'il existe des objets de multi-instance PID dans un tableau, la navigation vers ces objets de multi-instance dans la vue détaillée n'est possible que si le nombre d'éléments du tableau ne dépasse pas 100. Pour les tableaux comprenant plus de 100 éléments, les objets de multi-instance PID individuels ne sont pas affichés et les éditeurs de configuration et de mise en service ne sont pas disponibles.

Exemple d'utilisation d'objets de multi-instance PID

Pour utiliser des objets de multi-instance PID pour votre application, vous pouvez procéder comme suit :

1. Ajoutez un bloc de fonction à votre programme.
2. Appelez un ou plusieurs régulateurs PID appropriés avec l'option d'appel "Multi-instance" dans ce FB.
3. Ajoutez votre propre fonctionnalité dépendante de l'application dans le même FB, par exemple le prétraitement de la valeur de consigne.
4. Marquez le FB dans la navigation du projet, puis ouvrez les éditeurs de configuration des objets de multi-instance PID via l'onglet "Objets technologiques" de la vue détaillée.
5. Procédez à la configuration dans les éditeurs de configuration, celle-ci doit être identique pour toutes les instances du FB.
6. Fermez les éditeurs de configuration.
7. Instanciez le FB selon les besoins dans le programme utilisateur afin que les blocs de données d'instance soient créés.
8. Marquez l'un de ces blocs de données d'instance dans la navigation du projet, puis ouvrez les éditeurs de configuration des objets de multi-instance PID via l'onglet "Objets technologiques" de la vue détaillée.
9. Procédez à la configuration dans les éditeurs de configuration, celle-ci est individuelle pour ce bloc de données d'instance.
10. Fermez les éditeurs de configuration.
11. Répétez les étapes 8 à 10 également pour les autres blocs de données d'instance avec des objets de multi-instance PID.
12. Compilez le programme, chargez-le dans l'appareil et établissez une connexion en ligne.
13. Marquez un bloc de données d'instance qui contient des objets de multi-instance PID dans la navigation du projet, puis ouvrez les éditeurs de mise en service des objets de multi-instance PID via l'onglet "Objets technologiques" de la vue détaillée.
14. Effectuez la mise en service des objets de multi-instance PID.

15. Fermez les éditeurs de mise en service.

16. Répétez les étapes 13 à 15 également pour les autres blocs de données d'instance avec des objets de multi-instance PID.

4.10 Comparer valeurs

4.10.1 Visualisation de comparaison et autres conditions

La fonction "Comparer valeurs" offre les possibilités suivantes :

- Comparaison des valeurs initiales configurées du projet avec les valeurs initiales de la CPU et les valeurs en cours
- Traitement direct des valeurs en cours et des valeurs initiales du projet
- Détection immédiate et affichage des erreurs de saisie avec aide à la correction
- Sauvegarde des valeurs en cours dans le projet
- Transmission des valeurs initiales du projet à la CPU comme valeurs en cours

Icônes et éléments de commande

Les icônes et les éléments de commande suivants sont disponibles :

Icône	Fonction
	La Valeur initiale dans la CPU est égale à la Valeur initiale du projet configurée
	La Valeur initiale dans la CPU est différente de la Valeur initiale du projet configurée
	La comparaison Valeur initiale dans la CPU et Valeur initiale du projet configurée ne peut pas être effectuée.
	Au moins l'une des deux valeurs de comparaison est technologiquement ou syntaxiquement incorrecte
	Créer un instantané des valeurs de visualisation et adopter les valeurs de réglage de cet instantané comme valeurs initiales
	Charger les valeurs initiales des valeurs de réglage comme valeurs en cours (initialisation des valeurs de réglage)
	La boîte de dialogue "Comparaison valeurs" s'ouvre

Autres conditions

La fonction "Comparer valeurs" est disponible sans restriction pour S7-1200 et S7-1500.
La restriction suivante s'applique pour S7-300 et S7-400 :
En mode de visualisation, un S7-300/S7-400 ne peut pas transmettre les valeurs initiales dans la CPU. Ces valeurs ne peuvent pas être affichées en ligne avec "Comparer valeurs".
Les valeurs en cours de l'objet technologique s'affichent et peut être modifiées directement.

4.10.2 Comparer valeurs

La marche à suivre suivante est illustrée à l'aide des "Paramètres PID".

Conditions préalables

- Un projet comprenant un régulateur de logiciel est configuré.
- Le projet est chargé dans la CPU.
- La boîte de dialogue de configuration est ouverte dans le navigateur du projet.

Marche à suivre

1. Dans le navigateur du projet, ouvrez le régulateur de logiciel de votre choix.
2. Double-cliquez sur l'objet "Configuration".
3. Naviguez dans la fenêtre de configuration jusqu'à la boîte de dialogue "Paramètres PID".
4. Cliquez sur l'icône pour activer le mode de visualisation.
   Les icônes et éléments de commande (Page 52) de la fonction "Comparer valeurs" s'affichent derrière les paramètres.
5. Cliquez dans le champ de saisie du paramètre souhaité et modifiez manuellement les valeurs de paramètre par saisie directe.
   - Si l'arrière-fond du champ de saisie est grisé, les valeurs peuvent uniquement être lues, et pas modifiées.
   - Pour modifier les valeurs dans la boîte de dialogue "Paramètres PID", activez d'abord la saisie manuelle en cliquant sur la case à coher "Activer la saisie manuelle".
6. Cliquez sur l'icône pour ouvrir la boîte de dialogue des valeurs initiales.
   Cette boîte de dialogue indique deux valeurs du paramètres :
   - Valeur initiale dans la CPU : la valeur initiale dans la CPU s'affiche dans la partie supérieure.
   - Valeur initiale dans le projet : la valeur initiale configurée dans le projet s'affiche dans la partie inférieure.
7. Inscrivez la valeur de votre choix dans le champ de saisie pour le projet.

Détection de défaut

La saisie de valeurs incorrectes est détectée. Dans ce cas, des éléments d'aide sont proposés pour la correction.
Si vous saisissez une valeur incorrecte sur le plan syntaxique, un bandeau s'affiche sous le paramètre avec le message d'erreur correspondant. La valeur incorrecte n'est pas appliquée.
Si vous saisissez une valeur incorrecte sur le plan technologique, une boîte de dialogue signalant l'erreur s'ouvre et affiche une information relative à la correction :

- En cliquant sur "Non", vous pouvez accepter la correction et corriger votre saisie.
- En cliquant sur "OK", vous appliquez la valeur incorrecte.

IMPORTANT
Dysfonctionnement du régulateur Des valeurs incorrectes sur le plan technologique peuvent être entraîner des dysfonctionnements du régulateur.

Sauvegarder valeurs en cours

En cliquant sur l'icône , vous transférez les valeurs en cours du régulateur aux valeurs initiales de votre projet configuré.

Transférer les valeurs de projet à la CPU

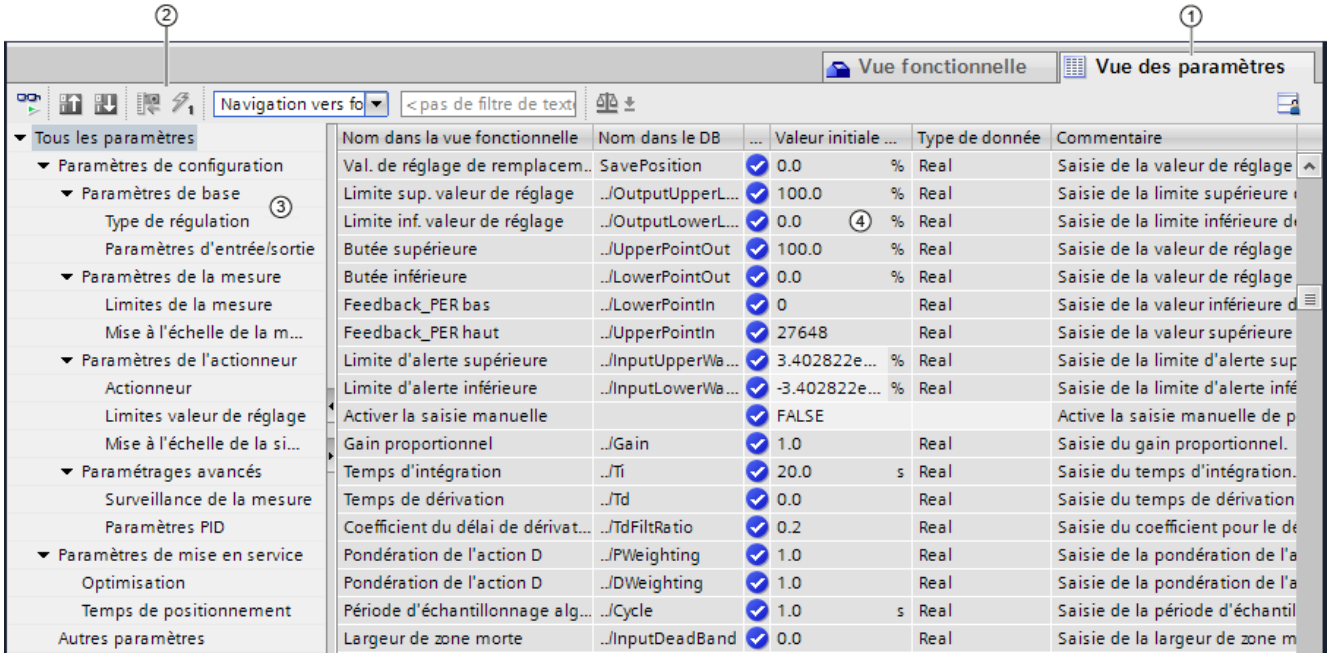
En cliquant sur l'icône , vous transférez les valeurs configurées de votre projet à la CPU.

⚠ PRUDENCE
Attention aux blessures et dommages matériels ! Le chargement et la réinitialisation du programme utilisateur alors que l'installation est en fonctionnement peut entraîner des dommages matériels et corporels graves en cas de dysfonctionnement ou d'erreur de programme ! Assurez-vous qu'aucun état dangereux ne puisse se produire avant de charger et réinitialiser le programme utilisateur !

4.11 Vue des paramètres

4.11.1 Introduction à la vue des paramètres

La vue des paramètres vous offre un aperçu général de tous les paramètres pertinents d'un objet technologique. Vous obtenez une vue d'ensemble sur les réglages des paramètres et vous pouvez les modifier facilement en mode hors ligne et en mode en ligne.



① Onglet "Vue des paramètres"
② Barre d'outils (Page 57)
③ Navigation (Page 58)
④ Table des paramètres (Page 58)

Fonctions

Les fonctions suivantes sont disponibles pour analyser, visualiser de manière ciblée et forcer les paramètres des objets technologiques.

Fonctions d'affichage :

- Affichage des valeurs des paramètres en mode hors ligne et en mode en ligne
- Affichage des informations d'état des paramètres
- Affichage des écarts de valeur et possibilité de correction directe
- Affichage des erreurs de configuration
- Affichage des modifications de valeur à la suite de dépendances de paramètres
- Affichage de toutes les valeurs d'enregistrement d'un paramètre : valeur initiale dans la CPU, valeur initiale dans le projet, valeur de visualisation
- Affichage de la comparaison de paramètres des valeurs d'enregistrement d'un paramètre

Fonctions de commande :

- Navigation pour basculer rapidement entre les paramètres et les structures de paramètres.
- Filtre de texte pour trouver plus rapidement certains paramètres.
- Fonction de tri pour adapter l'ordre des paramètres et des groupes de paramètres aux besoins.
- Fonction d'enregistrement afin de sauvegarder les paramètres structurels de la vue des paramètres.
- Visualisation en ligne et forçage des valeurs des paramètres
- Changement du format d'affichage de la valeur.
- Enregistrement d'un instantané des valeurs des paramètres de la CPU pour représenter des situations brèves et y réagir.
- Reprise de l'instantané de valeurs de paramètres comme valeurs initiales.
- Chargement des valeurs initiales modifiées dans la CPU.
- Fonctions de comparaison pour comparer des valeurs de paramètres les unes aux autres.

Validité

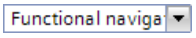
La vue des paramètres décrite ici est disponible pour les objets technologiques suivants :

- PID_Compact
- PID_3Step
- PID_Temp
- CONT_C (seulement S7-1500)
- CONT_S (seulement S7-1500)
- TCONT_CP (seulement S7-1500)
- TCONT_S (seulement S7-1500)
- TO_Axis_PTO (S7-1200 Motion Control)
- TO_Positioning_Axis (S7-1200 Motion Control)
- TO_CommandTable_PTO (S7-1200 Motion Control)
- TO_CommandTable (S7-1200 Motion Control)

4.11.2 Structure de la vue des paramètres

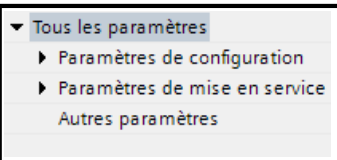
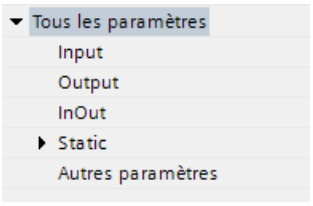
4.11.2.1 Barre d'outils

Les fonctions suivantes sont accessibles dans la barre d'outils de la vue des paramètres :

Icône	Fonction	Description
	Visualiser tout	Démarre la visualisation des paramètres visibles dans la vue des paramètres active (mode en ligne).
	Créer un instantané des valeurs de visualisation et adopter les valeurs de réglage de cet instantané comme valeurs initiales	Reprend les valeurs de visualisation actuelles dans la colonne "Instantané" et met à jour les valeurs initiales dans le projet. Seulement en mode en ligne pour PID_Compact, PID_3Step et PID_Temp.
	Charger les valeurs initiales comme valeurs en cours (initialisation des valeurs de réglage)	Transfère dans la CPU les valeurs initiales actualisées dans le projet. Seulement en mode en ligne pour PID_Compact, PID_3Step et PID_Temp.
	Créer un instantané des valeurs de visualisation	Reprend les valeurs de visualisation actuelles dans la colonne "Instantané". Seulement en mode en ligne.
	Forçage unique et immédiat de tous les paramètres sélectionnés	Cette commande est exécutée une seule fois et le plus rapidement possible sans référence à un emplacement particulier du programme utilisateur. Seulement en mode en ligne.
	Sélectionner la structure de navigation	Bascule entre la navigation vers fonctions et la navigation vers données.
	Filtre de texte...	Après la saisie d'une chaîne de caractères : affichage de tous les paramètres qui comprennent la chaîne de caractères saisie dans l'une des colonnes visibles actuellement.
	Sélectionner les valeurs de comparaison	Sélection des valeurs de paramètres qui doivent être comparées les unes aux autres en mode en ligne (Valeur initiale dans le projet, Valeur initiale dans la CPU, Instantané) Seulement en mode en ligne.
	Mémoriser la disposition	Enregistre les paramétrages que vous avez effectués en matière d'affichage dans la vue des paramètres (par ex. structure de navigation sélectionnée, colonnes de tableau activées, etc.)

4.11.2.2 Navigation

Les structures de navigation suivantes peuvent être sélectionnées de manière alternative dans l'onglet "Vue des paramètres" :

Navigation		Description
Navigation vers fonctions		Dans la navigation vers fonctions, la structure des paramètres est basée sur la structure dans la boîte de dialogue de configuration (onglet "Vue fonctionnelle"), la boîte de dialogue de mise en service et la boîte de dialogue de diagnostic. Le dernier groupe "Autres paramètres" contient tous les autres paramètres de l'objet technologique.
Navigation vers données		Dans la navigation vers données, la structure des paramètres est basée sur la structure dans le DB d'instance/DB technologique. Le dernier groupe "Autres paramètres" contient les paramètres qui ne sont pas compris dans le DB d'instance/DB technologique.

La liste déroulante "Sélectionner la structure de navigation" vous permet de changer de structure de navigation.

4.11.2.3 Table des paramètres

Le tableau suivant récapitule la signification des diverses colonnes de la table des paramètres. Vous pouvez afficher ou masquer les colonnes si besoin.

- Colonne "Hors ligne" = X : la colonne est visible en mode hors ligne.
- Colonne "En ligne" = X : la colonne est visible en mode en ligne (liaison en ligne à la CPU).

Colonne	Description	Hors ligne	En ligne
Nom dans la vue fonctionnelle	Nom du paramètre dans la vue fonctionnelle. Le champ d'affichage est vide pour les paramètres qui ne sont pas configurés par le biais de l'objet technologique.	X	X
Nom complet dans le DB	Chemin entier du paramètre dans le DB d'instance/DB technologique. Le champ d'affichage est vide pour les paramètres qui ne figurent pas dans le DB d'instance/DB technologique.	X	X
Nom dans le DB	Nom du paramètre dans le DB d'instance/DB technologique. Si le paramètre fait partie d'une structure ou d'un type de données utilisateur, le préfixe ". ./" est ajouté. Le champ d'affichage est vide pour les paramètres qui ne figurent pas dans le DB d'instance/DB technologique.	X	X
Etat de la configuration	Affichage de l'intégralité de la configuration avec des icônes d'état. voir Etat de la configuration (hors ligne) (Page 67)	X	
Résultat de la comparaison	Résultat de la fonction "Comparer des valeurs" Cette colonne est affichée s'il existe une liaison en ligne et que le bouton "Visualiser tout" est activé.		X
Valeur initiale dans le projet	Valeur initiale configurée dans le projet. Signalisation d'erreur en cas de saisie de valeurs syntaxiquement ou technologiquement incorrectes.	X	X

Colonne	Description	Hors ligne	En ligne
Valeur par défaut	Valeur par défaut du paramètre Le champ d'affichage est vide pour les paramètres qui ne figurent pas dans le DB d'instance/DB technologique.	X	X
Instantané	Instantané" des valeurs actuelles de la CPU (valeurs de visualisation). Signalisation d'erreur en cas de valeurs technologiquement incorrectes.	X	X
Valeur initiale dans la CPU	Valeur initiale dans la CPU. Cette colonne est affichée s'il existe une liaison en ligne et que le bouton "Visualiser tout" est activé. Signalisation d'erreur en cas de valeurs technologiquement incorrectes.		X
Valeur de visualisation	Valeur actuelle de la CPU. Cette colonne est affichée s'il existe une liaison en ligne et que le bouton "Visualiser tout" est activé. Signalisation d'erreur en cas de valeurs technologiquement incorrectes.		X
Valeur de forçage	Valeur avec laquelle la valeur de visualisation doit être modifiée. Cette colonne est affichée s'il existe une liaison en ligne et que le bouton "Visualiser tout" est activé. Signalisation d'erreur en cas de saisie de valeurs syntaxiquement ou technologiquement incorrectes.		X
Sélection de la valeur de forçage	Sélection des valeurs de forçage qui doivent être transférées à l'aide du bouton "Forçage unique et immédiat de tous les paramètres sélectionnés". Cette colonne est affichée avec la colonne "Valeur de forçage".		X
Valeur minimale	Valeur technologiquement la plus basse du paramètre. Si la valeur minimale dépend d'autres paramètres, elle est déterminée : • Hors ligne : par les valeurs initiales dans le projet. • En ligne : par les valeurs de visualisation.	X	X
Valeur maximale	Valeur technologiquement la plus haute du paramètre. Si la valeur maximale dépend d'autres paramètres, elle est déterminée : • Hors ligne : par les valeurs initiales dans le projet. • En ligne : par les valeurs de visualisation.	X	X
Valeur de réglage	Caractérise le paramètre comme valeur de réglage. Ces paramètres peuvent être initialisés en ligne.	X	X
Type de données	Type de données du paramètre. Le champ d'affichage est vide pour les paramètres qui ne figurent pas dans le DB d'instance/DB technologique.	X	X
Rémanence	Caractérise la valeur comme rémanente. Les valeurs de paramètres rémanents sont conservées même après une coupure de l'alimentation.	X	X
Accessible à partir de l'IHM	Indique si l'IHM peut accéder à ce paramètre pendant l'exécution.	X	X
Visible dans l'IHM	Indique si le paramètre est visible par défaut dans la liste de sélection de l'IHM.	X	X
Commentaire	Brève description du paramètre.	X	X

Voir aussi

Comparer valeurs (Page 52)

4.11.3 Ouvrir la vue des paramètres

Condition préalable

L'objet technologique est ajouté dans le navigateur du projet, c'est-à-dire que le DB d'instance/DB technologique correspondant de l'instruction est créé.

Marche à suivre

1. Ouvrez le dossier "Objets technologiques" dans la navigation du projet.
2. Ouvrez l'objet technologique dans la navigation du projet.
3. Effectuez un double-clic sur l'objet "Configuration".
4. Sélectionnez l'onglet "Vue des paramètres" dans le coin supérieur droit.

Résultat

La vue des paramètres s'ouvre. Chaque paramètre affiché est représenté par une ligne dans la table des paramètres.
Les propriétés de paramètres affichables (colonnes du tableau) dépendent du fonctionnement en mode hors ligne ou en mode en ligne de la vue des paramètres.
De plus, vous pouvez afficher ou masquer différentes colonnes du tableau de manière ciblée.

Voir aussi

Réglage par défaut de la vue des paramètres (Page 60)

4.11.4 Réglage par défaut de la vue des paramètres

Paramètres par défaut

Pour travailler efficacement avec la vue des paramètres, vous pouvez adapter l'affichage des paramètres et enregistrer les réglages effectués.
Les adaptations suivantes sont possibles et enregistrables :

- Afficher et masquer des colonnes
- Modifier la largeur d'une colonne
- Modifier l'ordre des colonnes
- Changer de navigation
- Sélectionner le groupe de paramètres dans la navigation
- Sélectionner les valeurs de comparaison

Afficher et masquer des colonnes

Procédez de la manière suivante pour afficher ou masquer des colonnes de la table des paramètres :

1. Positionnez le pointeur de la souris dans la ligne d'en-tête de la table des paramètres.
2. Sélectionnez la commande "Afficher/masquer" dans le menu contextuel.
   La sélection des colonnes disponibles s'affiche.
3. Pour afficher une colonne, cochez la case de cette colonne.
4. Pour masquer une colonne, décochez la case de cette colonne.

ou

1. Positionnez le pointeur de la souris dans la ligne d'en-tête de la table des paramètres.
2. Sélectionnez la commande "Afficher toutes les colonnes" dans le menu contextuel si toutes les colonnes du mode hors ligne ou du mode en ligne doivent être affichées.

Certaines colonnes ne peuvent être affichées qu'en mode en ligne : voir Table des paramètres (Page 58).

Modifier la largeur d'une colonne

Pour adapter la largeur d'une colonne au contenu de telle sorte que tous les textes soient lisibles dans les lignes, procédez comme suit :

1. Positionnez le pointeur de la souris dans la ligne d'en-tête de la table des paramètres à droite de la colonne à optimiser jusqu'à ce que le pointeur de la souris prenne la forme d'une croix.
2. Faites un double-clic à cet endroit.

ou

1. Ouvrez le menu contextuel sur la ligne d'en-tête de la table des paramètres.
2. Cliquez sur
   - "Optimiser la largeur de la colonne" ou
   - "Optimiser la largeur de toutes les colonnes".

Pour les colonnes trop étroites, le contenu complet des différentes cellules est affiché si vous laissez le pointeur de la souris sur la cellule concernée pendant un temps bref.

Modifier l'ordre des colonnes

Les colonnes de la table des paramètres peuvent être placées librement.

Pour modifier l'ordre des colonnes, procédez comme suit :

1. Cliquez sur l'en-tête d'une colonne et faites-la glisser à l'endroit souhaité.
   Lorsque vous relâchez le bouton de la souris, la colonne est ancrée à la nouvelle position.

Changer de navigation

Pour changer de structure d'affichage des paramètres, procédez comme suit :

1. Sélectionnez la navigation souhaitée dans la liste déroulante "Sélectionner la structure de navigation" :
   - Navigation vers données
   - Navigation vers fonctions

Voir aussi Navigation (Page 58).

Sélectionner le groupe de paramètres dans la navigation

Dans la navigation sélectionnée, vous pouvez choisir entre l'affichage "Tous les paramètres" ou l'affichage d'un groupe de paramètres sous-jacents de votre choix.

1. Cliquez sur le groupe de paramètres souhaité dans la navigation.
   Seuls les paramètres du groupe de paramètres sont affichés dans la table des paramètres.

Sélectionner les valeurs de comparaison (en ligne)

Pour paramétrer les valeurs de comparaison pour la fonction "Comparer des valeurs", procédez comme suit :

1. Sélectionnez les valeurs de comparaison souhaitées dans la liste déroulante "Sélectionner les valeurs de comparaison" :
   - Valeur initiale dans le projet / valeur initiale dans la CPU
   - Valeur initiale dans le projet / instantané
   - Valeur initiale dans la CPU / instantané

L'option "Valeur initiale dans le projet / valeur initiale dans la CPU" est réglée par défaut.

Enregistrer le préréglage de la Vue des paramètres

Pour enregistrer les adaptations ci-dessus de la vue des paramètres, procédez comme suit :

1. Adaptez la vue des paramètres selon vos besoins.
2. Cliquez sur le bouton  "Mémoriser la disposition" en haut à droite dans la vue des paramètres.

4.11.5 Utiliser la vue des paramètres

4.11.5.1 Vue d'ensemble

Le tableau suivant donne une vue d'ensemble des fonctions décrites ci-après de la vue des paramètres en mode en ligne et en mode hors ligne.

- Colonne "Hors ligne" = X : cette fonction est possible en mode hors ligne.
- Colonne "En ligne" = X : cette fonction est possible en mode en ligne.

Fonction/action	Hors ligne	En ligne
Filtrer la table des paramètres (Page 63)	X	X
Trier la table des paramètres (Page 64)	X	X
Reprendre les données des paramètres dans d'autres éditeurs (Page 64)	X	X
Signaler les erreurs (Page 65)	X	X
Editer les valeurs initiales dans le projet (Page 65)	X	X
Etat de la configuration (hors ligne) (Page 67)	X	
Visualiser en ligne des valeurs dans la vue des paramètres (Page 68)		X
Créer un instantané des valeurs de visualisation (Page 70)		X
Forcer des valeurs (Page 70)		X
Comparer des valeurs (Page 72)		X
Reprendre des valeurs du programme en ligne comme valeurs initiales (Page 73)		X
Initialiser des valeurs de réglage dans le programme en ligne (Page 74)		X

4.11.5.2 Filtrer la table des paramètres

Vous pouvez filtrer les paramètres de la table des paramètres de la façon suivante :

- Avec le filtre de texte
- Avec les sous-groupes de la navigation

Les deux méthodes de filtrage peuvent être utilisées simultanément.

Avec le filtre de texte

Le filtrage peut être effectué selon les textes visibles dans la table des paramètres. Cela signifie que le filtrage ne peut être effectué que selon des textes figurant dans des lignes de paramètre et des colonnes affichées.

1. Saisissez la chaîne de caractères souhaitée, selon laquelle le filtrage doit être effectué, dans le champ de saisie "Filtre de texte...".
   La table des paramètres n'affiche alors plus que les paramètres dans lesquels figure la chaîne de caractères.

Le filtrage de texte est réinitialisé :

- En cas de sélection d'un autre groupe de paramètres dans la navigation.
- En cas de commutation entre la navigation vers données et la navigation vers fonctions.

Avec les sous-groupes de la navigation

1. Cliquez sur le groupe de paramètres souhaité dans la navigation, par exemple "Static". Dans la table des paramètres, seuls les paramètres Static sont alors encore affichés. Pour certains groupes de la navigation, vous pouvez sélectionner d'autres sous-groupes.
2. Cliquez sur "Tous les paramètres" dans la navigation si tous les paramètres doivent être à nouveau affichés.

4.11.5.3 Trier la table des paramètres

Les valeurs des paramètres sont disposées en lignes. La table des paramètres peut être triée selon chaque colonne affichée.

- Les colonnes contenant des valeurs numériques sont triées par ordre de leurs valeurs numériques.
- Les colonnes de texte sont triées par ordre alphabétique.

Trier par colonne

1. Positionnez le pointeur de la souris dans la cellule d'en-tête de la colonne souhaitée. L'arrière-plan de cette cellule est surligné en bleu.
2. Cliquez sur l'en-tête de la colonne.

Résultat

Toute la table des paramètres est triée selon la colonne sélectionnée. Un triangle avec la pointe vers le haut apparaît dans l'en-tête de la colonne.
En cliquant plusieurs fois sur l'en-tête de la colonne, le tri est modifié comme suit :

- Symbole "▲" : la table des paramètres est triée par ordre croissant.
- Symbole "▼" : la table des paramètres est triée par ordre décroissant.
- Aucun symbole : le tri est annulé. La table des paramètres prend l'affichage par défaut.

Lors du tri, le préfixe "../" dans la colonne "Nom dans le DB" est ignoré.

4.11.5.4 Reprendre les données des paramètres dans d'autres éditeurs

Après avoir sélectionné une ligne de paramètre entière, vous pouvez reprendre des paramètres

- en les faisant glisser
- avec <Ctrl+C>/<Ctrl+V>
- par copier/coller dans le menu contextuel

dans les éditeurs suivants de TIA Portal :

- Editeur du programme
- Table de visualisation
- Table des signaux pour Trace

Le paramètre est inséré avec le nom complet : voir indication de la colonne "Nom complet dans le DB".

4.11.5.5 Signaler les erreurs

Signalisation d'erreur

Les erreurs de paramétrage qui entraînent des erreurs de compilation (par ex. dépassement de valeur limite) sont affichées dans la vue des paramètres.
À chaque saisie d'une valeur dans la vue des paramètres, l'exactitude technologique et syntaxique est immédiatement vérifiée et affichée.
Les valeurs erronées sont signalées par :

- un symbole d'erreur rouge dans les colonnes "Etat de la configuration" (mode hors ligne) ou "Résultat de la comparaison" (mode en ligne, selon le type de comparaison choisi)

et/ou

- une cellule ayant un arrière-plan rouge
  Lors d'un clic sur la cellule erronée : liste déroulante des messages d'erreur avec indication de la plage de valeurs admissible ou de la syntaxe nécessaire (format)

Erreur de compilation

La vue des paramètres avec le paramètre provoquant l'erreur peut être ouverte directement depuis le message d'erreur du compilateur (Navigation vers fonctions) pour les paramètres qui ne sont pas affichés dans la boîte de dialogue de configuration.

4.11.5.6 Editer les valeurs initiales dans le projet

La vue des paramètres vous permet d'éditer les valeurs initiales dans le projet en mode hors ligne et en mode en ligne :

- Vous modifiez les valeurs dans la colonne "Valeur initiale dans le projet" de la table des paramètres.
- La progression de la configuration est indiquée par les symboles d'état connus de la boîte de dialogue de configuration de l'objet technologique dans la colonne "Etat de la configuration" de la table des paramètres.

Autres conditions

- Si d'autres paramètres dépendent du paramètre dont la valeur initiale a été modifiée, la valeur initiale des paramètres dépendants est également adaptée.
- Si le paramètre d'un objet technologique ne peut pas être édité, il ne peut pas l'être non plus dans la vue des paramètres. La possibilité d'éditer un paramètre peut également dépendre des valeurs d'autres paramètres.

Définir de nouvelles valeurs initiales

Pour définir des valeurs initiales pour les paramètres dans la vue des paramètres, procédez comme suit :

1. Ouvrez la vue des paramètres de l'objet technologique.
2. Saisissez les valeurs initiales souhaitées dans la colonne "Valeur initiale dans le projet". La valeur doit correspondre au type de données du paramètre et ne peut pas dépasser la plage de valeurs du paramètre.
   Les valeurs limites de la plage de valeurs sont visibles dans les colonnes "Valeur maximale" et "Valeur minimale".

La "progression" de la configuration est indiquée par des symboles de couleur dans la colonne "Etat de la configuration".

Voir aussi Etat de la configuration (hors ligne) (Page 67)

Après l'adaptation des valeurs initiales et le chargement de l'objet technologique dans la CPU, les paramètres prennent la valeur définie au démarrage, pour autant qu'ils ne sont pas déclarés comme rémanents (colonne "Rémanence").

Signalisation d'erreur

Lors de la saisie d'une valeur initiale, l'exactitude technologique et syntaxique est immédiatement vérifiée et affichée :

Les valeurs initiales erronées sont indiquées par

- un symbole d'erreur rouge dans les colonnes "Etat de la configuration" (mode hors ligne) ou "Résultat de la comparaison" (mode en ligne, selon le type de comparaison choisi)

et/ou

- un arrière-plan rouge dans la cellule "Valeur initiale dans le projet"
  Lors d'un clic sur la cellule erronée : liste déroulante des messages d'erreur avec indication de la plage de valeurs admissible ou de la syntaxe nécessaire (format)

Correction de valeurs initiales erronées

1. Corrigez les valeurs initiales erronées à l'aide des informations de la liste déroulante des messages d'erreur.
   Le symbole d'erreur rouge, l'arrière-plan de cellule rouge et la liste déroulante des messages d'erreur ne sont plus affichés.

Le projet ne peut être compilé avec succès qu'avec des valeurs initiales correctes.

4.11.5.7 Etat de la configuration (hors ligne)

L'état de la configuration est indiqué par des icônes :

- Dans la colonne "Etat de la configuration" dans la table des paramètres
- Dans la structure de navigation de la navigation vers fonctions et la navigation vers données

Icône dans la colonne "Etat de la configuration"

Icône	Signification
	La valeur initiale du paramètre correspond à la valeur par défaut et elle est valide. Aucune valeur initiale n'est encore définie par l'utilisateur.
	La valeur initiale du paramètre contient une valeur définie par l'utilisateur ou modifiée automatiquement. La valeur initiale est différente de la valeur par défaut. La valeur initiale est correcte et valide.
	La valeur initiale du paramètre n'est pas valide (erreur syntaxique ou technologique). Le champ de saisie a un arrière-plan rouge. Un clic affiche la cause de l'erreur dans la liste déroulante des messages d'erreur.
	Seulement pour S7-1200 Motion Control : La valeur initiale du paramètre est valide mais comporte des avertissements. Le champ de saisie a un arrière-plan jaune.
	Le paramètre n'est pas pertinent dans la configuration actuelle.

Icône dans la navigation

Les icônes dans la navigation indiquent la "progression" de la configuration de la même façon que dans la boîte de dialogue de configuration de l'objet technologique.

Voir aussi

Configurer les objets technologiques (Page 46)

4.11.5.8 Visualiser en ligne des valeurs dans la vue des paramètres

Vous pouvez visualiser les valeurs que prennent actuellement les paramètres de l'objet technologique dans la CPU (valeurs de visualisation) directement dans la vue des paramètres.

Conditions préalables

- Il existe une liaison en ligne.
- L'objet technologique est chargé dans la CPU.
- Le traitement de programme est actif (la CPU est en "RUN").
- La vue des paramètres de l'objet technologique est ouverte.

Marche à suivre

1. Lancez la visualisation en cliquant sur l'icône .
   Dès que la vue des paramètres est en ligne, les colonnes supplémentaires suivantes s'affichent :
   - Résultat de la comparaison
   - Valeur initiale dans la CPU
   - Valeur de visualisation
   - Valeur de forçage
   - Sélection de la valeur de forçage

   La colonne "Valeur de visualisation" indique les valeurs de paramètre actuelles sur la CPU.
   Signification des colonnes supplémentaires : voir Table des paramètres (Page 58)
2. Vous fermez la visualisation en cliquant à nouveau sur l'icône .

Affichage

Toutes les colonnes disponibles en ligne exclusivement ont un arrière-plan orange :

- Les valeurs des cellules orange clair peuvent être modifiées.
- Les valeurs des cellules ayant un arrière-plan orange foncé ne peuvent pas être modifiées.

4.11.5.9 Changement du format d'affichage de la valeur

Le format d'affichage de la valeur peut être choisi dans le menu contextuel d'une ligne de tableau dans la vue des paramètres de l'objet technologique.

Le format d'affichage des valeurs suivantes peut être mis aussi bien en mode en ligne qu'en mode hors ligne :

- Valeur initiale dans le projet
- Valeur initiale dans la CPU
- Valeur maximale
- Valeur minimale
- Instantané
- Valeur de visualisation
- Valeur par défaut
- Valeur de forçage

Le format d'affichage paramétré s'applique à toutes les valeurs de la ligne du tableau.

Les formats d'affichage de la valeur suivants sont modifiables :

- Valeur par défaut
- Hexa
- Octal
- Bin
- Déc (+/-)
- DEC

En fonction du paramètre sélectionné dans la vue des paramètres, seuls les formats d'affichage pris en charge peuvent être sélectionnés.

Conditions préalables

- La vue des paramètres de l'objet technologique est ouverte.

Marche à suivre

Pour changer le format d'affichage de la valeur, procédez comme suit :

1. Sélectionnez une ou plusieurs lignes de tableau dans lesquelles vous souhaitez changer le format d'affichage.
2. Dans le menu contextuel, sélectionnez la commande "Format d'affichage".
3. Choisissez le format d'affichage souhaité.

REMARQUE

Pour modifier le format d'affichage d'un certain type de données dans plusieurs lignes du tableau, triez la vue des paramètres selon ce type de données. Ensuite, sélectionnez la première et la dernière ligne du tableau avec ce type de données en maintenant la touche <Maj> enfoncée et modifiez le format d'affichage pour les lignes du tableau sélectionnées.

4.11.5.10 Créer un instantané des valeurs de visualisation

Vous pouvez sauvegarder les valeurs actuelles de l'objet technologique dans la CPU (valeurs de visualisation) et les afficher dans la vue des paramètres.

Conditions préalables

- Il existe une liaison en ligne.
- L'objet technologique est chargé dans la CPU.
- Le traitement de programme est actif (la CPU est en "RUN").
- La vue des paramètres de l'objet technologique est ouverte.
- Le bouton "Visualiser tout" est activé.

Marche à suivre

Pour afficher les valeurs actuelles des paramètres, procédez de la manière suivante :

1. Cliquez sur l'icône "Créer un instantané des valeurs de visualisation" dans la vue des paramètres.

Résultat

Les valeurs de visualisation actuelles sont reprises une seule fois dans la colonne "Instantané" de la table des paramètres.
Vous pouvez analyser les valeurs ainsi "gelées" tandis que les valeurs de visualisation continuent d'être actualisées dans la colonne "Valeurs de visualisation".

4.11.5.11 Forcer des valeurs

La vue des paramètres vous permet de forcer des valeurs de l'objet technologique dans la CPU.
Vous pouvez affecter une fois unique des valeurs au paramètre (valeur de forçage) et les forcer immédiatement. Lors de l'exécution, la tâche de forçage est effectuée plus rapidement possible sans référence à un emplacement particulier du programme utilisateur.

⚠ DANGER
Risque lors du forçage : Une modification des valeurs des paramètres alors que l'installation est en fonctionnement peut entraîner des dommages matériels et blessures graves en cas de dysfonctionnement ou d'erreur de programme ! Assurez-vous qu'aucun état dangereux ne puisse se produire avant d'exécuter la fonction "Forçage".

Conditions préalables

- Il existe une liaison en ligne.
- L'objet technologique est chargé dans la CPU.
- Le traitement de programme est actif (la CPU est en "RUN").
- La vue des paramètres de l'objet technologique est ouverte.
- Le bouton "Visualiser tout" est activé.
- Le paramètre peut être forcé (la cellule correspondante de la colonne "valeur de forçage" a un arrière-plan orange clair).

Marche à suivre

Procédez de la manière suivante pour forcer immédiatement des paramètres :

1. Saisissez les valeurs de forçage souhaitées dans la colonne "Valeurs de forçage" de la table des paramètres.
2. Vérifiez que la case du forçage est cochée dans la colonne "Sélection de la valeur de forçage".
   Les valeurs de forçage et les cases à cocher correspondantes des paramètres dépendants sont automatiquement adaptées en même temps.
3. Cliquez sur l'icône "Forçage unique et immédiat de tous les paramètres sélectionnés".

Les paramètres sélectionnés sont forcés immédiatement et une seule fois avec les valeurs prédéfinies et peuvent être visualisés dans la colonne "Valeurs de visualisation". Les cases du forçage dans la colonne "Sélection de la valeur de forçage" sont automatiquement décochées après l'exécution de la tâche de forçage.

Signalisation d'erreur

Lors de la saisie d'une valeur de forçage, l'exactitude technologique et syntaxique est immédiatement vérifiée et affichée :

Les valeurs de forçage erronées sont indiquées par

- Un arrière-plan rouge dans la cellule "Valeur de forçage"

et

- Lors d'un clic sur la cellule erronée : liste déroulante des messages d'erreur avec indication de la plage de valeurs admissible ou de la syntaxe nécessaire (format)

Valeurs de forçage erronées

- Les valeurs de forçage technologiquement erronées peuvent être transférées.
- Les valeurs de forçage syntaxiquement erronées ne peuvent **pas** être transférées.

4.11.5.12 Comparer des valeurs

Les fonctions de comparaison vous permettent de comparer les valeurs d'enregistrement suivantes d'un paramètre :

- Valeur initiale dans le projet
- Valeur initiale dans la CPU
- Instantané

Conditions préalables

- Il existe une liaison en ligne.
- L'objet technologique est chargé dans la CPU.
- Le traitement de programme est actif (la CPU est en "RUN").
- La vue des paramètres de l'objet technologique est ouverte.
- Le bouton "Visualiser tout" est activé.

Marche à suivre

Pour comparer les valeurs initiales sur les différents systèmes cibles, procédez comme suit :

1. Cliquez sur l'icône "Sélectionner les valeurs de comparaison".
   Une liste de sélection avec les options de comparaison s'ouvre :
   - Valeur initiale dans le projet - Valeur initiale dans la CPU (paramétrage par défaut)
   - Valeur initiale dans le projet - Instantané
   - Valeur initiale dans la CPU - Instantané
2. Sélectionnez l'option de comparaison souhaitée.
   L'option de comparaison choisie est exécutée comme suit :
   - Un symbole de balance apparaît dans les cellules d'en-tête des deux colonnes sélectionnées pour la comparaison.
   - Le résultat de la comparaison des colonnes sélectionnées est indiqué par des icônes dans la colonne "Résultat de la comparaison".

Icône de la colonne "Résultat de la comparaison"

Icône	Signification
	Les valeurs de comparaison sont identiques et correctes.
	Les valeurs de comparaison sont différentes et correctes.
	Au moins l'une des deux valeurs de comparaison est technologiquement ou syntaxiquement incorrecte.
	La comparaison ne peut pas être effectuée. Au moins l'une des deux valeurs de comparaison n'est pas disponible (par ex. instantané).
	La valeur ne peut pas être comparée significativement, car elle n'est pas pertinente dans la configuration.

Icône dans la navigation

Les icônes sont affichées de manière identique dans la navigation lorsque le résultat de comparaison s'applique à au moins l'un des paramètres en dessous de la structure de navigation affichée.

4.11.5.13 Reprendre des valeurs du programme en ligne comme valeurs initiales

Pour reprendre, dans le projet, des valeurs optimisées de la CPU comme valeurs initiales en une seule étape, créez un instantané des valeurs de visualisation. Les valeurs marquées comme "valeur de réglage" de l'instantané sont ensuite reprises comme valeurs initiales dans le projet.

Conditions préalables

- L'objet technologique est du type "PID_Compact", "PID_3Step" ou "PID_Temp".
- Il existe une liaison en ligne.
- L'objet technologique est chargé dans la CPU.
- Le traitement de programme est actif (la CPU est en "RUN").
- La vue des paramètres de l'objet technologique est ouverte.
- Le bouton "Visualiser tout" est activé.

Marche à suivre

Pour reprendre les valeurs optimisées de la CPU, procédez comme suit :

1. Cliquez sur l'icône "Créer un instantané des valeurs de visualisation et adopter les valeurs de réglage de cet instantané comme valeurs initiales".

Résultat

Les valeurs de visualisation actuelles sont reprises dans la colonne "Instantané" et leurs valeurs de réglage sont copiées dans la colonne "Valeur initiale dans le projet" comme nouvelles valeurs initiales.

REMARQUE

Reprendre des valeurs de paramètres individuels

Vous pouvez également reprendre les valeurs de paramètres individuels, qui ne sont pas marquées comme valeur de réglage, dans la colonne "Valeurs initiales dans le projet" à partir de la colonne "Instantané". Copiez à cet effet les valeurs à l'aide des commandes "Copier" et "Coller" du menu contextuel et collez-les dans la colonne Valeur initiale dans le projet".

4.11.5.14 Initialiser des valeurs de réglage dans le programme en ligne

Vous pouvez initialiser, dans la CPU, tous les paramètres marqués comme "valeur de réglage" dans la vue des paramètres, avec de nouvelles valeurs en une seule étape. Ce faisant, les valeurs initiales du projet sont chargées dans la CPU. La CPU reste à l'état de fonctionnement "RUN".
Pour éviter toute perte de données sur la CPU en cas de démarrage à froid ou de redémarrage (démarrage à chaud), vous devez charger, de plus, l'objet technologique dans la CPU.

⚠ DANGER
Risque lors de la modification de valeurs de paramètres Une modification des valeurs des paramètres alors que l'installation est en fonctionnement peut entraîner des dommages matériels et blessures graves en cas de dysfonctionnement ou d'erreur de programme ! Assurez-vous qu'aucun état dangereux ne puisse se produire avant que vous ne réinitialisiez les valeurs de réglage.

Conditions préalables

- L'objet technologique est du type "PID_Compact", "PID_3Step" ou "PID_Temp".
- Il existe une liaison en ligne.
- L'objet technologique est chargé dans la CPU.
- Le traitement de programme est actif (la CPU est en "RUN").
- La vue des paramètres de l'objet technologique est ouverte.
- Le bouton "Visualiser tout" est activé.
- Les paramètres marqués comme " disposent d'une "valeur initiale dans le projet" technologiquement et syntaxiquement correcte.

Marche à suivre

Pour initialiser toutes les valeurs de réglage, procédez comme suit :

1. Saisissez les valeurs souhaitées dans la colonne "Valeur initiale dans le projet".
   Veillez à ce que les valeurs initiales soient technologiquement et syntaxiquement correctes.
2. Cliquez sur l'icone "Charger les valeurs initiales des valeurs de réglage comme valeurs en cours".

Résultat

Les valeurs de réglage dans la CPU sont initialisées avec les valeurs initiales du projet.

4.12 Afficher le DB d'instance d'un objet technologique

Un DB d'instance dans lequel sont enregistrés les paramètres et les variables statiques, est créé pour chaque objet technologique.

Marche à suivre

Pour afficher le DB d'instance d'un objet technologique, procédez de la manière suivante :

1. Dans le navigateur du projet, ouvrez le dossier de la CPU.
2. Ouvrez le dossier "Objets technologiques".
3. Sélectionnez l'objet technologique.
4. Dans le menu contextuel, sélectionner l'ordre "Ouvrir dans l'éditeur".

Utiliser PID_Compact

5

5.1 Objet technologique PID_Compact

L'objet technologique PID_Compact met à disposition un régulateur PID continu avec optimisation intégrée. De manière alternative, vous pouvez configurer un régulateur à impulsion. Les modes de fonctionnement manuel et automatique sont possibles.

Dans une boucle de régulation, PID-Compact réalise l'acquisition continue de la mesure et la compare à la consigne souhaitée. A partir du signal d'écart en résultant, l'instruction PID_Compact calcule une valeur de réglage par laquelle la mesure est ajustée à la consigne de la façon la plus rapide et la plus stable possible. Pour le régulateur PID, la valeur de réglage se compose de trois actions :

- Action **P**
  L'action P de la valeur de réglage augmente proportionnellement au signal d'écart.
- Action **I**
  L'action I de la valeur de réglage augmente jusqu'à ce que le signal d'écart soit compensé.
- Action **D**
  L'action D augmente avec la vitesse de modification du signal d'écart. La mesure est ajustée à la consigne le plus rapidement possible. Quand la vitesse de modification du signal d'écart ralentit, l'action D diminue également.

L'instruction PID_Compact calcule les paramètres P, I et D du système réglé de manière autonome pendant l'optimisation préalable. Une optimisation supplémentaire des paramètres peut être réalisée par une optimisation fine. Vous n'avez pas besoin de déterminer les paramètres manuellement.

Pour plus d'informations

- Présentation des régulateurs de logiciel (Page 43)
- Ajouter des objets technologiques (Page 45)
- Configurer les objets technologiques (Page 46)
- Configurer PID_Compact à partir de V2 (Page 77)
- Configurer PID_Compact V1 (Page 101)

FAQ

Pour plus d'informations à ce sujet, voir la FAQ suivante dans l'assistance en ligne de Siemens Industry :

- ID de contribution 79047707
  (https://support.industry.siemens.com/cs/ww/en/view/79047707)

5.2 PID_Compact à partir de V2

5.2.1 Configurer PID_Compact à partir de V2

5.2.1.1 Réglages de base à partir de V2

Configurez les propriétés suivantes de l'objet technologique "PID_Compact" dans la fenêtre d'inspection ou dans les "Paramètres de base" de la fenêtre de configuration.

- Grandeur physique
- Sens de régulation
- Comportement au démarrage après un Reset
- Consigne (seulement dans la fenêtre d'inspection)
- Mesure (seulement dans la fenêtre d'inspection)
- Valeur de réglage (seulement dans la fenêtre d'inspection)

Consigne, mesure et valeur de réglage

Vous ne pouvez configurer la consigne, la mesure et la valeur de réglage que dans la fenêtre d'inspection de l'éditeur de programmation. Sélectionnez la source pour chaque valeur :

- DB d'instance
  La valeur utilisée est celle qui est enregistrée dans le DB d'instance.
  La valeur doit être actualisée dans le DB d'instance par le programme utilisateur.
  L'instruction ne doit pas mentionner de valeur.
  Une modification via IHM est possible.
- Instruction
  La valeur utilisée est celle qui est interconnectée à l'instruction.
  La valeur est écrite dans le DB d'instance à chaque appel de l'instruction.
  Une modification via IHM n'est pas possible.

Grandeur physique

Sélectionnez la grandeur physique et l'unité pour la valeur de consigne et la mesure dans le groupe "Type de régulation". La valeur de consigne et la mesure s'afficheront dans cette unité.

Sens de régulation

La plupart du temps, une augmentation de la mesure doit être atteinte avec une augmentation de la valeur de réglage. Dans ce cas, on parle d'un sens de régulation normal.
PID_Compact ne fonctionne pas avec un gain proportionnel négatif. Pour réduire la mesure au moyen d'une valeur de réglage plus élevée, cochez la case "Inversion du sens de régulation".
Exemples

- L'ouverture d'une vanne d'écoulement fait baisser le niveau de remplissage d'un réservoir.
- En raison d'une plus grande performance de refroidissement, la température baisse.

Comportement au démarrage

1. Pour passer en mode de fonctionnement "Inactif" après un redémarrage de la CPU, décochez la case "Activer mode après redémarrage de CPU".
   Pour passer dans le mode de fonctionnement enregistré dans Mode après un redémarrage de la CPU, cochez la case "Activer mode après redémarrage de CPU".
2. Dans la liste déroulante "Mettre le mode à", sélectionnez le mode de fonctionnement qui doit être activé après un chargement complet dans l'appareil.
   Après un chargement complet dans l'appareil, PID_Compact démarre dans le mode de fonctionnement choisi. Lors de chaque redémarrage ultérieur, PID_Compact démarre dans le mode de fonctionnement qui a été enregistré en dernier dans Mode.

Exemple

Vous avez coché la case "Activer le mode après un redémarrage de la CPU" et choisi l'entrée "Optimisation préalable" dans la liste "Régler mode sur". Après un chargement complet dans l'appareil, PID_Compact démarre dans le mode de fonctionnement "Optimisation préalable". Si l'optimisation préalable est toujours activée, PID_Compact démarre à nouveau dans le mode de fonctionnement "Optimisation préalable" après le redémarrage de la CPU. Si l'optimisation préalable a été effectuée avec succès et que le mode automatique est activé, PID_Compact démarre en "mode automatique" après le redémarrage de la CPU.

Marche à suivre

Pour spécifier une consigne fixe, procédez de la manière suivante :

1. Sélectionnez "DB d'instance".
2. Entrez une consigne, par exemple 80 °C.
3. Supprimez éventuellement une entrée au niveau de l'instruction.

Pour spécifier une consigne variable, procédez de la manière suivante :

1. Sélectionnez "Instruction".
2. Entrez le nom de la variable REAL dans laquelle la consigne est enregistrée.
   Vous pouvez attribuer des valeurs différentes à la variable REAL dans le programme, par ex. pour une modification de la consigne déclenchée par horloge.

Si vous utilisez directement la valeur de l'entrée analogique, PID_Compact met la valeur de l'entrée analogique à l'échelle dans la grandeur physique.

Si vous souhaitez d'abord mettre en forme la valeur de l'entrée analogique, vous devez écrire un programme propre pour la mise en forme. Par exemple, la mesure n'est pas directement proportionnelle à la valeur de l'entrée analogique. La mesure mise en forme doit être disponible au format à virgule flottante.

Marche à suivre

Pour utiliser directement la valeur de l'entrée analogique, procédez comme suit :

1. Dans la liste déroulante "Input", sélectionnez l'entrée "Input_PER".
2. Sélectionnez "Instruction" comme source.
3. Entrez l'adresse de l'entrée analogique.

Pour utiliser la mesure mise au format à virgule flottante, procédez de la manière suivante :

1. Dans la liste déroulante "Input", sélectionnez l'entrée "Input".
2. Sélectionnez "Instruction" comme source.
3. Entrez le nom de la variable dans laquelle la mesure mise en forme est enregistrée.

PID_Compact met trois valeurs de réglage à disposition. La valeur de réglage que vous utilisez dépend de votre actionneur.

- Output_PER
  L'actionneur est adressé via une sortie analogique et est commandé à l'aide d'un signal continu, par exemple 0...10 V, 4...20 mA.
- Output
  La valeur de réglage doit être mise en forme dans le programme utilisateur, p. ex. parce que l'actionneur présente un comportement non linéaire.
- Output_PWM
  L'actionneur est commandé par une sortie TOR. Des temps d'activation et de désactivation variables sont formés à partir d'une modulation de largeur d'impulsion.

Marche à suivre

Pour utiliser la valeur de réglage analogique, procédez de la manière suivante :

1. Dans la liste déroulante "Output", sélectionnez l'entrée "Output_PER (analogique)".
2. Sélectionnez "Instruction".
3. Entrez l'adresse de la sortie analogique.

Pour mettre en forme la valeur de réglage dans le programme utilisateur, procédez de la manière suivante :

1. Dans la liste déroulante "Output", sélectionnez l'entrée "Output".
2. Sélectionnez "DB d'instance".
   La valeur de réglage calculée est enregistrée dans le DB d'instance.
3. Utilisez le paramètre de sortie Output pour la mise en forme de la valeur de réglage.
4. Transférez la valeur de réglage mise en forme à l'actionneur via une sortie TOR ou analogique de la CPU.

Pour utiliser la valeur de réglage TOR, procédez de la manière suivante :

1. Dans la liste déroulante "Output", sélectionnez l'entrée "Output_PWM".
2. Sélectionnez "Instruction".
3. Entrez l'adresse de la sortie TOR.

5.2.1.2 Paramètres de la mesure à partir de V2

Si vous avez configuré l'utilisation de Input_PER dans les paramètres de base, vous devez convertir la valeur de l'entrée analogique dans la grandeur physique de la mesure. La configuration actuelle est affichée dans le champ d'affichage Input_PER.
Si la mesure est directement proportionnelle à la valeur de l'entrée analogique, Input_PER est mis à l'échelle à l'aide d'une paire de valeurs supérieure et inférieure.

Marche à suivre

Pour mettre à l'échelle la mesure, procédez comme suit :

1. Indiquez la paire de valeurs inférieure dans les champs de saisie "Mesure inférieure à l'échelle" et "Bas".
2. Indiquez la paire de valeurs supérieure dans les champs de saisie "Mesure supérieure à l'échelle" et "Haut".

Des valeurs par défaut pour les paires de valeurs sont enregistrées dans la configuration matérielle. Pour utiliser les paires de valeurs de la configuration matérielle, procédez comme suit :

1. Sélectionnez l'instruction PID_Compact dans l'éditeur de programmation.
2. Dans les paramètres de base, reliez Input_PER à une entrée analogique.
3. Dans les paramètres de la mesure, cliquez sur le bouton "Paramétrage automatique".

Les valeurs existantes sont écrasées par les valeurs de la configuration matérielle.

Vous devez définir une limite absolue supérieure et inférieure significative de la mesure comme valeurs limites pour votre système réglé. Dès que ces limites sont dépassées, une erreur survient (ErrorBits = 0001h). L'optimisation est interrompue si les limites de la mesure sont dépassées. Configurez la réaction de PID_Compact en cas d'erreur en mode automatique dans les paramètres des valeurs de réglage.

5.2.1.3 Paramètres avancés à partir de V2

Dans la fenêtre de configuration "Surveillance de la mesure", configurez une limite d'alerte inférieure et une limite d'alerte supérieure de la mesure. Si l'une de ces limites d'alerte est dépassée ou n'est pas atteinte pendant le fonctionnement, l'instruction PID_Compact affiche un avertissement :

- Dans le paramètre de sortie InputWarning_H, lorsque la limite d'alerte supérieure a été dépassée
- Dans le paramètre de sortie InputWarning_L, lorsque la limite d'alerte inférieure n'est pas atteinte

Les limites d'alerte doivent se situer entre la limite supérieure et la limite inférieure de la mesure.
Si vous n'indiquez pas de valeur, les limites supérieure et inférieure de la mesure sont utilisées.

Exemple

Limite supérieure de la mesure = 98 °C ; limite d'alerte supérieure = 90 °C
Limite d'alerte inférieure = 10 °C ; limite inférieure de la mesure = 0 °C
PID_Compact se comporte comme suit :

Mesure	InputWarning_H	InputWarning_L	ErrorBits	Mode de fonctionnement
> 98 °C	TRUE	FALSE	0001h	Inactif ou Valeur de réglage de remplacement avec surveillance des erreurs
≤ 98 °C et > 90 °C	TRUE	FALSE	0000h	Mode automatique
≤ 90 °C et ≥ 10 °C	FALSE	FALSE	0000h	Mode automatique
≤ 10°C et ≥ 0 °C	FALSE	TRUE	0000h	Mode automatique
< 0 °C	FALSE	TRUE	0001h	Inactif ou Valeur de réglage de remplacement avec surveillance des erreurs

Configurez la réaction de PID_Compact en cas de dépassement de la limite supérieure ou inférieure de la mesure dans les paramètres des valeurs de réglage.

Voir aussi

Paramètres State et Mode à partir de V2 (Page 264)

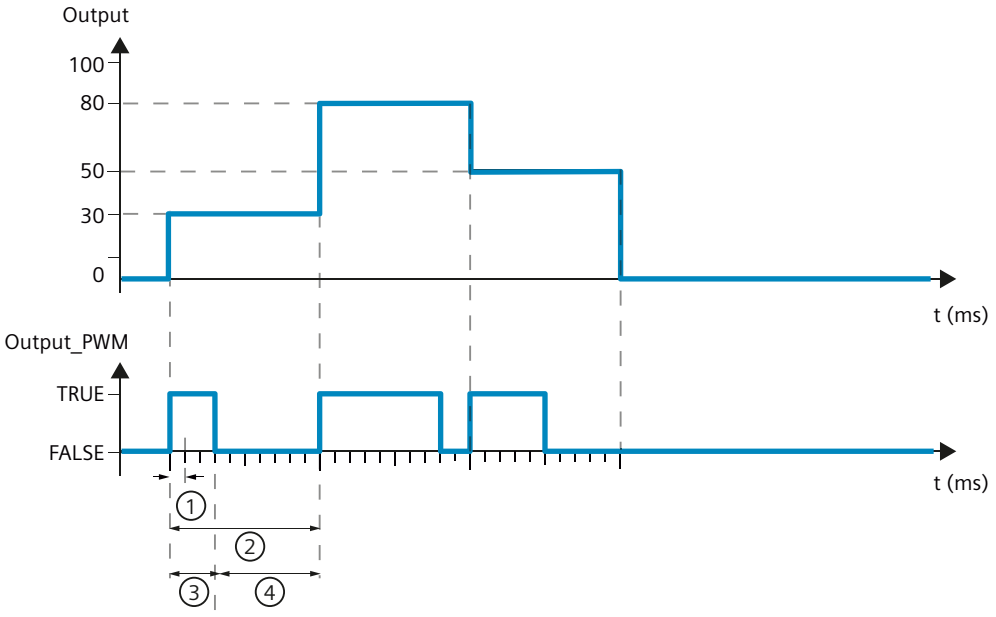
La valeur au paramètre de sortie Output est transformée via une modulation de largeur d'impulsions en une suite d'impulsions, fournie au paramètre de sortie Output_PWM.

Output est calculé dans la période d'échantillonnage de l'algorithme PID. La période d'échantillonnage est utilisée comme période de la modulation de largeur d'impulsion.

La période d'échantillonnage de l'algorithme PID est déterminée pendant l'optimisation préalable ou fine. Si vous réglez manuellement les paramètres PID, vous devez aussi configurer la période d'échantillonnage de l'algorithme PID.

Output_PWM est fourni dans la période d'échantillonnage PID_Compact. La période d'échantillonnage PID_Compact correspond au temps de cycle de l'OB appelant.

La durée d'impulsion est proportionnelle à la valeur à Output et s'élève toujours à un multiple entier de la période d'échantillonnage PID_Compact.



① Période d'échantillonnage PID_Compact

② Période d'échantillonnage de l'algorithme PID

③ Durée d'impulsion

④ Durée de pause

Le "plus petit temps ON" et le "plus petit temps OFF" sont arrondis à un multiple entier de la période d'échantillonnage PID_Compact.

Une impulsion ou une pause n'est jamais plus courte que le plus petit temps ON ou OFF. Les imprécisions qui en résultent sont totalisées et compensées au cycle suivant.

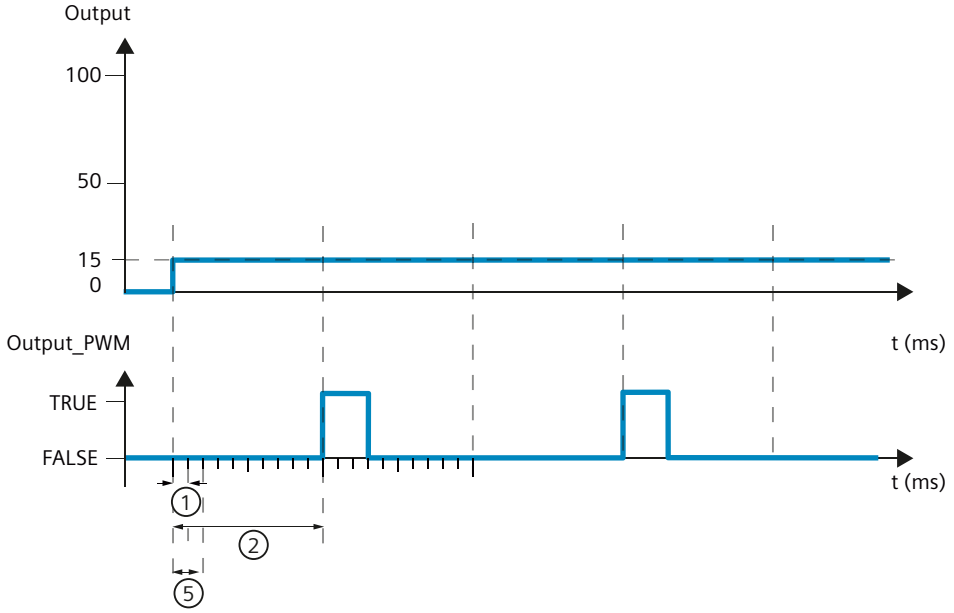
Exemple

Période d'échantillonnage PID_Compact (correspond au temps de cycle de l'OB appelant) = 100 ms

Période d'échantillonnage de l'algorithme PID (correspond à la durée de période) = 1000 ms

Plus petit temps ON = 200 ms

Output s'élève toujours à 15 %. La plus petite impulsion que PID_Compact peut fournir correspond à 20 %. Aucune impulsion n'est donnée dans le premier cycle. L'impulsion du premier cycle qui n'a pas été donnée est ajoutée à celle du deuxième cycle.



① Période d'échantillonnage PID_Compact

② Période d'échantillonnage de l'algorithme PID

⑤ Plus petit temps ON

Pour réduire la fréquence de commutation et pour ménager l'actionneur, rallongez les plus petits temps ON et OFF.
Si vous utilisez "Output" ou "Output_PER", vous devez configurer le plus petit temps ON et le plus petit temps OFF à la valeur 0.0.

REMARQUE

Les plus petits temps ON et OFF s'appliquent uniquement au paramètre de sortie Output_PWM et ne sont pas utilisés pour d'éventuels générateurs d'impulsions intégrés dans la CPU.

Limites de valeur de réglage

Dans la fenêtre de configuration "Limites de la valeur de réglage", configurez les limites absolues de votre valeur de réglage en pourcentage. Les limites absolues de la valeur de réglage ne seront pas dépassées, ni par le haut, ni par le bas, que ce soit en mode manuel ou en mode automatique. Si une valeur de réglage est prédéfinie en dehors des limites en mode manuel, la valeur effective est restreinte aux limites configurées dans la CPU.

Les limites de valeur de réglage doivent être dans le sens de régulation.

Les valeurs admissibles pour les limites de la valeur de réglage dépendent de Output utilisé.

Output	de -100.0 à 100.0 %
Output_PER	de -100.0 à 100.0 %
Output_PWM	de 0.0 à 100.0 %

Comportement en cas d'erreur

IMPORTANT
Votre installation peut être endommagée. En cas d'erreur, si vous fournissez "Valeur actuelle pour la durée de l'erreur" ou "Valeur de réglage de remplacement pour la durée de l'erreur", PID_Compact reste en mode automatique. Les limites de la mesure peuvent alors être dépassées et votre installation endommagée. Configurez un comportement en cas d'erreur pour votre système réglé, qui protège votre installation de tout endommagement.

PID_Compact est préréglé de telle façon qu'en cas d'erreur, la régulation reste active dans la plupart des cas. Lorsque des erreurs apparaissent fréquemment en mode régulation, cette valeur par défaut détériore le comportement de régulation. Vérifiez alors le paramètre Errorbits et éliminez la cause d'erreur.

En cas d'erreur, PID_Compact fournit une valeur de réglage configurable :

- Zéro (inactif)
  PID_Compact fournit pour toutes les erreurs 0.0 comme valeur de réglage et commute en mode de fonctionnement "Inactif". Le régulateur n'est réactivé que par un front descendant à Reset ou un front montant à ModeActivate.
- Valeur actuelle pour la durée de l'erreur
  Si les erreurs suivantes surviennent en **mode automatique**, PID_Compact repasse en mode automatique dès que les erreurs ont disparu.
  Si l'une ou plusieurs des erreurs suivantes apparaissent, PID_Compact reste en mode automatique :
  - 0001h : Le paramètre "Input" se trouve en dehors des limites de la mesure.
  - 0800h : Erreur de temps d'échantillonnage
  - 40000h : Valeur invalide au paramètre Disturbance.

Si l'une ou plusieurs des erreurs suivantes surviennent en **mode automatique**, PID_Compact passe en mode de fonctionnement "Valeur de réglage de remplacement avec surveillance des erreurs" et fournit la dernière valeur de réglage valide :

- 0002h : Valeur invalide au paramètre Input_PER.
- 0200h : Valeur invalide au paramètre Input.
- 0400h : Le calcul de la valeur de réglage a échoué.
- 1000h : Valeur invalide au paramètre Setpoint.

En cas d'erreur en **mode manuel**, PID_Compact continue d'utiliser la valeur manuelle comme valeur de réglage. Si la valeur manuelle est invalide, la valeur de réglage de remplacement est utilisée. Si la valeur manuelle et la valeur de réglage de remplacement sont invalides, la limite inférieure de la valeur de réglage est utilisée.

Si l'erreur suivante survient pendant une **optimisation préalable ou fine**, PID_Compact reste en mode de fonctionnement actif :

- 0020h : L'optimisation préalable n'est pas autorisée pendant l'optimisation fine.

Pour toutes les autres erreurs, PID_Compact interrompt l'optimisation et passe dans le mode de fonctionnement à partir duquel l'optimisation a été lancée.

Dès que toutes les erreurs ont disparu, PID_Compact repasse en mode automatique.

- Valeur de réglage de remplacement pour la durée de l'erreur

  PID_Compact utilise la valeur de remplacement.

  Si l'erreur suivante se produit, PID_Compact reste en mode de fonctionnement "Valeur de réglage de remplacement avec surveillance des erreurs" et fournit la limite inférieure de la valeur de réglage :

  - 20000h : Valeur invalide à la variable SubstituteOutput.

  Pour toutes les autres erreurs, PID_Compact se comporte comme décrit dans "Valeur actuelle pour la durée de l'erreur".

Voir aussi

Paramètres State et Mode à partir de V2 (Page 264)

Les paramètres PID sont affichés dans la fenêtre de configuration "Paramètres PID". Les paramètres PID sont adaptés à votre système réglé pendant l'optimisation. Vous n'avez pas besoin d'indiquer les paramètres PID manuellement.

REMARQUE

Les paramètres PID actuellement effectifs se trouvent pour PID_Compact V1 dans la structure sRet et pour PID_Compact à partir de V2 dans la structure Retain.CtrlParams.

Pour éviter un comportement erroné du régulateur PID, ne modifiez les paramètres PID actuellement effectifs en ligne que dans le mode de fonctionnement "Inactif".

Si vous souhaitez modifier les paramètres PID dans les modes de fonctionnement "Mode automatique" ou "Mode manuel", veuillez les modifier de la manière suivante :

- PID_Compact V1 : Modifiez les paramètres PID dans la structure sBackUp et appliquez ces modifications au moyen de sPid_Cmpt.b_LoadBackUp = TRUE dans la structure sRet.
- PID_Compact à partir de V2 : Modifiez les paramètres PID dans la structure CtrlParamsBackUp et appliquez ces modifications au moyen de LoadBackUp = TRUE dans la structure Retain.CtrlParams.

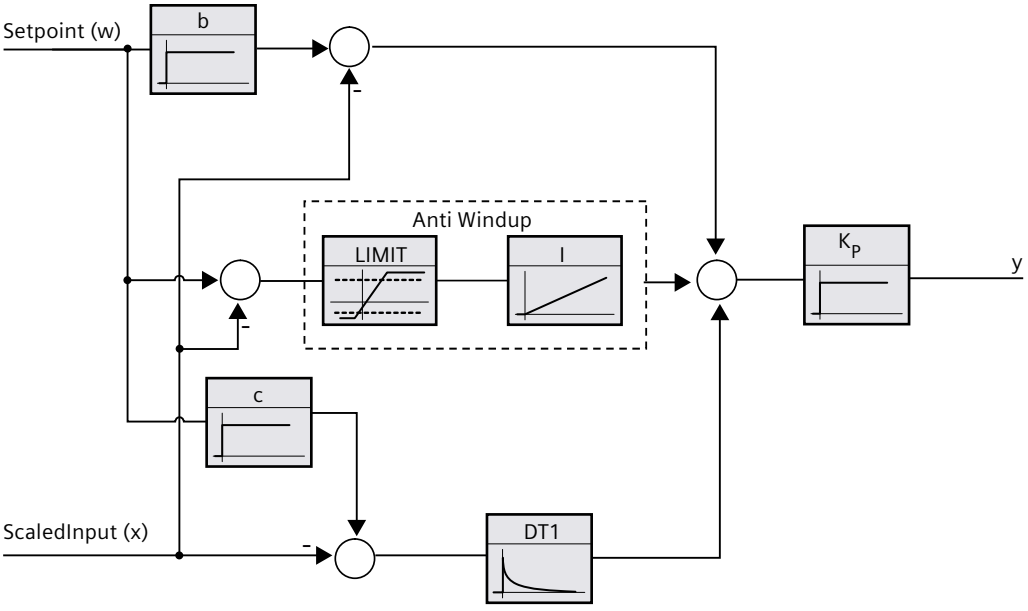
Apporter des modifications en ligne aux paramètres PID dans le mode de fonctionnement "Mode automatique" peut provoquer des sauts de la valeur de sortie.

L'algorithme PID fonctionne selon la formule suivante :

$$y = K_p \left[(b \cdot w - x) + \frac{1}{T_I \cdot s}(w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1}(c \cdot w - x) \right]$$

Symbole	Description
y	Valeur de sortie de l'algorithme PID
K_p	Gain proportionnel
s	Opérateur de Laplace
b	Pondération de l'action P
w	Consigne
x	Mesure
T_I	Temps d'intégration
a	Coefficient de l'action par dérivation (action par dérivation T1 = a × T_D)
T_D	Temps de dérivation
c	Pondération de l'action D

Le graphique suivant illustre l'intégration des paramètres dans l'algorithme PID.



Tous les paramètres PID sont rémanents. Si vous saisissez les paramètres PID manuellement, vous devez charger entièrement PID_Compact.
Charger des objets technologiques dans l'appareil (Page 48)

Gain proportionnel

La valeur indique le gain proportionnel du régulateur. PID_Compact ne fonctionne pas avec un gain proportionnel négatif. Inversez le sens de régulation dans Réglages de base > Type de régulation.

Temps d'intégration

Le temps d'intégration détermine le temps de réponse de l'action I. La désactivation de l'action I s'obtient avec un temps d'intégration = 0.0. Si le temps d'intégration est modifié en ligne en mode de fonctionnement "Mode automatique" en passant d'une autre valeur à 0.0, l'action I actuelle est supprimée et il se produit un saut de la valeur de sortie.

Temps de dérivation

Le temps de dérivation détermine le temps de réponse de l'action D. La désactivation de l'action D s'obtient avec un temps de dérivation = 0.0.

Coefficient de l'action par dérivation

L'effet de l'action D est retardé par le coefficient de l'action par dérivation.

Action par dérivation = Temps de dérivation x Coefficient de l'action par dérivation

- 0.0 : L'action D n'est active que pour un seul cycle et est donc quasiment inactive.
- 0.5 : Cette valeur a fait ses preuves dans la pratique pour les systèmes réglés avec **une** constante de temps dominante.
- > 1.0 : Plus le coefficient est grand, plus l'effet de l'action D est retardé.

Pondération de l'action P

En cas de modification de consigne, vous pouvez réduire l'action P.

Les valeurs judicieuses sont comprises entre 0.0 et 1.0.

- 1.0 : Action P totalement opérante si modification de la consigne
- 0.0 : Action P non opérante si modification de la consigne

En cas de variation de la mesure, l'action P est toujours totalement opérante.

Pondération de l'action D

En cas de modification de consigne, vous pouvez réduire l'action D.

Les valeurs judicieuses sont comprises entre 0.0 et 1.0.

- 1.0 : En cas de modification de la consigne, l'action D est totalement opérante
- 0.0 : En cas de modification de la consigne, l'action D n'est pas opérante

En cas de variation de la mesure, l'action D est toujours totalement opérante.

Période d'échantillonnage de l'algorithme PID

Comme le système réglé nécessite un certain temps pour réagir à une modification de la valeur de sortie, il est judicieux de ne pas calculer cette valeur à chaque cycle. La période d'échantillonnage de l'algorithme PID est le temps entre deux calculs de valeurs de sortie. Il est déterminé pendant l'optimisation et arrondi à un multiple du temps de cycle. Toutes les autres fonctions de PID_Compact sont exécutées lors de chaque appel.

Si vous utilisez Output_PWM, la période d'échantillonnage de l'algorithme PID est utilisée comme durée de la période de la modulation de largeur d'impulsions. La précision du signal de sortie est déterminée par le rapport entre la période d'échantillonnage de l'algorithme PID et le temps de cycle de l'OB. Pour cette raison, il est recommandé que le temps de cycle ne dépasse pas un dixième de la période d'échantillonnage de l'algorithme PID.

Règle pour l'optimisation

Dans la liste déroulante "Structure du régulateur", sélectionnez le calcul de paramètres PID ou PI.

- **PID**
  Des paramètres PID sont calculés pendant l'optimisation préalable et l'optimisation fine.
- **PI**
  Des paramètres PI sont calculés pendant l'optimisation préalable et l'optimisation fine.
- **Personnalisé**
  Si vous avez réglé des structures de régulateur différentes pour l'optimisation préalable et l'optimisation fine via le programme utilisateur, "Personnalisé" est affiché dans la liste déroulante.

Les paramètres PID sont affichés dans la fenêtre de configuration "Paramètres PID". Les paramètres PID sont adaptés à votre système réglé pendant l'optimisation, à l'exception de la largeur de la zone morte, qui doit être configurée manuellement.

REMARQUE

Les paramètres PID actuellement effectifs se trouvent dans la structure Retain.CtrlParams.

Pour éviter un comportement erroné du régulateur PID, ne modifiez les paramètres PID actuellement effectifs en ligne que dans le mode de fonctionnement "Inactif".

Si vous souhaitez modifier en ligne les paramètres PID dans les modes de fonctionnement "Mode automatique" ou "Mode manuel", modifiez les paramètres PID dans la structure CtrlParamsBackUp et appliquez ces modifications via LoadBackUp = TRUE dans la structure Retain.CtrlParams.

Apporter des modifications en ligne aux paramètres PID dans le mode de fonctionnement "Mode automatique" peut provoquer des sauts de la valeur de sortie.

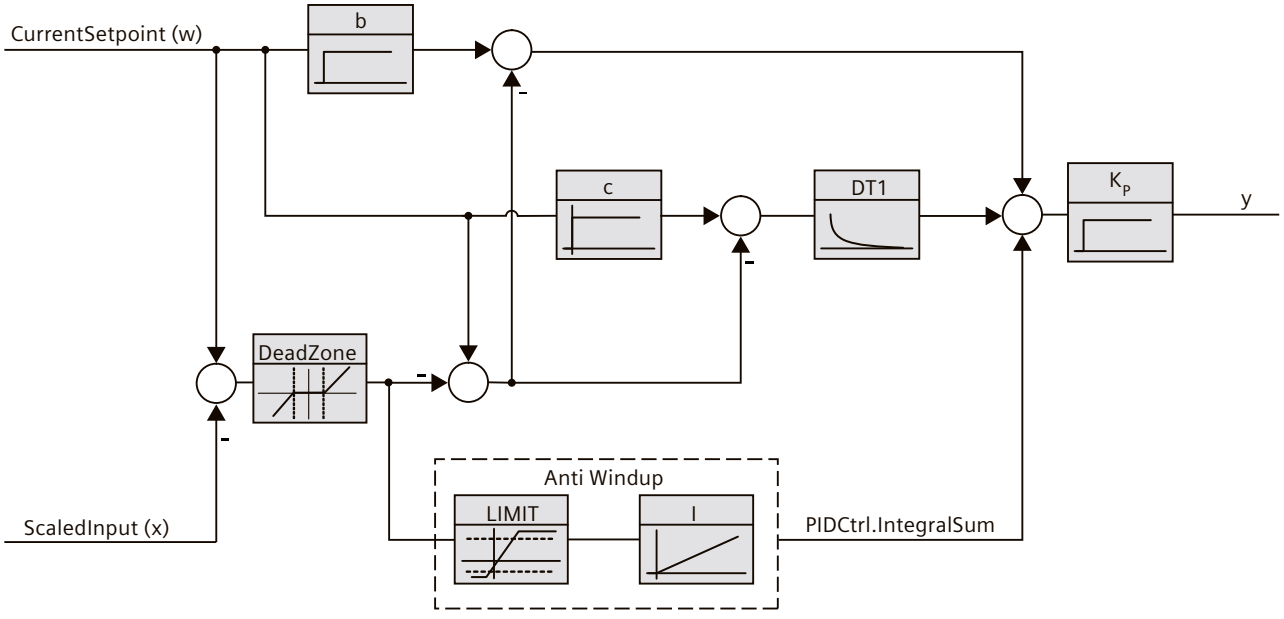
PID_Compact est un régulateur PIDT1 avec Anti-Windup et pondération de l'action P et D.

L'algorithme PID fonctionne selon la formule suivante (zone morte désactivée) :

$$y = K_p \left[(b \cdot w - x) + \frac{1}{T_I \cdot s}(w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1}(c \cdot w - x) \right]$$

Symbole	Description	Paramètres correspondants de l'instruction PID_Compact
y	Valeur de sortie de l'algorithme PID	-
K_p	Gain proportionnel	Retain.CtrlParams.Gain
s	Opérateur de Laplace	-
b	Pondération de l'action P	Retain.CtrlParams.PWeighting
w	Consigne	CurrentSetpoint
x	Mesure	ScaledInput
T_I	Temps d'intégration	Retain.CtrlParams.Ti
a	Coefficient de l'action par dérivation (action par dérivation T1 = a × T_D)	Retain.CtrlParams.TdFiltRatio
T_D	Temps de dérivation	Retain.CtrlParams.Td
c	Pondération de l'action D	Retain.CtrlParams.DWeighting
DeadZone	Largeur de zone morte	Retain.CtrlParams.DeadZone

Le graphique suivant illustre l'intégration des paramètres dans l'algorithme PID.



Tous les paramètres PID sont rémanents. Si vous saisissez les paramètres PID manuellement, vous devez charger entièrement PID_Compact.
Charger des objets technologiques dans l'appareil

Gain proportionnel

La valeur indique le gain proportionnel du régulateur. PID_Compact ne fonctionne pas avec un gain proportionnel négatif. Inversez le sens de régulation dans Réglages de base > Type de régulation.

Temps d'intégration

Le temps d'intégration détermine le temps de réponse de l'action I. La désactivation de l'action I s'obtient avec un temps d'intégration = 0.0. Si le temps d'intégration est modifié en ligne en mode de fonctionnement "Mode automatique" en passant d'une autre valeur à 0.0, l'action I actuelle est supprimée et il se produit un saut de la valeur de sortie.

Si la valeur de sortie atteint en mode automatique une limite de valeur de sortie, l'action I est arrêtée en fonction du sens (Anti-Windup). À partir de PID_Compact version 3.0, l'action I est de plus limitée activement, afin d'éviter un comportement de régulation différé, p. ex. lors de la modification des limites de valeur de sortie. La modification des variables suivantes peut entrainer une adaptation de l'action I en mode automatique :

- Limites de valeur de sortie (variables Config.OutputLowerLimit et Config.OutputUpperLimit)
- Consigne (variable Setpoint)
- Gain proportionnel (variable Retain.CtrlParams.Gain)
- Pondération de l'action P (variable Retain.CtrlParams.PWeighting)
- Grandeur perturbatrice (variable Disturbance)

Temps de dérivation

Le temps de dérivation détermine le temps de réponse de l'action D. La désactivation de l'action D s'obtient avec un temps de dérivation = 0.0.

Coefficient de l'action par dérivation

L'effet de l'action D est retardé par le coefficient de l'action par dérivation.

Action par dérivation = Temps de dérivation x Coefficient de l'action par dérivation

- 0.0 : L'action D n'est active que pour un seul cycle et est donc quasiment inactive.
- 0.5 : Cette valeur a fait ses preuves dans la pratique pour les systèmes réglés avec **une** constante de temps dominante.
- > 1.0 : Plus le coefficient est grand, plus l'effet de l'action D est retardé.

Pondération de l'action P

En cas de modification de consigne, vous pouvez réduire l'action P.

Les valeurs judicieuses sont comprises entre 0.0 et 1.0.

- 1.0 : Action P totalement opérante si modification de la consigne
- 0.0 : Action P non opérante si modification de la consigne

En cas de variation de la mesure, l'action P est toujours totalement opérante.

Pondération de l'action D

En cas de modification de consigne, vous pouvez réduire l'action D.
Les valeurs judicieuses sont comprises entre 0.0 et 1.0.

- 1.0 : En cas de modification de la consigne, l'action D est totalement opérante
- 0.0 : En cas de modification de la consigne, l'action D n'est pas opérante

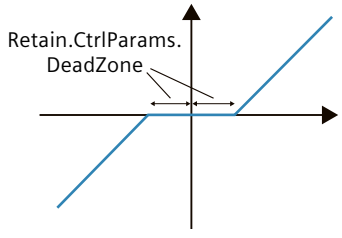
En cas de variation de la mesure, l'action D est toujours totalement opérante.

Période d'échantillonnage de l'algorithme PID

Comme le système réglé nécessite un certain temps pour réagir à une modification de la valeur de sortie, il est judicieux de ne pas calculer cette valeur à chaque cycle. La période d'échantillonnage de l'algorithme PID est le temps entre deux calculs de valeurs de sortie. Il est déterminé pendant l'optimisation et arrondi à un multiple du temps de cycle. Toutes les autres fonctions de PID_Compact sont exécutées lors de chaque appel.
Si vous utilisez Output_PWM, la période d'échantillonnage de l'algorithme PID est utilisée comme durée de la période de la modulation de largeur d'impulsions. La précision du signal de sortie est déterminée par le rapport entre la période d'échantillonnage de l'algorithme PID et le temps de cycle de l'OB. Pour cette raison, il est recommandé que le temps de cycle ne dépasse pas un dixième de la période d'échantillonnage de l'algorithme PID.

Largeur de zone morte

Si la mesure comporte des parasites, le taux de bruit a également un effet sur la valeur de sortie. La valeur de sortie peut osciller fortement si le gain de régulateur est élevé et l'action D activée. Si la mesure est comprise dans la zone morte autour de la consigne, le signal d'écart est réduit de telle sorte que l'algorithme PID ne réagisse pas et que les fluctuations inutiles de la valeur de sortie soient réduites.
La largeur de zone morte n'est pas réglée automatiquement pendant l'optimisation. Vous devez configurer correctement la largeur de zone morte manuellement. La désactivation de la zone morte s'obtient avec une largeur de zone morte = 0.0.
Lorsque la zone morte est activée, un signal d'écart (écart entre consigne et mesure) permanent peut s'établir. Cela peut avoir un effet négatif sur l'exécution d'une optimisation fine.
Si des valeurs différentes de 1.0 sont configurées pour la pondération de l'action P ou celle de l'action D, les variations de la consigne se répercutent également dans la zone morte sur la valeur de sortie.
Les variations de la mesure dans la zone morte ne se répercutent pas sur la valeur de sortie, et ce, indépendamment des pondérations.
Le graphique suivant montre l'impact de la zone morte. L'axe x / horizontal montre le signal d'écart = consigne - mesure. L'axe y / vertical montre le signal de sortie de la zone morte, qui est transmis à l'algorithme PID.



Règle pour l'optimisation

Dans la liste déroulante "Structure du régulateur", sélectionnez le calcul de paramètres PID ou PI.

- **PID**
  Des paramètres PID sont calculés pendant l'optimisation préalable et l'optimisation fine.
- **PI**
  Des paramètres PI sont calculés pendant l'optimisation préalable et l'optimisation fine.
- **Personnalisé**
  Si vous avez réglé des structures de régulateur différentes pour l'optimisation préalable et l'optimisation fine via le programme utilisateur, "Personnalisé" est affiché dans la liste déroulante.

Voir aussi

Sélection de la structure du régulateur pour un système réglé donné (Page 41)

5.2.2 Mettre en service PID_Compact à partir de V2

5.2.2.1 Optimisation préalable à partir de V2

L'optimisation préalable détermine la réponse du processus à un échelon de la valeur de réglage et recherche le point d'inflexion. Les paramètres PID sont calculés à partir de l'incrémentation maximale et du temps mort du système réglé. Les meilleurs paramètres PID sont obtenus pendant l'exécution d'une optimisation préalable et d'une optimisation fine.
Plus la mesure est stable, plus il sera facile de déterminer des paramètres PID précis. Un bruit de la mesure est acceptable tant que la croissance de la mesure est nettement supérieure au bruit. Cela est plutôt le cas en mode de fonctionnement "Inactif" ou "Mode manuel". Les paramètres PID sont sauvegardés avant qu'ils ne soient recalculés.

Condition

- L'instruction "PID_Compact" est appelée dans un OB d'alarme cyclique.
- ManualEnable = FALSE
- Reset = FALSE
- PID_Compact se trouve en mode de fonctionnement "Mode manuel", "Inactif" ou "Mode automatique".
- La consigne et la mesure se trouvent dans les limites configurées (voir configuration "Surveillance de la mesure").
- La différence entre la consigne et la mesure représente plus de 30 % de la différence entre la limite supérieure et la limite inférieure de la mesure.
- L'écart entre la consigne et la mesure est > 50% de la consigne.

Marche à suivre

Pour réaliser l'"optimisation préalable", procédez de la manière suivante :

1. Dans la navigation de projet, double-cliquez sur l'entrée "PID_Compact > Mise en service".
2. Dans la liste déroulante "Type d'optimisation", sélectionnez l'entrée "Optimisation préalable".
3. Cliquez sur l'icône "Start".
   - Une liaison en ligne est établie.
   - L'enregistrement des valeurs démarre.
   - L'optimisation préalable est lancée.
   - Les étapes actuelles et éventuelles erreurs s'affichent dans le champ "Etat". La barre de progression affiche la progression de l'étape actuelle.

REMARQUE

Cliquez sur l'icône "Stop" lorsque la barre de progression a atteint 100 % et qu'il faut supposer un blocage de l'optimisation. Vérifiez la configuration de l'objet technologique et redémarrez éventuellement l'optimisation.

Résultat

Si l'optimisation préalable a été réalisée sans message d'erreur, les paramètres PID ont été optimisés. PID_Compact passe en mode automatique et utilise les paramètres optimisés. Les paramètres PID optimisés sont conservés lors d'une mise hors tension et d'un redémarrage de la CPU.

Si une optimisation préalable n'est pas possible, PID_Compact se comporte comme cela a été défini sous Comportement en cas d'erreur.

Voir aussi

Paramètres State et Mode à partir de V2 (Page 264)

5.2.2.2 Optimisation fine à partir de V2

L'optimisation fine génère une oscillation constante limitée de la mesure. Les paramètres PID sont optimisés, pour le point de fonctionnement, à partir de l'amplitude et de la fréquence de cette oscillation. Tous les paramètres PID sont recalculés à partir des résultats. Les paramètres PID de l'optimisation fine montrent généralement un meilleur comportement de référence et de perturbation que les paramètres PID de l'optimisation préalable. Les meilleurs paramètres PID sont obtenus pendant l'exécution d'une optimisation préalable et d'une optimisation fine.

PID_Compact essaie automatiquement de créer une oscillation supérieure au bruit de la mesure. La stabilité de la mesure n'influence l'optimisation fine que de manière insignifiante. Les paramètres PID sont sauvegardés avant qu'ils ne soient recalculés.

Condition

- L'instruction PID_Compact est appelée dans un OB d'alarme cyclique.
- ManualEnable = FALSE
- Reset = FALSE
- La consigne et la mesure se trouvent dans les limites configurées.
- La boucle de régulation est en régime stationnaire au point de fonctionnement. Le point de fonctionnement est atteint lorsque la mesure correspond à la consigne.
- Aucune perturbation n'est attendue.
- PID_Compact se trouve en mode de fonctionnement Inactif, Mode automatique ou Mode manuel.

Déroulement dépendant de la situation de départ

Vous pouvez démarrer l'optimisation fine à partir des modes de fonctionnement "Inactif", "Mode automatique" ou "Mode manuel". L'optimisation fine se déroule de la manière suivante au démarrage :

- Mode automatique
  Si vous souhaitez améliorer les paramètres PID existants à l'aide de l'optimisation, démarrez l'optimisation fine à partir du mode automatique.
  PID_Compact régule avec les paramètres PID existants jusqu'à ce que la boucle de régulation soit en régime établi et que les conditions pour une optimisation fine soient remplies. C'est seulement après cela que l'optimisation fine commence.
- Inactif ou mode manuel
  Une optimisation préalable est lancée lorsque les conditions correspondantes sont réunies. Une régulation est effectuée avec les paramètres PID déterminés jusqu'à ce que la boucle de régulation soit en régime établi et que les conditions pour une optimisation fine soient remplies. C'est seulement après cela que l'optimisation fine commence. Si une optimisation préalable n'est pas possible, PID_Compact se comporte comme cela a été défini sous Comportement en cas d'erreur.
  Quand la mesure est déjà trop proche de la consigne pour une optimisation préalable, le système essaie d'atteindre la consigne avec la valeur de réglage mini ou maxi. Cela peut entraîner une suroscillation élevée.

Marche à suivre

Pour réaliser l'"optimisation fine", procédez de la manière suivante :

1. Dans la liste déroulante "Type d'optimisation", sélectionnez l'entrée "Optimisation fine".
2. Cliquez sur l'icône "Start".
   - Une liaison en ligne est établie.
   - L'enregistrement des valeurs démarre.
   - Le déroulement de l'optimisation fine démarre.
   - Les étapes actuelles et éventuelles erreurs s'affichent dans le champ "Etat". La barre de progression affiche la progression de l'étape actuelle.

REMARQUE

Cliquez sur l'icône "Stop" dans le groupe "Type d'optimisation" lorsque la barre de progression a atteint 100 % et qu'il faut supposer un blocage de l'optimisation. Vérifiez la configuration de l'objet technologique et redémarrez éventuellement l'optimisation.

Résultat

Si aucune erreur n'est apparue pendant l'optimisation fine, les paramètres PID ont été optimisés. PID_Compact passe en mode automatique et utilise les paramètres optimisés. Les paramètres PID optimisés sont conservés lors d'une mise hors tension et d'un redémarrage de la CPU.

Si des erreurs sont apparues au cours de l'"optimisation fine", PID_Compact se comporte comme cela a été défini dans Comportement en cas d'erreur.

Voir aussi

Paramètres State et Mode à partir de V2 (Page 264)

5.2.2.3 Mode de fonctionnement "mode manuel" à partir de V2

Ce paragraphe décrit comment utiliser le mode de fonctionnement "Mode manuel" dans la fenêtre de mise en service de l'objet technologique "PID_Compact. En cas d'erreur, le mode manuel est également possible.

Condition

- L'instruction "PID_Compact" est appelée dans un OB d'alarme cyclique.
- Une connexion en ligne est établie avec la CPU et celle-ci se trouve à l'état de fonctionnement "RUN".

Marche à suivre

Utilisez "Mode Manuel" dans la fenêtre de mise en service quand vous souhaitez tester le système réglé en spécifiant une valeur manuelle, Pour spécifier une valeur manuelle, procédez comme suit :

1. Cliquez sur l'icône "Start".
2. Cochez la case "Mode manuel" dans la zone "État en ligne du régulateur".
   PID_Compact travaille en mode manuel. La dernière valeur de réglage actuelle reste active.
3. Dans le champ "Output", écrivez la valeur de réglage souhaitée dans l'unité %.
4. Cliquez sur l'icône  .

Résultat

La valeur manuelle est écrite dans la CPU et elle est opérante immédiatement.
Retirez la coche de la case "Mode manuel" pour que la valeur de réglage soit à nouveau spécifiée par le régulateur PID. Le passage au mode automatique s'effectue sans à-coups.

Voir aussi

Paramètres State et Mode à partir de V2 (Page 264)

5.2.3 Régulation en mode alternatif avec PID_Compact à partir de V2

Régulation en mode alternatif

Dans la régulation en mode alternatif, deux ou plusieurs régulateurs agissent sur un seul actionneur commun. À chaque instant, un seul régulateur accède à l'actionneur et agit sur le processus.
Une logique décide du régulateur qui a accès à l'actionneur. Ce choix se base généralement sur une comparaison des valeurs de réglage de tous les régulateurs ; avec une sélection MAX, par ex., c'est le régulateur ayant la valeur de réglage maximale qui accède à l'actionneur.
Le choix basé sur la valeur de réglage requiert que tous les régulateurs fonctionnent en mode automatique. Les régulateurs qui n'agissent pas sur l'actionneur sont alignés (poursuite). Cela est nécessaire pour éviter les effets de windup et leur influence négative sur le comportement de régulation et sur la commutation entre les régulateurs.

PID_Compact prend en charge la régulation en mode alternatif à partir de la version 2.3 ; à cet effet, le régulateur offre un procédé simple pour aligner les régulateurs qui ne sont pas actifs :

- Les variables OverwriteInitialOutputValue et PIDCtrl.PIDInit vous permettent de configurer par défaut l'action I du régulateur en mode automatique comme si l'algorithme PID avait calculé Output = OverwriteInititalOutputValue dans le dernier cycle pour la valeur de réglage.
- Pour ce faire, OverwriteInitialOutputValue est connecté à la valeur de réglage du régulateur qui a actuellement accès à l'actionneur.
- En mettant le bit PIDCtrl.PIDInit à 1, vous activez le paramétrage par défaut de l'action I ainsi que le redémarrage du cycle du régulateur et de la période de modulation de largeur d'impulsion.
- Le calcul de la valeur de réglage dans le cycle actuel est ensuite effectué sur la base de l'action I paramétrée par défaut (et alignée pour tous les régulateurs) ainsi que de l'action P et de l'action I à partir du signal d'écart actuel.
- L'action D n'est pas active pendant l'appel avec PIDCtrl.PIDInit = TRUE et ne contribue donc pas à la valeur de réglage.

Ce procédé garantit que le calcul de la valeur de réglage actuelle et donc le choix du régulateur agissant sur l'actionneur n'est effectué que sur la base de l'état actuel du processus et des paramètres PI. On évite de cette manière les effets de windup sur les régulateurs non actifs, de même que les décisions incorrectes de la logique de commutation.

Conditions préalables

- PIDCtrl.PIDInit n'est effectif que si l'action I est activée (variables Retain.CtrlParams.Ti > 0.0).
- Vous devez affecter vous-même des valeurs à PIDCtrl.PIDInit et OverwriteInitialOutputValue dans votre programme utilisateur (voir l'exemple ci-dessous). PID_Compact n'effectue aucune modification automatique de ces variables.
- PIDCtrl.PIDInit n'a d'effet que si PID_Compact est en mode automatique (paramètre State = 3).
- Si possible, choisissez une période d'échantillonnage de l'algorithme PID (variable Retain.CtrlParams.Cycle) identique pour tous les régulateurs et appelez tous les régulateurs dans le même OB d'alarme cyclique. Vous garantissez ainsi que la commutation n'a pas lieu au sein d'un cycle du régulateur ou d'une période de modulation de largeur d'impulsion.

REMARQUE

Adaptation continue des limites de valeur de réglage

Au lieu de l'alignement actif des régulateurs sans action sur l'actionneur décrit ici, d'autres systèmes de régulation proposent une solution consistant à adapter en continu les limites de valeur de réglage.

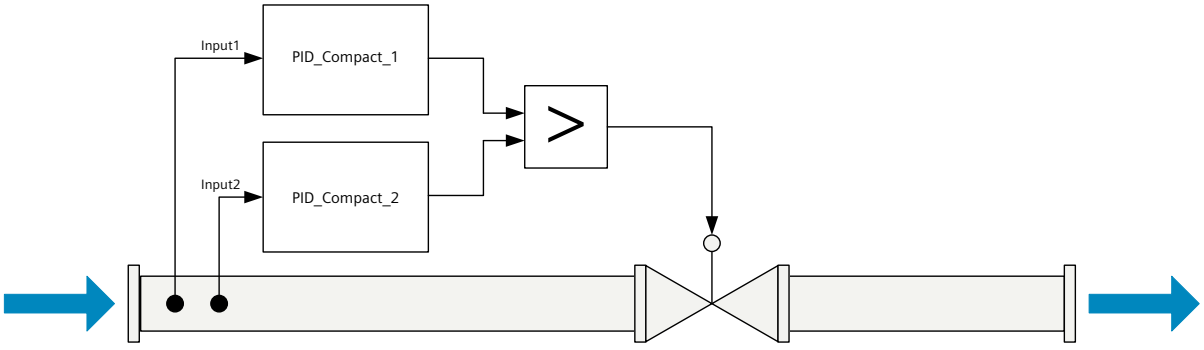
Cela n'est pas possible avec PID_Compact, car une modification des limites de valeur de réglage n'est pas prise en charge en mode automatique.

Exemple : Régulation d'un gazoduc

PID_Compact est utilisé pour la régulation d'un gazoduc.

L'objectif principal est la régulation du débit Input1. Le régulateur PID_Compact_1 est utilisé pour ce faire. En outre, le régulateur limiteur PID_Compact_2 doit maintenir la pression Input2 (mesurée en amont de la vanne dans le sens de l'écoulement) en dessous d'une limite supérieure.

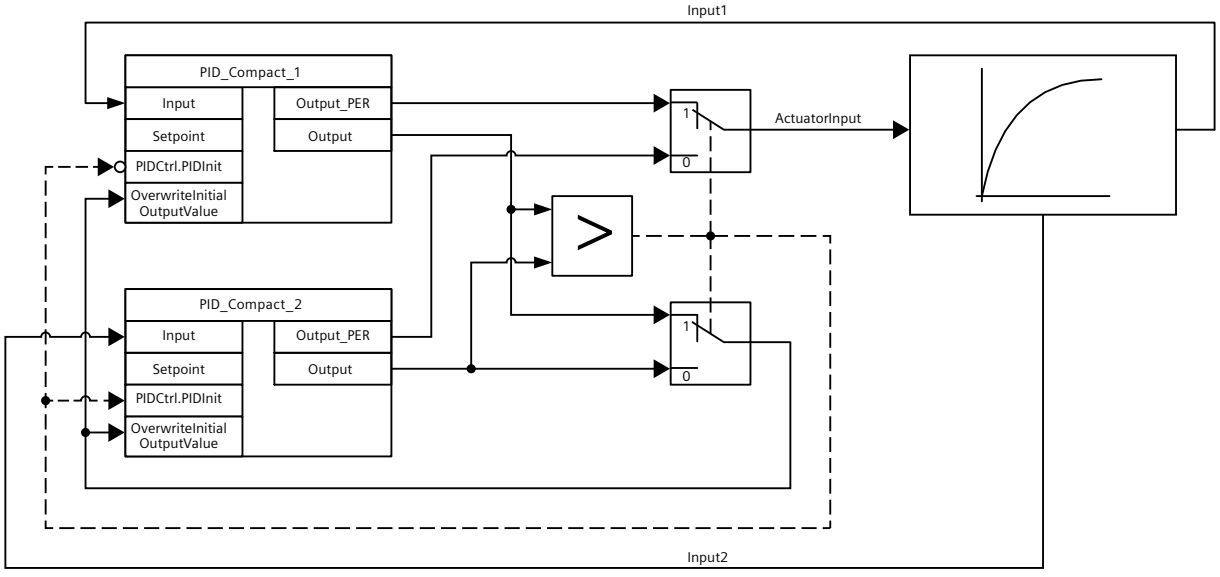
Le débit et la pression sont modifiés à l'aide d'une seule électrovanne. La valeur de réglage du régulateur correspond à l'ouverture de la vanne : si la valeur de réglage augmente, la vanne est ouverte. Le débit augmente alors (sens de régulation normal) tandis que la pression diminue (sens de régulation inverse).



La vanne est commandée via la valeur de réglage de PID_Compact au format de périphérie (paramètre Output_PER) en écrivant la variable de programme ActuatorInput.

La consigne de débit est transmise sur le paramètre PID_Compact_1.Setpoint.

La limite supérieure de pression est transmise comme consigne sur le paramètre PID_Compact_2.Setpoint.



Les deux régulateurs ne disposent que de la seule vanne comme actionneur commun. La logique de choix du régulateur agissant sur l'actionneur est réalisée ici par un sélecteur MAX de la valeur de réglage (au format Real, paramètre Output). Comme la valeur de réglage correspond à l'ouverture de la vanne, le régulateur qui demande la plus grande ouverture de la vanne est le régulateur actif.

REMARQUE

Activer l'inversion du sens de régulation

Étant donné que, pour le régulateur de pression PID_Compact_2, une augmentation de la grandeur de réglage (ouverture de la vanne) est destinée à réduire la valeur de mesure (pression), il faut activer l'inversion du sens de régulation :
PID_Compact_2.Config.InvertControl = TRUE.

En mode normal de l'installation, la valeur de mesure du débit correspond à la consigne. Le régulateur de débit PID_Compact_1 s'est stabilisé sur une valeur de réglage stationnaire PID_Compact_1.Output. En fonctionnement normal, la mesure de pression est largement inférieure à la limite supérieure qui est prescrite comme consigne pour PID_Compact_2. C'est pourquoi le régulateur de pression va vouloir fermer davantage la vanne en vue d'augmenter la pression, il va donc calculer une valeur de réglage PID_Compact_2.Output inférieure à celle du régulateur de débit PID_Compact_1.Output. Le sélecteur MAX de la logique de commutation laisse par conséquent le régulateur de débit PID_Compact_1 agir sur l'actionneur. En outre, les affectations PID_Compact_2.OverwriteInitialOutputValue = PID_Compact_1.Output et PID_Compact_2.PIDCtrl.PIDInit = TRUE garantissent que PID_Compact_2 soit aligné.

Si la pression se rapproche de la limite supérieure ou la dépasse, par ex. à la suite d'une anomalie, le régulateur de pression PID_Compact_2 va calculer une valeur de réglage supérieure afin d'ouvrir davantage la vanne et de réduire la pression. Si PID_Compact_2.Output est supérieur à PID_Compact_1.Output, le régulateur de pression PID_Compact_2 se voit attribuer l'accès à l'actionneur par le sélecteur MAX, et ouvre la vanne. En outre, les affectations PID_Compact_1.OverwriteInitialOutputValue = PID_Compact_2.Output et PID_Compact_1.PIDCtrl.PIDInit = TRUE garantissent que PID_Compact_1 soit aligné.

La pression est réduite tandis que le débit augmente et ne peut plus être maintenu à la valeur de consigne.

Une fois le défaut éliminé, la pression va encore diminuer et l'ouverture de la vanne sera réduite par le régulateur de pression. Lorsque le régulateur de débit calcule une ouverture plus grande comme valeur de réglage, l'installation retourne en mode de fonctionnement normal, si bien que le régulateur de débit PID_Compact_1 reprend l'accès à l'actionneur.

Cet exemple est réalisable à l'aide du code programme SCL suivant :

```
"PID_Compact_1"(Input := "Input1");
"PID_Compact_2"(Input := "Input2");
IF "PID_Compact_1".Output >= "PID_Compact_2".Output THEN
  "ActuatorInput" := "PID_Compact_1".Output_PER;
  "PID_Compact_1".PIDCtrl.PIDInit := FALSE;
  "PID_Compact_2".PIDCtrl.PIDInit := TRUE;
  "PID_Compact_2".OverwriteInitialOutputValue :=
  "PID_Compact_1".Output;
ELSE
  "ActuatorInput" := "PID_Compact_2".Output_PER;
  "PID_Compact_1".PIDCtrl.PIDInit := TRUE;
  "PID_Compact_2".PIDCtrl.PIDInit := FALSE;
  "PID_Compact_1".OverwriteInitialOutputValue :=
  "PID_Compact_2".Output;
END_IF;
```

5.2.4 Simuler PID_Compact à partir de V2 avec PLCSIM

REMARQUE

Simulation avec PLCSIM

La simulation de PID_Compact V2.x avec PLCSIM n'est pas prise en charge pour la CPU S7-1200.

PID_Compact V2.x peut être simulé avec PLCSIM uniquement pour la CPU S7-1500.

Lors de la simulation avec PLCSIM, le comportement dans le temps de l'API simulé n'est pas exactement le même que celui d'un API "réel". Le temps de cycle effectif d'un OB d'alarme cyclique peut présenter de plus grandes fluctuations pour un API simulé que pour un API "réel".

Dans la configuration standard, PID_Compact calcule automatiquement le temps entre les appels et le surveille vis-à-vis des fluctuations.

Lors de la simulation de PID_Compact avec PLCSIM, il est par conséquent possible de détecter une erreur de période d'échantillonnage (ErrorBits = DW#16#00000800).

Cela entraîne l'interruption d'optimisations en cours.

La réaction en mode automatique dépend de la valeur de la variable ActivateRecoverMode.

Pour éviter cela, vous devez configurer PID_Compact de la manière suivante lors de la simulation avec PLCSIM :

- CycleTime.EnEstimation = FALSE
- CycleTime.EnMonitoring = FALSE
- CycleTime.Value : affectez à cette variable le temps de cycle de l'OB d'alarme cyclique d'appel en secondes.

5.3 PID_Compact V1

5.3.1 Configurer PID_Compact V1

5.3.1.1 Paramètres de base V1

Configurez les propriétés suivantes de l'objet technologique "PID_Compact" dans la fenêtre d'inspection ou dans les "Paramètres de base" de la fenêtre de configuration.

- Grandeur physique
- Sens de régulation
- Comportement au démarrage après un Reset
- Consigne (seulement dans la fenêtre d'inspection)
- Mesure (seulement dans la fenêtre d'inspection)
- Valeur de réglage (seulement dans la fenêtre d'inspection)

Consigne, mesure et valeur de réglage

Vous ne pouvez configurer la consigne, la mesure et la valeur de réglage que dans la fenêtre d'inspection de l'éditeur de programmation. Sélectionnez la source pour chaque valeur :

- DB d'instance
  La valeur utilisée est celle qui est enregistrée dans le DB d'instance.
  La valeur doit être actualisée dans le DB d'instance par le programme utilisateur.
  L'instruction ne doit pas mentionner de valeur.
  Une modification via IHM est possible.
- Instruction
  La valeur utilisée est celle qui est interconnectée à l'instruction.
  La valeur est écrite dans le DB d'instance à chaque appel de l'instruction.
  Une modification via IHM n'est pas possible.

Grandeur physique

Sélectionnez la grandeur physique et l'unité pour la consigne et la mesure dans le groupe "Type de régulation". La consigne et la mesure sont affichées dans cette unité.

Sens de régulation

La plupart du temps, une augmentation de la mesure doit être atteinte avec une augmentation de la valeur de réglage. Dans ce cas, on parle d'un sens de régulation normal.
PID_Compact ne fonctionne pas avec un gain proportionnel négatif. Pour réduire la mesure au moyen d'une valeur de réglage plus élevée, cochez la case "Inversion du sens de régulation".
Exemples

- L'ouverture d'une vanne d'écoulement fait baisser le niveau de remplissage d'un réservoir.
- En raison d'une plus grande performance de refroidissement, la température baisse.

Comportement au démarrage après un Reset

En cas de redémarrage de la CPU, cochez l'option "Activer le dernier mode de fonctionnement au démarrage de la CPU" pour passer directement au dernier mode de fonctionnement actif.
Si la case n'est pas cochée, PID_Compact reste dans le mode de fonctionnement "Inactif".

Marche à suivre

Pour spécifier une consigne fixe, procédez de la manière suivante :

1. Sélectionnez "DB d'instance".
2. Entrez une consigne, par exemple 80 °C.
3. Supprimez éventuellement une entrée au niveau de l'instruction.

Pour spécifier une consigne variable, procédez de la manière suivante :

1. Sélectionnez "Instruction".
2. Entrez le nom de la variable REAL dans laquelle la consigne est enregistrée.
   Vous pouvez attribuer des valeurs différentes à la variable REAL dans le programme, par ex. pour une modification de la consigne déclenchée par horloge.

Si vous utilisez directement la valeur de l'entrée analogique, PID_Compact met la valeur de l'entrée analogique à l'échelle dans la grandeur physique.
Si vous souhaitez d'abord mettre en forme la valeur de l'entrée analogique, vous devez écrire un programme propre pour la mise en forme. Par exemple, la mesure n'est pas directement proportionnelle à la valeur de l'entrée analogique. La mesure mise en forme doit être disponible au format à virgule flottante.

Marche à suivre

Pour utiliser directement la valeur de l'entrée analogique, procédez comme suit :

1. Dans la liste déroulante "Input", sélectionnez l'entrée "Input_PER".
2. Sélectionnez "Instruction" comme source.
3. Entrez l'adresse de l'entrée analogique.

Pour utiliser la mesure mise au format à virgule flottante, procédez de la manière suivante :

1. Dans la liste déroulante "Input", sélectionnez l'entrée "Input".
2. Sélectionnez "Instruction" comme source.
3. Entrez le nom de la variable dans laquelle la mesure mise en forme est enregistrée.

PID_Compact met trois valeurs de réglage à disposition. La valeur de réglage que vous utilisez dépend de votre actionneur.

- Output_PER
  L'actionneur est adressé via une sortie analogique et est commandé à l'aide d'un signal continu, par exemple 0...10 V, 4...20 mA.
- Output
  La valeur de réglage doit être mise en forme dans le programme utilisateur, p. ex. parce que l'actionneur présente un comportement non linéaire.
- Output_PWM
  L'actionneur est commandé par une sortie TOR. Des temps d'activation et de désactivation variables sont formés à partir d'une modulation de largeur d'impulsion.

Marche à suivre

Pour utiliser la valeur de réglage analogique, procédez de la manière suivante :

1. Dans la liste déroulante "Output", sélectionnez l'entrée "Output_PER (analogique)".
2. Sélectionnez "Instruction".
3. Entrez l'adresse de la sortie analogique.

Pour mettre en forme la valeur de réglage dans le programme utilisateur, procédez de la manière suivante :

1. Dans la liste déroulante "Output", sélectionnez l'entrée "Output".
2. Sélectionnez "DB d'instance".
   La valeur de réglage calculée est enregistrée dans le DB d'instance.
3. Utilisez le paramètre de sortie Output pour la mise en forme de la valeur de réglage.
4. Transférez la valeur de réglage mise en forme à l'actionneur via une sortie TOR ou analogique de la CPU.

Pour utiliser la valeur de réglage TOR, procédez de la manière suivante :

1. Dans la liste déroulante "Output", sélectionnez l'entrée "Output_PWM".
2. Sélectionnez "Instruction".
3. Entrez l'adresse de la sortie TOR.

5.3.1.2 Paramétrage de la mesure V1

Configurez la normalisation de votre mesure dans la fenêtre de configuration "Paramètres de la mesure" et déterminez les limites absolues de la mesure.

Mise à l'échelle de la mesure

Si vous avez configuré l'utilisation de Input_PER dans les paramètres de base, vous devez convertir la valeur de l'entrée analogique dans la grandeur physique de la mesure. La configuration actuelle est affichée dans le champ d'affichage Input_PER.

Si la mesure est directement proportionnelle à la valeur de l'entrée analogique, Input_PER est mis à l'échelle à l'aide d'une paire de valeurs supérieure et inférieure.

1. Indiquez la paire de valeurs inférieure dans les champs de saisie "Mesure inférieure à l'échelle" et "Bas".
2. Indiquez la paire de valeurs supérieure dans les champs de saisie "Mesure supérieure à l'échelle" et "Haut".

Des valeurs par défaut pour les paires de valeurs sont enregistrées dans la configuration matérielle. Pour utiliser les paires de valeurs de la configuration matérielle, procédez comme suit :

1. Sélectionnez l'instruction PID_Compact dans l'éditeur de programmation.
2. Dans les paramètres de base, reliez Input_PER à une entrée analogique.
3. Dans les paramètres de la mesure, cliquez sur le bouton "Paramétrage automatique".

Les valeurs existantes sont écrasées par les valeurs de la configuration matérielle.

Surveiller la mesure

Fixez les limites supérieure et inférieure absolues de la mesure. Dès que ces limites sont dépassées par le haut ou par le bas durant le fonctionnement, la régulation est désactivée et la valeur de réglage est mise à 0%. Vous devez saisir des valeurs judicieuses comme valeurs limites pour votre système réglé. Pendant l'optimisation, des valeurs limites judicieuses sont importantes pour obtenir des paramètres PID optimaux.
La valeur par défaut de la "limite supérieure de la mesure" est de 120 %. A l'entrée de périphérie, la mesure peut dépasser de 18 % au plus la plage normée (dépassement haut). Un dépassement de la "limite supérieure de la mesure" ne provoque plus le signalement d'une erreur. Seuls la rupture de fil et le court-circuit sont détectés et PID_Compact passe en mode "Inactif".

⚠ ATTENTION

Si vous réglez des valeurs très élevées (par ex. $-3{,}4*10^{38}...+3{,}4*10^{38}$) comme limites de la mesure, la surveillance de la mesure sera désactivée. Une erreur peut alors entraîner des dommages sur votre installation.

Voir aussi

Surveillance de la mesure V1 (Page 104)
Limitations de la modulation de largeur d'impulsion V1 (Page 106)
Limites de la valeur de réglage V1 (Page 108)
Paramètres PID V1 (Page 108)

5.3.1.3 Paramètres avancés V1

Dans la fenêtre de configuration "Surveillance de la mesure", configurez une limite d'alerte inférieure et une limite d'alerte supérieure de la mesure. Si l'une de ces limites d'alerte est dépassée ou n'est pas atteinte pendant le fonctionnement, l'instruction PID_Compact affiche un avertissement :

- Dans le paramètre de sortie InputWarning_H, lorsque la limite d'alerte supérieure a été dépassée
- Dans le paramètre de sortie InputWarning_L, lorsque la limite d'alerte inférieure n'est pas atteinte

Les limites d'alerte doivent se situer entre la limite supérieure et la limite inférieure de la mesure.
Si vous n'indiquez pas de valeur, les limites supérieure et inférieure de la mesure sont utilisées.

Exemple

Limite supérieure de la mesure = 98 °C ; limite d'alerte supérieure = 90 °C
Limite d'alerte inférieure = 10 °C ; limite inférieure de la mesure = 0 °C
PID_Compact se comporte comme suit :

Mesure	InputWarning_H	InputWarning_L	Mode de fonctionnement
> 98 °C	TRUE	FALSE	Inactif
≤ 98 °C et > 90 °C	TRUE	FALSE	Mode automatique
≤ 90 °C et ≥ 10 °C	FALSE	FALSE	Mode automatique
≤ 10°C et ≥ 0 °C	FALSE	TRUE	Mode automatique
< 0 °C	FALSE	TRUE	Inactif

Voir aussi

Paramétrage de la mesure V1 (Page 103)
Limitations de la modulation de largeur d'impulsion V1 (Page 106)
Limites de la valeur de réglage V1 (Page 108)
Paramètres PID V1 (Page 108)

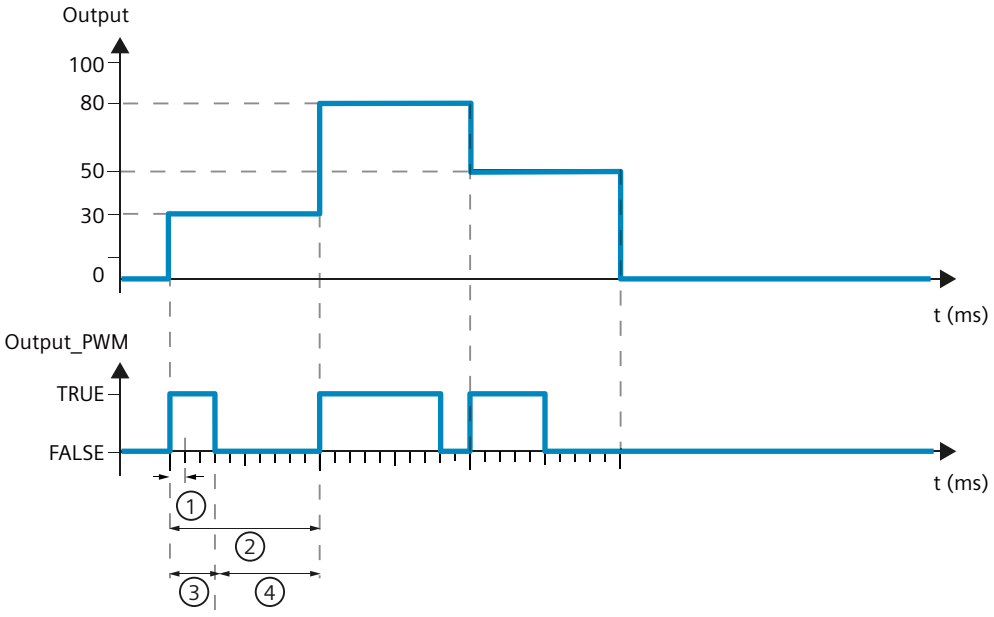
La valeur au paramètre de sortie Output est transformée via une modulation de largeur d'impulsions en une suite d'impulsions, fournie au paramètre de sortie Output_PWM.

Output est calculé dans la période d'échantillonnage de l'algorithme PID. La période d'échantillonnage est utilisée comme période de la modulation de largeur d'impulsion.

La période d'échantillonnage de l'algorithme PID est déterminée pendant l'optimisation préalable ou fine. Si vous réglez manuellement les paramètres PID, vous devez aussi configurer la période d'échantillonnage de l'algorithme PID.

Output_PWM est fourni dans la période d'échantillonnage PID_Compact. La période d'échantillonnage PID_Compact correspond au temps de cycle de l'OB appelant.

La durée d'impulsion est proportionnelle à la valeur à Output et s'élève toujours à un multiple entier de la période d'échantillonnage PID_Compact.



① Période d'échantillonnage PID_Compact

② Période d'échantillonnage de l'algorithme PID

③ Durée d'impulsion

④ Durée de pause

Le "plus petit temps ON" et le "plus petit temps OFF" sont arrondis à un multiple entier de la période d'échantillonnage PID_Compact.

Une impulsion ou une pause n'est jamais plus courte que le plus petit temps ON ou OFF. Les imprécisions qui en résultent sont totalisées et compensées au cycle suivant.

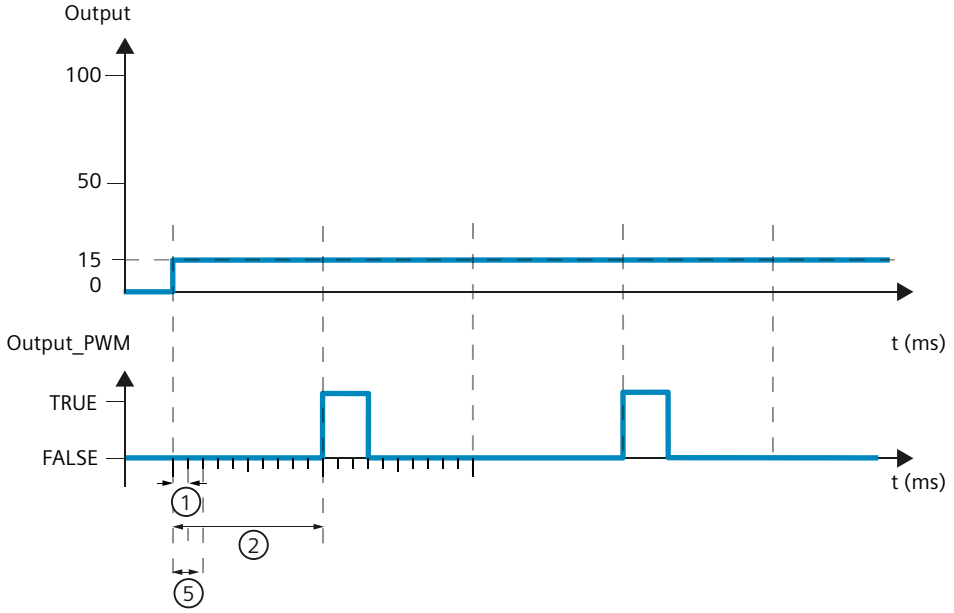
Exemple

Période d'échantillonnage PID_Compact (correspond au temps de cycle de l'OB appelant) = 100 ms

Période d'échantillonnage de l'algorithme PID (correspond à la durée de période) = 1000 ms

Plus petit temps ON = 200 ms

Output s'élève toujours à 15 %. La plus petite impulsion que PID_Compact peut fournir correspond à 20 %. Aucune impulsion n'est donnée dans le premier cycle. L'impulsion du premier cycle qui n'a pas été donnée est ajoutée à celle du deuxième cycle.



① Période d'échantillonnage PID_Compact

② Période d'échantillonnage de l'algorithme PID

⑤ Plus petit temps ON

Pour réduire la fréquence de commutation et pour ménager l'actionneur, rallongez les plus petits temps ON et OFF.
Si vous utilisez "Output" ou "Output_PER", vous devez configurer le plus petit temps ON et le plus petit temps OFF à la valeur 0.0.

REMARQUE

Les plus petits temps ON et OFF s'appliquent uniquement au paramètre de sortie Output_PWM et ne sont pas utilisés pour d'éventuels générateurs d'impulsions intégrés dans la CPU.

Voir aussi

Paramétrage de la mesure V1 (Page 103)

Surveillance de la mesure V1 (Page 104)

Limites de la valeur de réglage V1 (Page 108)

Paramètres PID V1 (Page 108)

Dans la fenêtre de configuration "Limites de la valeur de réglage", configurez les limites absolues de votre valeur de réglage en pourcentage. Les limites absolues de la valeur de réglage ne seront pas dépassées, ni par le haut, ni par le bas, que ce soit en mode manuel ou en mode automatique. Si vous spécifiez en mode manuel une valeur de réglage en dehors des limites, la valeur effective sera restreinte aux limites configurées dans la CPU.
Les valeurs admissibles pour les limites de la valeur de réglage dépendent de Output utilisé.

Output	-100.0 à 100.0
Output_PER	-100.0 à 100.0
Output_PWM	0.0 à 100.0

En cas d'erreur, PID_Compact met la valeur de réglage sur 0.0. C'est pour cela que 0.0 doit toujours se trouver à l'intérieur des limites de valeur de réglage. Pour obtenir une limite inférieure de valeur de réglage qui soit supérieure à 0.0, vous devez ajouter un décalage à Output et Output_PER dans le programme utilisateur.

Voir aussi

Paramétrage de la mesure V1 (Page 103)

Surveillance de la mesure V1 (Page 104)

Limitations de la modulation de largeur d'impulsion V1 (Page 106)

Paramètres PID V1 (Page 108)

Les paramètres PID sont affichés dans la fenêtre de configuration "Paramètres PID". Les paramètres PID sont adaptés à votre système réglé pendant l'optimisation. Vous n'avez pas besoin d'indiquer les paramètres PID manuellement.

REMARQUE

Les paramètres PID actuellement effectifs se trouvent pour PID_Compact V1 dans la structure sRet et pour PID_Compact à partir de V2 dans la structure Retain.CtrlParams.

Pour éviter un comportement erroné du régulateur PID, ne modifiez les paramètres PID actuellement effectifs en ligne que dans le mode de fonctionnement "Inactif".

Si vous souhaitez modifier les paramètres PID dans les modes de fonctionnement "Mode automatique" ou "Mode manuel", veuillez les modifier de la manière suivante :

- PID_Compact V1 : Modifiez les paramètres PID dans la structure sBackUp et appliquez ces modifications au moyen de sPid_Cmpt.b_LoadBackUp = TRUE dans la structure sRet.
- PID_Compact à partir de V2 : Modifiez les paramètres PID dans la structure CtrlParamsBackUp et appliquez ces modifications au moyen de LoadBackUp = TRUE dans la structure Retain.CtrlParams.

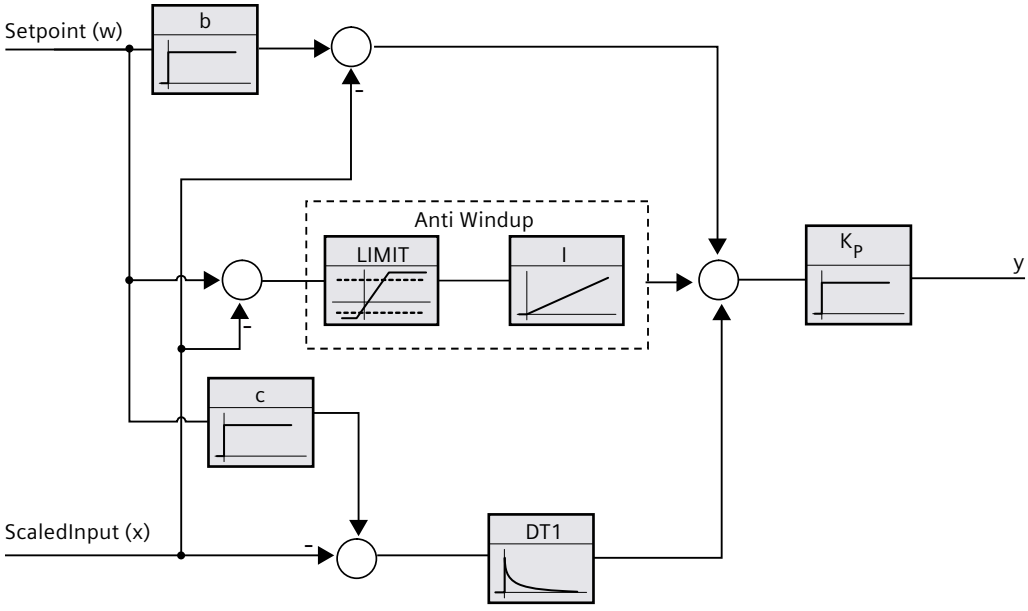
Apporter des modifications en ligne aux paramètres PID dans le mode de fonctionnement "Mode automatique" peut provoquer des sauts de la valeur de sortie.

L'algorithme PID fonctionne selon la formule suivante :

$$y = K_p \left[(b \cdot w - x) + \frac{1}{T_I \cdot s}(w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1}(c \cdot w - x) \right]$$

Symbole	Description
y	Valeur de sortie de l'algorithme PID
K_p	Gain proportionnel
s	Opérateur de Laplace
b	Pondération de l'action P
w	Consigne
x	Mesure
T_I	Temps d'intégration
a	Coefficient de l'action par dérivation (action par dérivation T1 = a × T_D)
T_D	Temps de dérivation
c	Pondération de l'action D

Le graphique suivant illustre l'intégration des paramètres dans l'algorithme PID.



Tous les paramètres PID sont rémanents. Si vous saisissez les paramètres PID manuellement, vous devez charger entièrement PID_Compact.
Charger des objets technologiques dans l'appareil (Page 48)

Gain proportionnel

La valeur indique le gain proportionnel du régulateur. PID_Compact ne fonctionne pas avec un gain proportionnel négatif. Inversez le sens de régulation dans Réglages de base > Type de régulation.

Temps d'intégration

Le temps d'intégration détermine le temps de réponse de l'action I. La désactivation de l'action I s'obtient avec un temps d'intégration = 0.0. Si le temps d'intégration est modifié en ligne en mode de fonctionnement "Mode automatique" en passant d'une autre valeur à 0.0, l'action I actuelle est supprimée et il se produit un saut de la valeur de sortie.

Temps de dérivation

Le temps de dérivation détermine le temps de réponse de l'action D. La désactivation de l'action D s'obtient avec un temps de dérivation = 0.0.

Coefficient de l'action par dérivation

L'effet de l'action D est retardé par le coefficient de l'action par dérivation.

Action par dérivation = Temps de dérivation x Coefficient de l'action par dérivation

- 0.0 : L'action D n'est active que pour un seul cycle et est donc quasiment inactive.
- 0.5 : Cette valeur a fait ses preuves dans la pratique pour les systèmes réglés avec **une** constante de temps dominante.
- > 1.0 : Plus le coefficient est grand, plus l'effet de l'action D est retardé.

Pondération de l'action P

En cas de modification de consigne, vous pouvez réduire l'action P.

Les valeurs judicieuses sont comprises entre 0.0 et 1.0.

- 1.0 : Action P totalement opérante si modification de la consigne
- 0.0 : Action P non opérante si modification de la consigne

En cas de variation de la mesure, l'action P est toujours totalement opérante.

Pondération de l'action D

En cas de modification de consigne, vous pouvez réduire l'action D.

Les valeurs judicieuses sont comprises entre 0.0 et 1.0.

- 1.0 : En cas de modification de la consigne, l'action D est totalement opérante
- 0.0 : En cas de modification de la consigne, l'action D n'est pas opérante

En cas de variation de la mesure, l'action D est toujours totalement opérante.

Période d'échantillonnage de l'algorithme PID

Comme le système réglé nécessite un certain temps pour réagir à une modification de la valeur de sortie, il est judicieux de ne pas calculer cette valeur à chaque cycle. La période d'échantillonnage de l'algorithme PID est le temps entre deux calculs de valeurs de sortie. Il est déterminé pendant l'optimisation et arrondi à un multiple du temps de cycle. Toutes les autres fonctions de PID_Compact sont exécutées lors de chaque appel.

Si vous utilisez Output_PWM, la période d'échantillonnage de l'algorithme PID est utilisée comme durée de la période de la modulation de largeur d'impulsions. La précision du signal de sortie est déterminée par le rapport entre la période d'échantillonnage de l'algorithme PID et le temps de cycle de l'OB. Pour cette raison, il est recommandé que le temps de cycle ne dépasse pas un dixième de la période d'échantillonnage de l'algorithme PID.

Règle pour l'optimisation

Dans la liste déroulante "Structure du régulateur", sélectionnez le calcul de paramètres PID ou PI.

- **PID**
  Des paramètres PID sont calculés pendant l'optimisation préalable et l'optimisation fine.
- **PI**
  Des paramètres PI sont calculés pendant l'optimisation préalable et l'optimisation fine.
- **Personnalisé**
  Si vous avez réglé des structures de régulateur différentes pour l'optimisation préalable et l'optimisation fine via le programme utilisateur, "Personnalisé" est affiché dans la liste déroulante.

5.3.2 Mettre en service PID_Compact V1

5.3.2.1 Mise en service V1

La fenêtre de mise en service vous assiste lors de la mise en service du régulateur PID. L'affichage de courbes permet de visualiser la consigne, la mesure et la valeur de réglage sur l'axe du temps. Les fonctions suivantes sont prises en charge dans la fenêtre de mise en service :

- Optimisation préalable du régulateur
- Optimisation fine du régulateur
  Servez-vous de l'optimisation fine si vous souhaitez un ajustement détaillé des paramètres PID.
- Visualisation de la régulation en cours dans la fenêtre des courbes
- Test du système réglé en spécifiant une valeur de réglage manuelle

Une liaison en ligne doit être établie avec la CPU pour toutes les fonctions.

Commande fondamentale

- Sélectionnez le temps d'actualisation souhaité dans la liste déroulante "Temps d'actualisation".
  Toutes les valeurs de la fenêtre de mise en service sont actualisées durant le temps d'actualisation sélectionné.
- Si vous souhaitez utiliser les fonctions de mise en service, cliquez sur l'icône "Démarrage" du groupe Mesure.
  L'enregistrement des valeurs démarre. Les valeurs actuelles pour la consigne, la mesure et la valeur de réglage sont écrites dans l'affichage de courbes. La commande de la fenêtre de mise en service est validée.
- Si vous souhaitez mettre fin aux fonctions de mise en service, cliquez sur l'icône "Arrêt".
  L'analyse des valeurs tracées dans l'affichage des courbes peut continuer.

Lorsque vous fermez la fenêtre de mise en service, l'enregistrement prend fin dans l'affichage des courbes et les valeurs enregistrées sont effacées.

Voir aussi

Optimisation préalable V1 (Page 112)

Optimisation fine V1 (Page 114)

Mode de fonctionnement "Mode manuel" V1 (Page %getreference)

5.3.2.2 Optimisation préalable V1

L'optimisation préalable détermine la réponse du processus à un échelon de la valeur de réglage et recherche le point d'inflexion. Les paramètres PID optimisés sont calculés à partir de l'incrémentation maximale et du temps mort du système réglé.
Plus la mesure est stable, plus il sera facile de déterminer des paramètres PID précis. Un bruit de la mesure est acceptable tant que la croissance de la mesure est nettement supérieure au bruit. Les paramètres PID sont sauvegardés avant qu'ils ne soient recalculés.

Condition

- L'instruction "PID_Compact" est appelée dans un OB d'alarme cyclique.
- ManualEnable = FALSE
- PID_Compact se trouve en mode de fonctionnement "Inactif" ou "Mode manuel".
- La valeur de consigne ne doit pas être modifiée pendant l'optimisation. Sinon, PID_Compact est désactivé.
- La consigne et la mesure se trouvent dans les limites configurées (voir configuration "Surveillance de la mesure").
- La différence entre la consigne et la mesure représente plus de 30 % de la différence entre la limite supérieure et la limite inférieure de la mesure.
- L'écart entre la consigne et la mesure est > 50% de la consigne.

Marche à suivre

Pour réaliser l'"optimisation préalable", procédez de la manière suivante :

1. Dans la navigation de projet, double-cliquez sur l'entrée "PID_Compact > Mise en service".
2. Dans la liste déroulante "Type d'optimisation", sélectionnez l'entrée "Optimisation préalable".
3. Cliquez sur l'icône "Start".
   - Une liaison en ligne est établie.
   - L'enregistrement des valeurs démarre.
   - L'optimisation préalable est lancée.
   - Les étapes actuelles et éventuelles erreurs s'affichent dans le champ "Etat". La barre de progression affiche la progression de l'étape actuelle.

REMARQUE

Cliquez sur l'icône "Stop" lorsque la barre de progression a atteint 100% et qu'il faut supposer un blocage de l'optimisation. Vérifiez la configuration de l'objet technologique et redémarrez éventuellement l'optimisation.

Résultat

Si l'optimisation préalable a été réalisée sans message d'erreur, les paramètres PID ont été optimisés. PID_Compact passe en mode automatique et utilise les paramètres optimisés. Les paramètres PID optimisés sont conservés lors d'une mise hors tension et d'un redémarrage de la CPU.

Si l'optimisation préalable n'est pas possible, PID_Compact passe en mode de fonctionnement "Inactif".

Voir aussi

Paramètres State et sRet.i_Mode V1 (Page 289)

Mise en service V1 (Page 111)

Optimisation fine V1 (Page 114)

Mode de fonctionnement "Mode manuel" V1 (Page %getreference)

5.3.2.3 Optimisation fine V1

L'optimisation fine génère une oscillation constante limitée de la mesure. Les paramètres PID sont optimisés, pour le point de fonctionnement, à partir de l'amplitude et de la fréquence de cette oscillation. Tous les paramètres PID sont recalculés à partir des résultats. Les paramètres PID de l'optimisation fine montrent généralement un meilleur comportement de référence et de perturbation que les paramètres PID de l'optimisation préalable.

PID_Compact essaie automatiquement de créer une oscillation supérieure au bruit de la mesure. La stabilité de la mesure n'influence l'optimisation fine que de manière insignifiante. Les paramètres PID sont sauvegardés avant qu'ils ne soient recalculés.

Condition

- L'instruction PID_Compact est appelée dans un OB d'alarme cyclique.
- ManualEnable = FALSE
- La consigne et la mesure se trouvent dans les limites configurées (voir configuration "Surveillance de la mesure").
- La boucle de régulation est en régime stationnaire au point de fonctionnement. Le point de fonctionnement est atteint lorsque la mesure correspond à la consigne.
- Aucune perturbation n'est attendue.
- La valeur de consigne ne doit pas être modifiée pendant l'optimisation.
- PID_Compact se trouve en mode de fonctionnement Inactif, Mode automatique ou Mode manuel.

Déroulement dépendant de la situation de départ

Vous pouvez démarrez l'optimisation fine à partir des modes de fonctionnement "Inactif", "Mode automatique" ou "Mode manuel". L'optimisation fine se déroule de la manière suivante au démarrage :

- Mode automatique
  Si vous souhaitez améliorer les paramètres PID existants à l'aide de l'optimisation, démarrez l'optimisation fine à partir du mode automatique.
  PID_Comact effectue un réglage avec les paramètres PID existants jusqu'à ce que la boucle de régulation soit en régime stationnaire et que les conditions pour une optimisation fine soient remplies. C'est seulement après cela que l'optimisation fine commence.
- Inactif ou mode manuel
  Une optimisation préalable est lancée lorsque les conditions correspondantes sont réunies. Un réglage a lieu avec les paramètres PID déterminés jusqu'à ce que la boucle de régulation soit en régime stationnaire et que les conditions pour une optimisation fine soient remplies. C'est seulement après cela que l'optimisation fine commence. Si l'optimisation préalable n'est pas possible, PID_Compact passe en mode de fonctionnement "Inactif".
  Quand la mesure est déjà trop proche de la consigne pour une optimisation préalable, le système essaie d'atteindre la consigne avec la valeur de réglage mini ou maxi. Cela peut entraîner une suroscillation élevée.

Marche à suivre

Pour réaliser l'"optimisation fine", procédez de la manière suivante :

1. Dans la liste déroulante "Type d'optimisation", sélectionnez l'entrée "Optimisation fine".
2. Cliquez sur l'icône "Start".
   - Une liaison en ligne est établie.
   - L'enregistrement des valeurs démarre.
   - Le déroulement de l'optimisation fine démarre.
   - Les étapes actuelles et éventuelles erreurs s'affichent dans le champ "Etat". La barre de progression affiche la progression de l'étape actuelle.

REMARQUE

Cliquez sur l'icône "Stop" dans le groupe "Type d'optimisation" lorsque la barre de progression a atteint 100% et qu'il faut supposer un blocage de l'optimisation. Vérifiez la configuration de l'objet technologique et redémarrez éventuellement l'optimisation.

Résultat

Si l'optimisation fine a été réalisée sans message d'erreur, les paramètres PID ont été optimisés. PID_Compact passe en mode automatique et utilise les paramètres optimisés. Les paramètres PID optimisés sont conservés lors d'une mise hors tension et d'un redémarrage de la CPU.

Si des erreurs sont apparues au cours de l'"optimisation fine", PID_Compact passe en mode de fonctionnement "Inactif".

Voir aussi

Paramètres State et sRet.i_Mode V1 (Page 289)

Mise en service V1 (Page 111)

Optimisation préalable V1 (Page 112)

Mode de fonctionnement "Mode manuel" V1 (Page %getreference)

5.3.2.4 Mode de fonctionnement "Mode manuel" V1

Ce paragraphe décrit comment utiliser le mode de fonctionnement "Mode manuel" dans la fenêtre de mise en service de l'objet technologique "PID Compact".

Condition

- L'instruction "PID_Compact" est appelée dans un OB d'alarme cyclique.
- Une connexion en ligne est établie avec la CPU et celle-ci se trouve à l'état de fonctionnement "RUN".
- Les fonctions de la fenêtre de mise en service sont validées par l'icône "Mesure Marche".

Marche à suivre

Utilisez "Mode Manuel" dans la fenêtre de mise en service quand vous souhaitez tester le système réglé en spécifiant une valeur manuelle, Pour spécifier une valeur manuelle, procédez comme suit :

1. Cochez la case "Mode manuel" dans la zone "État en ligne du régulateur".
   PID_Compact travaille en mode manuel. La dernière valeur de réglage actuelle reste active.
2. Dans le champ "Output", écrivez la valeur de réglage souhaitée dans l'unité %.
3. Cliquez sur l'icône de commande  .

Résultat

La valeur manuelle est écrite dans la CPU et elle est opérante immédiatement.

REMARQUE

PID_Compact continue à surveiller la mesure. Si les limites de la mesure se trouvent dépassées, PID_Compact est désactivé.

Retirez la coche de la case "Mode manuel" pour que la valeur de réglage soit à nouveau spécifiée par le régulateur PID. Le passage au mode automatique s'effectue sans à-coups.

Voir aussi

Paramètres State et sRet.i_Mode V1 (Page 289)
Mise en service V1 (Page 111)
Optimisation préalable V1 (Page 112)
Optimisation fine V1 (Page 114)

5.3.3 Simuler PID_Compact V1 avec PLCSIM

REMARQUE

Simulation avec PLCSIM

Lors de la simulation avec PLCSIM, le comportement dans le temps de l'API simulé n'est pas exactement le même que celui d'un API "réel". Le temps de cycle effectif d'un OB d'alarme cyclique peut présenter de plus grandes fluctuations pour un API simulé que pour un API "réel".

Dans la configuration standard, PID_Compact calcule automatiquement le temps entre les appels et le surveille vis-à-vis des fluctuations.

Lors de la simulation de PID_Compact avec PLCSIM, il est par conséquent possible de détecter une erreur de période d'échantillonnage (ErrorBits = DW#16#00000800).

PID_Compact passe dans ce cas au mode de fonctionnement Inactif (State = 0).

Pour éviter cela, vous devez configurer PID_Compact de la manière suivante lors de la simulation avec PLCSIM :

- sb_EnCyclEstimation = FALSE
- sb_EnCyclMonitoring = FALSE
- sPid_Calc.r_Cycle : affectez à cette variable le temps de cycle de l'OB d'alarme cyclique d'appel en secondes.

Utiliser PID_3Step

6

6.1 Objet technologique PID_3Step

L'objet technologique PID_3Step met à disposition un régulateur PID avec optimisation pour vannes ou actionneurs au comportement intégral.

Vous pouvez configurer les régulateurs suivants :

- Régulateur pas à pas à trois échelons avec signalisation de position
- Régulateur pas à pas à trois échelons sans signalisation de position
- Régulateur de vanne avec valeur de réglage analogique

Dans une boucle de régulation, PID_3Step réalise l'acquisition continue de la mesure et la compare à la consigne. PID_3Step calcule, à partir du signal d'écart en résultant, une valeur de réglage permettant à la mesure d'atteindre la consigne de la façon la plus rapide et la plus stable possible. Pour le régulateur PID, la valeur de réglage se compose de trois actions :

- Action **P**
  L'action P de la valeur de réglage augmente proportionnellement au signal d'écart.
- Action **I**
  L'action I de la valeur de réglage augmente jusqu'à ce que le signal d'écart soit compensé.
- Action **D**
  L'action D augmente avec la vitesse de modification du signal d'écart. La mesure est ajustée à la consigne le plus rapidement possible. Quand la vitesse de modification du signal d'écart ralentit, l'action D diminue également.

L'instruction PID_3Step calcule les paramètres P, I et D du système réglé de manière autonome pendant l'optimisation préalable. Une optimisation supplémentaire des paramètres peut être réalisée par une optimisation fine. Vous n'avez pas besoin de déterminer les paramètres manuellement.

Pour plus d'informations

- Présentation des régulateurs de logiciel (Page 43)
- Ajouter des objets technologiques (Page 45)
- Configurer les objets technologiques (Page 46)
- Configurer PID_3Step V2 (Page 119)
- Configurer PID_3Step V1 (Page 137)

Principe

Pour plus d'informations à ce sujet, voir la FAQ suivante dans l'assistance en ligne de Siemens Industry :

- ID de contribution 68011827 (https://support.industry.siemens.com/cs/ww/en/view/68011827)

6.2 PID_3Step V2

6.2.1 Configurer PID_3Step V2

6.2.1.1 Paramètres de base V2

Configurez les propriétés suivantes de l'objet technologique "PID_3Step" dans la fenêtre d'inspection ou dans les "Paramètres de base" de la fenêtre de configuration.

- Grandeur physique
- Sens de régulation
- Comportement au démarrage après un Reset
- Consigne (seulement dans la fenêtre d'inspection)
- Mesure (seulement dans la fenêtre d'inspection)
- Valeur de réglage (seulement dans la fenêtre d'inspection)
- Signalisation de position (seulement dans la fenêtre d'inspection)

Consigne, mesure, valeur de réglage et signalisation de position

Vous ne pouvez configurer la consigne, la mesure, la valeur de réglage et la signalisation de position que dans la fenêtre d'inspection de l'éditeur de programmation. Sélectionnez la source pour chaque valeur :

- DB d'instance
  La valeur utilisée est celle qui est enregistrée dans le DB d'instance.
  La valeur doit être actualisée dans le DB d'instance par le programme utilisateur.
  L'instruction ne doit pas mentionner de valeur.
  Une modification via IHM est possible.
- Instruction
  La valeur utilisée est celle qui est interconnectée à l'instruction.
  La valeur est écrite dans le DB d'instance à chaque appel de l'instruction.
  Une modification via IHM n'est pas possible.

Grandeur physique

Sélectionnez la grandeur physique et l'unité pour la valeur de consigne et la mesure dans le groupe "Type de régulation". La valeur de consigne et la mesure s'afficheront dans cette unité.

Sens de régulation

La plupart du temps, une augmentation de la mesure doit être atteinte avec une augmentation de la valeur de réglage. Dans ce cas, on parle d'un sens de régulation normal.
PID_3Step ne fonctionne pas avec un gain proportionnel négatif. Pour réduire la mesure au moyen d'une valeur de réglage plus élevée, cochez la case "Inversion du sens de régulation".
Exemples

- L'ouverture d'une vanne d'écoulement fait baisser le niveau de remplissage d'un réservoir.
- En raison d'une plus grande performance de refroidissement, la température baisse.

Comportement au démarrage

1. Pour passer en mode de fonctionnement "Inactif" après un redémarrage de la CPU, décochez la case "Activer mode après redémarrage de CPU".
   Pour passer dans le mode de fonctionnement enregistré dans Mode après un redémarrage de la CPU, cochez la case "Activer mode après redémarrage de CPU".
2. Dans la liste déroulante "Mettre le mode à", sélectionnez le mode de fonctionnement qui doit être activé après un chargement complet dans l'appareil.
   Après un chargement complet dans l'appareil, PID_3Step démarre dans le mode de fonctionnement choisi. Lors de chaque redémarrage ultérieur, PID_3Step démarre dans le mode de fonctionnement qui a été enregistré en dernier dans Mode.

Exemple

Vous avez coché la case "Activer le mode après un redémarrage de la CPU" et choisi l'entrée "Optimisation préalable" dans la liste "Régler le mode à". Après un chargement complet dans l'appareil, PID_3Step démarre dans le mode de fonctionnement "Optimisation préalable". Si l'optimisation préalable est toujours activée, PID_3Step démarre à nouveau dans le mode de fonctionnement "Optimisation préalable" après le redémarrage de la CPU. Si l'optimisation préalable a été effectuée avec succès et que le mode automatique est activé, PID_3Step démarre en "mode automatique" après le redémarrage de la CPU.

Marche à suivre

Pour spécifier une consigne fixe, procédez de la manière suivante :

1. Sélectionnez "DB d'instance".
2. Entrez une consigne, par exemple 80 °C.
3. Supprimez éventuellement une entrée au niveau de l'instruction.

Pour spécifier une consigne variable, procédez de la manière suivante :

1. Sélectionnez "Instruction".
2. Entrez le nom de la variable REAL dans laquelle la consigne est enregistrée.
   Vous pouvez attribuer des valeurs différentes à la variable REAL dans le programme, par ex. pour une modification de la consigne déclenchée par horloge.

Si vous utilisez directement la valeur de l'entrée analogique, PID_3Step met la valeur de l'entrée analogique à l'échelle dans la grandeur physique.

Si vous souhaitez d'abord mettre en forme la valeur de l'entrée analogique, vous devez écrire un programme propre pour la mise en forme. Par exemple, la mesure n'est pas directement proportionnelle à la valeur de l'entrée analogique. La mesure mise en forme doit être disponible au format à virgule flottante.

Marche à suivre

Pour utiliser directement la valeur de l'entrée analogique, procédez comme suit :

1. Dans la liste déroulante "Input", sélectionnez l'entrée "Input_PER".
2. Sélectionnez "Instruction" comme source.
3. Entrez l'adresse de l'entrée analogique.

Pour utiliser la mesure mise au format à virgule flottante, procédez de la manière suivante :

1. Dans la liste déroulante "Input", sélectionnez l'entrée "Input".
2. Sélectionnez "Instruction" comme source.
3. Entrez le nom de la variable dans laquelle la mesure mise en forme est enregistrée.

La configuration de la signalisation de position dépend de l'actionneur utilisé.

- Actionneur sans signalisation de position
- Actionneur avec signaux de butée TOR
- Actionneur avec signalisation de position analogique
- Actionneur avec signalisation de position analogique et signaux de butée

Actionneur sans signalisation de position

Pour configurer PID_3Step pour un actionneur sans signalisation de position, procédez comme suit :

1. Dans la liste déroulante "Feedback", sélectionnez l'entrée "Pas de Feedback".

Actionneur avec signaux de butée TOR

Pour configurer PID_3Step pour un actionneur avec signaux de butée, procédez comme suit :

1. Dans la liste déroulante "Feedback", sélectionnez l'entrée "Pas de Feedback".
2. Cochez la case "Signaux de butée actionneur".
3. Sélectionnez "Instruction" comme source pour Actuator_H et Actuator_L.
4. Saisissez les adresses des entrées TOR pour Actuator_H et Actuator_L.

Actionneur avec signalisation de position analogique

Pour configurer PID_3Step pour un actionneur avec signalisation de position analogique, procédez comme suit :

1. Dans la liste déroulante "Feedback", sélectionnez l'entrée "Feedback" ou "Feedback_PER".
   - Avec Feedback_PER, vous utilisez directement la valeur de l'entrée analogique. Vous configurez la mise à l'échelle de Feedback_PER dans les paramètres de l'actionneur.
   - Avec Feedback, vous mettez en forme la valeur de l'entrée analogique avec votre programme utilisateur.
2. Sélectionnez "Instruction" comme source.
3. Entrez l'adresse de l'entrée analogique ou la variable de votre programme utilisateur.

Actionneur avec signalisation de position analogique et signaux de butée

Pour configurer PID_3Step pour un actionneur avec signalisation de position analogique et signaux de butée, procédez comme suit :

1. Dans la liste déroulante "Feedback", sélectionnez l'entrée "Feedback" ou "Feedback_PER".
2. Sélectionnez "Instruction" comme source.
3. Entrez l'adresse de l'entrée analogique ou la variable de votre programme utilisateur.
4. Cochez la case "Signaux de butée actionneur".
5. Sélectionnez "Instruction" comme source pour Actuator_H et Actuator_L.
6. Saisissez les adresses des entrées TOR pour Actuator_H et Actuator_L.

PID_3Step met à disposition une valeur de réglage analogique (Output_PER) et des valeurs de réglage TOR (Output_UP, Output_DN). La valeur de réglage que vous utilisez dépend de votre actionneur.

- Output_PER
  L'actionneur dispose d'un temps de positionnement du moteur pertinent et est adressé via une sortie analogique et commandé à l'aide d'un signal continu, par exemple 0...10 V, 4...20 mA. La valeur dans Output_PER correspond à la position finale de la vanne, p. ex. Output_PER = 13824, lorsque la vanne doit s'ouvrir à 50 %.
  PID_3Step tient compte, par ex. pour l'auto-optimisation et le comportement anti-windup, du fait que la valeur de réglage analogique agit sur le processus avec un retard dû au temps de positionnement du moteur. Si aucun temps de positionnement du moteur pertinent n'est actif (par ex. dans le cas d'électrovannes), si bien que la valeur de réglage est appliquée directement et entièrement au processus, utilisez à la place PID_Compact.
- Output_UP, Output_DN
  L'actionneur dispose d'un temps de positionnement du moteur pertinent et est commandé à l'aide de deux sorties TOR.
  Output_UP manœuvre la vanne en sens ouvert.
  Output_DN manœuvre la vanne en sens fermé.

Le temps de positionnement du moteur est pris en compte dans le calcul de la valeur de réglage analogique ainsi que dans le calcul des valeurs de réglage TOR. Il est surtout nécessaire pour un fonctionnement correct lors de l'auto-optimisation et du comportement anti-windup. Configurez par conséquent le temps de positionnement du moteur, sous "Paramètres de l'actionneur", à la valeur nécessaire au moteur pour faire passer l'actionneur de l'état fermé à l'état ouvert.

Marche à suivre

Pour utiliser la valeur de réglage analogique, procédez de la manière suivante :

1. Dans la liste déroulante "Output", sélectionnez l'entrée "Output (analogique)".
2. Sélectionnez "Instruction".
3. Entrez l'adresse de la sortie analogique.

Pour utiliser la valeur de réglage TOR, procédez de la manière suivante :

1. Dans la liste déroulante "Output", sélectionnez l'entrée "Output (TOR)".
2. Sélectionnez "Instruction" pour Output_UP et Output_DN.
3. Entrez les adresses des sorties TOR.

Pour mettre en forme la valeur de réglage dans le programme utilisateur, procédez de la manière suivante :

1. Dans la liste déroulante "Output", sélectionnez l'entrée adaptée à l'actionneur.
2. Sélectionnez "Instruction".
3. Indiquez le nom de la variable que vous utilisez pour la mise en forme de la valeur de réglage.
4. Transférez la valeur de réglage mise en forme à l'actionneur via une sortie analogique ou TOR de la CPU.

6.2.1.2 Paramètres de la mesure V2

Si vous avez configuré l'utilisation de Input_PER dans les paramètres de base, vous devez convertir la valeur de l'entrée analogique dans la grandeur physique de la mesure. La configuration actuelle est affichée dans le champ d'affichage Input_PER.

Si la mesure est directement proportionnelle à la valeur de l'entrée analogique, Input_PER est mis à l'échelle à l'aide d'une paire de valeurs supérieure et inférieure.

Marche à suivre

Pour mettre à l'échelle la mesure, procédez comme suit :

1. Indiquez la paire de valeurs inférieure dans les champs de saisie "Mesure inférieure à l'échelle" et "Bas".
2. Indiquez la paire de valeurs supérieure dans les champs de saisie "Mesure supérieure à l'échelle" et "Haut".

Des valeurs par défaut pour les paires de valeurs sont enregistrées dans la configuration matérielle. Pour utiliser les paires de valeurs de la configuration matérielle, procédez comme suit :

1. Sélectionnez l'instruction PID_3Step dans l'éditeur de programmation.
2. Dans les paramètres de base, reliez Input_PER à une entrée analogique.
3. Dans les paramètres de la mesure, cliquez sur le bouton "Paramétrage automatique".

Les valeurs existantes sont écrasées par les valeurs de la configuration matérielle.

Vous devez définir une limite absolue supérieure et inférieure significative de la mesure comme valeurs limites pour votre système réglé. Dès que ces limites sont dépassées, une erreur survient (ErrorBits = 0001h). L'optimisation est interrompue si les limites de la mesure sont dépassées. Définissez la réaction de PID_3Step en cas d'erreur en mode automatique dans les paramètres de l'actionneur.

6.2.1.3 Paramètres d'actionneur V2

Durées spécifiques à l'actionneur

Afin de protéger l'actionneur de tout endommagement, vous configurez le temps de positionnement du moteur, le plus petit temps ON et le plus petit temps OFF. Vous trouverez ces données dans la fiche technique de l'actionneur.

Le temps de positionnement du moteur est le temps en secondes requis par le moteur pour faire passer l'actionneur de l'état fermé à l'état ouvert. Le temps de positionnement du moteur peut être défini pendant la mise en service.

Le temps de positionnement du moteur est pris en compte dans le calcul de la valeur de réglage analogique ainsi que dans le calcul des valeurs de réglage TOR. Il est surtout nécessaire pour un fonctionnement correct lors de l'auto-optimisation et du comportement anti-windup.

Si aucun temps de positionnement du moteur pertinent n'est actif (par ex. dans le cas d'électrovannes), si bien que la valeur de réglage est appliquée directement et entièrement au processus, utilisez à la place PID_Compact.

Le temps de positionnement du moteur est rémanent. Si vous modifiez manuellement le temps de positionnement du moteur, vous devez charger entièrement PID_3Step.

Charger des objets technologiques dans l'appareil (Page 48)

Si vous utilisez Output_UP et Output_DN réduisez la fréquence de commutation avec les plus petits temps ON et OFF.

En mode automatique, les temps ON ou OFF calculés sont cumulés et n'ont d'effet que quand la somme est supérieure ou égale au plus petit temps ON ou OFF.

En mode manuel, l'actionneur est commandé, au moins pour le plus petit temps ON ou OFF, par Manual_UP = TRUE ou Manual_DN = TRUE.

Si vous avez choisi la valeur analogique de sortie Output_PER dans les paramètres de base, le plus petit temps ON et le plus petit temps OFF ne sont pas évalués et ne peuvent pas être modifiés.

Comportement en cas d'erreur

PID_3Step est préréglé de telle façon qu'en cas d'erreur, la régulation reste active dans la plupart des cas. Lorsque des erreurs apparaissent fréquemment en mode régulation, cette valeur par défaut détériore le comportement de régulation. Vérifiez alors le paramètre Errorbits et éliminez la cause d'erreur.

IMPORTANT
Votre installation peut être endommagée. En cas d'erreur, si vous fournissez "Valeur actuelle pour la durée de l'erreur" ou "Valeur de réglage de remplacement pour la durée de l'erreur", PID_3Step reste en mode automatique aussi en cas de dépassement des limites de la mesure. Cela peut endommager votre installation. Configurez un comportement en cas d'erreur pour votre système réglé, qui protège votre installation de tout endommagement.

En cas d'erreur, PID_3Step fournit une valeur de réglage configurable :

- Valeur actuelle
  PID_3Step est désactivé et ne modifie plus la position de l'actionneur.
- Valeur actuelle pour la durée de l'erreur
  Les fonctions de régulateur de PID_3Step sont désactivées et la position de l'actionneur n'est plus modifiée.
  Si les erreurs suivantes sont apparues en mode automatique, PID_3Step revient en mode automatique dès que les erreurs ont disparu.
  - 0002h : Valeur invalide au paramètre Input_PER.
  - 0200h : Valeur invalide au paramètre Input.
  - 0400h : Le calcul de la valeur de réglage a échoué.
  - 1000h : Valeur invalide au paramètre Setpoint.
  - 2000h : Valeur invalide au paramètre Feedback_PER.
  - 4000h : Valeur invalide au paramètre Feedback.
  - 8000h : Erreur dans la signalisation de position TOR.
  - 20000h : Valeur invalide à la variable SavePosition.

  Si l'une ou plusieurs des erreurs suivantes apparaissent, PID_3Step reste en mode automatique :
  - 0001h : Le paramètre Input est en dehors des limites de la mesure.
  - 0800h : Erreur de temps d'échantillonnage
  - 40000h : Valeur invalide au paramètre Disturbance.

Si une erreur survient en mode manuel, PID_3Step reste en mode manuel.
Si une erreur survient pendant l'optimisation ou la mesure du temps de positionnement, PID_3Step passe au mode de fonctionnement dans lequel l'optimisation ou la mesure du temps de positionnement a été démarrée. L'optimisation n'est pas interrompue uniquement pour les erreurs suivantes :
– 0020h : L'optimisation préalable n'est pas autorisée pendant l'optimisation fine.

- Valeur de réglage de remplacement
  PID_3Step met l'actionneur sur la valeur de réglage de remplacement et s'éteint.
- Valeur de réglage de remplacement pour la durée de l'erreur
  PID_3Step met l'actionneur sur la valeur de réglage de remplacement. Une fois la valeur de réglage de remplacement atteinte, PID_3Step se comporte comme décrit au point "Valeur actuelle pour la durée de l'erreur".

Vous entrez la valeur de réglage de remplacement en "%".
Pour l'utilisation de Output_UP et Output_DN sans signalisation en retour de position (Config.OutputPerOn = FALSE et Config.FeedbackOn = FALSE), seules les valeurs de réglage de remplacement 0 % et 100 % peuvent être configurées. Afin que la butée supérieure ou inférieure soit atteinte, l'actionneur est déplacé dans le sens correspondant. Lorsque des signaux de butée sont présents (Config.ActuatorEndStopOn = TRUE), Output_UP et Output_DN sont remis à 0 pour Actuator_H = TRUE ou Actuator_L = TRUE. Si aucun signal de butée n'est disponible (Config.ActuatorEndStopOn = FALSE), Output_UP et Output_DN sont remis à 0 après un temps de course de Config.VirtualActuatorLimit * Retain.TransitTime/100.
Pour l'utilisation de Output_PER ou d'une signalisation en retour de position (Config.OutputPerOn = TRUE ou Config.FeedbackOn = TRUE), toutes les valeurs de réglage de remplacement souhaitées peuvent être configurées dans les limites de la valeur de réglage.

Mise à l'échelle de la signalisation de position

Si vous avez configuré l'utilisation de Feedback_PER dans les paramètres de base, vous devez convertir la valeur de l'entrée analogique en %. La configuration actuelle est affichée dans le champ d'affichage "Feedback".
Feedback_PER est mis à l'échelle avec une paire de valeurs supérieure et inférieure.

1. Indiquez la paire de valeurs inférieure dans les champs de saisie "Butée inférieure" et "Bas".
2. Indiquez la paire de valeurs supérieure dans les champs de saisie "Butée supérieure" et "Haut".

"Butée inférieure" doit être plus petite que "Butée supérieure" ; "Bas" doit être plus petit que "Haut".
Les valeurs valables de la "Butée supérieure" et de la "Butée inférieure" dépendent de :

- Pas de Feedback, Feedback, Feedback_PER
- Output (analogique), Output (TOR)

Output	Feedback	Butée inférieure	Butée supérieure
Output (TOR)	Pas de Feedback	non réglable (0.0 %)	non réglable (100.0 %)
Output (TOR)	Feedback	-100.0 % ou 0.0 %	0.0 % ou +100.0 %
Output (TOR)	Feedback_PER	-100.0 % ou 0.0 %	0.0 % ou +100.0 %
Output (analogique)	Pas de Feedback	non réglable (0.0 %)	non réglable (100.0 %)
Output (analogique)	Feedback	-100.0 % ou 0.0 %	0.0 % ou +100.0 %
Output (analogique)	Feedback_PER	-100.0 % ou 0.0 %	0.0 % ou +100.0 %

Limiter la valeur de réglage

Pendant la mesure du temps de positionnement et avec mode = 10, les limites de valeur de réglage peuvent être dépassées par le haut ou par le bas. Dans tous les autres modes de fonctionnement, la valeur de réglage est limitée à ces valeurs.

Dans les champs de saisie "Limite supérieure de la valeur de réglage" et "Limite inférieure de la valeur de réglage", entrez les limites absolues de valeur de réglage. Les limites de la valeur de réglage doivent se trouver entre la "butée inférieure" et la "butée supérieure".

En l'absence de Feedback et si Output (TOR) est paramétré, vous ne pouvez pas limiter la valeur de réglage. Output_UP et Output_DN sont alors remis à 0 si Actuator_H = TRUE ou Actuator_L = TRUE. Si aucun signal de butée n'est disponible, Output_UP et Output_DN sont remis à 0 après un temps de course de 150 %.

La valeur par défaut 150 % peut être adaptée via la variable Config.VirtualActuatorLimit. À partir de PID_3Step version 2.3, il est possible de désactiver la surveillance et la limitation du temps de course avec Config.VirtualActuatorLimit = 0.0.

6.2.1.4 Paramètres avancés V2

Dans la fenêtre de configuration "Surveillance de la mesure", configurez une limite d'alerte inférieure et une limite d'alerte supérieure de la mesure. Si l'une de ces limites d'alerte est dépassée ou n'est pas atteinte pendant le fonctionnement, l'instruction PID_3Step affiche un avertissement :

- Dans le paramètre de sortie InputWarning_H, lorsque la limite d'alerte supérieure a été dépassée
- Dans le paramètre de sortie InputWarning_L, lorsque la limite d'alerte inférieure n'est pas atteinte

Les limites d'alerte doivent se situer entre la limite supérieure et la limite inférieure de la mesure.

Si vous n'indiquez pas de valeur, les limites supérieure et inférieure de la mesure sont utilisées.

Exemple

Limite supérieure de la mesure = 98 °C ; limite d'alerte supérieure = 90 °C
Limite d'alerte inférieure = 10 °C ; limite inférieure de la mesure = 0 °C
PID_3Step se comporte comme suit :

Mesure	InputWarning_H	InputWarning_L	ErrorBits	Mode de fonctionnement
> 98 °C	TRUE	FALSE	0001h	Comme configuré
≤ 98 °C et > 90 °C	TRUE	FALSE	0000h	Mode automatique
≤ 90 °C et ≥ 10 °C	FALSE	FALSE	0000h	Mode automatique
≤ 10℃ et ≥ 0 ℃	FALSE	TRUE	0000h	Mode automatique
< 0 °C	FALSE	TRUE	0001h	Comme configuré

Configurez la réaction de PID_3Step en cas de dépassement de la limite supérieure ou inférieure de la mesure dans les paramètres de l'actionneur.

Les paramètres PID sont affichés dans la fenêtre de configuration "Paramètres PID". Les paramètres PID sont adaptés à votre système réglé pendant l'optimisation. Vous n'avez pas besoin d'indiquer les paramètres PID manuellement.

REMARQUE

Les paramètres PID actuellement effectifs se trouvent dans la structure Retain.CtrlParams.

Pour éviter un comportement erroné du régulateur PID, ne modifiez les paramètres PID actuellement effectifs en ligne que dans le mode de fonctionnement "Inactif".

Si vous souhaitez modifier en ligne les paramètres PID dans les modes de fonctionnement "Mode automatique" ou "Mode manuel", modifiez les paramètres PID dans la structure CtrlParamsBackUp et appliquez ces modifications dans la structure Retain.CtrlParams comme suit :

- PID_3Step V1 : Appliquez les modifications via Config.LoadBackUp = TRUE
- PID_3Step V2 : Appliquez les modifications via LoadBackUp = TRUE

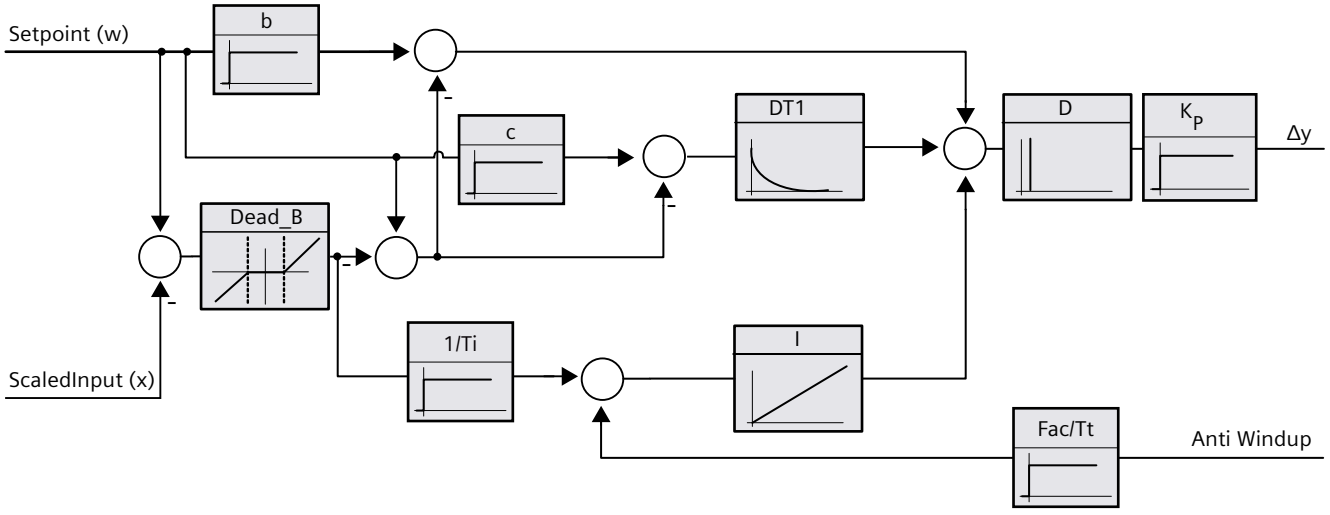
Apporter des modifications en ligne aux paramètres PID dans le mode de fonctionnement "Mode automatique" peut provoquer des sauts de la valeur de réglage.

L'algorithme PID fonctionne selon la formule suivante :

$$\Delta y = K_p \cdot s \cdot \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

Δy	Valeur de réglage de l'algorithme PID
K_p	Gain proportionnel
s	Opérateur de Laplace
b	Pondération de l'action P
w	Consigne
x	Mesure
T_I	Temps d'intégration
a	Coefficient de l'action par dérivation (action par dérivation T1 = a × T_D)
T_D	Temps de dérivation
c	Pondération de l'action D

Le graphique suivant illustre l'intégration des paramètres dans l'algorithme PID.



Tous les paramètres PID sont rémanents. Si vous saisissez les paramètres PID manuellement, vous devez charger entièrement PID_3Step.
Charger des objets technologiques dans l'appareil (Page 48)

Gain proportionnel

La valeur indique le gain proportionnel du régulateur. PID_3Step ne fonctionne pas avec un gain proportionnel négatif. Inversez le sens de régulation dans Réglages de base > Type de régulation.

Temps d'intégration

Le temps d'intégration détermine le temps de réponse de l'action I. La désactivation de l'action I s'obtient avec un temps d'intégration = 0.0. Si le temps d'intégration est modifié en ligne en mode de fonctionnement "Mode automatique" en passant d'une autre valeur à 0.0, l'action I actuelle est supprimée et il se produit un saut de la valeur de réglage.

Temps de dérivation

Le temps de dérivation détermine le temps de réponse de l'action D. La désactivation de l'action D s'obtient avec un temps de dérivation = 0.0.

Coefficient de l'action par dérivation

L'effet de l'action D est retardé par le coefficient de l'action par dérivation.
Action par dérivation = Temps de dérivation x Coefficient de l'action par dérivation

- 0.0 : L'action D n'est active que pour un seul cycle et est donc quasiment inactive.
- 0.5 : Cette valeur a fait ses preuves dans la pratique pour les systèmes réglés avec **une** constante de temps dominante.
- > 1.0 : Plus le coefficient est grand, plus l'effet de l'action D est retardé.

Pondération de l'action P

En cas de modification de consigne, vous pouvez réduire l'action P.
Les valeurs comprises entre 0.0 et 1.0 sont judicieuses.

- 1.0 : Action P totalement opérante si modification de la consigne
- 0.0 : Action P non opérante si modification de la consigne

En cas de variation de la mesure, l'action P est toujours totalement opérante.

Pondération de l'action D

En cas de modification de consigne, vous pouvez réduire l'action D.
Les valeurs comprises entre 0.0 et 1.0 sont judicieuses.

- 1.0 : En cas de modification de la consigne, l'action D est totalement opérante
- 0.0 : En cas de modification de la consigne, l'action D n'est pas opérante

En cas de variation de la mesure, l'action D est toujours totalement opérante.

Période d'échantillonnage de l'algorithme PID

Comme le système réglé nécessite un certain temps pour réagir à une modification de la valeur de réglage, il est judicieux de ne pas calculer cette valeur à chaque cycle. Le temps d'échantillonnage de l'algorithme PID est le temps entre deux calculs de valeurs de réglage. Il est déterminé pendant l'optimisation et arrondi à un multiple de la période d'échantillonnage PID_3Step. Toutes les autres fonctions de PID_3Step sont exécutées lors de chaque appel.

Largeur de zone morte

La zone morte réduit le taux de bruit lorsque le régulateur est à l'état stationnaire. La largeur de la zone morte indique la taille de la zone morte. Lorsque la largeur de la zone morte est 0.0, la zone morte est désactivée.
Si des valeurs différentes de 1.0 sont configurées pour la pondération de l'action P ou la pondération de l'action D, des variations de la consigne se répercutent également dans la zone morte sur la valeur de sortie.
Des variations de la valeur réelle dans la zone morte ne se répercutent pas sur la valeur de sortie, et ce, indépendamment des pondérations.

6.2.2 Mise en service de PID_3Step V2

6.2.2.1 Optimisation préalable V2

L'optimisation préalable détermine la réponse du processus à une impulsion de la valeur de réglage et recherche le point d'inflexion. Les paramètres PID optimisés sont calculés à partir de l'incrémentation maximale et du temps mort du système réglé. Les meilleurs paramètres PID sont obtenus pendant l'exécution d'une optimisation préalable et d'une optimisation fine.
Plus la mesure est stable, plus il sera facile de déterminer des paramètres PID précis. Un bruit de la mesure est acceptable tant que la croissance de la mesure est nettement supérieure au bruit. Cela est plutôt le cas en mode de fonctionnement "Inactif" ou "Mode manuel". Les paramètres PID sont sauvegardés avant qu'ils ne soient recalculés.
La consigne est gelée pendant l'optimisation préalable.

Condition

- L'instruction PID_3Step est appelée dans un OB d'alarme cyclique.
- ManualEnable = FALSE
- Reset = FALSE
- Le temps de positionnement du moteur est configuré ou mesuré.
- PID_3Step se trouve en mode de fonctionnement "Inactif", "Mode manuel" ou "Mode automatique".
- La consigne et la mesure se trouvent dans les limites configurées (voir configuration "Paramètres de la mesure").

Marche à suivre

Pour réaliser l'optimisation préalable, procédez de la manière suivante :

1. Dans la navigation de projet, double-cliquez sur l'entrée "PID_3Step > Mise en service".
2. Dans la zone de travail "Optimisation", dans la liste déroulante "Type d'optimisation", sélectionnez l'entrée "Optimisation préalable".
3. Cliquez sur l'icône "Start".
   - Une liaison en ligne est établie.
   - L'enregistrement des valeurs démarre.
   - L'optimisation préalable est lancée.
   - Les étapes actuelles et éventuelles erreurs s'affichent dans le champ "Etat". La barre de progression affiche la progression de l'étape actuelle.

REMARQUE

Cliquez sur l'icône "Stop" lorsque la barre de progression a atteint 100 % et qu'il faut supposer un blocage de l'optimisation. Vérifiez la configuration de l'objet technologique et redémarrez éventuellement l'optimisation.

Résultat

Si l'optimisation préalable a été réalisée sans message d'erreur, les paramètres PID ont été optimisés. PID_3Step passe en mode automatique et utilise les paramètres optimisés. Les paramètres PID optimisés sont conservés lors d'une mise hors tension et d'un redémarrage de la CPU.
Si une optimisation préalable n'est pas possible, PID_3Step se comporte comme cela a été défini sous Comportement en cas d'erreur.

6.2.2.2 Optimisation fine V2

L'optimisation fine génère une oscillation constante limitée de la mesure. Les paramètres PID sont optimisés, pour le point de fonctionnement, à partir de l'amplitude et de la fréquence de cette oscillation. Tous les paramètres PID sont recalculés à partir des résultats. Les paramètres PID de l'optimisation fine montrent généralement un meilleur comportement de référence et de perturbation que les paramètres PID de l'optimisation préalable. Les meilleurs paramètres PID sont obtenus pendant l'exécution d'une optimisation préalable et d'une optimisation fine.
PID_3Step essaie automatiquement de créer une oscillation supérieure au bruit de la mesure. La stabilité de la mesure n'influence l'optimisation fine que de manière insignifiante. Les paramètres PID sont sauvegardés avant qu'ils ne soient recalculés.
La consigne est gelée pendant l'optimisation fine.

Condition

- L'instruction PID_3Step est appelée dans un OB d'alarme cyclique.
- ManualEnable = FALSE
- Reset = FALSE
- Le temps de positionnement du moteur est configuré ou mesuré.
- La consigne et la mesure se trouvent dans les limites configurées (voir configuration "Paramètres de la mesure").
- La boucle de régulation est en régime stationnaire au point de fonctionnement. Le point de fonctionnement est atteint lorsque la mesure correspond à la consigne.
- Aucune perturbation n'est attendue.
- PID_3Step se trouve en mode de fonctionnement Inactif, Mode automatique ou Mode manuel.

Déroulement dépendant de la situation de départ

L'optimisation fine se déroule de la manière suivante au démarrage :

- Mode automatique
  Si vous souhaitez améliorer les paramètres PID existants à l'aide de l'optimisation, démarrez l'optimisation fine à partir du mode automatique.
  PID_3Step effectue un réglage avec les paramètres PID existants jusqu'à ce que la boucle de régulation soit en régime stationnaire et que les conditions pour une optimisation fine soient remplies. C'est seulement après cela que l'optimisation fine commence.
- Inactif ou mode manuel
  Ue optimisation préalable est toujours lancée en premier. Une régulation est effectuée avec les paramètres PID déterminés jusqu'à ce que la boucle de régulation soit en régime établi et que les conditions pour une optimisation fine soient remplies. C'est seulement après cela que l'optimisation fine commence.

Marche à suivre

Pour réaliser l'"optimisation fine", procédez de la manière suivante :

1. Dans la liste déroulante "Type d'optimisation", sélectionnez l'entrée "Optimisation fine".
2. Cliquez sur l'icône "Start".
   – Une liaison en ligne est établie.
   – L'enregistrement des valeurs démarre.
   – Le déroulement de l'optimisation fine démarre.
   – Les étapes actuelles et éventuelles erreurs s'affichent dans le champ "Etat". La barre de progression affiche la progression de l'étape actuelle.

REMARQUE

Cliquez sur l'icône "Stop" dans le groupe "Type d'optimisation" lorsque la barre de progression a atteint 100% et qu'il faut supposer un blocage de l'optimisation. Vérifiez la configuration de l'objet technologique et redémarrez éventuellement l'optimisation.

Résultat

Si aucune erreur n'est apparue pendant l'optimisation fine, les paramètres PID ont été optimisés. PID_3Step passe en mode automatique et utilise les paramètres optimisés. Les paramètres PID optimisés sont conservés lors d'une mise hors tension et d'un redémarrage de la CPU.

Si des erreurs sont apparues au cours de l'optimisation fine, PID_3Step se comporte comme cela a été défini sous Comportement en cas d'erreur.

6.2.2.3 Mise en service avec des paramètres PID manuels V2

Condition

- L'instruction PID_3Step est appelée dans un OB d'alarme cyclique.
- ManualEnable = FALSE
- Reset = FALSE
- Le temps de positionnement du moteur est configuré ou mesuré.
- PID_3Step se trouve en mode de fonctionnement "Inactif".
- La consigne et la mesure se trouvent dans les limites configurées (voir configuration "Paramètres de la mesure").

Marche à suivre

Pour mettre PID_3Step en service avec des paramètres PID manuels, procédez comme suit :

1. Dans la navigation de projet, double-cliquez sur l'entrée "PID_3Step > Configuration".
2. Dans la fenêtre de configuration, cliquez sur "Paramètres avancés > Paramètres PID".
3. Cochez la case "Activer la saisie manuelle".
4. Saisissez les paramètres PID.
5. Dans la navigation de projet, double-cliquez sur l'entrée "PID_3Step > Mise en service".
6. Etablissez une liaison en ligne avec la CPU.
7. Chargez les paramètres PID dans la CPU.
8. Cliquez sur l'icône "Start PID_3Step".

Résultat

PID_3Step passe en mode automatique et utilise les paramètres PID actuels pour la régulation.

Voir aussi

Paramètres PID V2 (Page 127)

6.2.2.4 Mesurer le temps de positionnement du moteur V2

Introduction

PID_3Step a besoin d'un temps de positionnement du moteur aussi exact que possible pour obtenir un bon résultat de régulation. La documentation de l'actionneur indique des valeurs moyennes pour ce type d'actionneur. L'actionneur utilisé réellement peut avoir une valeur différente.

Lorsque vous utilisez des actionneurs avec signalisation de position ou avec signaux de butée, vous pouvez mesurer le temps de positionnement du moteur pendant la mise en service. Les limites de valeur de réglage ne sont pas prises en compte lors de la mesure du temps de positionnement du moteur. Il est possible de déplacer l'actionneur jusqu'à la butée supérieure ou inférieure.

En l'absence de signalisation de position ou de signaux de butée, il n'est pas possible de mesurer le temps de positionnement du moteur.

Actionneurs avec signalisation de position analogique

Pour mesurer le temps de positionnement du moteur avec signalisation de position, procédez comme suit :

Condition

- Feedback ou Feedback_PER est sélectionné dans les paramètres de base et le signal est connecté.
- Une liaison en ligne avec la CPU est établie.

1. Cochez la case "Utiliser la signalisation de position".
2. Dans le champ de saisie "Position cible", entrez l'endroit où l'actionneur doit être amené.
   La signalisation de position actuelle (position de départ) s'affiche. La différence entre la "Position cible" et la "Signalisation de position" doit être au moins égale à 50 % de la plage de valeurs de réglage autorisée.
3. Cliquez sur l'icône "Start".

Résultat

L'actionneur est déplacé de la position de départ à la position cible. La mesure de temps est immédiatement lancée et se termine lorsque l'actionneur a atteint la position cible. Le temps de positionnement du moteur est calculé selon la formule :
Temps de positionnement du moteur = (limite supérieure valeur de réglage – limite inférieure valeur de réglage) x temps de mesure / VALEUR (position cible – position de départ).
La progression et l'état de la mesure du temps de positionnement sont affichés. Le temps de positionnement mesuré est enregistré dans le bloc de données d'instance sur la CPU et affiché dans le champ "Temps de positionnement mesuré". Lorsque la mesure du temps de positionnement est terminée et si ActivateRecoverMode = TRUE, PID_3Step passe dans le mode de fonctionnement à partir duquel la mesure a été lancée. Lorsque la mesure du temps de positionnement est terminée et si ActivateRecoverMode = FALSE, PID_3Step passe en mode de fonctionnement "Inactif".

REMARQUE

Pour intégrer le temps de positionnement du moteur dans le projet, cliquez sur l'icône "Charger le temps de positionnement mesuré".

Actionneurs avec signaux de butée

Pour mesurer le temps de positionnement d'actionneurs avec signaux de butée, procédez comme suit :

Condition

- La case "Signaux de butée" est activée dans les paramètres de base et Actuator_H et Actuator_L sont connectés.
- Une liaison en ligne avec la CPU est établie.

Pour mesurer le temps de positionnement du moteur avec signaux de butée, procédez comme suit :

1. Cochez la case "Utiliser les signaux de butée de l'actionneur".
2. Sélectionnez le sens dans lequel l'actionneur doit être déplacé.
   - Ouverture - fermeture - ouverture
     L'actionneur est d'abord déplacé jusqu'à la butée supérieure, puis à la butée inférieure et de nouveau à la butée supérieure.
   - Fermeture - ouverture - fermeture
     L'actionneur est d'abord déplacé jusqu'à la butée inférieure, puis à la butée supérieure et de nouveau à la butée inférieure.
3. Cliquez sur l'icône "Start".

Résultat

L'actionneur est déplacé dans le sens sélectionné. La mesure du temps démarre lorsque l'actionneur a atteint la première butée et se termine lorsque l'actionneur atteint cette butée pour la deuxième fois. Le temps mesuré divisé par deux donne le temps de positionnement du moteur.
La progression et l'état de la mesure du temps de positionnement sont affichés. Le temps de positionnement mesuré est enregistré dans le bloc de données d'instance sur la CPU et affiché dans le champ "Temps de positionnement mesuré". Lorsque la mesure du temps de positionnement est terminée et si ActivateRecoverMode = TRUE, PID_3Step passe dans le mode de fonctionnement à partir duquel la mesure a été lancée. Lorsque la mesure du temps de positionnement est terminée et si ActivateRecoverMode = FALSE, PID_3Step passe en mode de fonctionnement "Inactif".

Annuler la mesure du temps de positionnement

Si vous annulez la mesure du temps de positionnement avec le bouton Stop, PID_3Step passe en mode de fonctionnement "Inactif".

6.2.3 Simuler PID_3Step V2 avec PLCSIM

REMARQUE
Simulation avec PLCSIM

La simulation de PID_3Step V2.x avec PLCSIM n'est pas prise en charge pour la CPU S7-1200.

PID_3Step V2.x peut être simulé avec PLCSIM uniquement pour la CPU S7-1500.

Lors de la simulation avec PLCSIM, le comportement dans le temps de l'API simulé n'est pas exactement le même que celui d'un API "réel". Le temps de cycle effectif d'un OB d'alarme cyclique peut présenter de plus grandes fluctuations pour un API simulé que pour un API "réel".

Dans la configuration standard, PID_3Step calcule automatiquement le temps entre les appels et le surveille vis-à-vis des fluctuations.

Lors de la simulation de PID_3Step avec PLCSIM, il est par conséquent possible de détecter une erreur de période d'échantillonnage (ErrorBits = DW#16#00000800).

Cela entraîne l'interruption d'optimisations en cours.

La réaction en mode automatique dépend de la valeur de la variable ActivateRecoverMode.

Pour éviter cela, vous devez configurer PID_3Step de la manière suivante lors de la simulation avec PLCSIM :

- CycleTime.EnEstimation = FALSE
- CycleTime.EnMonitoring = FALSE
- CycleTime.Value : affectez à cette variable le temps de cycle de l'OB d'alarme cyclique d'appel en secondes.

6.3 PID_3Step V1

6.3.1 Configurer PID_3Step V1

6.3.1.1 Paramètres de base V1

Configurez les propriétés suivantes de l'objet technologique "PID_3Step" dans la fenêtre d'inspection ou dans les "Paramètres de base" de la fenêtre de configuration.

- Grandeur physique
- Sens de régulation
- Comportement au démarrage après un Reset
- Consigne (seulement dans la fenêtre d'inspection)
- Mesure (seulement dans la fenêtre d'inspection)
- Valeur de réglage (seulement dans la fenêtre d'inspection)
- Signalisation de position (seulement dans la fenêtre d'inspection)

Consigne, mesure, valeur de réglage et signalisation de position

Vous ne pouvez configurer la consigne, la mesure, la valeur de réglage et la signalisation de position que dans la fenêtre d'inspection de l'éditeur de programmation. Sélectionnez la source pour chaque valeur :

- DB d'instance
  La valeur utilisée est celle qui est enregistrée dans le DB d'instance.
  La valeur doit être actualisée dans le DB d'instance par le programme utilisateur.
  L'instruction ne doit pas mentionner de valeur.
  Une modification via IHM est possible.
- Instruction
  La valeur utilisée est celle qui est interconnectée à l'instruction.
  La valeur est écrite dans le DB d'instance à chaque appel de l'instruction.
  Une modification via IHM n'est pas possible.

Grandeur physique

Sélectionnez la grandeur physique et l'unité pour la consigne et la mesure dans le groupe "Type de régulation". La consigne et la mesure sont affichées dans cette unité.

Sens de régulation

La plupart du temps, une augmentation de la mesure doit être atteinte avec une augmentation de la valeur de réglage. Dans ce cas, on parle d'un sens de régulation normal.
PID_3Step ne fonctionne pas avec un gain proportionnel négatif. Pour réduire la mesure au moyen d'une valeur de réglage plus élevée, cochez la case "Inversion du sens de régulation".
Exemples

- L'ouverture d'une vanne d'écoulement fait baisser le niveau de remplissage d'un réservoir.
- En raison d'une plus grande performance de refroidissement, la température baisse.

Comportement au démarrage après un Reset

En cas de redémarrage de la CPU, pour passer directement au dernier mode de fonctionnement actif, cochez l'option "Activer le dernier mode de fonctionnement après un redémarrage de la CPU".
Si la case n'est pas cochée, PID_3Step reste dans le mode de fonctionnement "Inactif".

Marche à suivre

Pour spécifier une consigne fixe, procédez de la manière suivante :

1. Sélectionnez "DB d'instance".
2. Entrez une consigne, par exemple 80 °C.
3. Supprimez éventuellement une entrée au niveau de l'instruction.

Pour spécifier une consigne variable, procédez de la manière suivante :

1. Sélectionnez "Instruction".
2. Entrez le nom de la variable REAL dans laquelle la consigne est enregistrée.
   Vous pouvez attribuer des valeurs différentes à la variable REAL dans le programme, par ex. pour une modification de la consigne déclenchée par horloge.

Si vous utilisez directement la valeur de l'entrée analogique, PID_3Step met la valeur de l'entrée analogique à l'échelle dans la grandeur physique.
Si vous souhaitez d'abord mettre en forme la valeur de l'entrée analogique, vous devez écrire un programme propre pour la mise en forme. Par exemple, la mesure n'est pas directement proportionnelle à la valeur de l'entrée analogique. La mesure mise en forme doit être disponible au format à virgule flottante.

Marche à suivre

Pour utiliser directement la valeur de l'entrée analogique, procédez comme suit :

1. Dans la liste déroulante "Input", sélectionnez l'entrée "Input_PER".
2. Sélectionnez "Instruction" comme source.
3. Entrez l'adresse de l'entrée analogique.

Pour utiliser la mesure mise au format à virgule flottante, procédez de la manière suivante :

1. Dans la liste déroulante "Input", sélectionnez l'entrée "Input".
2. Sélectionnez "Instruction" comme source.
3. Entrez le nom de la variable dans laquelle la mesure mise en forme est enregistrée.

La configuration de la signalisation de position dépend de l'actionneur utilisé.

- Actionneur sans signalisation de position
- Actionneur avec signaux de butée TOR
- Actionneur avec signalisation de position analogique
- Actionneur avec signalisation de position analogique et signaux de butée

Actionneur sans signalisation de position

Pour configurer PID_3Step pour un actionneur sans signalisation de position, procédez comme suit :

1. Dans la liste déroulante "Feedback", sélectionnez l'entrée "Pas de Feedback".

Actionneur avec signaux de butée TOR

Pour configurer PID_3Step pour un actionneur avec signaux de butée, procédez comme suit :

1. Dans la liste déroulante "Feedback", sélectionnez l'entrée "Pas de Feedback".
2. Cochez la case "Signaux de butée actionneur".
3. Sélectionnez "Instruction" comme source pour Actuator_H et Actuator_L.
4. Saisissez les adresses des entrées TOR pour Actuator_H et Actuator_L.

Actionneur avec signalisation de position analogique

Pour configurer PID_3Step pour un actionneur avec signalisation de position analogique, procédez comme suit :

1. Dans la liste déroulante "Feedback", sélectionnez l'entrée "Feedback" ou "Feedback_PER".
   - Avec Feedback_PER, vous utilisez directement la valeur de l'entrée analogique. Vous configurez la mise à l'échelle de Feedback_PER dans les paramètres de l'actionneur.
   - Avec Feedback, vous mettez en forme la valeur de l'entrée analogique avec votre programme utilisateur.
2. Sélectionnez "Instruction" comme source.
3. Entrez l'adresse de l'entrée analogique ou la variable de votre programme utilisateur.

Actionneur avec signalisation de position analogique et signaux de butée

Pour configurer PID_3Step pour un actionneur avec signalisation de position analogique et signaux de butée, procédez comme suit :

1. Dans la liste déroulante "Feedback", sélectionnez l'entrée "Feedback" ou "Feedback_PER".
2. Sélectionnez "Instruction" comme source.
3. Entrez l'adresse de l'entrée analogique ou la variable de votre programme utilisateur.
4. Cochez la case "Signaux de butée actionneur".
5. Sélectionnez "Instruction" comme source pour Actuator_H et Actuator_L.
6. Saisissez les adresses des entrées TOR pour Actuator_H et Actuator_L.

PID_3Step met à disposition une valeur de réglage analogique (Output_PER) et des valeurs de réglage TOR (Output_UP, Output_DN). La valeur de réglage que vous utilisez dépend de votre actionneur.

- Output_PER
  L'actionneur dispose d'un temps de positionnement du moteur pertinent et est adressé via une sortie analogique et commandé à l'aide d'un signal continu, par exemple 0...10 V, 4...20 mA. La valeur dans Output_PER correspond à la position finale de la vanne, p. ex. Output_PER = 13824, lorsque la vanne doit s'ouvrir à 50 %.
  PID_3Step tient compte, par ex. pour l'auto-optimisation et le comportement anti-windup, du fait que la valeur de réglage analogique agit sur le processus avec un retard dû au temps de positionnement du moteur. Si aucun temps de positionnement du moteur pertinent n'est actif (par ex. dans le cas d'électrovannes), si bien que la valeur de réglage est appliquée directement et entièrement au processus, utilisez à la place PID_Compact.

- Output_UP, Output_DN
  L'actionneur dispose d'un temps de positionnement du moteur pertinent et est commandé à l'aide de deux sorties TOR.
  Output_UP manœuvre la vanne en sens ouvert.
  Output_DN manœuvre la vanne en sens fermé.

Le temps de positionnement du moteur est pris en compte dans le calcul de la valeur de réglage analogique ainsi que dans le calcul des valeurs de réglage TOR. Il est surtout nécessaire pour un fonctionnement correct lors de l'auto-optimisation et du comportement anti-windup. Configurez par conséquent le temps de positionnement du moteur, sous "Paramètres de l'actionneur", à la valeur nécessaire au moteur pour faire passer l'actionneur de l'état fermé à l'état ouvert.

Marche à suivre

Pour utiliser la valeur de réglage analogique, procédez de la manière suivante :

1. Dans la liste déroulante "Output", sélectionnez l'entrée "Output (analogique)".
2. Sélectionnez "Instruction".
3. Entrez l'adresse de la sortie analogique.

Pour utiliser la valeur de réglage TOR, procédez de la manière suivante :

1. Dans la liste déroulante "Output", sélectionnez l'entrée "Output (TOR)".
2. Sélectionnez "Instruction" pour Output_UP et Output_DN.
3. Entrez les adresses des sorties TOR.

Pour mettre en forme la valeur de réglage dans le programme utilisateur, procédez de la manière suivante :

1. Dans la liste déroulante "Output", sélectionnez l'entrée adaptée à l'actionneur.
2. Sélectionnez "Instruction".
3. Indiquez le nom de la variable que vous utilisez pour la mise en forme de la valeur de réglage.
4. Transférez la valeur de réglage mise en forme à l'actionneur via une sortie analogique ou TOR de la CPU.

6.3.1.2 Paramétrage de la mesure V1

Configurez la normalisation de votre mesure dans la fenêtre de configuration "Paramètres de la mesure" et déterminez les limites absolues de la mesure.

Mise à l'échelle de la mesure

Si vous avez configuré l'utilisation de Input_PER dans les paramètres de base, vous devez convertir la valeur de l'entrée analogique dans la grandeur physique de la mesure. La configuration actuelle est affichée dans le champ d'affichage Input_PER.

Si la mesure est directement proportionnelle à la valeur de l'entrée analogique, Input_PER est mis à l'échelle à l'aide d'une paire de valeurs supérieure et inférieure.

1. Indiquez la paire de valeurs inférieure dans les champs de saisie "Mesure inférieure à l'échelle" et "Bas".
2. Indiquez la paire de valeurs supérieure dans les champs de saisie "Mesure supérieure à l'échelle" et "Haut".

Des valeurs par défaut pour les paires de valeurs sont enregistrées dans la configuration matérielle. Pour utiliser les paires de valeurs de la configuration matérielle, procédez comme suit :

1. Sélectionnez l'instruction PID_3Step dans l'éditeur de programmation.
2. Dans les paramètres de base, reliez Input_PER à une entrée analogique.
3. Dans les paramètres de la mesure, cliquez sur le bouton "Paramétrage automatique". Les valeurs existantes sont écrasées par les valeurs de la configuration matérielle.

Surveiller la mesure

Fixez les limites supérieure et inférieure absolues de la mesure. Vous devez saisir des valeurs judicieuses comme valeurs limites pour votre système réglé. Pendant l'optimisation, des valeurs limites judicieuses sont importantes pour obtenir des paramètres PID optimaux. La valeur par défaut de la "limite supérieure de la mesure" est de 120 %. A l'entrée de périphérie, la mesure peut dépasser de 18 % au plus la plage normée (dépassement haut). Un dépassement de la "limite supérieure de la mesure" ne provoque plus le signalement d'aucune erreur avec ce réglage. Seuls la rupture de fil et le court-circuit sont détectés et PID_3Step se comporte comme vous en avez décidé sous Comportement en cas d'erreur.

IMPORTANT
Votre installation peut être endommagée. Si vous réglez des valeurs très élevées (par ex. $-3{,}4*10^{38}$...$+3{,}4*10^{38}$) comme limites de la mesure, la surveillance de la mesure sera désactivée. Une erreur peut alors entraîner des dommages sur votre installation. Configurez des limites judicieuses de la mesure pour votre système réglé.

6.3.1.3 Paramétrage de l'actionneur V1

Durées spécifiques à l'actionneur

Afin de protéger l'actionneur de tout endommagement, vous configurez le temps de positionnement du moteur, le plus petit temps ON et le plus petit temps OFF. Vous trouverez ces données dans la fiche technique de l'actionneur.

Le temps de positionnement du moteur est le temps en secondes requis par le moteur pour faire passer l'actionneur de l'état fermé à l'état ouvert. L'actionneur est déplacé dans un sens d'au maximum 110 % du temps de positionnement du moteur. Vous pouvez mesurer le temps de positionnement du moteur pendant la mise en service.

Le temps de positionnement du moteur est pris en compte dans le calcul de la valeur de réglage analogique ainsi que dans le calcul des valeurs de réglage TOR. Il est surtout nécessaire pour un fonctionnement correct lors de l'auto-optimisation et du comportement anti-windup.

Si aucun temps de positionnement du moteur pertinent n'est actif (par ex. dans le cas d'électrovannes), si bien que la valeur de réglage est appliquée directement et entièrement au processus, utilisez à la place PID_Compact.

Si vous utilisez Output_UP et Output_DN réduisez la fréquence de commutation avec les plus petits temps ON et OFF.

En mode automatique, les temps ON ou OFF calculés sont cumulés et n'ont d'effet que quand la somme est supérieure ou égale au plus petit temps ON ou OFF.

En mode manuel, l'actionneur est commandé, au moins pour le plus petit temps ON ou OFF, par un front montant à Manual_UP ou Manual_DN.

Si vous avez choisi la valeur analogique de réglage Output_PER dans les paramètres de base, le plus petit temps ON et le plus petit temps OFF ne sont pas évalués et ne peuvent pas être modifiés.

Comportement en cas d'erreur

PID_3Step est préréglé de telle façon qu'en cas d'erreur, la régulation reste active dans la plupart des cas. Lorsque des erreurs apparaissent fréquemment en mode régulation, cette valeur par défaut détériore le comportement de régulation. Vérifiez alors le paramètre Errorbits et éliminez la cause d'erreur.

En cas d'erreur, PID_3Step fournit une valeur de réglage configurable :

- Valeur actuelle
  PID_3Step est désactivé et ne modifie plus la position de l'actionneur.
- Valeur actuelle pour la durée de l'erreur
  Les fonctions de régulateur de PID_3Step sont désactivées et la position de l'actionneur n'est plus modifiée.
  Si les erreurs suivantes sont apparues en mode automatique, PID_3Step revient en mode automatique dès que les erreurs ont disparu.
  - 0002h : Valeur invalide au paramètre Input_PER.
  - 0200h : Valeur invalide au paramètre Input.
  - 0800h : Erreur de temps d'échantillonnage
  - 1000h : Valeur invalide au paramètre Setpoint.
  - 2000h : Valeur invalide au paramètre Feedback_PER.
  - 4000h : Valeur invalide au paramètre Feedback.
  - 8000h : Erreur dans la signalisation de position TOR.

  Lorsqu'une de ces erreurs apparaît en mode manuel, PID_3Step reste en mode manuel.
  PID_3Step est désactivé si une erreur apparaît pendant l'optimisation ou la mesure du temps de positionnement.
- Valeur de réglage de remplacement
  PID_3Step met l'actionneur sur la valeur de réglage de remplacement et s'éteint.
- Valeur de réglage de remplacement pour la durée de l'erreur
  PID_3Step met l'actionneur sur la valeur de réglage de remplacement. Une fois la valeur de réglage de remplacement atteinte, PID_3Step se comporte comme décrit au point "Valeur actuelle pour la durée de l'erreur".

Vous entrez la valeur de réglage de remplacement en "%".

Pour l'utilisation de Output_UP et Output_DN sans signalisation en retour de position (Config.OutputPerOn = FALSE et Config.FeedbackOn = FALSE), seules les valeurs de réglage de remplacement 0 % et 100 % peuvent être configurées. Afin que la butée supérieure ou inférieure soit atteinte, l'actionneur est déplacé dans un sens à hauteur de 110 % du temps de positionnement du moteur. Les signaux de butée sont pris en compte en priorité.

Pour l'utilisation de Output_PER ou d'une signalisation en retour de position (Config.OutputPerOn = TRUE ou Config.FeedbackOn = TRUE), toutes les valeurs de réglage de remplacement souhaitées peuvent être configurées dans les limites de la valeur de réglage.

Mise à l'échelle de la signalisation de position

Si vous avez configuré l'utilisation de Feedback_PER dans les paramètres de base, vous devez convertir la valeur de l'entrée analogique en %. La configuration actuelle est affichée dans le champ d'affichage "Feedback".

Feedback_PER est mis à l'échelle avec une paire de valeurs supérieure et inférieure.

1. Indiquez la paire de valeurs inférieure dans les champs de saisie "Butée inférieure" et "Bas".
2. Indiquez la paire de valeurs supérieure dans les champs de saisie "Butée supérieure" et "Haut".

"Butée inférieure" doit être plus petite que "Butée supérieure" ; "Bas" doit être plus petit que "Haut".

Les valeurs valables de la "Butée supérieure" et de la "Butée inférieure" dépendent de :

- Pas de Feedback, Feedback, Feedback_PER
- Output (analogique), Output (TOR)

Output	Feedback	Butée inférieure	Butée supérieure
Output (TOR)	Pas de Feedback	non réglable (0,0 %)	non réglable (100,0 %)
Output (TOR)	Feedback	-100.0 % ou 0.0 %	0.0 % ou +100.0 %
Output (TOR)	Feedback_PER	-100.0 % ou 0.0 %	0.0 % ou +100.0 %
Output (analogique)	Pas de Feedback	non réglable (0,0 %)	non réglable (100,0 %)
Output (analogique)	Feedback	-100.0 % ou 0.0 %	0.0 % ou +100.0 %
Output (analogique)	Feedback_PER	-100.0 % ou 0.0 %	0.0 % ou +100.0 %

Limiter la valeur de réglage

Les limites de valeur de réglage ne peuvent être dépassées par le haut ou par le bas que pendant la mesure du temps de positionnement. Dans tous les autres modes de fonctionnement, la valeur de réglage est limitée à ces valeurs.

Dans les champs de saisie "Limite supérieure de la valeur de réglage" et "Limite inférieure de la valeur de réglage", entrez les limites absolues de valeur de réglage. Les limites de la valeur de réglage doivent se trouver entre la "butée inférieure" et la "butée supérieure".

En l'absence de Feedback et si Output (TOR) est paramétré, vous ne pouvez pas limiter la valeur de réglage. Les sorties TOR sont remises à zéro soit avec Actuator_H = TRUE ou Actuator_L = TRUE, soit après un temps de course de 110 % du temps de positionnement du moteur.

6.3.1.4 Paramètres avancés V1

Dans la fenêtre de configuration "Surveillance de la mesure", configurez une limite d'alerte inférieure et une limite d'alerte supérieure de la mesure. Si l'une de ces limites d'alerte est dépassée ou n'est pas atteinte pendant le fonctionnement, l'instruction PID_3Step affiche un avertissement :

- Dans le paramètre de sortie InputWarning_H, lorsque la limite d'alerte supérieure a été dépassée
- Dans le paramètre de sortie InputWarning_L, lorsque la limite d'alerte inférieure n'est pas atteinte

Les limites d'alerte doivent se situer entre la limite supérieure et la limite inférieure de la mesure.
Si vous n'indiquez pas de valeur, les limites supérieure et inférieure de la mesure sont utilisées.

Exemple

Limite supérieure de la mesure = 98 °C ; limite d'alerte supérieure = 90 °C
Limite d'alerte inférieure = 10 °C ; limite inférieure de la mesure = 0 °C
PID_3Step se comporte comme suit :

Mesure	InputWarning_H	InputWarning_L	Mode de fonctionnement
> 98 °C	TRUE	FALSE	Inactif
≤ 98 °C et > 90 °C	TRUE	FALSE	Mode automatique
≤ 90 °C et ≥ 10 °C	FALSE	FALSE	Mode automatique
≤ 10°C et ≥ 0 °C	FALSE	TRUE	Mode automatique
< 0 °C	FALSE	TRUE	Inactif

Les paramètres PID sont affichés dans la fenêtre de configuration "Paramètres PID". Les paramètres PID sont adaptés à votre système réglé pendant l'optimisation. Vous n'avez pas besoin d'indiquer les paramètres PID manuellement.

REMARQUE

Les paramètres PID actuellement effectifs se trouvent dans la structure Retain.CtrlParams.

Pour éviter un comportement erroné du régulateur PID, ne modifiez les paramètres PID actuellement effectifs en ligne que dans le mode de fonctionnement "Inactif".

Si vous souhaitez modifier en ligne les paramètres PID dans les modes de fonctionnement "Mode automatique" ou "Mode manuel", modifiez les paramètres PID dans la structure CtrlParamsBackUp et appliquez ces modifications dans la structure Retain.CtrlParams comme suit :

- PID_3Step V1 : Appliquez les modifications via Config.LoadBackUp = TRUE
- PID_3Step V2 : Appliquez les modifications via LoadBackUp = TRUE

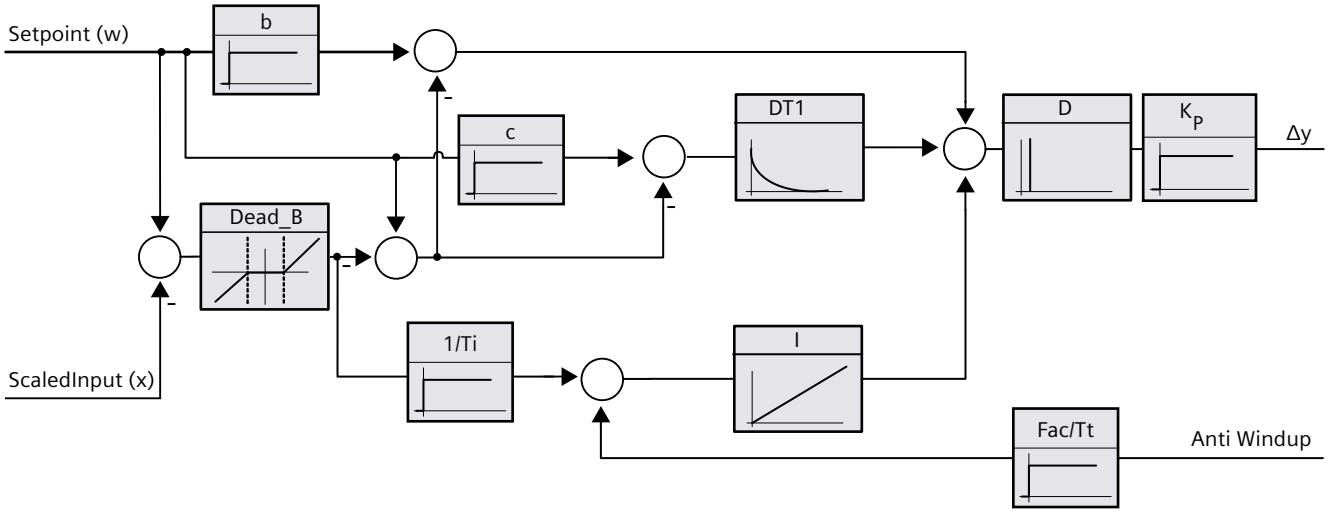
Apporter des modifications en ligne aux paramètres PID dans le mode de fonctionnement "Mode automatique" peut provoquer des sauts de la valeur de réglage.

L'algorithme PID fonctionne selon la formule suivante :

$$\Delta y = K_p \cdot s \cdot \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

Δy	Valeur de réglage de l'algorithme PID
K_p	Gain proportionnel
s	Opérateur de Laplace
b	Pondération de l'action P
w	Consigne
x	Mesure
T_I	Temps d'intégration
a	Coefficient de l'action par dérivation (action par dérivation T1 = a × T_D)
T_D	Temps de dérivation
c	Pondération de l'action D

Le graphique suivant illustre l'intégration des paramètres dans l'algorithme PID.



Tous les paramètres PID sont rémanents. Si vous saisissez les paramètres PID manuellement, vous devez charger entièrement PID_3Step.
Charger des objets technologiques dans l'appareil (Page 48)

Gain proportionnel

La valeur indique le gain proportionnel du régulateur. PID_3Step ne fonctionne pas avec un gain proportionnel négatif. Inversez le sens de régulation dans Réglages de base > Type de régulation.

Temps d'intégration

Le temps d'intégration détermine le temps de réponse de l'action I. La désactivation de l'action I s'obtient avec un temps d'intégration = 0.0. Si le temps d'intégration est modifié en ligne en mode de fonctionnement "Mode automatique" en passant d'une autre valeur à 0.0, l'action I actuelle est supprimée et il se produit un saut de la valeur de réglage.

Temps de dérivation

Le temps de dérivation détermine le temps de réponse de l'action D. La désactivation de l'action D s'obtient avec un temps de dérivation = 0.0.

Coefficient de l'action par dérivation

L'effet de l'action D est retardé par le coefficient de l'action par dérivation.
Action par dérivation = Temps de dérivation x Coefficient de l'action par dérivation

- 0.0 : L'action D n'est active que pour un seul cycle et est donc quasiment inactive.
- 0.5 : Cette valeur a fait ses preuves dans la pratique pour les systèmes réglés avec **une** constante de temps dominante.
- > 1.0 : Plus le coefficient est grand, plus l'effet de l'action D est retardé.

Pondération de l'action P

En cas de modification de consigne, vous pouvez réduire l'action P.
Les valeurs comprises entre 0.0 et 1.0 sont judicieuses.

- 1.0 : Action P totalement opérante si modification de la consigne
- 0.0 : Action P non opérante si modification de la consigne

En cas de variation de la mesure, l'action P est toujours totalement opérante.

Pondération de l'action D

En cas de modification de consigne, vous pouvez réduire l'action D.
Les valeurs comprises entre 0.0 et 1.0 sont judicieuses.

- 1.0 : En cas de modification de la consigne, l'action D est totalement opérante
- 0.0 : En cas de modification de la consigne, l'action D n'est pas opérante

En cas de variation de la mesure, l'action D est toujours totalement opérante.

Période d'échantillonnage de l'algorithme PID

Comme le système réglé nécessite un certain temps pour réagir à une modification de la valeur de réglage, il est judicieux de ne pas calculer cette valeur à chaque cycle. Le temps d'échantillonnage de l'algorithme PID est le temps entre deux calculs de valeurs de réglage. Il est déterminé pendant l'optimisation et arrondi à un multiple de la période d'échantillonnage PID_3Step. Toutes les autres fonctions de PID_3Step sont exécutées lors de chaque appel.

Largeur de zone morte

La zone morte réduit le taux de bruit lorsque le régulateur est à l'état stationnaire. La largeur de la zone morte indique la taille de la zone morte. Lorsque la largeur de la zone morte est 0.0, la zone morte est désactivée.
Si des valeurs différentes de 1.0 sont configurées pour la pondération de l'action P ou la pondération de l'action D, des variations de la consigne se répercutent également dans la zone morte sur la valeur de sortie.
Des variations de la valeur réelle dans la zone morte ne se répercutent pas sur la valeur de sortie, et ce, indépendamment des pondérations.

6.3.2 Mise en service de PID_3Step V1

6.3.2.1 Mise en service V1

Dans la zone de travail "Optimisation", vous pouvez observer la consigne, la mesure et la valeur de réglage en fonction du temps. Les fonctions de mise en service suivantes sont prises en charge dans le traceur de courbes :

- Optimisation préalable du régulateur
- Optimisation fine du régulateur
- Visualisation de la régulation en cours dans la fenêtre des courbes

Une liaison en ligne doit être établie avec la CPU pour toutes les fonctions.

Commande fondamentale

- Sélectionnez le temps d'actualisation souhaité dans la liste déroulante "Temps d'actualisation".
  Toutes les valeurs de la zone de travail Optimisation sont actualisées durant le temps d'actualisation sélectionné.
- Si vous souhaitez utiliser les fonctions de mise en service, cliquez sur l'icône "Démarrage" du groupe Mesure.
  L'enregistrement des valeurs démarre. Les valeurs actuelles pour la consigne, la mesure et la valeur de réglage sont écrites dans l'affichage de courbes. La commande de la fenêtre de mise en service est validée.
- Si vous souhaitez mettre fin aux fonctions de mise en service, cliquez sur l'icône "Arrêt".
  L'analyse des valeurs tracées dans l'affichage des courbes peut continuer.
- Lorsque vous fermez la fenêtre de mise en service, l'enregistrement prend fin dans l'affichage des courbes et les valeurs enregistrées sont effacées.

6.3.2.2 Optimisation préalable V1

L'optimisation préalable détermine la réponse du processus à une impulsion de la valeur de réglage et recherche le point d'inflexion. Les paramètres PID optimisés sont calculés à partir de l'incrémentation maximale et du temps mort du système réglé.
Plus la mesure est stable, plus il sera facile de déterminer des paramètres PID précis. Un bruit de la mesure est acceptable tant que la croissance de la mesure est nettement supérieure au bruit. Les paramètres PID sont sauvegardés avant qu'ils ne soient recalculés.
La consigne est gelée pendant l'optimisation préalable.

Condition

- L'instruction PID_3Step est appelée dans un OB d'alarme cyclique.
- ManualEnable = FALSE
- PID_3Step se trouve en mode de fonctionnement "Inactif" ou "Mode manuel".
- La consigne et la mesure se trouvent dans les limites configurées (voir configuration "Paramètres de la mesure").

Marche à suivre

Pour réaliser l'optimisation préalable, procédez de la manière suivante :

1. Dans la navigation de projet, double-cliquez sur l'entrée "PID_3Step > Mise en service".
2. Dans la zone de travail "Optimisation", dans la liste déroulante "Type d'optimisation", sélectionnez l'entrée "Optimisation préalable".
3. Cliquez sur l'icône "Start".
   - Une liaison en ligne est établie.
   - L'enregistrement des valeurs démarre.
   - L'optimisation préalable est lancée.
   - Les étapes actuelles et éventuelles erreurs s'affichent dans le champ "Etat". La barre de progression affiche la progression de l'étape actuelle.

REMARQUE

Cliquez sur l'icône "Stop" lorsque la barre de progression a atteint 100 % et qu'il faut supposer un blocage de l'optimisation. Vérifiez la configuration de l'objet technologique et redémarrez éventuellement l'optimisation.

Résultat

Si l'optimisation préalable a été réalisée sans message d'erreur, les paramètres PID ont été optimisés. PID_3Step passe en mode automatique et utilise les paramètres optimisés. Les paramètres PID optimisés sont conservés lors d'une mise hors tension et d'un redémarrage de la CPU.
Si l'optimisation préalable n'est pas possible, PID_3Step passe en mode de fonctionnement "Inactif".

6.3.2.3 Optimisation fine V1

L'optimisation fine génère une oscillation constante limitée de la mesure. Les paramètres PID sont optimisés, pour le point de fonctionnement, à partir de l'amplitude et de la fréquence de cette oscillation. Tous les paramètres PID sont recalculés à partir des résultats. Les paramètres PID de l'optimisation fine montrent généralement un meilleur comportement de référence et de perturbation que les paramètres PID de l'optimisation préalable.

PID_3Step essaie automatiquement de créer une oscillation supérieure au bruit de la mesure. La stabilité de la mesure n'influence l'optimisation fine que de manière insignifiante. Les paramètres PID sont sauvegardés avant qu'ils ne soient recalculés.

La consigne est gelée pendant l'optimisation fine.

Condition

- L'instruction PID_3Step est appelée dans un OB d'alarme cyclique.
- ManualEnable = FALSE
- Le temps de positionnement du moteur est configuré ou mesuré.
- La consigne et la mesure se trouvent dans les limites configurées (voir configuration "Paramètres de la mesure").
- La boucle de régulation est en régime stationnaire au point de fonctionnement. Le point de fonctionnement est atteint lorsque la mesure correspond à la consigne.
- Aucune perturbation n'est attendue.
- PID_3Step se trouve en mode de fonctionnement Inactif, Mode automatique ou Mode manuel.

Déroulement dépendant de la situation de départ

L'optimisation fine se déroule de la manière suivante au démarrage :

- Mode automatique
  Si vous souhaitez améliorer les paramètres PID existants à l'aide de l'optimisation, démarrez l'optimisation fine à partir du mode automatique.
  PID_3Step effectue un réglage avec les paramètres PID existants jusqu'à ce que la boucle de régulation soit en régime stationnaire et que les conditions pour une optimisation fine soient remplies. C'est seulement après cela que l'optimisation fine commence.
- Inactif ou mode manuel
  Ue optimisation préalable est toujours lancée en premier. Un réglage a lieu avec les paramètres PID déterminés jusqu'à ce que la boucle de régulation soit en régime stationnaire et que les conditions pour une optimisation fine soient remplies. C'est seulement après cela que l'optimisation fine commence.

Marche à suivre

Pour réaliser l'"optimisation fine", procédez de la manière suivante :

1. Dans la liste déroulante "Type d'optimisation", sélectionnez l'entrée "Optimisation fine".
2. Cliquez sur l'icône "Start".
   - Une liaison en ligne est établie.
   - L'enregistrement des valeurs démarre.
   - Le déroulement de l'optimisation fine démarre.
   - Les étapes actuelles et éventuelles erreurs s'affichent dans le champ "Etat". La barre de progression affiche la progression de l'étape actuelle.

REMARQUE

Cliquez sur l'icône "Stop" dans le groupe "Type d'optimisation" lorsque la barre de progression a atteint 100% et qu'il faut supposer un blocage de l'optimisation. Vérifiez la configuration de l'objet technologique et redémarrez éventuellement l'optimisation.

Résultat

Si l'optimisation fine a été réalisée sans message d'erreur, les paramètres PID ont été optimisés. PID_3Step passe en mode automatique et utilise les paramètres optimisés. Les paramètres PID optimisés sont conservés lors d'une mise hors tension et d'un redémarrage de la CPU.

Si des erreurs sont apparues au cours de l'optimisation fine, PID_3Step passe en mode de fonctionnement "Inactif".

6.3.2.4 Mise en service avec des paramètres PID manuels V1

Marche à suivre

Pour mettre PID_3Step en service avec des paramètres PID manuels, procédez comme suit :

1. Dans la navigation de projet, double-cliquez sur l'entrée "PID_3Step > Configuration".
2. Dans la fenêtre de configuration, cliquez sur "Paramètres avancés > Paramètres PID".
3. Cochez la case "Activer la saisie manuelle".
4. Saisissez les paramètres PID.
5. Dans la navigation de projet, double-cliquez sur l'entrée "PID_3Step > Mise en service".
6. Etablissez une liaison en ligne avec la CPU.
7. Chargez les paramètres PID dans la CPU.
8. Cliquez sur l'icône "Activer le régulateur".

Résultat

PID_3Step passe en mode automatique et utilise les paramètres PID actuels pour la régulation.

6.3.2.5 Mesurer le temps de positionnement du moteur V1

Introduction

PID_3Step a besoin d'un temps de positionnement du moteur aussi exact que possible pour obtenir un bon résultat de régulation. La documentation de l'actionneur indique des valeurs moyennes pour ce type d'actionneur. L'actionneur utilisé réellement peut avoir une valeur différente.

Lorsque vous utilisez des actionneurs avec signalisation de position ou avec signaux de butée, vous pouvez mesurer le temps de positionnement du moteur pendant la mise en service. Les limites de valeur de réglage ne sont pas prises en compte lors de la mesure du temps de positionnement du moteur. Il est possible de déplacer l'actionneur jusqu'à la butée supérieure ou inférieure.

En l'absence de signalisation de position ou de signaux de butée, il n'est pas possible de mesurer le temps de positionnement du moteur.

Actionneurs avec signalisation de position analogique

Pour mesurer le temps de positionnement du moteur avec signalisation de position, procédez comme suit :

Condition

- Feedback ou Feedback_PER est sélectionné dans les paramètres de base et le signal est connecté.
- Une liaison en ligne avec la CPU est établie.

1. Cochez la case "Utiliser la signalisation de position".
2. Dans le champ de saisie "Position cible", entrez l'endroit où l'actionneur doit être amené.
   La signalisation de position actuelle (position de départ) s'affiche. La différence entre la "Position cible" et la "Signalisation de position" doit être au moins égale à 50 % de la plage de valeurs de réglage autorisée.
3. Cliquez sur l'icône "Démarrer la mesure du temps de positionnement".

Résultat

L'actionneur est déplacé de la position de départ à la position cible. La mesure de temps est immédiatement lancée et se termine lorsque l'actionneur a atteint la position cible. Le temps de positionnement du moteur est calculé selon la formule :

Temps de positionnement du moteur = (limite supérieure valeur de réglage – limite inférieure valeur de réglage) x temps de mesure / VALEUR (position cible – position de départ).

La progression et l'état de la mesure du temps de positionnement sont affichés. Le temps de positionnement mesuré est enregistré dans le bloc de données d'instance sur la CPU et affiché dans le champ "Temps de positionnement mesuré". Lorsque la mesure du temps de positionnement est terminée, PID_3Step passe en mode de fonctionnement "Inactif".

REMARQUE

Pour intégrer le temps de positionnement du moteur dans le projet, cliquez sur l'icône "Charger le temps de positionnement mesuré".

Actionneurs avec signaux de butée

Pour mesurer le temps de positionnement d'actionneurs avec signaux de butée, procédez comme suit :

Condition

- La case "Signaux de butée" est activée dans les paramètres de base et Actuator_H et Actuator_L sont connectés.
- Une liaison en ligne avec la CPU est établie.

Pour mesurer le temps de positionnement du moteur avec signaux de butée, procédez comme suit :

1. Cochez la case "Utiliser les signaux de butée de l'actionneur".
2. Sélectionnez le sens dans lequel l'actionneur doit être déplacé.
   - Ouverture - fermeture - ouverture
     L'actionneur est d'abord déplacé jusqu'à la butée supérieure, puis à la butée inférieure et de nouveau à la butée supérieure.
   - Fermeture - ouverture - fermeture
     L'actionneur est d'abord déplacé jusqu'à la butée inférieure, puis à la butée supérieure et de nouveau à la butée inférieure.
3. Cliquez sur l'icône "Démarrer la mesure du temps de positionnement".

Résultat

L'actionneur est déplacé dans le sens sélectionné. La mesure du temps démarre lorsque l'actionneur a atteint la première butée et se termine lorsque l'actionneur atteint cette butée pour la deuxième fois. Le temps mesuré divisé par deux donne le temps de positionnement du moteur.

La progression et l'état de la mesure du temps de positionnement sont affichés. Le temps de positionnement mesuré est enregistré dans le bloc de données d'instance sur la CPU et affiché dans le champ "Temps de positionnement mesuré". Lorsque la mesure du temps de positionnement est terminée, PID_3Step passe en mode de fonctionnement "Inactif".

Annuler la mesure du temps de positionnement

Si vous annulez la mesure du temps de positionnement, PID_3Step passe immédiatement en mode de fonctionnement "Inactif". L'actionneur n'est plus déplacé. Vous pouvez réactiver PID-3Step dans le traceur de courbes.

6.3.3 Simuler PID_3Step V1 avec PLCSIM

REMARQUE

Simulation avec PLCSIM

Lors de la simulation avec PLCSIM, le comportement dans le temps de l'API simulé n'est pas exactement le même que celui d'un API "réel". Le temps de cycle effectif d'un OB d'alarme cyclique peut présenter de plus grandes fluctuations pour un API simulé que pour un API "réel".

Dans la configuration standard, PID_3Step calcule automatiquement le temps entre les appels et le surveille vis-à-vis des fluctuations.

Lors de la simulation de PID_3Step avec PLCSIM, il est par conséquent possible de détecter une erreur de période d'échantillonnage (ErrorBits = DW#16#00000800).

Cela entraîne l'interruption d'optimisations en cours.

La réaction en mode automatique dépend de la valeur de la variable ActivateRecoverMode.

Pour éviter cela, vous devez configurer PID_3Step de la manière suivante lors de la simulation avec PLCSIM :

- CycleTime.EnEstimation = FALSE
- CycleTime.EnMonitoring = FALSE
- CycleTime.Value : affectez à cette variable le temps de cycle de l'OB d'alarme cyclique d'appel en secondes.

Utiliser PID_Temp

7

7.1 Objet technologique PID_Temp

L'objet technologique PID_Temp met à disposition un régulateur PID continu avec optimisation intégrée. PID_Temp est spécialement conçu pour la régulation de la température et est adapté aux applications de chauffage ou de chauffage/refroidissement. Deux sorties sont disponibles à cet effet, une pour le chauffage et une pour le refroidissement. PID_Temp peut également être utilisé pour d'autres tâches de régulation. PID_Temp peut être mis en cascade et peut être utilisé en mode manuel ou automatique.

Dans une boucle de régulation, PID_Temp réalise l'acquisition continue de la mesure et la compare à la consigne paramétrée. A partir des signaux d'écart qui en résultent, l'instruction PID_Temp calcule la valeur de réglage pour le chauffage et/ou le refroidissement grâce à laquelle la mesure est adaptée à la consigne. Pour le régulateur PID, les valeurs de réglage se composent de trois actions :

- Action P
  L'action P de la valeur de réglage augmente proportionnellement au signal d'écart.
- Action I
  L'action I de la valeur de réglage augmente jusqu'à ce que le signal d'écart soit compensé.
- Action D
  L'action D augmente avec la vitesse de modification du signal d'écart. La mesure est ajustée à la consigne le plus rapidement possible. Quand la vitesse de modification du signal d'écart ralentit, l'action D diminue également.

L'instruction PID_Temp calcule les paramètres P, I et D de votre système réglé de manière autonome pendant "l'optimisation préalable". Une optimisation supplémentaire des paramètres peut être réalisée par une "optimisation fine". Vous n'avez pas besoin de déterminer les paramètres manuellement.

Pour les applications de chauffage/refroidissement, il est possible d'utiliser soit un facteur de refroidissement fixe soit deux jeux de paramètres PID.

Pour plus d'informations

- Présentation des régulateurs de logiciel (Page 43)
- Ajouter des objets technologiques (Page 45)
- Configurer les objets technologiques (Page 46)
- Configurer PID_Temp (Page 155)

7.2 Configurer PID_Temp

7.2.1 Paramètres de base

7.2.1.1 Introduction

Configurez les propriétés suivantes de l'objet technologique "PID_Temp" dans la fenêtre d'inspection ou dans les "Paramètres de base" de la fenêtre de configuration.

- Grandeur physique
- Comportement au démarrage après un Reset
- Source et saisie de la consigne (seulement dans la fenêtre d'inspection)
- Sélection de la mesure
- Source et saisie de la mesure (seulement dans la fenêtre d'inspection)
- Sélection de la valeur de réglage pour le chauffage
- Source et saisie de la valeur de réglage pour le chauffage (seulement dans la fenêtre d'inspection)
- Activation et sélection de la valeur de réglage pour le refroidissement
- Source et saisie de la valeur de réglage pour le refroidissement (seulement dans la fenêtre d'inspection)
- Activer PID_Temp comme maître ou esclave d'une cascade
- Nombre d'esclaves
- Sélection du maître (seulement dans la fenêtre d'inspection)

Consigne, mesure, valeur de réglage pour le chauffage et valeur de réglage pour le refroidissement

Pour la consigne, la mesure, la valeur de réglage pour le chauffage et la valeur de réglage pour le refroidissement, vous pouvez sélectionner la source dans la fenêtre d'inspection de l'éditeur de programmation et saisir des valeurs ou des variables.

Sélectionnez la source pour chaque valeur :

- DB d'instance :
  La valeur utilisée est celle qui est enregistrée dans le DB d'instance. La valeur doit être actualisée dans le DB d'instance par le programme utilisateur. L'instruction ne doit pas mentionner de valeur. Une modification via IHM est possible.
- Instruction :
  La valeur utilisée est celle qui est interconnectée à l'instruction. La valeur est écrite dans le DB d'instance à chaque appel de l'instruction. Une modification via IHM n'est pas possible.

7.2.1.2 Type de régulation

Grandeur physique

Sélectionnez la grandeur physique et l'unité pour la consigne et la mesure dans le groupe "Type de régulation". La consigne et la mesure s'afficheront dans cette unité.

Comportement au démarrage

1. Pour passer au mode de fonctionnement "Inactif" après un redémarrage de la CPU, décochez la case "Activer Mode après le redémarrage de la CPU".
   Pour passer au mode de fonctionnement enregistré dans Mode après un redémarrage de la CPU, cochez la case "Activer Mode après le redémarrage de la CPU".
2. Dans la liste déroulante "Mettre mode à", sélectionnez le mode de fonctionnement qui doit être activé après un chargement complet dans l'appareil.
   Après un "Chargement dans l'appareil" complet, PID_Temp démarre dans le mode de fonctionnement choisi. A chaque redémarrage ultérieur, PID_Temp démarre dans le mode de fonctionnement qui a été enregistré en dernier dans Mode.
   Lors du choix de l'optimisation préalable ou de l'optimisation fine, vous devez, en plus, mettre à 1 ou à 0 les variables Heat.EnableTuning et Cool.EnableTuning pour choisir entre l'optimisation du chauffage et l'optimisation du refroidissement.

Exemple :

Vous avez coché la case "Activer Mode après le redémarrage de la CPU" et choisi l'entrée "Optimisation préalable" dans la liste "Mettre Mode à". Après un "chargement dans l'appareil" complet, PID_Temp démarre en mode de fonctionnement "Optimisation préalable". Si l'optimisation préalable est encore active, PID_Temp démarre à nouveau en mode de fonctionnement "Optimisation préalable" après le redémarrage de la CPU (chauffage/refroidissement en fonction des variables Heat.EnableTuning et Cool.EnableCooling). Si l'optimisation préalable a été terminée correctement et que le mode automatique est actif, PID_Temp démarre en "Mode automatique" après le redémarrage de la CPU.

7.2.1.3 Consigne

Marche à suivre

Pour spécifier une consigne fixe, procédez de la manière suivante :

1. Sélectionnez "DB d'instance".
2. Entrez une consigne, par exemple 80°C.
3. Supprimez éventuellement une entrée au niveau de l'instruction.

Pour spécifier une consigne variable, procédez de la manière suivante :

1. Sélectionnez "Instruction".
2. Entrez le nom de la variable REAL dans laquelle la consigne est enregistrée.
   Vous pouvez attribuer des valeurs différentes à la variable REAL dans le programme, par ex. pour une modification de la consigne déclenchée par horloge.

7.2.1.4 Mesure

Si vous utilisez directement la valeur de l'entrée analogique, PID_Temp met la valeur de l'entrée analogique à l'échelle dans la grandeur physique.
Si vous souhaitez d'abord mettre en forme la valeur de l'entrée analogique, vous devez écrire un programme propre pour la mise en forme. Par exemple, la mesure n'est pas directement proportionnelle à la valeur de l'entrée analogique. La mesure mise en forme doit être disponible au format à virgule flottante.

Marche à suivre

Pour utiliser directement la valeur de l'entrée analogique, procédez comme suit :

1. Dans la liste déroulante "Input", sélectionnez l'entrée "Input_PER".
2. Sélectionnez "Instruction" comme source.
3. Entrez l'adresse de l'entrée analogique.

Pour utiliser la mesure mise au format à virgule flottante, procédez de la manière suivante :

1. Dans la liste déroulante "Input", sélectionnez l'entrée "Input".
2. Sélectionnez "Instruction" comme source.
3. Entrez le nom de la variable dans laquelle la mesure mise en forme est enregistrée.

7.2.1.5 Valeur de réglage pour le chauffage et le refroidissement

L'instruction PID_Temp met à disposition un régulateur PID avec optimisation intégrée pour la régulation de température. PID_Temp se prête aux applications de chauffage ou de chauffage/refroidissement.
PID_Temp met les valeurs de réglage suivantes à disposition. La valeur de réglage que vous utilisez dépend de votre actionneur.

- OutputHeat
  Valeur de réglage pour le chauffage (format à virgule flottante) : La valeur de réglage pour le chauffage doit être mise en forme dans le programme utilisateur, par ex. parce que l'actionneur présente un comportement non linéaire.
- OutputHeat_PER
  Valeur de réglage analogique pour le chauffage : L'actionneur pour le chauffage est adressé via une sortie analogique et est commandé à l'aide d'un signal continu, par exemple 0...10 V, 4...20 mA.
- OutputHeat_PWM
  Valeur de réglage modulée en largeur d'impulsion pour le chauffage : L'actionneur pour le chauffage est commandé par une sortie TOR. Des temps d'activation et de désactivation variables sont formés à partir d'une modulation de largeur d'impulsions.
- OutputCool
  Valeur de réglage pour le refroidissement (format à virgule flottante) : La valeur de réglage pour le refroidissement doit être mise en forme dans le programme utilisateur, par ex. parce que l'actionneur présente un comportement non linéaire.

- OutputCool_PER
  Valeur de réglage analogique pour le refroidissement : L'actionneur pour le refroidissement est adressé via une sortie analogique et est commandé à l'aide d'un signal continu, par exemple 0...10 V, 4...20 mA.
- OutputCool_PWM
  Valeur de réglage modulée en largeur d'impulsion pour le refroidissement : L'actionneur pour le refroidissement est commandé par une sortie TOR. Des temps d'activation et de désactivation variables sont formés à partir d'une modulation de largeur d'impulsions.

La sortie de refroidissement n'est disponible que si la case "Activer refroidissement" a été cochée.

- Si la case est décochée, la valeur de réglage de l'algorithme PID (PidOutputSum) est mise à l'échelle et est fournie aux sorties pour le chauffage.
- Si la case est cochée, des valeurs de réglage positives de l'algorithme PID (PidOutputSum) sont mises à l'échelle et fournies aux sorties pour le chauffage. Des valeurs de réglage négatives de l'algorithme PID sont mises à l'échelle et fournies aux sorties pour le refroidissement. Dans les paramètres de sortie, il est possible de choisir entre deux méthodes de calcul de la valeur de réglage.

REMARQUE

Tenez compte des points suivants :

- Les sorties OutputHeat_PWM, OutputHeat_PER, OutputCool_PWM, OutputCool_PER ne sont calculées que si vous les sélectionnez en conséquence dans la liste déroulante.
- La sortie OutputHeat est toujours calculée.
- La sortie OutputCool est calculée si la case pour le refroidissement est cochée.
- La case à cocher "Activer refroidissement" n'est disponible que si le régulateur n'est pas configuré comme maître dans une cascade.

Marche à suivre

Pour utiliser la valeur de réglage analogique, procédez de la manière suivante :

1. Dans la liste déroulante "OutputHeat" ou "OutputCool", sélectionnez l'entrée "OutputHeat_PER" ou "OutputCool_PER".
2. Sélectionnez "Instruction".
3. Entrez l'adresse de la sortie analogique.

Pour utiliser la valeur de réglage modulée en largeur d'impulsion, procédez de la manière suivante :

1. Dans la liste déroulante "OutputHeat" ou "OutputCool", sélectionnez l'entrée "OutputHeat_PWM" ou "OutputCool_PWM".
2. Sélectionnez "Instruction".
3. Entrez l'adresse de la sortie TOR.

Pour mettre en forme la valeur de réglage dans le programme utilisateur, procédez de la manière suivante :

1. Dans la liste déroulante "OutputHeat" ou "OutpuCool", sélectionnez l'entrée "OutputHeat" ou "OutputCool".
2. Sélectionnez "Instruction".
3. Indiquez le nom de la variable que vous utilisez pour la mise en forme de la valeur de réglage.
4. Transférez la valeur de réglage mise en forme à l'actionneur via une sortie analogique ou TOR de la CPU.

7.2.1.6 Cascade

Si une instance de PID_Temp reçoit sa consigne d'un régulateur maître de niveau supérieur et transmet sa valeur de réglage elle-même à un régulateur esclave de niveau inférieur, cette instance de PID_Temp est simultanément un régulateur maître et un régulateur esclave. Pour une instance de PID_Temp de ce type, il faut alors effectuer les deux configurations figurant ci-dessous. C'est par exemple le cas pour l'instance de PID_Temp centrale d'une régulation en cascade avec trois grandeurs de mesure enchaînées et trois instances de PID_Temp.

Configurer un régulateur comme maître dans une cascade

Un régulateur maître spécifie la consigne d'un régulateur esclave avec sa sortie.

Pour utiliser PID_Temp comme maître dans une cascade, vous devez désactiver le refroidissement dans les paramètres de base. Pour configurer cette instance de PID_Temp comme régulateur maître dans une cascade, cochez la case "Le régulateur est le maître". Ce faisant, la sélection de la valeur de réglage pour le chauffage est automatiquement mise sur OutputHeat.

OutputHeat_PWM et OutputHeat_PER ne peuvent pas être utilisées pour un maître dans une cascade.

Indiquez ensuite le nombre de régulateurs esclaves directement asservis qui reçoivent leur consigne de ce régulateur maître.

Si aucune fonction de mise à l'échelle spécifique n'est utilisée lors de l'affectation du paramètre OutputHeat du maître au paramètre Setpoint de l'esclave, il peut être nécessaire d'adapter les limites de la valeur de réglage et la mise à l'échelle de la sortie du maître à la plage de la consigne/mesure de l'esclave. Vous pouvez le faire dans les paramètres de sortie du maître dans la zone "OutputHeat / OutputCool".

Configuration d'un régulateur comme esclave dans une cascade

Un régulateur esclave reçoit sa consigne (paramètre Setpoint) de la sortie de son régulateur maître (paramètre OutputHeat).

Pour configurer cette instance de PID_Temp comme régulateur esclave dans une cascade, cochez la case "Le régulateur est esclave" dans les paramètres de base.

Sélectionnez ensuite, dans la fenêtre d'inspection de l'éditeur de programmation, l'instance de PID_Temp qui doit être utilisée comme régulateur maître pour ce régulateur esclave. La sélection permet d'interconnecter les paramètres Master et Setpoint du régulateur esclave avec le régulateur maître choisi (les interconnexions précédentes à ces paramètres sont écrasées). Cette interconnexion permet les échanges d'information et la transmission de la consigne entre le maître et l'esclave. Si besoin, vous pouvez modifier ultérieurement

l'interconnexion au paramètre Setpoint du régulateur esclave, par ex. pour insérer un filtre supplémentaire. L'interconnexion au paramètre Master ne peut pas être modifiée a posteriori.

La case "Le régulateur est le maître" doit être cochée sur le régulateur maître sélectionné et le nombre d'esclaves doit être correctement configuré. Le régulateur maître doit être appelé avant le régulateur esclave dans le même OB d'alarme cyclique.

Pour plus d'informations

Vous trouverez plus informations sur la programmation, la configuration et la mise en service lors de l'utilisation de PID_Temp dans des régulations en cascade sous Fonction cascade avec PID_Temp (Page 186).

7.2.2 Paramètres de la mesure

7.2.2.1 Limites de la mesure

Vous devez définir des limites absolues supérieure et inférieure judicieuses de la mesure, comme valeurs limites pour votre système réglé. Dès que ces limites sont dépassées par le haut ou par le bas, une erreur apparaît (ErrorBits = 0001h). L'optimisation est abandonnée quand les limites de la mesure sont dépassées. Vous déterminez dans les paramètres de sortie la réaction de PID_Temp en cas d'erreur en mode automatique.

7.2.2.2 Mise à l'échelle de la mesure

Si vous avez configuré l'utilisation d'Input_PER dans les paramètres de base, vous devez convertir la valeur de l'entrée analogique dans la grandeur physique de la mesure. La configuration actuelle s'affiche dans le champ d'affichage Input_PER.

Quand la mesure est directement proportionnelle à la valeur de l'entrée analogique, Input_PER est mis à l'échelle à l'aide de paires de valeurs supérieure et inférieure.

Marche à suivre

Pour mettre la mesure à l'échelle, procédez comme suit :

1. Indiquez la paire de valeurs inférieure dans les champs de saisie "Mesure inférieure à l'échelle" et "Bas".
2. Indiquez la paire de valeurs supérieure dans les champs de saisie "Mesure supérieure à l'échelle" et "Haut".

Des valeurs par défaut pour les paires de valeurs sont enregistrées dans la configuration matérielle. Pour utiliser les paires de valeurs de la configuration matérielle, procédez comme suit :

1. Sélectionnez l'instruction PID_Temp dans l'éditeur de programmation.
2. Dans les paramètres de base, interconnectez Input_PER avec une entrée analogique.
3. Dans les paramètres de la mesure, cliquez sur le bouton "Paramétrage automatique".

Les valeurs existantes sont écrasées par les valeurs de la configuration matérielle.

7.2.3 Paramètres de sortie

7.2.3.1 Paramètres de base de la sortie

Méthode pour le chauffage et le refroidissement

Si le refroidissement est activé dans les paramètres de base, deux méthodes sont disponibles pour le calcul de la valeur de réglage PID :

- Commutation des paramètres PID (Config.AdvancedCooling = TRUE) :
  la valeur de réglage pour le refroidissement est calculée à l'aide d'un jeu de paramètres PID spécifique. L'algorithme PID décide à l'aide de la valeur de réglage calculée et du signal d'écart, si les paramètres PID pour le chauffage ou le refroidissement sont utilisés. Cette méthode est adaptée si un actionneur de chauffage et un actionneur de refroidissement présentent des temps de réponse et des gains différents.
  Ce n'est que si cette méthode est sélectionnée, que l'optimisation préalable et l'optimisation fine pour le refroidissement sont disponibles.
- Facteur de refroidissement (Config.AdvancedCooling = FALSE) :
  la valeur de réglage pour le refroidissement est calculée avec les paramètres PID pour le chauffage en tenant compte du facteur de refroidissement configurable Config.CoolFactor. Cette méthode est adaptée si un actionneur de chauffage et un actionneur de refroidissement présentent un temps de réponse similaire mais des gains différents. Lorsque cette méthode est sélectionnée, l'optimisation préalable et l'optimisation fine pour le refroidissement ainsi que le jeu de paramètres PID pour le refroidissement ne sont pas disponibles. Seules les optimisations pour le chauffage sont exécutées.

Facteur de refroidissement

Si la méthode choisie pour le chauffage/refroidissement est le facteur de refroidissement, celui-ci est pris en compte comme facteur dans le calcul de la valeur de réglage pour le refroidissement. De cette façon, il est possible de prendre en compte différents gains de l'actionneur de chauffage et l'actionneur de refroidissement.

Le facteur de refroidissement n'est pas automatiquement paramétré ou adapté pendant l'optimisation. Vous devez configurer manuellement le facteur de refroidissement correctement avec le rapport "Gain actionneur de chauffage /gain actionneur de refroidissement.

Exemple : Facteur de refroidissement = 2.0, signifie que le gain de l'actionneur de chauffage est le double de celui de l'actionneur de refroidissement.

Le facteur de refroidissement n'est effectif et modifiable que si le "facteur de refroidissement" est choisi comme méthode pour le chauffage/refroidissement.

Comportement en cas d'erreur

IMPORTANT
Votre installation peut être endommagée. En cas d'erreur, si vous fournissez "Valeur actuelle pour la durée de l'erreur" ou "Valeur de réglage de remplacement pour la durée de l'erreur", PID_Temp reste en mode automatique ou en mode manuel. Les limites de la mesure peuvent, de ce fait, être dépassées et votre installation endommagée. Configurez un comportement en cas d'erreur pour votre système réglé, qui protège votre installation de tout endommagement.

PID_Temp est préréglé de telle façon qu'en cas d'erreur, la régulation reste active dans la plupart des cas.

Lorsque des erreurs apparaissent fréquemment en mode régulation, cette valeur par défaut détériore le comportement de régulation. Vérifiez alors le paramètre ErrorBits et éliminez la cause de l'erreur.

En cas d'erreur, PID_Temp fournit une valeur de réglage configurable :

- Zéro (inactif)
  PID_Temp commute en mode "Inactif" pour toutes les erreurs et fournit les valeurs suivantes :
  - 0,0 comme valeur de réglage PID (PidOutputSum)
  - 0,0 comme valeur de réglage pour le chauffage (OutputHeat) et valeur de réglage pour le refroidissement (OutputCool)
  - 0 comme valeur de réglage analogique pour le chauffage (OutputHeat_PER) et valeur de réglage analogique pour le refroidissement (OutputCool_PER)
  - FALSE comme valeur de réglage modulée en largeur d'impulsion pour le chauffage (OutputHeat_PWM) et valeur de réglage modulée en largeur d'impulsion pour le refroidissement (OutputCool_PWM)

  Cela est indépendant des limites et de la mise à l'échelle de la valeur de réglage configurées. Le régulateur n'est réactivé que par un front descendant à Reset ou un front montant à ModeActivate.
- Valeur actuelle pour la durée de l'erreur
  La réaction en cas d'erreur dépend de l'erreur survenue et du mode de fonctionnement.
  Si l'une ou plusieurs des erreurs suivantes apparaissent en mode automatique, PID_Temp reste en mode automatique :
  - 0000001h : Le paramètre Input est en dehors des limites de la mesure.
  - 0000800h : Erreur de temps d'échantillonnage
  - 0040000h : Valeur invalide au paramètre Disturbance.
  - 8000000h : Erreur lors du calcul des paramètres PID.

  Si l'une ou plusieurs des erreurs suivantes surviennent en mode automatique, PID_Temp passe en mode de fonctionnement "Valeur de réglage de remplacement avec surveillance des erreurs" et fournit la dernière valeur de réglage PID valide (PidOutputSum) :
  - 0000002h : Valeur invalide au paramètre Input_PER.
  - 0000200h : Valeur invalide au paramètre Input.
  - 0000400h : Le calcul de la valeur de réglage a échoué.
  - 0001000h : Valeur invalide au paramètre Setpoint ou au paramètre SubstituteSetpoint.

  Les valeurs résultant de la valeur de réglage PID aux sorties pour le chauffage et le refroidissement découlent de la mise à l'échelle de la sortie configurée.

Dès que les erreurs ont disparu, PID_Temp repasse en mode automatique.
Si une erreur survient en mode manuel, PID_Temp reste en mode manuel et continue d'utiliser la valeur manuelle comme valeur de réglage PID.
Si la valeur manuelle est invalide, la valeur de réglage de remplacement configurée est utilisée.
Si la valeur manuelle et la valeur de réglage de remplacement sont invalides, la limite inférieure de la valeur de réglage PID pour le chauffage est utilisée (Config.Output.Heat.PidLowerLimit).
Si l'erreur suivante survient pendant une optimisation préalable ou fine, PID_Temp reste en mode de fonctionnement actif :

– 0000020h : L'optimisation préalable n'est pas autorisée pendant l'optimisation fine.

Pour toutes les autres erreurs, PID_Temp abandonne l'optimisation et passe au mode de fonctionnement à partir duquel l'optimisation a été lancée.

- Valeur de réglage de remplacement pour la durée de l'erreur
  PID_Temp se comporte tel que décrit dans "Valeur actuelle pour la durée de l'erreur", mais fournit la valeur de réglage de remplacement configurée (SubstituteOutput) comme valeur de réglage PID (PidOutputSum) en mode "Valeur de réglage de remplacement avec surveillance des erreurs".
  Les valeurs résultant de la valeur de réglage PID aux sorties pour le chauffage et le refroidissement découlent de la mise à l'échelle de la sortie configurée.
  Pour le régulateur avec sortie de refroidissement activée (Config.ActivateCooling = TRUE) vous entrez
  – une valeur de réglage de remplacement positive pour fournir la valeur aux sorties pour le chauffage.
  – une valeur de réglage de remplacement négative pour fournir la valeur aux sorties pour le refroidissement.

  Si l'erreur suivante se produit, PID_Temp reste en mode de fonctionnement "Valeur de réglage de remplacement avec surveillance des erreurs" et fournit la limite inférieure de la valeur de réglage PID pour le chauffage (Config.Output.Heat.PidLowerLimit) :
  – 0020000h : Valeur invalide à la variable SubstituteOutput.

7.2.3.2 Limites et mise à l'échelle de la valeur de réglage

La valeur de réglage PID (PidOutputSum) est, selon le mode de fonctionnement, calculée automatiquement par l'algorithme PID ou prédéfinie par la valeur manuelle (ManualValue) ou la valeur de réglage de remplacement configurée (SubstituteOutput).
La valeur de réglage PID est limitée en fonction de la configuration :

- Si le refroidissement est désactivé dans les paramètres de base (Config.ActivateCooling = FALSE), la valeur est limitée à la limite supérieure de la valeur de réglage PID (chauffage) (Config.Output.Heat.PidUpperLimit) et à la limite inférieure de la valeur de réglage PID (chauffage) (Config.Output.Heat.PidLowerLimit).
  Vous pouvez configurer les deux valeurs limites dans la section "OutputHeat / OutputCool" sur l'axe horizontal de la courbe caractéristique de mise à l'échelle. Ces valeurs sont affichées dans les sections "OutputHeat_PWM / OutputCool_PWM" et "OutputHeat_PER / OutputCool_PER" mais ne peuvent pas être modifiées.

- Si le refroidissement est activé dans les paramètres de base (Config.ActivateCooling = TRUE), la valeur est limitée à la limite supérieure de la valeur de réglage PID (Config.Output.Heat.PidUpperLimit) et à la limite inférieure de la valeur de réglage PID (refroidissement) (Config.Output.Cool.PidLowerLimit).

  Vous pouvez configurer les deux valeurs limites dans la section "OutputHeat / OutputCool" sur l'axe horizontal de la courbe caractéristique de mise à l'échelle. Ces valeurs sont affichées dans les sections "OutputHeat_PWM / OutputCool_PWM" et "OutputHeat_PER / OutputCool_PER" mais ne peuvent pas être modifiées.

  La limite inférieure de la valeur de réglage PID (chauffage) (Config.Output.Heat.PidLowerLimit) et la limite supérieure de la valeur de réglage PID (refroidissement) (Config.Output.Cool.PidUpperLimit) ne peuvent pas être modifiées et la valeur 0.0 doit leur être attribuée.

La valeur de réglage PID est mise à l'échelle et est fournie aux sorties pour le chauffage et le refroidissement. La mise à l'échelle peut être spécifiée séparément pour chaque sortie et est déterminée à l'aide de 2 paires de valeurs respectivement, constituées d'une limite de la valeur de réglage PID et d'une valeur de mise à l'échelle :

Sortie	**Paire de valeurs**	**Paramètre**
OutputHeat	Paire de valeurs 1	Limite supérieure de la valeur de réglage PID (chauffage) Config.Output.Heat.PidUpperLimit, Valeur de réglage supérieure mise à l'échelle (chauffage) Config.Output.Heat.UpperScaling
	Paire de valeurs 2	Limite inférieure de la valeur de réglage PID (chauffage) Config.Output.Heat.PidLowerLimit, Valeur de réglage inférieure mise à l'échelle (chauffage) Config.Output.Heat.LowerScaling
OutputHeat_PWM	Paire de valeurs 1	Limite supérieure de la valeur de réglage PID (chauffage) Config.Output.Heat.PidUpperLimit, Valeur de réglage modulée en largeur d'impulsion supérieure mise à l'échelle (chauffage) Config.Output.Heat.PwmUpperScaling
	Paire de valeurs 2	Limite inférieure de la valeur de réglage PID (chauffage) Config.Output.Heat.PidLowerLimit, Valeur de réglage modulée en largeur d'impulsion inférieure mise à l'échelle (chauffage) Config.Output.Heat.PwmLowerScaling
OutputHeat_PER	Paire de valeurs 1	Limite supérieure de la valeur de réglage PID (chauffage) Config.Output.Heat.PidUpperLimit, Valeur de réglage analogique supérieure mise à l'échelle (chauffage) Config.Output.Heat.PerUpperScaling
	Paire de valeurs 2	Limite inférieure de la valeur de réglage PID (chauffage) Config.Output.Heat.PidLowerLimit, Valeur de réglage analogique inférieure mise à l'échelle (chauffage) Config.Output.Heat.PerLowerScaling

La limite inférieure de la valeur de réglage PID (chauffage) (Config.Output.Heat.PidLowerLimit) doit avoir la valeur 0.0, si le refroidissement est activé (Config.ActivateCooling = TRUE).

La limite supérieure de la valeur de réglage PID (refroidissement) (Config.Output.Cool.PidUpperLimit) doit toujours avoir la valeur 0.0.

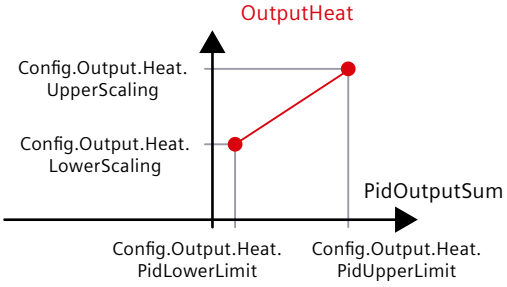
Sortie	Paire de valeurs	Paramètre
OutputCool	Paire de valeurs 1	Limite inférieure de la valeur de réglage PID (refroidissement) Config.Output.Cool.PidLowerLimit, Valeur de réglage supérieure mise à l'échelle (refroidissement) Config.Output.Cool.UpperScaling
	Paire de valeurs 2	Limite supérieure de la valeur de réglage PID (refroidissement) Config.Output.Cool.PidUpperLimit, Valeur de réglage inférieure mise à l'échelle (refroidissement) Config.Output.Cool.LowerScaling
OutputCool_PWM	Paire de valeurs 1	Limite inférieure de la valeur de réglage PID (refroidissement) Config.Output.Cool.PidLowerLimit, Valeur de réglage modulée en largeur d'impulsion supérieure mise à l'échelle (refroidissement) Config.Output.Cool.PwmUpperScaling
	Paire de valeurs 2	Limite supérieure de la valeur de réglage PID (refroidissement) Config.Output.Cool.PidUpperLimit, Valeur de réglage modulée en largeur d'impulsion inférieure mise à l'échelle (refroidissement) Config.Output.Cool.PwmLowerScaling
OutputCool_PER	Paire de valeurs 1	Limite inférieure de la valeur de réglage PID (refroidissement) Config.Output.Cool.PidLowerLimit, Valeur de réglage analogique supérieure mise à l'échelle (refroidissement) Config.Output.Cool.PerUpperScaling
	Paire de valeurs 2	Limite supérieure de la valeur de réglage PID (refroidissement) Config.Output.Cool.PidUpperLimit, Valeur de réglage analogique inférieure mise à l'échelle (refroidissement) Config.Output.Cool.PerLowerScaling

La limite inférieure de la valeur de réglage PID (chauffage) (Config.Output.Heat.PidLowerLimit) doit avoir la valeur 0.0, si le refroidissement est activé (Config.ActivateCooling = TRUE).

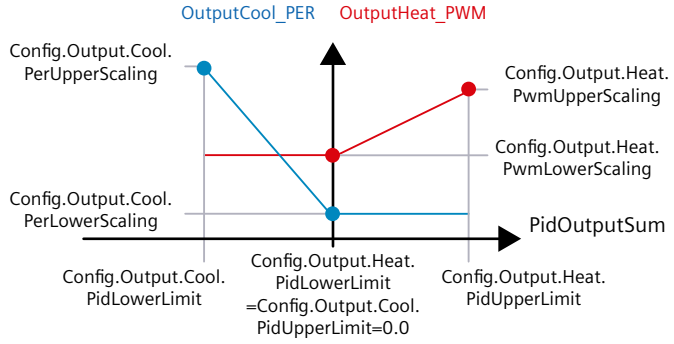
La limite supérieure de la valeur de réglage PID (refroidissement) (Config.Output.Cool.PidUpperLimit) doit toujours avoir la valeur 0.0.

Exemple :

Mise à l'échelle de la sortie lors de l'utilisation de la sortie OutputHeat (refroidissement désactivé ; la limite inférieure de la valeur de réglage PID (chauffage) (Config.Output.Heat.PidLowerLimit) peut ne pas être égale à 0.0) :



Exemple :
Mise à l'échelle de la sortie lors de l'utilisation des sorties OutputHeat_PWM et OutputCool_PER (refroidissement activé ; la limite inférieure de la valeur de réglage PID (chauffage) (Config.Output.Heat.PidLowerLimit) doit être égale à 0.0) :



A l'exception du mode de fonctionnement "Inactif", la valeur à une sortie est toujours comprise entre sa valeur de réglage supérieure mise à l'échelle et sa valeur de réglage inférieure mise à l'échelle, par exemple pour OutputHeat, elle est toujours comprise entre la valeur de réglage supérieure mise à l'échelle (chauffage) (Config.Output.Heat.UpperScaling) et la valeur de réglage inférieure mise à l'échelle (chauffage) (Config.Output.Heat.LowerScaling).

Si vous voulez limiter la valeur à la sortie correspondante, vous devez alors adapter également ces valeurs de mise à l'échelle.

Vous pouvez configurer les valeurs de mise à l'échelle d'une sortie sur l'axe vertical de la courbe de mise à l'échelle. Chaque sortie dispose de deux valeurs de mise à l'échelle propres. Pour OutputHeat_PWM, OutputCool_PWM, OutputHeat_PER et OutputCool_PER, ces valeurs ne peuvent être modifiées que si la sortie correspondante est sélectionnée dans les paramètres de base. Pour toutes les sorties de refroidissement, le refroidissement doit, en plus, être activé dans les paramètres de base.

La vue de courbes dans la boîte de dialogue de mise en service n'enregistre que les valeurs de OutputHeat et OutputCool, quelle que soit la sortie sélectionnée dans les paramètres de base. Adaptez donc également les valeurs de mise à l'échelle pour OutputHeat ou OutputCool, si besoin, si vous utilisez OutputHeat_PWM ou OutputHeat_PER ou bien OutputCool_PWM ou OutputCool_PER et que vous voulez utiliser la vue de courbes dans la boîte de dialogue de mise en service.

7.2.4 Paramètres avancés

7.2.4.1 Surveillance de la mesure

Dans la fenêtre de configuration "Surveillance de la mesure", configurez une limite d'alerte inférieure et une limite d'alerte supérieure de la mesure. Quand l'une de ces limites d'alerte est dépassée par le haut ou par le bas pendant le fonctionnement, l'instruction PID_Temp affiche un avertissement :

- Dans le paramètre de sortie InputWarning_H quand la limite d'alerte supérieure a été dépassée
- Dans le paramètre de sortie InputWarning_L quand la limite d'alerte inférieure a été dépassée par le bas

Les limites d'alerte doivent se situer entre la limite supérieure et la limite inférieure de la mesure.
Si vous n'indiquez pas de valeur, les limites supérieure et inférieure de la mesure seront utilisées.

Exemple

Limite supérieure de la mesure = 98 °C ; limite d'alerte supérieure = 90 °C
Limite d'alerte inférieure = 10 °C ; limite inférieure de la mesure = 0 °C
PID_Temp se comporte comme suit :

Mesure	InputWarning_H	InputWarning_L	ErrorBits
> 98 °C	TRUE	FALSE	0001h
≤ 98 °C et > 90 °C	TRUE	FALSE	0000h
≤ 90 °C et ≥ 10 °C	FALSE	FALSE	0000h
< 10°C et ≥ 0 °C	FALSE	TRUE	0000h
< 0 °C	FALSE	TRUE	0001h

Vous configurez dans les paramètres de sortie la réaction de PID_Temp au dépassement par le haut de la limite supérieure ou par le bas de la limite inférieure de la mesure.

7.2.4.2 Limites de modulation de largeur d'impulsions

La valeur de réglage PID PidOutputSum est normalisée et transformée via une modulation de largeur d'impulsions en une suite d'impulsions fournie au paramètre de sortie OutputHeat_PWM ou OutputCool_PWM.
La "Période d'échantillonnage algorithme PID" est l'écart entre deux calculs de la valeur de réglage PID. La période d'échantillonnage est utilisée comme période de la modulation de largeur d'impulsion.
Pendant le chauffage, la valeur de réglage PID est toujours calculée dans la "Période d'échantillonnage algorithme PID (chauffage)".
Le calcul de la valeur de réglage PID pendant le refroidissement dépend du type de refroidissement choisi dans "Paramètres de base Sortie"

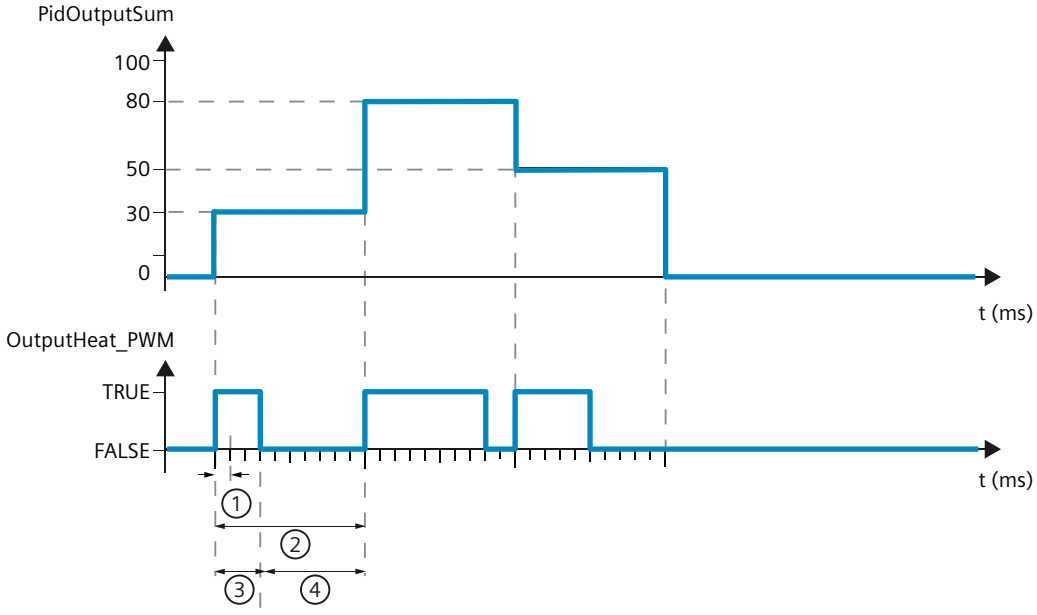
- Si le facteur de refroidissement est utilisé, la "Période d'échantillonnage de l'algorithme PID pour le chauffage " s'applique.
- Si la commutation de paramètres PID est utilisée, la "Période d'échantillonnage de l'algorithme PID pour le refroidissement" s'applique.

La période d'échantillonnage de l'algorithme PID pour le chauffage ou le refroidissement est déterminée pendant l'optimisation préalable ou fine. Si vous réglez manuellement les paramètres PID, vous devez aussi configurer la période d'échantillonnage de l'algorithme PID pour le chauffage ou le refroidissement.

OutputHeat_PWM et OutputCool_PWM sont fournis dans la période d'échantillonnage PID_Temp. La période d'échantillonnage PID_Temp correspond au temps de cycle de l'OB appelant.

La durée d'impulsion est proportionnelle à la valeur de réglage PID et s'élève toujours à un multiple entier de la période d'échantillonnage PID_Temp.

Exemple de OutputHeat_PWM



① Période d'échantillonnage PID_Temp

② Période d'échantillonnage de l'algorithme PID pour le chauffage

③ Durée d'impulsion

④ Durée de pause

Le "plus petit temps ON" et le "plus petit temps OFF", arrondis à un multiple entier de la période d'échantillonnage PID_Temp, peuvent être paramétrés séparément pour le chauffage et pour le refroidissement.

Une impulsion ou une pause n'est jamais plus courte que le plus petit temps ON ou OFF. Les imprécisions qui en résultent sont totalisées et compensées au cycle suivant.

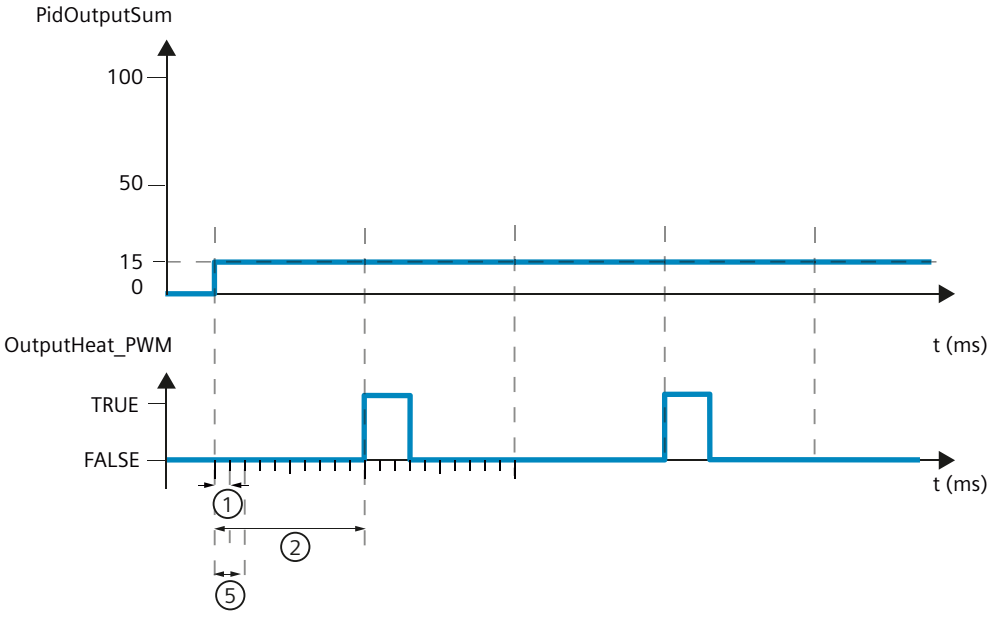
Exemple de OutputHeat_PWM

Période d'échantillonnage PID_Temp (correspond au temps de cycle de l'OB appelant) = 100 ms

Période d'échantillonnage de l'algorithme PID (correspond à la durée de période) = 1000 ms

Plus petit temps ON = 200 ms

La valeur de réglage PID PidOutputSum s'élève toujours à 15 %. La plus petite impulsion que PID_Temp peut fournir correspond à 20 %. Aucune impulsion n'est donnée dans le premier cycle. L'impulsion du premier cycle qui n'a pas été donnée est ajoutée à celle du deuxième cycle.



① Période d'échantillonnage PID_Temp

② Période d'échantillonnage de l'algorithme PID pour le chauffage

⑤ Plus petit temps ON

Pour réduire la fréquence de commutation et pour ménager l'actionneur, rallongez les plus petits temps ON et OFF.

Si vous avez choisi OutputHeat ou OutputCool, ou bien OutputHeat_PER ou OutputCool_PER comme sortie dans les paramètres de base, le plus petit temps ON et le plus petit temps OFF ne sont pas évalués et ne peuvent pas être modifiés.

Si, lors de l'utilisation de OutputHeat_PWM ou de OutputCool_PWM la "Période d'échantillonnage de l'algorithme PID" (Retain.CtrlParams.Heat.Cycle ou Retain.CtrlParams.Cool.Cycle), et par conséquent la durée de la période de la modulation de largeur d'impulsions, est très grande, vous pouvez spécifier une durée de période différente plus courte aux paramètres Config.Output.Heat.PwmPeriode ou Config.Output.Cool.PwmPeriode pour améliorer le lissage de la mesure (voir aussi Variable PwmPeriode (Page 417)).

REMARQUE

Les plus petits temps ON et OFF s'appliquent uniquement aux paramètres de sortie OutputHeat_PWM ou OutputCool_PWM et ne sont pas utilisés pour d'éventuels générateurs d'impulsions intégrés dans la CPU.

7.2.4.3 Paramètres PID

Les paramètres PID sont affichés dans la fenêtre de configuration "Paramètres PID".

Si le refroidissement est activé dans les paramètres de base et que la commutation des paramètres PID est sélectionnée comme méthode de chauffage/refroidissement dans les paramètres de sortie, deux jeux de paramètres sont disponibles : un pour le chauffage et un pour le refroidissement.

Dans ce cas, l'algorithme PID décide à l'aide de la valeur de réglage calculée et du signal d'écart, si les paramètres PID pour le chauffage ou le refroidissement sont utilisés.

Si le refroidissement est désactivé ou que le facteur de refroidissement est choisi comme méthode de chauffage/refroidissement, le jeu de paramètres pour le chauffage est toujours utilisé.

Les paramètres PID sont adaptés à votre système réglé pendant l'optimisation, à l'exception de la largeur de la zone morte, qui doit être configurée manuellement.

REMARQUE

Les paramètres PID actuellement effectifs se trouvent dans la structure Retain.CtrlParams.

Pour éviter un comportement erroné du régulateur PID, ne modifiez les paramètres PID actuellement effectifs en ligne que dans le mode de fonctionnement "Inactif".

Si vous souhaitez modifier en ligne les paramètres PID dans les modes de fonctionnement "Mode automatique" ou "Mode manuel", modifiez les paramètres PID dans la structure CtrlParamsBackUp et appliquez ces modifications dans la structure Retain.CtrlParams via LoadBackUp = TRUE.

Apporter des modifications en ligne aux paramètres PID dans le mode de fonctionnement "Mode automatique" peut provoquer des sauts vers la valeur de réglage.

PID_Temp est un régulateur PIDT1 avec anti-saturation et pondération de l'action P et D.

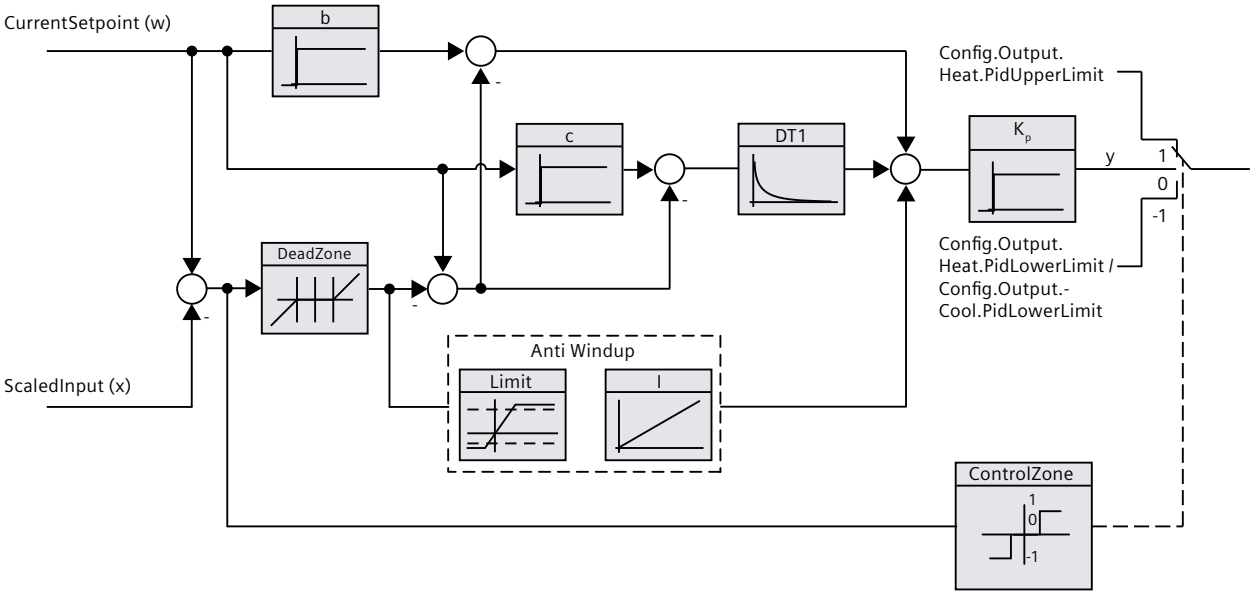
L'algorithme PID fonctionne selon la formule suivante (plage de régulation et zone morte désactivées) :

$$y = K_p \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

Symbole	Description	Paramètres correspondants de l'instruction PID_Temp
y	Valeur de réglage de l'algorithme PID	-
K_p	Gain proportionnel	Retain.CtrlParams.Heat.Gain Retain.CtrlParams.Cool.Gain CoolFactor
s	Opérateur de Laplace	-
b	Pondération de l'action P	Retain.CtrlParams.Heat.PWeighting Retain.CtrlParams.Cool.PWeighting
w	Consigne	CurrentSetpoint
x	Mesure	ScaledInput
T_I	Temps d'intégration	Retain.CtrlParams.Heat.Ti Retain.CtrlParams.Cool.Ti
T_D	Temps de dérivation	Retain.CtrlParams.Heat.Td Retain.CtrlParams.Cool.Td

Symbole	Description	Paramètres correspondants de l'instruction PID_Temp
a	Coefficient pour le retard de dérivation (Retard de dérivation T1 = a × T_D)	Retain.CtrlParams.Heat.TdFiltRatio Retain.CtrlParams.Cool.TdFiltRatio
c	Pondération de l'action D	Retain.CtrlParams.Heat.DWeighting Retain.CtrlParams.Cool.DWeighting
DeadZone	Largeur de zone morte	Retain.CtrlParams.Heat.DeadZone Retain.CtrlParams.Cool.DeadZone
ControlZone	Largeur de la plage de régulation	Retain.CtrlParams.Heat.ControlZone Retain.CtrlParams.Cool.ControlZone

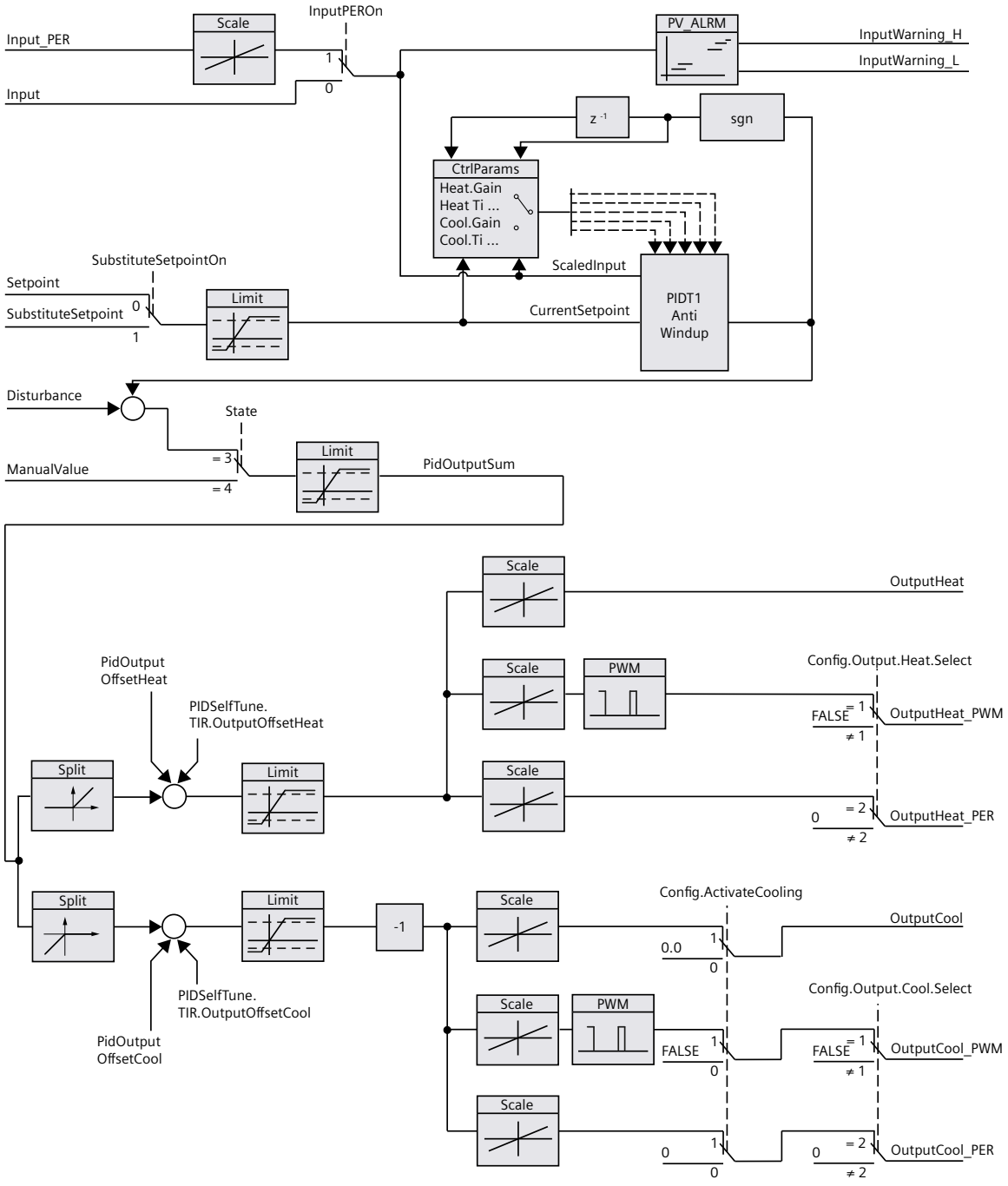
Le graphique suivant illustre l'intégration des paramètres dans l'algorithme PID.



Tous les paramètres PID sont rémanents. Si vous saisissez les paramètres PID manuellement, vous devez charger PID_Temp entièrement (Charger des objets technologiques dans l'appareil (Page 48)).

Schéma fonctionnel PID_Temp

Le schéma fonctionnel suivant montre comment l'algorithme PID est intégré dans PID_Temp.



Gain proportionnel

La valeur indique le gain proportionnel du régulateur. PID_Temp ne fonctionne pas avec un gain proportionnel négatif et ne prend en charge que le sens de régulation normal, c'est-à-dire qu'une augmentation de la valeur de réglage PID (PidOutputSum) doit aboutir à une augmentation de la mesure.

Temps d'intégration

Le temps d'intégration détermine le temps de réponse de l'action I. La désactivation de l'action I s'obtient avec un temps d'intégration = 0.0. Si le temps d'intégration est modifié en ligne en mode de fonctionnement "Mode automatique" en passant d'une autre valeur à 0.0, l'action I actuelle est supprimée et il se produit un saut de la valeur de réglage.

Temps de dérivation

Le temps de dérivation détermine le temps de réponse de l'action D. La désactivation de l'action D s'obtient avec un temps de dérivation = 0.0.

Coefficient de l'action par dérivation

L'effet de l'action D est retardé par le coefficient de l'action par dérivation.

Action par dérivation = Temps de dérivation x Coefficient de l'action par dérivation

- 0.0 : L'action D n'est active que pour un seul cycle et est donc quasiment inactive.
- 0.5 : Cette valeur a fait ses preuves dans la pratique pour les systèmes réglés avec une constante de temps dominante.
- > 1.0 : Plus le coefficient est grand, plus l'effet de l'action D est retardé.

Pondération de l'action P

En cas de modification de consigne, vous pouvez réduire l'action P.

Les valeurs comprises entre 0.0 et 1.0 sont judicieuses.

- 1.0 : Action P totalement opérante si modification de la consigne
- 0.0 : Action P non opérante si modification de la consigne

En cas de variation de la mesure, l'action P est toujours totalement opérante.

Pondération de l'action D

En cas de modification de consigne, vous pouvez réduire l'action D.

Les valeurs comprises entre 0.0 et 1.0 sont judicieuses.

- 1.0 : En cas de modification de la consigne, l'action D est totalement opérante
- 0.0 : En cas de modification de la consigne, l'action D n'est pas opérante

En cas de variation de la mesure, l'action D est toujours totalement opérante.

Période d'échantillonnage de l'algorithme PID

Comme le système réglé nécessite un certain temps pour réagir à une modification de la valeur de réglage, il est judicieux de ne pas calculer cette valeur à chaque cycle. La période d'échantillonnage "Algorithme PID" est l'écart entre deux calculs de la valeur de réglage PID. Il est déterminé pendant l'optimisation et arrondi à un multiple de la période d'échantillonnage PID_Temp (temps de cycle de l'OB d'alarme cyclique). Toutes les autres fonctions de PID_Temp sont exécutées lors de chaque appel.

Si vous utilisez OutputHeat_PWM ou. OutputCool_PWM, la période d'échantillonnage de l'algorithme PID est utilisée comme durée de la période de la modulation de largeur d'impulsions. La précision du signal de sortie est déterminée par le rapport entre la période d'échantillonnage de l'algorithme PID et le temps de cycle de l'OB. Le temps de cycle doit s'élever au plus à un dixième de la période d'échantillonnage de l'algorithme PID.

La période d'échantillonnage de l'algorithme PID qui est utilisée comme durée de la période de la modulation de largeur d'impulsions pour OutputCool_PWM dépend de la méthode de chauffage/refroidissement choisie dans les "Paramètres de base Sortie" :

- Si le facteur de refroidissement est utilisé, la "période d'échantillonnage de l'algorithme PID pour le chauffage" s'applique également à OutputCool_PWM.
- Si la commutation de paramètres PID est utilisée, la "période d'échantillonnage de l'algorithme PID pour le refroidissement" s'applique comme durée de la période à OutputCool_PWM.

Si, lors de l'utilisation de OutputHeat_PWM ou de OutputCool_PWM la période d'échantillonnage de l'algorithme PID, et par conséquent, la durée de la période de la modulation de largeur d'impulsions, est très grande, vous pouvez spécifier une durée de période différente, plus courte aux paramètres Config.Output.Heat.PwmPeriode ou Config.Output.Cool.PwmPeriode pour améliorer le lissage de la mesure.

Largeur de zone morte

Si la mesure comporte des parasites, le taux de bruit a également un effet sur la valeur de réglage. Lorsque le gain de régulateur est élevé et l'action D, activée, la valeur de réglage peut osciller fortement. Si la mesure est comprise dans la zone morte autour de la consigne, le signal d'écart est réduit de telle sorte que l'algorithme PID ne réagisse pas et que les fluctuations inutiles de la valeur de réglage soient réduites.

La largeur de zone morte pour le chauffage ou le refroidissement n'est pas paramétrée automatiquement au cours de l'optimisation. Vous devez configurer correctement la largeur de zone morte manuellement. La désactivation de la zone morte s'obtient avec une largeur de zone morte = 0.0.

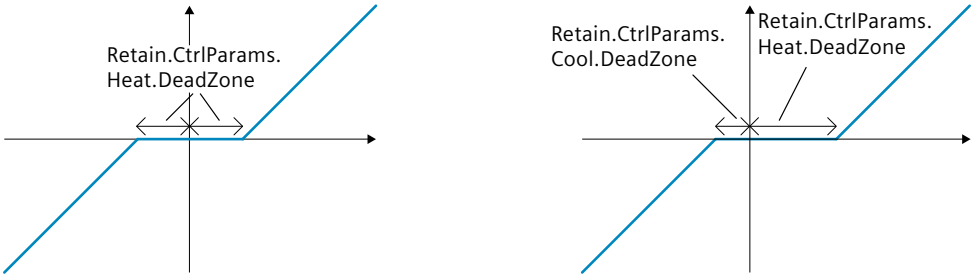
Lorsque la zone morte est activée, un signal d'écart (écart entre consigne et mesure) permanent peut s'établir. Cela peut avoir un effet négatif sur l'exécution d'une optimisation fine.

Si le refroidissement est activé dans les paramètres de base et que la commutation de paramètres PID est sélectionnée comme méthode de chauffage/refroidissement dans les paramètres de base, la zone morte est comprise entre "Consigne - largeur de zone morte (chauffage)" et "Consigne + largeur de zone morte (refroidissement)".

Si le refroidissement est désactivé dans les paramètres de base ou que le facteur de refroidissement est utilisé, la zone morte est comprise symétriquement entre "Consigne - largeur de zone morte (chauffage)" et "Consigne + largeur de zone morte (chauffage)".

Si des valeurs différentes de 1.0 sont configurées pour la pondération de l'action P ou la pondération de l'action D, des variations de la consigne se répercutent également dans la zone morte sur la valeur de sortie.

Des variations de la valeur réelle dans la zone morte ne se répercutent pas sur la valeur de sortie, et ce, indépendamment des pondérations.



Zone morte avec refroidissement désactivé ou facteur de refroidissement (à gauche) ou alors refroidissement activé et commutation de paramètres PID (à droite). L'axe x/horizontal montre le signal d'écart = consigne - mesure. L'axe y/vertical montre le signal de sortie de la zone morte, qui est transmis à l'algorithme PID.

Largeur de la plage de régulation

Si la mesure quitte la plage de régulation autour de la consigne, la valeur de réglage minimum ou maximum est fournie. Ainsi, la mesure peut atteindre la consigne plus rapidement.

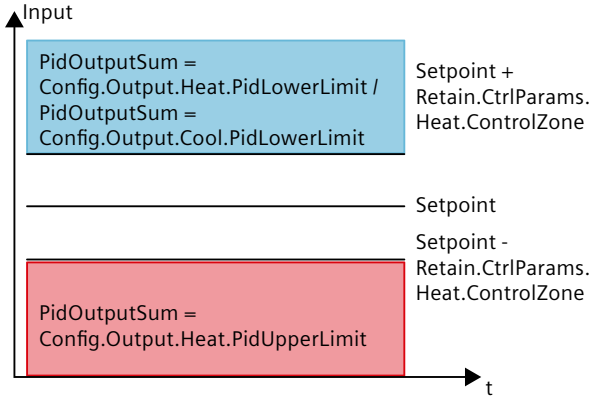
Si la mesure se trouve dans la plage de régulation autour de la consigne, la valeur de réglage de l'algorithme PID est calculée.

La largeur de la plage de régulation pour le chauffage ou le refroidissement n'est automatiquement paramétrée que pendant l'optimisation préalable, si "PID (température)" est choisi comme structure de régulateur pour le chauffage ou le refroidissement.

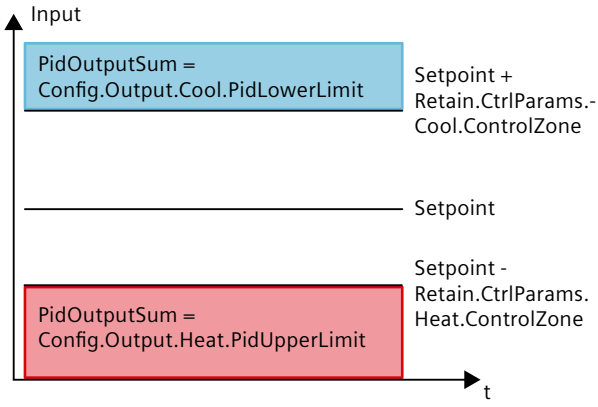
La désactivation de la plage de régulation s'obtient avec une largeur de plage de régulation = 3.402822e+38.

Si le refroidissement est désactivé dans les paramètres de base ou que le facteur de refroidissement est utilisé, la plage de régulation est comprise symétriquement entre "Consigne - largeur de la plage de régulation (chauffage)" et "Consigne + largeur de la plage de régulation (chauffage)".

Si le refroidissement est activé dans les paramètres de base et que la commutation de paramètres PID est sélectionnée comme méthode de chauffage/refroidissement dans les paramètres de base, la plage de régulation est comprise entre "Consigne - largeur de la plage de régulation (chauffage)" et "Consigne + largeur de plage de régulation (refroidissement)".



Plage de régulation avec refroidissement désactivé ou facteur de refroidissement.



Plage de régulation avec refroidissement activé et commutation de paramètres PID.

Règle pour l'optimisation

Dans la liste déroulante "Structure du régulateur", sélectionnez si des paramètres PID ou PI sont calculés. Vous pouvez spécifier les règles d'optimisation pour le chauffage et d'optimisation pour le refroidissement séparément.

- PID (température)
  Des paramètres PID sont calculés pendant l'optimisation préalable et l'optimisation fine. Cependant, l'optimisation préalable est conçue pour des procédés thermiques et produit un comportement de régulation plus lent et plutôt asymptotique avec une suroscillation plus faible comparé à l'option "PID". L'optimisation fine est identique à l'option "PID".
  La largeur de la plage de régulation n'est automatiquement déterminée pendant l'optimisation préalable que lorsque cette option est choisie.
- PID
  Des paramètres PID sont calculés pendant l'optimisation préalable et l'optimisation fine.
- PI
  Des paramètres PI sont calculés pendant l'optimisation préalable et l'optimisation fine.
- Personnalisé
  Si vous avez réglé des structures de régulateur différentes pour l'optimisation préalable et l'optimisation fine via le programme utilisateur ou la vue des paramètres, "Personnalisé" est affiché dans la liste déroulante.

7.3 Mise en service de PID_Temp

7.3.1 Mise en service

La fenêtre de mise en service vous assiste lors de la mise en service du régulateur PID. La vue de courbes permet de visualiser les valeurs de la consigne, la mesure et les valeurs de réglage pour le chauffage et le refroidissement sur l'axe du temps. Les fonctions suivantes sont prises en charge dans la fenêtre de mise en service :

- Optimisation préalable du régulateur
- Optimisation fine du régulateur
  Servez-vous de l'optimisation fine pour une adaptation fine des paramètres PID.
- Visualisation de la régulation en cours dans la fenêtre des courbes
- Test du système réglé en spécifiant une valeur de réglage PID manuelle et une consigne de remplacement
- Sauvegarde des valeurs effectives des paramètres PID dans le projet hors ligne.

Une liaison en ligne doit être établie avec la CPU pour toutes les fonctions.

Les boutons "Visualiser tout" ou "Démarrage" de la vue de courbes permettent d'établir la liaison en ligne avec la CPU, si elle ne l'est pas encore, et de débloquer l'utilisation de la fenêtre de mise en service.

Utilisation de la vue de courbes

- Sélectionnez le temps d'échantillonnage souhaité dans la liste déroulante "Période d'échantillonnage".
  Toutes les valeurs dans la vue de courbes sont actualisées pendant la période d'échantillonnage choisie.
- Si vous souhaitez utiliser la vue de courbes, cliquez sur l'icône "Démarrage" du groupe Mesure.
  L'enregistrement des valeurs démarre. Les valeurs actuelles pour la consigne, la mesure et les valeurs de réglage pour le chauffage et le refroidissement sont écrites dans la vue de courbes.
- Si vous souhaitez terminer la vue de courbes, cliquez sur l'icône "Arrêt".
  L'analyse des valeurs tracées dans la vue de courbes peut continuer.

Lorsque vous fermez la fenêtre de mise en service, l'enregistrement prend fin dans la vue de courbes et les valeurs enregistrées sont effacées.

7.3.2 Optimisation préalable

L'optimisation préalable détermine la réponse du processus à un échelon de la valeur de réglage et recherche le point d'inflexion. Les paramètres PID optimisés sont calculés à partir de l'incrémentation maximale et du temps mort du système réglé. Les meilleurs paramètres PID sont obtenus pendant l'exécution d'une optimisation préalable et d'une optimisation fine.

Plus la mesure est stable, plus il sera facile de déterminer des paramètres PID précis. Un bruit de la mesure est acceptable tant que la croissance de la mesure est nettement supérieure au bruit. Cela est plutôt le cas en mode de fonctionnement "Inactif" ou "Mode manuel". Les paramètres PID sont sauvegardés avant qu'ils ne soient recalculés.

PID_Temp propose différents types d'optimisation préalable selon la configuration :

- Optimisation préalable du chauffage
  Un échelon est appliqué à la valeur de réglage pour le chauffage, les paramètres PID pour le chauffage sont calculés puis la régulation est effectuée en fonction de la consigne en mode automatique.
- Optimisation préalable du chauffage et du refroidissement
  Un échelon est appliqué à la valeur de réglage pour le chauffage.
  Dès que la mesure s'approche de la consigne, un échelon est appliqué à la valeur de réglage du refroidissement.
  Les paramètres PID pour le chauffage (structure Retain.CtrlParams.Heat) et le refroidissement (structure Retain.CtrlParams.Cool) sont calculés puis la régulation est effectuée en fonction de la consigne en mode automatique.
- Optimisation préalable du refroidissement
  Un échelon est appliqué à la valeur de réglage pour le refroidissement.
  Les paramètres PID pour le refroidissement sont calculés puis la régulation est effectuée en fonction de la consigne en mode automatique.

Si vous voulez optimiser les paramètres PID pour le chauffage et le refroidissement, vous pouvez compter sur un meilleur comportement de régulation si vous effectuez une "Optimisation préalable du chauffage" puis une "Optimisation préalable du refroidissement" que si vous effectuez une "Optimisation préalable du chauffage et du refroidissement". L'exécution de l'optimisation préalable en deux étapes requiert toutefois plus de temps.

Conditions générales

- L'instruction PID_Temp est appelée dans un OB d'alarme cyclique.
- ManualEnable = FALSE
- Reset = FALSE
- PID_Temp se trouve en mode de fonctionnement "Inactif", "Mode manuel" ou "Mode automatique".
- La consigne et la mesure se trouvent dans les limites configurées (voir configuration Surveillance de la mesure (Page 167)).

Conditions pour une optimisation préalable du chauffage

- La différence entre la consigne et la mesure représente plus de 30 % de la différence entre la limite supérieure et la limite inférieure de la mesure.
- L'écart entre la consigne et la mesure est supérieur à 50 % de la consigne.
- La consigne est supérieure à la mesure.

Conditions pour une optimisation préalable du chauffage et du refroidissement

- La sortie de refroidissement est activée dans les "Paramètres de base" (Config.ActivateCooling = TRUE).
- La commutation de paramètres PID est activée dans les "Paramètres de base de la valeur de réglage" (Config.AdvancedCooling = TRUE).
- La différence entre la consigne et la mesure représente plus de 30 % de la différence entre la limite supérieure et la limite inférieure de la mesure.
- L'écart entre la consigne et la mesure est supérieur à 50 % de la consigne.
- La consigne est supérieure à la mesure.

Conditions pour une optimisation préalable du refroidissement

- La sortie de refroidissement est activée dans les "Paramètres de base" (Config.ActivateCooling = TRUE).
- La commutation de paramètres PID est activée dans les "Paramètres de base de la valeur de réglage" (Config.AdvancedCooling = TRUE).
- Une "Optimisation préalable du chauffage" ou une "Optimisation préalable du chauffage et du refroidissement" a été effectuée correctement (PIDSelfTune.SUT.ProcParHeatOk = TRUE). La même consigne doit être utilisée pour toutes les optimisations.
- La différence entre la consigne et la mesure représente moins de 5 % de la différence entre la limite supérieure et la limite inférieure de la mesure.

Marche à suivre

Pour réaliser une optimisation préalable, procédez de la manière suivante :

1. Dans la navigation de projet, double-cliquez sur l'entrée "PID_Temp > Mise en service".
2. Actionnez le bouton "Visualiser tout" ou démarrez la vue de courbes.
   Une liaison en ligne est établie.
3. Sélectionnez l'optimisation préalable souhaitée dans la liste déroulante "Type d'optimisation".
4. Cliquez sur l'icône "Start".
   – L'optimisation préalable est lancée.
   – Les étapes actuelles et éventuelles erreurs s'affichent dans le champ "Etat". La barre de progression affiche la progression de l'étape actuelle.

REMARQUE

Cliquez sur l'icône "Stop" lorsque la barre de progression (variable "progress") ne change pas pendant un long moment et qu'il faut supposer un blocage de l'optimisation. Vérifiez la configuration de l'objet technologique et redémarrez éventuellement l'optimisation.

Résultat

Si l'optimisation préalable a été réalisée sans message d'erreur, les paramètres PID ont été optimisés. PID_Temp passe en mode automatique et utilise les paramètres optimisés. Les paramètres PID optimisés sont conservés lors d'une mise hors tension et d'un redémarrage de la CPU.
Si une optimisation préalable n'est pas possible, PID_Temp se comporte comme cela a été configuré sous Comportement en cas d'erreur.

7.3.3 Optimisation fine

L'optimisation fine génère une oscillation constante limitée de la mesure. Les paramètres PID sont optimisés, pour le point de fonctionnement, à partir de l'amplitude et de la fréquence de cette oscillation. Les paramètres PID sont recalculés à partir des résultats. Les paramètres PID de l'optimisation fine montrent généralement un meilleur comportement de référence et de perturbation que les paramètres PID de l'optimisation préalable. Les meilleurs paramètres PID sont obtenus pendant l'exécution d'une optimisation préalable et d'une optimisation fine.
PID_Temp tente automatiquement de générer une oscillation supérieure au bruit de la mesure. La stabilité de la mesure n'influence l'optimisation fine que de manière insignifiante. Les paramètres PID sont sauvegardés avant qu'ils ne soient recalculés.
PID_Temp propose différents types d'optimisation fine selon la configuration :

- Optimisation fine du chauffage :
  PID_Temp génère une oscillation de la mesure par des modifications périodiques de la valeur de réglage du chauffage et calcule les paramètres PID pour le chauffage.
- Optimisation fine du refroidissement :
  PID_Temp génère une oscillation de la mesure par des modifications périodiques de la valeur de réglage du refroidissement et calcule les paramètres PID pour le refroidissement.

Décalage d'optimisation temporaire pour le régulateur de chauffage et de refroidissement

Si PID_Temp est utilisé comme régulateur de chauffage et de refroidissement (Config.ActivateCooling = TRUE), la valeur de réglage PID (PidOutputSum) à la consigne doit remplir la condition suivante pour qu'une oscillation de la mesure puisse être générée et que l'optimisation fine puisse être effectuée correctement :

- Valeur de réglage PID positive pour l'optimisation fine du chauffage
- Valeur de réglage PID négative pour l'optimisation fine du refroidissement

Si cette condition n'est pas remplie, vous pouvez spécifier un décalage temporaire pour l'optimisation fine qui est fourni à la sortie ayant l'action contraire.

- Décalage pour la sortie de refroidissement (PIDSelfTune.TIR.OutputOffsetCool) lors de l'optimisation fine du chauffage.
  Avant le démarrage de l'optimisation, spécifiez un décalage d'optimisation de refroidissement négatif, inférieur à la valeur de réglage PID (PidOutputSum) à la consigne à l'état stationnaire.
- Décalage pour la sortie de chauffage (PIDSelfTune.TIR.OutputOffsetHeat) lors de l'optimisation fine du refroidissement
  Avant le démarrage de l'optimisation, spécifiez un décalage d'optimisation de chauffage positif, supérieur à la valeur de réglage PID (PidOutputSum) à la consigne à l'état stationnaire.

Le décalage spécifié est alors compensé par l'algorithme PID de sorte que la mesure reste proche de la consigne. La valeur du décalage permet ainsi d'adapter la valeur de réglage PID en conséquence, pour que cette dernière remplisse la condition susmentionnée.

Pour éviter des suroscillations plus fortes de la mesure lors de la spécification du décalage, il est possible d'augmenter celui-ci en plusieurs étapes.

Si PID_Temp quitte le mode de fonctionnement optimisation fine, le décalage de l'optimisation est réinitialisé.

Exemple : Spécification d'un décalage pour l'optimisation fine du refroidissement

- Sans décalage
  - Consigne (Setpoint) = mesure (ScaledInput) = 80 °C
  - Valeur de réglage PID (PidOutputSum) = 30,0
  - Valeur de réglage du chauffage (OutputHeat) = 30,0
  - Valeur de réglage du refroidissement (OutputCool) = 0,0
    Une oscillation de la mesure autour de la consigne ne peut pas être générée uniquement avec la sortie de refroidissement. L'optimisation fine échouerait dans ce cas.
- Avec décalage pour la sortie de chauffage (PIDSelfTune.TIR.OutputOffsetHeat) = 80,0
  - Consigne (Setpoint) = mesure (ScaledInput) = 80 °C
  - Valeur de réglage PID (PidOutputSum) = -50,0
  - Valeur de réglage du chauffage (OutputHeat) = 80,0
  - Valeur de réglage du refroidissement (OutputCool) = -50,0
    Grâce à la spécification d'un décalage pour la sortie de chauffage, la sortie de refroidissement peut désormais générer une oscillation de la mesure autour de la consigne. L'optimisation fine peut ainsi être effectuée correctement.

Conditions générales

- L'instruction PID_Temp est appelée dans un OB d'alarme cyclique.
- ManualEnable = FALSE
- Reset = FALSE
- La consigne et la mesure se trouvent dans les limites configurées (voir configuration "Paramètres de la mesure").
- La boucle de régulation est en régime stationnaire au point de fonctionnement. Le point de fonctionnement est atteint lorsque la mesure correspond à la consigne.
  Lorsque la zone morte est activée, un signal d'écart (écart entre consigne et mesure) permanent peut s'établir. Cela peut avoir un effet négatif sur l'exécution de l'optimisation fine.
- Aucune perturbation n'est attendue.
- PID_Temp se trouve en mode de fonctionnement Inactif, Mode automatique ou Mode manuel.

Conditions pour une optimisation fine du chauffage

- Heat.EnableTuning = TRUE
- Cool.EnableTuning = FALSE
- Si PID_Temp est configuré comme régulateur de chauffage et de refroidissement (Config.ActivateCooling = TRUE), la sortie de chauffage doit être active au point de fonctionnement auquel l'optimisation doit être effectuée.
  PidOutputSum > 0,0 (voir décalage d'optimisation)

Conditions pour une optimisation fine du refroidissement

- Heat.EnableTuning = FALSE
- Cool.EnableTuning = TRUE
- La sortie de refroidissement est activée (Config.ActivateCooling = TRUE).
- La commutation de paramètres PID est activée (Config.AdvancedCooling = TRUE).
- La sortie de refroidissement doit être active au point de fonctionnement auquel l'optimisation doit être effectuée.
  PidOutputSum < 0,0 (voir décalage d'optimisation)

Déroulement dépendant de la situation de départ

Vous pouvez démarrer l'optimisation fine à partir des modes de fonctionnement "Inactif", "Mode automatique" ou "Mode manuel".

L'optimisation fine se déroule de la manière suivante au démarrage :

- Mode automatique avec PIDSelfTune.TIR.RunIn = FALSE (préréglage)
  Si vous souhaitez améliorer les paramètres PID existants à l'aide de l'optimisation, démarrez l'optimisation fine à partir du mode automatique.
  PID_Temp utilise les paramètres PID existants pour la régulation jusqu'à ce que la boucle de régulation soit en régime établi et que les conditions pour une optimisation fine soient remplies. C'est seulement après cela que l'optimisation fine commence.
- Mode inactif, manuel ou automatique avec PIDSelfTune.TIR.RunIn = TRUE
  Le système essaie d'atteindre la consigne avec la valeur de réglage minimum ou maximum (régulation à deux échelons) :
  - avec la valeur de réglage minimum ou maximum pour le chauffage pour l'optimisation fine du chauffage.
  - avec la valeur de réglage minimum ou maximum pour le refroidissement pour l'optimisation fine du refroidissement.

  Cela peut entraîner une suroscillation élevée. Si la consigne est atteinte, l'optimisation fine démarre.
  Si la consigne ne peut pas être atteinte, PID_Temp n'abandonne pas automatiquement l'optimisation.

Marche à suivre

Pour réaliser l'"optimisation fine", procédez de la manière suivante :

1. Dans la navigation de projet, double-cliquez sur l'entrée "PID_Temp > Mise en service".
2. Actionnez le bouton "Visualiser tout" ou démarrez la vue de courbes.
   Une liaison en ligne est établie.
3. Sélectionnez l'entrée d'optimisation fine souhaitée dans la liste déroulante "Type d'optimisation".
4. Spécifiez un décalage d'optimisation si besoin (voir décalage d'optimisation) et attendez jusqu'à ce que l'état stationnaire soit établi.
5. Cliquez sur l'icône "Start".
   - Le déroulement de l'optimisation fine démarre.
   - Les étapes actuelles et éventuelles erreurs s'affichent dans le champ "Etat".
     La barre de progression affiche la progression de l'étape actuelle.

REMARQUE

Dans le groupe "Type d'optimisation, cliquez sur l'icône "Stop" si la barre de progression (variable "progress") ne change pas pendant un long moment et qu'il faut supposer un blocage de l'optimisation. Vérifiez la configuration de l'objet technologique et redémarrez éventuellement l'optimisation.

Lors des phases suivantes notamment, l'optimisation n'est pas automatiquement abandonnée si la consigne ne peut pas être atteinte.

- "Essayer d'atteindre la consigne avec la régulation de chauffage deux points."
- "Essayer d'atteindre la consigne avec la régulation de refroidissement deux points."

Résultat

Si aucune erreur n'est apparue pendant l'optimisation fine, les paramètres PID ont été optimisés. PID_Temp passe en mode automatique et utilise les paramètres optimisés. Les paramètres PID optimisés sont conservés lors d'une mise hors tension et d'un redémarrage de la CPU.

Si des erreurs sont apparues au cours de l'optimisation fine, PID_Temp se comporte comme cela a été configuré sous Comportement en cas d'erreur.

7.3.4 Mode de fonctionnement "Mode manuel"

Ce paragraphe décrit comment utiliser le mode de fonctionnement "Mode manuel" dans la fenêtre de mise en service de l'objet technologique "PID_Temp".
En cas d'erreur, le mode manuel est également possible.

Condition préalable

- L'instruction "PID_Temp" est appelée dans un OB d'alarme cyclique.
- Une liaison en ligne avec la CPU est établie.
- La CPU est à l'état de fonctionnement "RUN".

Marche à suivre

Si vous souhaitez tester le système réglé en spécifiant une valeur manuelle, utilisez "Mode Manuel" dans la fenêtre de mise en service
Pour spécifier une valeur manuelle, procédez comme suit :

1. Dans la navigation de projet, double-cliquez sur l'entrée "PID_Temp > Mise en service".
2. Actionnez le bouton "Visualiser tout" ou démarrez la vue de courbes.
   Une liaison en ligne est établie.
3. Cochez la case "Mode manuel" dans la zone "Etat en ligne du régulateur".
   PID_Temp fonctionne en mode manuel. La dernière valeur de réglage actuelle reste active.
4. Dans le champ éditable, saisissez la valeur manuelle dans l'unité %.
   Si le refroidissement est activé dans les paramètres de base, saisissez la valeur manuelle comme suit :
   - pour fournir la valeur aux sorties de chauffage, saisissez une valeur manuelle positive.
   - pour fournir la valeur aux sorties de refroidissement, saisissez une valeur manuelle négative.
5. Cliquez sur l'icône .

Résultat

La valeur manuelle est écrite dans la CPU et elle est opérante immédiatement.
Décochez la case "Mode manuel" pour que la valeur de réglage soit à nouveau spécifiée par le régulateur PID.
Le passage au mode automatique s'effectue sans à-coups.

7.3.5 Consigne de remplacement

Ce paragraphe décrit comment utiliser la consigne de remplacement dans la fenêtre de mise en service de l'objet technologique "PID_Temp".

Condition préalable

- L'instruction "PID_Temp" est appelée dans un OB d'alarme cyclique.
- Une liaison en ligne avec la CPU est établie.
- La CPU est à l'état de fonctionnement "RUN".

Marche à suivre

Si vous souhaitez utiliser temporairement une autre valeur que celle du paramètre "Setpoint" comme consigne (par exemple pour optimiser un esclave dans une cascade), utilisez la consigne de remplacement dans la fenêtre de mise en service.

Pour spécifier une consigne de remplacement, procédez comme suit :

1. Dans la navigation de projet, double-cliquez sur l'entrée "PID_Temp > Mise en service".
2. Actionnez le bouton "Visualiser tout" ou démarrez la vue de courbes.
   Une liaison en ligne est établie.
3. Cochez la case "Subst.Setpoint" dans la zone "Etat en ligne du régulateur".
   La consigne de remplacement (variable SubstituteSetpoint) est initialisée avec la dernière consigne actuelle et est désormais utilisée.
4. Dans le champ éditable, saisissez la consigne de remplacement.
5. Cliquez sur l'icône .

Résultat

La consigne de remplacement est écrite dans la CPU et elle est opérante immédiatement.

Décochez la case "Subst.Setpoint" pour que la valeur du paramètre "Setpoint" soit à nouveau utilisée comme consigne.

Le changement ne s'effectue pas sans à-coups.

7.3.6 Mise en service de cascades

Vous trouverez des informations sur la mise en service de cascades avec PID_Temp sous Mise en service (Page 190).

7.4 Fonction cascade avec PID_Temp

7.4.1 Introduction

En cas de régulation en cascade, plusieurs boucles de régulation sont imbriquées les unes dans les autres. Les esclaves reçoivent leur consigne (Setpoint) à partir de la valeur de réglage (OutputHeat) du maître supérieur correspondant.

La condition pour l'établissement d'une régulation en cascade est que le système réglé puisse être subdivisé en systèmes partiels avec chacun une grandeur de mesure spécifique.

La transmission de la consigne pour la grandeur réglée est effectuée sur le maître le plus extérieur.

La valeur de réglage de l'esclave le plus intérieur est appliquée sur l'actionneur et agit ainsi sur le système réglé.

Les avantages essentiels suivants découlent de l'utilisation d'une régulation en cascade comparée à une boucle de régulation unique :

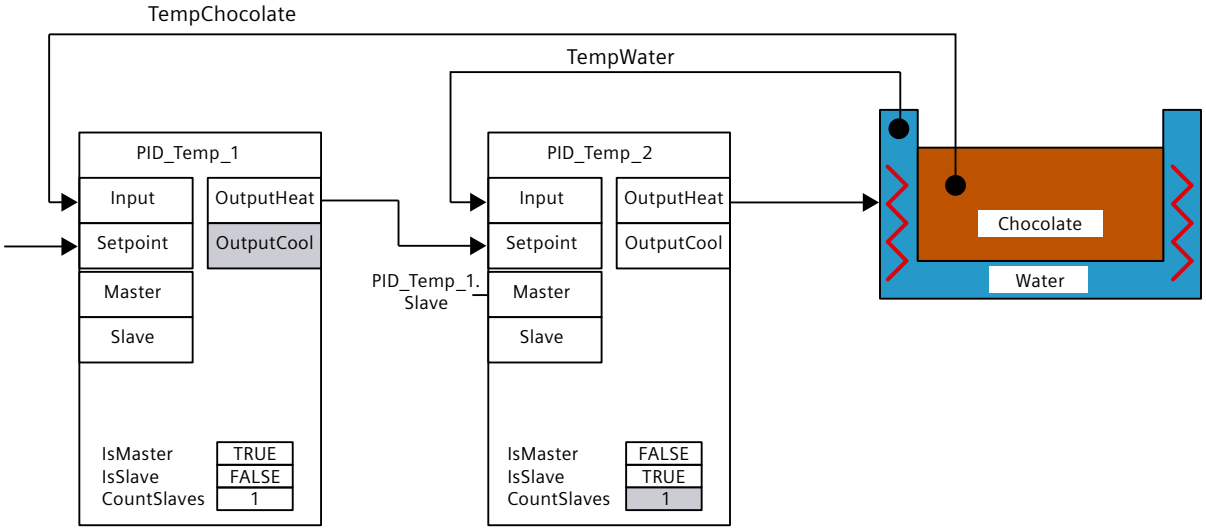
- Les boucles de régulation esclaves supplémentaires permettent d'éliminer rapidement les perturbations qui surviennent. Par conséquent, l'influence de celles-ci sur la grandeur réglée est fortement réduite. Le comportement de perturbation est donc amélioré.
- Les boucles de régulation esclaves exercent un effet de linéarisation. Cette action atténue les effets négatifs des non-linéarités sur la grandeur réglée.

PID_Temp offre la fonctionnalité suivante spécialement pour l'utilisation dans des régulations en cascade :

- Transmission d'une consigne de remplacement
- Echange d'informations d'état entre maître et esclave (par ex. mode de fonctionnement actuel)
- Différents modes Anti-Wind-Up (réaction du maître à la limitation de ses esclaves)

Exemple

Le schéma fonctionnel suivant montre une régulation en cascade avec PID_Temp à l'aide de l'exemple simplifié de la fonte du chocolat :



Le PID_Temp_1 maître compare la mesure de la température du chocolat (TempChocolate) à la consigne transmise par l'utilisateur à son paramètre Setpoint. Sa valeur de réglage OutputHeat forme la consigne de l'esclave PID_Temp_2.

PID_Temp_2 essaie de réguler la mesure de la température du bain marie (TempWater) en fonction de cette consigne. La valeur de réglage de PID_Temp_2 est directement appliquée sur l'actionneur du système réglé (chauffage du bain-marie) et influence ainsi la température du bain-marie. La température du bain-marie agit à son tour sur la température du chocolat.

FAQ

Pour plus d'informations à ce sujet, voir la FAQ suivante dans l'assistance en ligne de Siemens Industry :

- ID de contribution 103526819 (https://support.industry.siemens.com/cs/ww/en/view/103526819)

Voir aussi

Programmation (Page 188)

7.4.2 Programmation

Veuillez respecter les points suivants lors de la programmation :

- Nombre d'instances PID_Temp
  Il faut appeler autant d'instances différentes de PID_Temp dans un OB d'alarme cyclique qu'il existe de grandeurs de mesure enchaînées dans le processus.
  Dans l'exemple, il existe deux grandeurs de mesure enchaînées : TempChocolate et TempWater. Deux instances de PID_Temp sont donc nécessaires.
- Ordre d'appel
  Un maître doit être appelé avant ses esclaves dans le même OB d'alarme cyclique.
  Le maître le plus extérieur, sur lequel la consigne de l'utilisateur est spécifiée, est appelé en premier.
  L'esclave recevant sa consigne du maître le plus extérieur est appelé en deuxième et ainsi de suite.
  L'esclave le plus intérieur, qui agit avec sa valeur de réglage sur l'actionneur du processus, est appelé en dernier.
  Dans l'exemple, PID_Temp_1 est appelé avant PID_Temp_2.
- Interconnexion des grandeurs de mesure
  Le maître le plus extérieur est interconnecté avec la grandeur de mesure la plus extérieure, devant être régulée selon la consigne de l'utilisateur.
  L'esclave le plus intérieur est interconnecté à la grandeur de mesure la plus intérieure, qui est directement influencée par l'actionneur.
  L'interconnexion des grandeurs de mesure à PID_Temp s'effectue avec les paramètres Input ou Input_PER.
  Dans l'exemple, la grandeur de mesure extérieure TempChocolate est interconnectée avec PID_Temp_1 et la grandeur de mesure intérieure TempWater, avec PID_Temp_2.
- Interconnexion de la valeur de réglage du maître à la consigne de l'esclave
  La valeur de réglage (OutputHeat) d'un maître doit être affectée à la consigne (Setpoint) de son esclave.
  Vous pouvez effectuer cette interconnexion manuellement dans l'éditeur de programme ou la faire effectuer automatiquement dans la fenêtre d'inspection de l'esclave dans les paramètres de base via la sélection du maître.
  Si besoin, vous pouvez insérer des fonctions de filtrage ou de mise à l'échelle spécifiques, par ex. pour adapter la plage de la valeur de réglage du maître à la plage de la consigne/mesure de l'esclave.
  Dans l'exemple, OutputHeat de PID_Temp_1 est affecté à Setpoint de PID_Temp_2.
- Interconnexion de l'interface pour l'échange d'informations entre maître et esclave
  Le paramètre "Slave" d'un maître doit être affecté, pour tous ses esclaves directement subordonnés (qui reçoivent leur consigne de ce maître), aux paramètres "Master" de ces derniers. Pour permettre l'interconnexion d'un maître avec plusieurs esclaves et l'affichage de l'interconnexion dans la fenêtre d'inspection de l'esclave dans les paramètres de base, l'affectation doit être réalisée via l'interface de l'esclave.
  Vous pouvez effectuer cette interconnexion manuellement dans l'éditeur de programme ou la faire effectuer automatiquement dans la fenêtre d'inspection de l'esclave dans les paramètres de base via la sélection du maître.
  Ce n'est que si cette interconnexion est effectuée que la fonctionnalité Anti-Wind-Up et l'évaluation des modes de fonctionnement de l'esclave peuvent correctement fonctionner chez le maître.
  Dans l'exemple, le paramètre "Slave" de PID_Temp_1 est affecté au paramètre "Master" de PID_Temp_2.

Code de programme de l'exemple dans le langage SCL (sans affectation de la valeur de réglage de l'esclave à l'actionneur) :

```
"PID_Temp_1"(Input:="TempChocolate");
"PID_Temp_2"(Input:="TempWater", Master := "PID_Temp_1".Slave,
Setpoint := "PID_Temp_1".OutputHeat);
```

Voir aussi

Variable ActivateRecoverMode de PID_Temp (Page 414)

7.4.3 Configuration

Vous pouvez effectuer la configuration via votre programme utilisateur, l'éditeur de configuration ou la fenêtre d'inspection de l'appel de PID_Temp.
Veillez, lors de l'utilisation de PID_Temp dans une régulation en cascade, à la configuration correcte des paramétrages indiqués ci-après.
Si une instance de PID_Temp reçoit sa consigne d'un maître de niveau supérieur et transmet sa valeur de réglage à un esclave de niveau inférieur, cette instance de PID_Temp est simultanément maître et esclave. Pour une instance de PID_Temp de ce type, il faut effectuer les deux configurations figurant ci-dessous. C'est par exemple le cas pour l'instance de PID_Temp centrale d'une régulation en cascade avec trois grandeurs de mesure enchaînées et trois instances de PID_Temp.

Configuration d'un maître

Paramétrage dans l'éditeur de configuration ou la fenêtre d'inspection	Paramètre DB	Description
Paramètres de base → Cascade : Cocher la case "Le régulateur est maître"	Config.Cascade.IsMaster = TRUE	Active ce régulateur comme maître dans une cascade
Paramètres de base → Cascade : Nombre d'esclaves	Config.Cascade.CountSlaves	Nombre d'esclaves directement asservis recevant leur consigne de ce maître.
Paramètres de base → Paramètres d'entrée/de sortie : Sélection de la valeur de réglage (chauffage) = OutputHeat	Config.Output.Heat.Select = 0	Le maître n'utilise que le paramètre de sortie OutputHeat. OutputHeat_PWM et OutputHeat_PER sont désactivés.
Paramètres de base → Paramètres d'entrée/de sortie : Décocher la case "Activer refroidissement"	Config.ActivateCooling = FALSE	Chez un maître, le refroidissement doit être désactivé.
Paramètres de sortie → Limites et mise à l'échelle de la valeur de réglage → OutputHeat / OutputCool : limite inférieure de la valeur de réglage PID (chauffage), limite supérieure de la valeur de réglage PID (chauffage), valeur de réglage inférieure mise à l'échelle (chauffage), valeur de réglage supérieure mise à l'échelle (chauffage)	Config.Output.Heat.PidLowerLimit, Config.Output.Heat.PidUpperLimit, Config.Output.Heat.LowerScaling, Config.Output.Heat.UpperScaling	Si aucune fonction de mise à l'échelle spécifique n'est utilisée lors de l'affectation de OutputHeat du maître au Setpoint de l'esclave, il peut être nécessaire d'adapter les limites et la mise à l'échelle de la valeur de réglage du maître à la plage de la consigne/mesure de l'esclave.

Paramétrage dans l'éditeur de configuration ou la fenêtre d'inspection	Paramètre DB	Description
Cette variable n'est pas disponible dans la fenêtre d'inspection ou dans la vue fonctionnelle de l'éditeur de configuration. Vous pouvez les modifier via la vue des paramètres de l'éditeur de configuration.	Config.Cascade.AntiWindUp-Mode	Le mode Anti-Wind-Up définit la façon dont l'action I de ce maître est traitée si des esclaves directement asservis atteignent les limites de leur valeur de réglage. Sont possibles : • AntiWindUpMode = 0: La fonctionnalité AntiWindUp est désactivée. Le maître ne réagit pas à la limitation de ses esclaves. • AntiWindUpMode = 1 (par défaut) : L'action I du maître est réduite selon le rapport "Esclaves limités / nombre d'esclaves". Cela diminue les effets de la limitation sur le comportement de régulation. • AntiWindUpMode = 2: L'action I du maître s'arrête dès qu'un esclave subit une limitation.

Configuration d'un esclave

Paramétrage dans l'éditeur de configuration ou la fenêtre d'inspection	Paramètre DB	Description
Paramètres de base → Cascade : Cocher la case "Le régulateur est esclave"	Config.Cascade.IsSlave = TRUE	Active ce régulateur comme esclave dans une cascade

7.4.4 Mise en service

Après la compilation et le chargement du programme, vous pouvez démarrer la mise en service de la régulation en cascade.

Pour la mise en service (exécution d'une optimisation ou passage en mode automatique avec les paramètres PID actuels), commencez par l'esclave le plus intérieur et continuez vers l'extérieur, jusqu'à atteindre le maître le plus extérieur.

Dans l'exemple ci-dessus, la mise en service démarre avec PID_Temp_2 et est ensuite poursuivie avec PID_Temp_1.

Optimisation de l'esclave

L'optimisation de PID_Temp requiert une consigne constante. Activez donc la consigne de remplacement d'un esclave pour son optimisation (variables SubstituteSetpoint et SubstituteSetpointOn) ou faites passer le maître correspondant en mode manuel avec la valeur manuelle correspondante. De cette manière, vous garantissez que la consigne de l'esclave restera constante pendant l'optimisation.

Optimisation du maître

Pour qu'un maître puisse influer sur le processus ou effectuer une optimisation, tous les esclaves suivants doivent se trouver en mode automatique et avoir désactivé la consigne de remplacement. Via l'interface d'échange d'informations entre maître et esclave (paramètre Master et paramètre Slave), un maître évalue ces conditions et affiche l'état actuel dans les variables AllSlaveAutomaticState et NoSlaveSubstituteSetpoint. Des alarmes d'état sont émises en conséquence dans l'éditeur de mise en service.

Alarme d'état dans l'éditeur de mise en service du maître	**Paramètre DB du maître**	**Solution**
Un ou plusieurs esclaves ne sont pas en mode automatique.	AllSlaveAutomaticState = FALSE, NoSlaveSubstituteSetpoint = TRUE	Effectuez d'abord la mise en service de tous les esclaves suivants. Assurez-vous que les conditions suivantes soient remplies avant d'effectuer une optimisation ou d'activer le mode manuel ou automatique du maître : • Tous les esclaves suivants se trouvent en mode automatique (State = 3). • La consigne de remplacement est désactivée chez tous les esclaves suivants (SubstituteSetpointOn = FALSE).
La consigne de remplacement est activée chez un ou plusieurs esclaves.	AllSlaveAutomaticState = TRUE, NoSlaveSubstituteSetpoint = FALSE	
Un ou plusieurs esclaves ne sont pas en mode automatique et ont activé la consigne de remplacement.	AllSlaveAutomaticState = FALSE, NoSlaveSubstituteSetpoint = FALSE	

Si l'optimisation préalable ou l'optimisation fine a été démarrée pour un maître, PID_Temp abandonne l'optimisation dans les cas suivants et affiche une erreur avec ErrorBits = DW#16#0200000 :

- un ou plusieurs esclaves ne sont pas en mode automatique (AllSlaveAutomaticState = FALSE)
- un ou plusieurs esclaves ont activé la consigne de remplacement (NoSlaveSubstituteSetpoint = FALSE).

Le changement de mode de fonctionnement suivant dépend de ActivateRecoverMode.

7.4.5 Consigne de remplacement

Pour la transmission d'une consigne, PID_Temp propose, en plus du paramètre Setpoint, une consigne de remplacement dans la variable SubstituteSetpoint. Cette consigne de remplacement peut être activée avec SubstituteSetpointOn = TRUE ou en cochant la case appropriée dans l'éditeur de mise en service.

La consigne de remplacement vous permet de transmettre temporairement la consigne directement à l'esclave, par ex. pour la mise en service ou l'optimisation.

Pour cela, il faut que l'interconnexion de la valeur de réglage du maître, requise pour le fonctionnement normal de la régulation en cascade à la consigne de l'esclave ne soit pas modifiée dans le programme.

Pour qu'un maître puisse influer sur le processus ou effectuer une optimisation, tous les esclaves suivants doivent avoir désactivé la consigne de remplacement.

Vous pouvez voir la consigne actuellement opérante, telle qu'elle est utilisée par l'algorithme PID pour le calcul, au niveau de la variable CurrentSetpoint.

7.4.6 Modes de fonctionnement et réaction en cas d'erreur

Le maître ou l'esclave d'une instance PID_Temp ne modifie pas le mode de fonctionnement de cette instance PID_Temp.

Si une erreur survient chez un de ses esclaves, le maître reste dans son mode de fonctionnement actuel.

Si une erreur survient chez son maître, l'esclave reste dans son mode de fonctionnement actuel. Toutefois, le fonctionnement ultérieur de l'esclave dépend alors de l'erreur et la réaction configurée en cas d'erreur du maître, car la valeur de réglage du maître est utilisée comme consigne de l'esclave :

- Si la réaction ActivateRecoverMode = TRUE est configurée chez le maître et que l'erreur n'empêche pas le calcul de OutputHeat, l'erreur n'a pas d'impact sur l'esclave.
- Si la réaction ActivateRecoverMode = TRUE est configurée chez le maître et que l'erreur empêche le calcul de OutputHeat, le maître fournit la dernière valeur de réglage valide ou la valeur de réglage de remplacement configurée SubstituteOutput selon SetSubstituteOutput. Cette valeur est alors utilisée comme consigne par l'esclave.

  PID_Temp est préréglé de telle façon que, dans ce cas, la valeur de réglage de remplacement 0.0 soit fournie (ActivateRecoverMode = TRUE, SetSubstituteOutput = TRUE, SubstituteOutput = 0.0). Configurez une valeur de réglage de remplacement adéquate pour votre application ou activez l'utilisation de la dernière valeur de réglage PID valide (SetSubstituteOutput = FALSE).
- Si ActivateRecoverMode = FALSE est configuré chez le maître, ce dernier passe en cas d'erreur en mode "Inactif" et fournit OutputHeat = 0.0. L'esclave utilise alors 0.0 comme consigne.

Vous trouverez la réaction en cas d'erreur dans les paramètres de sortie dans l'éditeur de configuration.

7.5 Réglage multi-zones avec PID_Temp

Introduction

Lors d'un réglage multi-zones, plusieurs parties, dites zones, d'une installation sont régulées simultanément selon différentes températures. L'influence mutuelle des zones de température par couplage thermique est caractéristique du réglage multi-zones, c'est-à-dire que la mesure d'une zone peut influer sur la mesure d'une autre zone par couplage thermique. L'intensité avec laquelle s'exerce cette influence dépend de la structure de l'installation et des points de fonctionnement choisis des zones.

Exemple : installation d'extrusion utilisée, entres autres, dans la transformation des matières plastiques.

Le mélange de matières qui traverse l'extrudeuse doit être régulé selon différentes températures pour une traitement optimal. Ainsi, il est possible que d'autres températures soient requises au point de remplissage de l'extrudeuse qu'à sa buse de sortie. Ce faisant, les différentes zones de températures influent les unes sur les autres par couplage thermique.

Lors de l'utilisation de PID_Temp dans des réglages multi-zones, chaque zone de température est régulée par une instance de PID_Temp qui lui est propre.

Veuillez tenir compte des explications suivantes si vous utilisez PID_Temp dans un réglage multi-zones.

Optimisation préalable séparée du chauffage et du refroidissement

La première mise en service d'une installation commence, en règle générale, par l'exécution d'une optimisation préalable pour effectuer un premier réglage des paramètres PID et la régulation au point de fonctionnement. L'optimisation préalable pour les réglages multi-zones est souvent effectuée pour toutes les zones simultanément.

PID_Temp offre la possibilité d'effectuer l'optimisation préalable du chauffage et du refroidissement en une seule étape (Mode = 1, Heat.EnableTuning = TRUE, Cool.EnableTuning = TRUE) pour les régulateurs pour lesquels le refroidissement est activé et la commutation de paramètres PID est activée comme méthode de chauffage/refroidissement (Config.ActivateCooling = TRUE, Config.AdvancedCooling = TRUE).

Il est toutefois recommandé de ne pas utiliser cette optimisation pour l'optimisation préalable simultanée de plusieurs instances de PID_Temp dans un réglage multi-zones. A la place, exécutez séparément l'optimisation préalable du chauffage (Mode = 1, Heat.EnableTuning = TRUE, Cool.EnableTuning = FALSE) et l'optimisation préalable du refroidissement (Mode = 1, Heat.EnableTuning = FALSE, Cool.EnableTuning = TRUE).

L'optimisation préalable du refroidissement ne doit être démarrée que lorsque l'optimisation préalable du chauffage est terminée pour toutes les zones et que celles-ci ont atteint leur point de fonctionnement.

Cela permet de réduire les influences réciproques par couplages thermiques entre les zones pendant l'optimisation.

Adaptation du délai

Si PID_Temp est utilisé dans un réglage multi-zones avec des couplages thermiques forts entre les zones, vous devez vous assurer que l'adaptation du délai est désactivée pour l'optimisation préalable avec PIDSelfTune.SUT.AdaptDelayTime = 0. En cas contraire, la détermination du délai peut être faussée si le refroidissement de cette zone est empêché par un flux de chaleur arrivant d'autres zones pendant l'adaptation du délai (le chauffage est désactivé pendant cette phase).

Désactivation temporaire du refroidissement

PID_Temp offre la possibilité, pour les régulateurs avec refroidissement activé (Config.ActivateCooling = TRUE), de désactiver temporairement le refroidissement en mode automatique en réglant DisableCooling = TRUE.

Vous pouvez ainsi empêcher pendant la mise en service que ce régulateur refroidisse en mode automatique tandis que les régulateurs d'autres zones n'ont pas terminé l'optimisation du chauffage. En cas contraire, l'optimisation peut être influencée négativement du fait du couplage thermique entre les zones.

Marche à suivre

Vous pouvez procéder comme suit pour la mise en service de réglages multi-zones avec des couplages thermiques importants :

1. Réglez DisableCooling = TRUE pour tous les régulateurs avec refroidissement activé.
2. Réglez PIDSelfTune.SUT.AdaptDelayTime = 0 pour tous les régulateurs.
3. Spécifiez les consignes souhaitées (paramètre Setpoint) et démarrez l'optimisation du chauffage simultanément pour tous les régulateurs (Mode = 1, Heat.EnableTuning = TRUE, Cool.EnableTuning = FALSE).
4. Attendez que tous les régulateurs aient terminé l'optimisation préalable du chauffage.
5. Réglez DisableCooling = FALSE pour tous les régulateurs avec refroidissement activé.
6. Attendez que les mesures de toutes les zones soient stabilisées et proches de leur consigne respective.
   Si la consigne ne peut pas être atteinte durablement pour une zone, la conception de l'actionneur de chauffage ou de refroidissement est trop faible.
7. Démarrez, pour tous les régulateurs avec refroidissement activé, l'optimisation préalable du refroidissement (Mode = 1, Heat.EnableTuning = FALSE, Cool.EnableTuning = TRUE).

REMARQUE

Dépassement de valeur limite de la mesure

Si le refroidissement est désactivé en mode automatique avec DisableCooling = TRUE, il se peut que la mesure dépasse la consigne et les limites de la mesure tant que DisableCooling = TRUE. Observez les mesures et intervenez éventuellement si vous utilisez DisableCooling.

REMARQUE

Réglages multi-zones

Lors de réglages multi-zones, les couplages thermiques entre les zones peuvent entraîner des suroscillations plus fortes, un dépassement durable ou temporaire de valeurs limites et des écarts de régulation durables ou temporaires durant la mise en service et le fonctionnement. Observez les mesures et soyez prêt à intervenir. Selon l'installation, il peut être nécessaire de s'écarter de la marche à suivre décrite ci-dessus.

Synchronisation de plusieurs optimisations fines

Si l'optimisation fine est démarrée à partir du mode automatique avec PIDSelfTune.TIR.RunIn = FALSE, PID_Temp tente d'atteindre la consigne avec la régulation PID et les paramètres PID actuels. L'optimisation à proprement parler ne démarre que lorsque la consigne est atteinte. Le temps nécessaire pour atteindre la consigne peut être différent pour les différentes zones d'un réglage multi-zones.

Si vous voulez effectuer l'optimisation fine pour plusieurs zones simultanément, PID_Temp offre la possibilité de les synchroniser. Un temps d'attente est pour cela respecté après que la consigne est atteinte avant de poursuivre les autres étapes d'optimisation.

Marche à suivre

De cette façon, vous pouvez garantir que tous les régulateurs ont atteint leur consigne avant que les étapes d'optimisation soient démarrées. Cela permet de réduire les influences mutuelles par couplages thermiques entre les zones pendant l'optimisation.
Procédez comme suit pour les régulateurs pour les zones desquels vous voulez effectuer l'optimisation fine simultanément :

1. Réglez PIDSelfTune.TIR.WaitForControlIn = TRUE pour tous les régulateurs.
   Ces régulateurs doivent se trouver en mode automatique avec PIDSelfTune.TIR.RunIn = FALSE.
2. Spécifiez les consignes souhaitées (paramètre Setpoint) et démarrez l'optimisation fine pour tous les régulateurs.
3. Attendez que PIDSelfTune.TIR.ControlInReady = TRUE soit réglé pour tous les régulateurs.
4. Réglez PIDSelfTune.TIR.FinishControlIn = TRUE pour tous les régulateurs.

Ainsi, tous les régulateurs démarrent simultanément l'optimisation à proprement parler.

7.6 Régulation en mode alternatif avec PID_Temp

Régulation en mode alternatif

Dans la régulation en mode alternatif, deux ou plusieurs régulateurs agissent sur un seul actionneur commun. À chaque instant, un seul régulateur accède à l'actionneur et agit sur le processus.
Une logique décide du régulateur qui a accès à l'actionneur. Ce choix se base généralement sur une comparaison des valeurs de réglage de tous les régulateurs ; avec une sélection MAX, par ex., c'est le régulateur ayant la valeur de réglage maximale qui accède à l'actionneur.
Le choix basé sur la valeur de réglage requiert que tous les régulateurs fonctionnent en mode automatique. Les régulateurs qui n'agissent pas sur l'actionneur sont alignés (poursuite). Cela est nécessaire pour éviter les effets de windup et leur influence négative sur le comportement de régulation et sur la commutation entre les régulateurs.
PID_Temp prend en charge la régulation en mode alternatif à partir de la version 1.1 ; à cet effet, le régulateur offre un procédé simple pour aligner les régulateurs qui ne sont pas actifs : Les variables OverwriteInitialOutputValue et PIDCtrl.PIDInit vous permettent de configurer par défaut l'action I du régulateur en mode automatique comme si l'algorithme PID avait calculé dans le dernier cycle pour la valeur de réglage PID PidOutputSum = OverwriteInititalOutputValue. Pour ce faire, OverwriteInitialOutputValue est connecté à la valeur de réglage PID du régulateur qui a actuellement accès à l'actionneur. En mettant le bit PIDCtrl.PIDInit à 1, vous activez le paramétrage par défaut de l'action I ainsi que le redémarrage du cycle du régulateur et de la période de modulation de largeur d'impulsion. Le calcul de la valeur de réglage PID dans le cycle actuel est ensuite effectué sur la base de l'action I paramétrée par défaut (et alignée pour tous les régulateurs) ainsi que de l'action P et de l'action I à partir du signal d'écart actuel. L'action D n'est pas active pendant l'appel avec PIDCtrl.PIDInit = TRUE et ne contribue donc pas à la valeur de réglage.
Ce procédé garantit que le calcul de la valeur de réglage PID actuelle et donc le choix du régulateur agissant sur l'actionneur n'est effectué que sur la base de l'état actuel du processus et des paramètres PI. On évite de cette manière les effets de windup sur les régulateurs non actifs, de même que les décisions incorrectes de la logique de commutation.

Condition

- PIDCtrl.PIDInit n'est effectif que si l'action I est activée (variables Retain.CtrlParams.Heat.Ti et Retain.CtrlParams.Cool.Ti > 0.0).
- Vous devez affecter vous-même des valeurs à PIDCtrl.PIDInit et OverwriteInitialOutputValue dans votre programme utilisateur (voir l'exemple ci-dessous). PID_Temp n'effectue aucune modification automatique de ces variables.
- PIDCtrl.PIDInit n'a d'effet que si PID_Temp est en mode automatique (paramètre State = 3).
- Si possible, choisissez une période d'échantillonnage de l'algorithme PID (variables Retain.CtrlParams.Heat.Cycle et Retain.CtrlParams.Cool.Cycle) identique pour tous les régulateurs et appelez tous les régulateurs dans le même OB d'alarme cyclique. Vous garantissez ainsi que la commutation n'a pas lieu au sein d'un cycle du régulateur ou d'une période de modulation de largeur d'impulsion.

REMARQUE

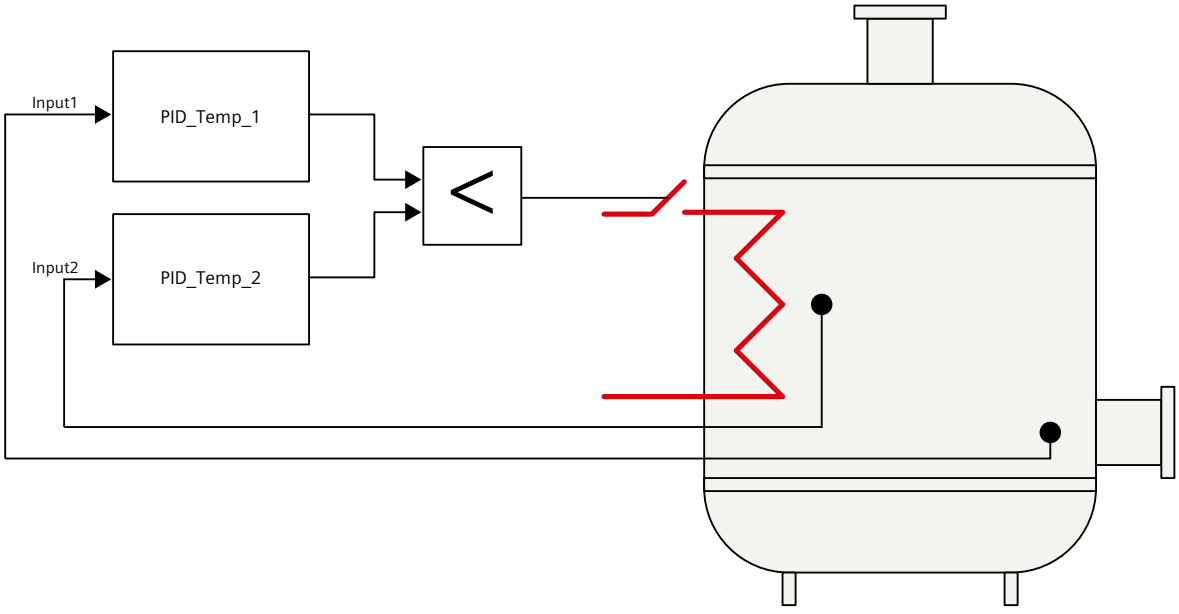
Adaptation continue des limites de valeur de réglage

Au lieu de l'alignement actif des régulateurs sans action sur l'actionneur décrit ici, d'autres systèmes de régulation proposent une solution consistant à adapter en continu les limites de valeur de réglage.

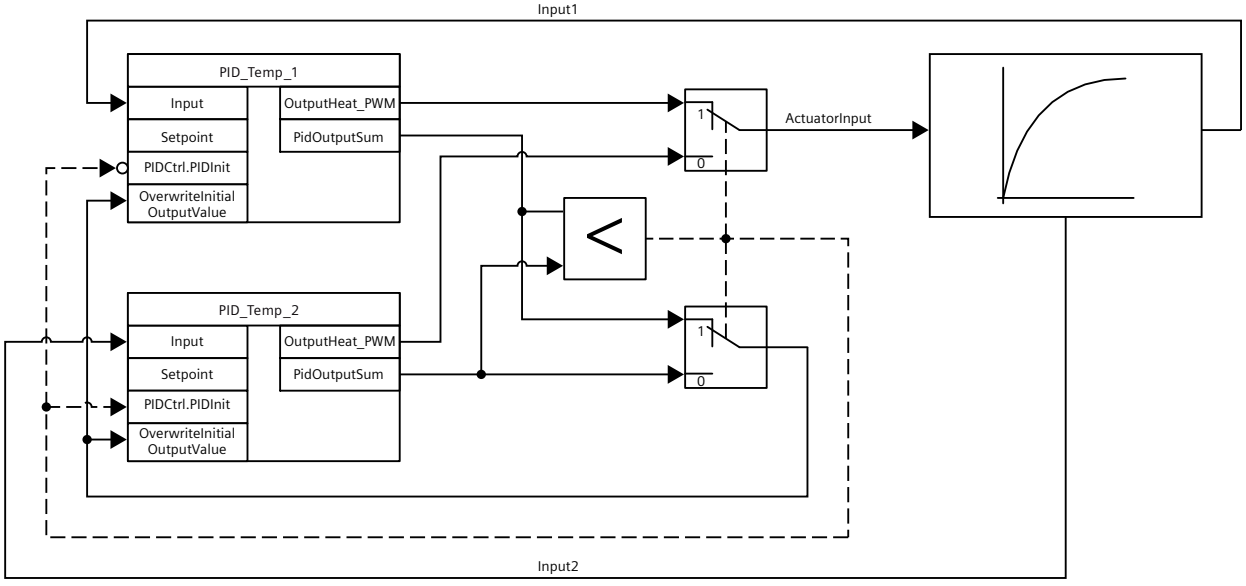
Cela n'est pas possible avec PID_Temp, car une modification des limites de valeur de réglage n'est pas prise en charge en mode automatique.

Exemple : Régulation d'une grande chaudière

PID_Temp est utilisé pour la régulation d'une grande chaudière.
L'objectif principal est la régulation de la température Input1. Le régulateur PID_Temp_1 est utilisé pour ce faire. En outre, le régulateur limiteur PID_Temp_2 doit maintenir la température Input2 d'un point de mesure supplémentaire en dessous d'une limite supérieure.
Les deux températures sont modifiées à l'aide d'un seul chauffage. La valeur de réglage du régulateur correspond à la puissance de chauffage.



Le chauffage est commandé via la valeur de réglage à largeur d'impulsions modulée de PID_Temp (paramètre OutputHeat_PWM) en écrivant la variable de programme ActuatorInput. La consigne de température Input1 est transmise sur le paramètre PID_Temp_1.Setpoint. La limite supérieure de température pour le point de mesure supplémentaire est transmise comme consigne sur le paramètre PID_Temp_2.Setpoint.



Les deux régulateurs ne disposent que d'un chauffage comme actionneur commun. La logique de choix du régulateur agissant sur l'actionneur est réalisée ici par un sélecteur MIN de la valeur de réglage PID (au format Real, paramètre PidOutputSum). Comme la valeur de réglage PID correspond à la puissance calorifique, le régulateur qui demande la plus petite puissance calorifique est le régulateur actif.

En mode normal de l'installation, la valeur de mesure de la grandeur réglée principale correspond à la consigne. Le régulateur principal PID_Temp_1 s'est stabilisé à une valeur de réglage PID stationnaire PID_Temp_1.PidOutputSum. En fonctionnement normal, la mesure du régulateur limiteur Input2 est largement inférieure à la valeur limite supérieure qui est prescrite comme valeur de consigne pour PID_Temp_2. C'est pourquoi le régulateur limiteur va vouloir augmenter la puissance calorifique en vue d'augmenter sa mesure, il va donc calculer une valeur de réglage PID PID_Temp_2.PidOutputSum supérieure à celle du régulateur principal PID_Temp_1.PidOutputSum. Le sélecteur MIN de la logique de commutation laisse par conséquent le régulateur principal PID_Temp_1 agir sur l'actionneur. En outre, les affectations PID_Temp_2.OverwriteInitialOutputValue = PID_Temp_1.PidOutputSum et PID_Temp_2.PIDCtrl.PIDInit = TRUE garantissent que PID_Temp_2 soit aligné.

Si Input2 se rapproche de la valeur limite supérieure ou la dépasse, par exemple à la suite d'un défaut, le régulateur limiteur PID_Temp_2 va calculer une valeur de réglage PID inférieure afin de diminuer la puissance calorifique et de réduire ainsi Input2. Si PID_Temp_2.PidOutputSum est inférieur à PID_Temp_1.PidOutputSum, le régulateur limiteur PID_Temp_2 se voit attribuer l'accès à l'actionneur par le sélecteur MIN, et réduit la puissance de chauffage. En outre, les affectations PID_Temp_1.OverwriteInitialOutputValue = PID_Temp_2.PidOutputSum et PID_Temp_1.PIDCtrl.PIDInit = TRUE garantissent que PID_Temp_1 soit aligné.

La température au point de mesure supplémentaire Input2 baisse. La température de la grandeur réglée principale Input1 baisse également et ne peut plus être maintenue à la valeur de consigne.

Une fois le défaut éliminé, Input2 va encore diminuer et la puissance de chauffage sera encore augmentée par le régulateur limiteur. Lorsque le régulateur principal calcule une puissance de chauffage plus faible comme valeur de réglage, l'installation retourne en mode de fonctionnement normal, si bien que le régulateur principal PID_Temp_1 reprend l'accès à l'actionneur. Cet exemple est réalisable à l'aide du code programme SCL suivant :

```
"PID_Temp_1"(Input := "Input1");
"PID_Temp_2"(Input := "Input2");
IF "PID_Temp_1".PidOutputSum <= "PID_Temp_2".PidOutputSum THEN
  "ActuatorInput" := "PID_Temp_1".OutputHeat_PWM;
  "PID_Temp_1".PIDCtrl.PIDInit := FALSE;
  "PID_Temp_2".PIDCtrl.PIDInit := TRUE;
  "PID_Temp_2".OverwriteInitialOutputValue := "PID_Temp_1".PidOutputSum;
ELSE
  "ActuatorInput" := "PID_Temp_2".OutputHeat_PWM;
  "PID_Temp_1".PIDCtrl.PIDInit := TRUE;
  "PID_Temp_2".PIDCtrl.PIDInit := FALSE;
  "PID_Temp_1".OverwriteInitialOutputValue := "PID_Temp_2".PidOutputSum;
END_IF;
```

7.7 Simuler PID_Temp avec PLCSIM

REMARQUE

Simulation avec PLCSIM

La simulation de PID_Temp avec PLCSIM n'est pas prise en charge pour la CPU S7-1200.

PID_Temp peut être simulé avec PLCSIM uniquement pour la CPU S7-1500.

Lors de la simulation avec PLCSIM, le comportement dans le temps de l'API simulé n'est pas exactement le même que celui d'un API "réel". Le temps de cycle effectif d'un OB d'alarme cyclique peut présenter de plus grandes fluctuations pour un API simulé que pour un API "réel".

Dans la configuration standard, PID_Temp calcule automatiquement le temps entre les appels et le surveille vis-à-vis des fluctuations.

Lors de la simulation de PID_Temp avec PLCSIM, il est par conséquent possible de détecter une erreur de période d'échantillonnage (ErrorBits = DW#16#00000800).

Cela entraîne l'interruption d'optimisations en cours.

La réaction en mode automatique dépend de la valeur de la variable ActivateRecoverMode.

Pour éviter cela, vous devez configurer PID_Temp de la manière suivante lors de la simulation avec PLCSIM :

- CycleTime.EnEstimation = FALSE
- CycleTime.EnMonitoring = FALSE
- CycleTime.Value : affectez à cette variable le temps de cycle de l'OB d'alarme cyclique d'appel en secondes.

Utiliser les fonctions de base PID

8

8.1 CONT_C

8.1.1 Objet technologique CONT_C

L'objet technologique CONT_C met à disposition un régulateur PID continu pour le mode automatique et le mode manuel. Il correspond au DB d'instance de l'instruction CONT_C. L'instruction PULSEGEN permet de configurer un régulateur à impulsions.

Les actions proportionnelle, par intégration (INT) et par dérivation (DIF) sont montées en parallèle et peuvent être activées et désactivées individuellement. Les régulateurs P, I, PI, PD et PID peuvent ainsi être paramétrés.

S7-1500

Toutes les variables et tous les paramètres de l'objet technologique sont rémanents et peuvent être modifiés uniquement lors du chargement dans l'appareil si vous avez chargé complètement CONT_C.

Voir aussi

Présentation des régulateurs de logiciel (Page 43)

Ajouter des objets technologiques (Page 45)

Configurer les objets technologiques (Page 46)

Charger des objets technologiques dans l'appareil (Page 48)

CONT_C (Page 420)

8.1.2 Configurer le signal d'écart CONT_C

Utiliser la mesure de périphérie

Pour utiliser la mesure au format de périphérie au niveau du paramètre d'entrée PV_PER, procédez de la manière suivante :

1. Cochez la case "Activer la périphérie".
2. Lorsque la mesure existe sous forme de grandeur physique, enregistrez le facteur et le décalage pour la normalisation en pourcentage.
   La mesure se calcule ensuite à l'aide de la formule suivante :
   PV = PV_PER × PV_FAC + PV_OFF

Utiliser la mesure interne

Pour utiliser la mesure en format à virgule flottante au niveau du paramètre d'entrée PV_IN, procédez de la manière suivante :

1. Décochez la case "Activer la périphérie".

Signal d'écart

Dans les conditions suivantes, vous réglez une largeur de zone morte :

- Le signal de la mesure est brouillé.
- Le gain du régulateur est élevé.
- L'action D est activée.

La composante de bruit de la mesure occasionne dans ce cas de fortes variations de la valeur de réglage. La zone morte réduit le taux de bruit lorsque le régulateur est à l'état stationnaire. La largeur de la zone morte indique la taille de la zone morte. Lorsque la largeur de la zone morte est 0.0, la zone morte est désactivée.

Voir aussi

Fonctionnement de CONT_C (Page 421)

8.1.3 Configurer l'algorithme de régulation CONT_C

Généralités

Pour déterminer les composantes actives de l'algorithme, procédez de la manière suivante :

1. Sélectionnez une entrée dans la liste "Structure du régulateur".
   Vous pouvez uniquement saisir les paramètres nécessaires de la structure de régulateur sélectionnée.

Action P

1. Lorsque la structure du régulateur contient une action P, saisissez le "Gain proportionnel".

Action I

1. Lorsque la structure du régulateur contient une action I, saisissez le temps d'intégration.
2. Pour affecter une valeur d'initialisation à l'action I, cochez la case "Initialisation de l'action I" et saisissez la valeur d'initialisation.
3. Pour régler de manière permanente l'action I sur cette valeur d'initialisation, cochez la case "Suspendre l'action I".

Action D

1. Lorsque la structure du régulateur contient une action D, saisissez le temps de dérivation, la pondération de l'action D et le temps de retard.

Voir aussi

Fonctionnement de CONT_C (Page 421)

8.1.4 Configurer la valeur de réglage CONT_C

Généralités

Vous pouvez régler CONT_C en mode manuel ou automatique.

1. Pour saisir une valeur de réglage manuelle, cochez la case "Activer le mode manuel".
   Vous pouvez saisir une valeur de réglage manuelle au niveau du paramètre d'entrée MAN.

Limites de valeur de réglage

La valeur de réglage est limitée en haut et en bas de sorte qu'elle puisse uniquement accepter des valeurs valides. Vous ne pouvez pas désactiver la limitation. Le dépassement des limites est affiché par les paramètres de sortie QLMN_HLM et QLMN_LLM.

1. Saisissez une valeur pour la limite de valeur de réglage supérieure et la limite de valeur de réglage inférieure.
   Lorsque la valeur de réglage est une grandeur physique, les unités des limites de valeur de réglage supérieure et inférieure doivent correspondre.

Normalisation

La valeur de réglage peut être normalisée à partir d'un facteur et d'un décalage, pour la transmission en tant que valeur à virgule flottante et valeur périphérique, avec la formule suivante.

Valeur de réglage normalisée = valeur de réglage x facteur + décalage

Un facteur de 1.0 et un décalage de 0.0 sont préréglés.

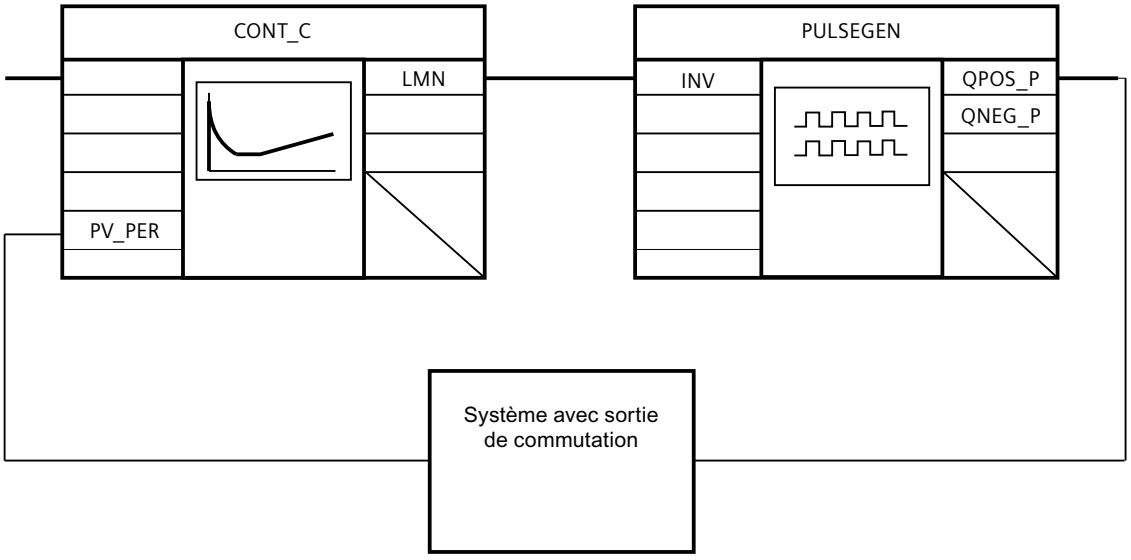
1. Saisissez une valeur de facteur et une valeur de décalage.

Voir aussi

Fonctionnement de CONT_C (Page 421)

8.1.5 Programmer le régulateur à impulsions

Le régulateur continu CONT_C et le conformateur d'impulsions PULSEGEN permettent de réaliser un régulateur à consigne fixe avec sortie commutante pour actionneurs proportionnels. L'image ci-dessous montre l'évolution du signal de la boucle de régulation.



Le régulateur continu CONT_C forme la valeur de réglage LMN, que le conformateur d'impulsions PULSEGEN transforme en signaux impulsion-pause QPOS_P et QNEG_P.

Voir aussi

PULSEGEN (Page 430)

8.1.6 Mise en service de CONT_C

Conditions

- L'instruction et l'objet technologique sont chargés sur la CPU.

Marche à suivre

Pour déterminer manuellement les paramètres PID optimaux, procédez de la manière suivante :

1. Cliquez sur l'icône "Start".
   Si aucune connexion en ligne n'existe encore, une connexion est établie. Les valeurs actuelles de la consigne, de la mesure et de la valeur de réglage sont enregistrées.
2. Saisissez de nouveaux paramètres PID dans les champs "P", "I", "D" et "Temps de retard".
3. Dans le groupe "Optimisation", cliquez sur l'icône "Envoyer les paramètres à la CPU".
4. Dans le groupe "Valeurs actuelles", cochez la case "Spécifier la consigne".
5. Saisissez une nouvelle consigne et cliquez sur l'icône, dans le groupe "Valeurs actuelles".

6. Le cas échéant, décochez la case "Mode manuel".
   Le régulateur fonctionne avec les nouveaux paramètres PID et régule en fonction de la nouvelle consigne.
7. Contrôlez la qualité des paramètres PID en observant l'allure des courbes.
8. Répétez les étapes 2 à 6 jusqu'à ce que vous soyez satisfait du résultat du régulateur.

8.2 CONT_S

8.2.1 Objet technologique CONT_S

L'objet technologique CONT_S fournit un régulateur pas à pas pour un actionneur à comportement intégral et sert à régler les processus thermiques techniques avec des signaux de sortie TOR de la valeur de réglage. L'objet technologique correspond au DB de données d'instance de l'instruction CONT_S. Son fonctionnement est basé sur l'algorithme de régulation PI du régulateur à échantillonnage. Ce régulateur pas à pas fonctionne sans signalisation de position. Les modes de fonctionnement manuel et automatique sont possibles.
S7-1500
Toutes les variables et tous les paramètres de l'objet technologique sont rémanents et peuvent être modifiés uniquement lors du chargement dans l'appareil si vous avez chargé complètement CONT_S.

Voir aussi

Présentation des régulateurs de logiciel (Page 43)
Ajouter des objets technologiques (Page 45)
Configurer les objets technologiques (Page 46)
Charger des objets technologiques dans l'appareil (Page 48)
CONT_S (Page 425)

8.2.2 Configurer le signal d'écart CONT_S

Utiliser la mesure de périphérie

Pour utiliser la mesure au format de périphérie au niveau du paramètre d'entrée PV_PER, procédez de la manière suivante :

1. Cochez la case "Activer la périphérie".
2. Lorsque la mesure existe sous forme de grandeur physique, enregistrez le facteur et le décalage pour la normalisation en pourcentage.
   La mesure se calcule ensuite à l'aide de la formule suivante :
   PV = PV_PER × PV_FAC + PV_OFF

Utiliser la mesure interne

Pour utiliser la mesure en format à virgule flottante au niveau du paramètre d'entrée PV_IN, procédez de la manière suivante :

1. Décochez la case "Activer la périphérie".

Signal d'écart

Dans les conditions suivantes, vous réglez une largeur de zone morte :

- Le signal de la mesure est brouillé.
- Le gain du régulateur est élevé.
- L'action D est activée.

La part de bruit de la mesure occasionne dans ce cas de fortes variations de la valeur de réglage. La zone morte réduit le taux de bruit lorsque le régulateur est à l'état stationnaire. La largeur de la zone morte indique la taille de la zone morte. Lorsque la largeur de la zone morte est 0.0, la zone morte est désactivée.

Voir aussi

Fonctionnement CONT_S (Page 426)

8.2.3 Configurer l'algorithme de régulation CONT_S

Algorithme PI

1. Saisissez le "Gain proportionnel" pour l'action P.
2. Saisissez le temps d'intégration pour la temporisation de l'action I.
   Pour un temps d'intégration de 0.0, l'action I est désactivée.

Voir aussi

Fonctionnement CONT_S (Page 426)

8.2.4 Configurer la valeur de réglage CONT_S

Généralités

Vous pouvez régler CONT_S en mode manuel ou automatique.

1. Pour saisir une valeur de réglage manuelle, cochez la case "Activer le mode manuel".
   Saisissez une valeur de réglage manuelle pour les paramètres d'entrée LMNUP et LMNDN.

Générateur d'impulsions

1. Saisissez la durée minimale d'impulsion et la durée minimale de pause.
   Les valeurs doivent être supérieures ou égales au temps de cycle du paramètre d'entrée CYCLE. La fréquence de commutation en est réduite.
2. Saisissez le temps de positionnement du moteur.
   La valeur doit être supérieure ou égale au temps de cycle du paramètre d'entrée CYCLE.

Voir aussi

Fonctionnement CONT_S (Page 426)

8.2.5 Mise en service de CONT_S

Conditions

- L'instruction et l'objet technologique sont chargés sur la CPU.

Marche à suivre

Pour déterminer manuellement les paramètres PID optimaux, procédez de la manière suivante :

1. Cliquez sur l'icône "Start".
   Si aucune connexion en ligne n'existe encore, une connexion est établie. Les valeurs actuelles de la consigne, de la mesure et de la valeur de réglage sont enregistrées.
2. Saisissez un nouveau coefficient d'action proportionnelle et un nouveau temps d'intégration dans les champs "P" et "I".
3. Dans le groupe "Optimisation", cliquez sur l'icône "Envoyer les paramètres à la CPU".
4. Dans le groupe "Valeurs actuelles", cochez la case "Spécifier la consigne".
5. Saisissez une nouvelle consigne et cliquez sur l'icône, dans le groupe "Valeurs actuelles".
6. Le cas échéant, décochez la case "Mode manuel".
   Le régulateur fonctionne avec les nouveaux paramètres et régule en fonction de la nouvelle consigne.
7. Contrôlez la qualité des paramètres PID en observant l'allure des courbes.
8. Répétez les étapes 2 à 6 jusqu'à ce que vous soyez satisfait du résultat du régulateur.

8.3 TCONT_CP

8.3.1 Objet technologique TCONT_CP

L'objet technologique TCONT_CP met à disposition un régulateur de température continu avec générateur d'impulsions. Il correspond au DB d'instance de l'instruction TCONT_CP. Son fonctionnement est basé sur l'algorithme de régulation PID du régulateur à échantillonnage. Les modes de fonctionnement manuel et automatique sont possibles.

L'instruction TCONT_CP calcule les paramètres P, I et D de son système réglé de manière autonome pendant l'"optimisation préalable". Une optimisation supplémentaire des paramètres peut être réalisée par une "optimisation fine". Vous pouvez aussi déterminer les paramètres PID manuellement.

S7-1500

Toutes les variables et tous les paramètres de l'objet technologique sont rémanents et peuvent être modifiés uniquement lors du chargement dans l'appareil si vous avez chargé complètement TCONT_CP.

Voir aussi

Présentation des régulateurs de logiciel (Page 43)
Ajouter des objets technologiques (Page 45)
Configurer les objets technologiques (Page 46)
Charger des objets technologiques dans l'appareil (Page 48)
TCONT_CP (Page 439)

8.3.2 Configurer TCONT_CP

8.3.2.1 Signal d'écart

Utiliser la mesure de périphérie

Pour utiliser le paramètre d'entrée PV_PER, procédez de la manière suivante :

1. Sélectionnez l'entrée "Périphérie" dans la liste "Source".
2. Sélectionnez le "Type de capteur".
   Suivant le type de capteur, la mesure est normalisée selon différentes formules.
   - Standard
     Thermocouples ; PT100/NI100
     PV = 0.1 × PV_PER × PV_FAC + PV_OFFS
   - Climatique ;
     PT100/NI100
     PV = 0.01 × PV_PER × PV_FAC + PV_OFFS
   - Courant/tension
     PV = 100/27648 × PV_PER × PV_FAC + PV_OFFS
3. Enregistrez le facteur et le décalage pour la normalisation de la mesure de périphérie.

Utiliser la mesure interne

Pour utiliser le paramètre d'entrée PV_IN, procédez de la manière suivante :

1. Sélectionnez l'entrée "Interne" dans la liste "Source".

Signal d'écart

Dans les conditions suivantes, vous réglez une largeur de zone morte :

- Le signal de la mesure est brouillé.
- Le gain du régulateur est élevé.
- L'action D est activée.

La part de bruit de la mesure occasionne dans ce cas de fortes variations de la valeur de réglage. La zone morte réduit la part de bruit lorsque le régulateur est à l'état stationnaire. La largeur de la zone morte indique la taille de la zone morte. Lorsque la largeur de la zone morte est 0.0, la zone morte est désactivée.

Voir aussi

Fonctionnement TCONT_CP (Page 440)

8.3.2.2 Algorithme de régulation

Généralités

1. Saisissez le "Temps d'échantillonnage de l'algorithme PID".
   Le temps d'échantillonnage du régulateur ne doit pas dépasser 10 % du temps d'intégration du régulateur (TI).
2. Lorsque la structure du régulateur contient une action P, saisissez le "Gain proportionnel".
   Un gain proportionnel négatif inverse le sens de régulation.

Action P

En cas de modification de la consigne, des suroscillations peuvent se produire au niveau de l'action P. La pondération de l'action P permet de choisir le degré de l'effet de l'action P en cas de modifications de consigne. L'atténuation de l'action P s'obtient par compensation de l'action I.

1. Pour affaiblir l'action P lors de la modification des valeurs de consigne, vous entrez la "pondération de l'action P".
   - 1.0: Action P totalement opérante si modification de la consigne
   - 0.0: Action P non opérante si modification de la consigne

Action I

En cas de limitation de la valeur de réglage, l'action I est arrêtée. Elle est à nouveau activée lorsque le signal d'écart rapproche l'action par intégration I de la plage de réglage interne.

1. Lorsque la structure du régulateur contient une action I, saisissez le "Temps d'intégration". Pour un temps d'intégration de 0.0, l'action I est désactivée.
2. Pour affecter une valeur d'initialisation à l'action I, cochez la case "Initialisation de l'action I" et saisissez la "Valeur d'initialisation".
   Lors du redémarrage ou si COM_RST = TRUE, l'action I est mise à cette valeur.

Action D

1. Lorsque la structure du régulateur contient une action D, entrez le temps de dérivation (TD) et le coefficient DT1 (D_F).
   Lorsque l'action D est activée, l'équation suivante doit être respectée :
   TD = 0.5 × CYCLE× D_F.
   Le temps de retard est calculé à partir de là selon la formule :
   Temps de retard = TD/D_F

Paramétrer le régulateur PD avec le point de fonctionnement

1. Saisissez le temps d'intégration 0.0.
2. Cochez la case "Initialisation de l'action I".
3. Saisissez le point de fonctionnement comme valeur d'initialisation.

Paramétrer le régulateur P avec le point de fonctionnement

1. Paramétrez un régulateur PD avec le point de fonctionnement.
2. Saisissez le temps de dérivation 0.0.
   L'action D est désactivée.

Plage de régulation

La plage de régulation limite la plage de valeurs du signal d'écart. Lorsque le signal d'écart se trouve en dehors de la plage de valeurs, les limites des valeurs de réglage sont utilisées.

À l'entrée dans la plage de régulation, l'action D activée entraîne une réduction très rapide de la grandeur réglante. La plage de régulation n'est donc utile que lorsque l'action D est activée. Sans plage de régulation, seule l'action P diminuant entraînerait une réduction de la grandeur réglante. La plage de régulation conduit plus rapidement à un régime transitoire sans sur ou sous-oscillation lorsque la grandeur réglante minimale ou maximale fournie est très éloignée de la grandeur réglante stationnaire qui est requise pour le nouveau point de fonctionnement.

1. Cochez la case "Activer" dans le groupe "Plage de régulation".
2. Dans le champ de saisie "Largeur", saisissez la valeur dont la mesure pourra s'écarter de la consigne vers le haut et vers le bas.

Voir aussi

Fonctionnement TCONT_CP (Page 440)

8.3.2.3 Valeur de réglage du régulateur continu

Limites de valeur de réglage

La valeur de réglage est limitée en haut et en bas de sorte qu'elle puisse uniquement accepter des valeurs valides. Vous ne pouvez pas désactiver la limitation. Le dépassement des limites est affiché par les paramètres de sortie QLMN_HLM et QLMN_LLM.

1. Saisissez une valeur pour la limite de valeur de réglage supérieure et la limite de valeur de réglage inférieure.

Normalisation

La valeur de réglage peut être normalisée à partir d'un facteur et d'un décalage, pour la transmission en tant que valeur à virgule flottante et valeur périphérique, avec la formule suivante.

Valeur de réglage normalisée = valeur de réglage x facteur + décalage

Un facteur de 1.0 et un décalage de 0.0 sont préréglés.

1. Saisissez une valeur de facteur et une valeur de décalage.

Générateur d'impulsions

Pour un régulateur continu, le générateur d'impulsions doit être désactivé.

1. Décochez la case "Activer" dans le groupe "Générateur d'impulsions".

Voir aussi

Fonctionnement TCONT_CP (Page 440)

8.3.2.4 Valeur de réglage du régulateur d'impulsions

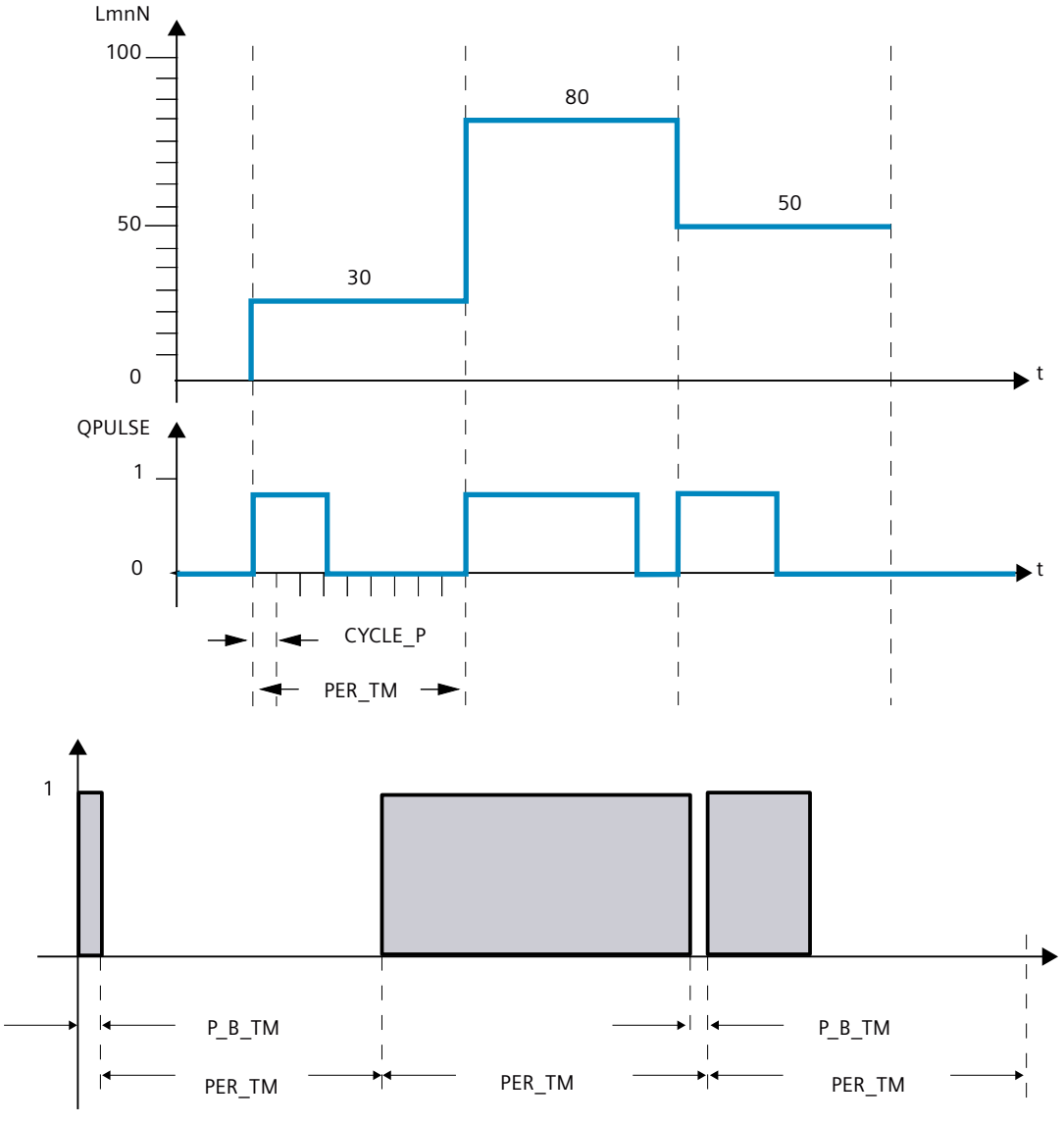
Générateur d'impulsions

La valeur de réglage analogique (LmnN) peut être fournie comme train d'impulsions au paramètre de sortie QPULSE par modulation de la largeur d'impuslion.

Pour utiliser le générateur d'impulsions, procédez de la manière suivante :

1. Cochez la case "Activer" dans le groupe "Générateur d'impulsions".
2. Saisissez le "Temps d'échantillonnage du générateur d'impulsion", la "Durée minimale d'impulsion / de pause" et la "Durée de période".

Les graphiques suivants montrent le rapport entre le "Temps d'échantillonnage du générateur d'impulsions" (CYCLE_P), la "Durée minimale d'impulsion / de pause" (P_B_TM) et la "Période"(PER_TM) :



Temps d'échantillonnage du générateur d'impulsions

Le temps d'échantillonnage du générateur d'impulsions doit correspondre avec le cycle de synchronisation de l'OB d'alarme cyclique émetteur de l'appel. La durée de l'impulsion générée est toujours égale à un multiple entier de cette valeur. Pour une résolution suffisamment précise de la valeur de réglage, le lien suivant doit s'appliquer :
CYCLE_P ≤ PER_TM/50

Durée minimale d'impulsion / de pause

La durée minimale d'impulsion / de pause permet d'éviter de courts temps d'activation ou de désactivation des actionneurs. Une impulsion inférieure à P_B_TM est supprimée.
Les valeurs recommandées sont P_B_TM ≤ 0.1 × PER_TM.

Durée de période

La période ne doit pas dépasser 20 % du temps d'intégration calculé (TI) du régulateur :
PER_TM ≤ TI/5

Exemple d'effet des paramètres CYCLE_P, CYCLE et PER_TM :

Période PER_TM = 10 s
Temps d'échantillonnage de l'algorithme PID CYCLE = 1 s
Temps d'échantillonnage du générateur d'impulsions CYCLE_P = 100 ms.
Toutes les secondes, une nouvelle valeur de réglage est calculée, toutes les 100 ms, la valeur de réglage est comparée avec les longueurs d'impulsion ou de pause affichées jusque là.

- Lorsqu'une impulsion est émise, il y a 2 possibilités :
  - La valeur de réglage calculée est supérieure à la longueur d'impulsion/PER_TM précédente. Dans ce cas, l'impulsion est allongée.
  - La valeur de réglage calculée est inférieure ou égale à la longueur d'impulsion/PER_TM précédente. Aucun signal d'impulsion n'est plus affiché.
- Lorsque aucune impulsion n'est émise, il y a également 2 possibilités :
  - La valeur (100 % - valeur de réglage calculée) est supérieure à la longueur de pause / PER_TM précédente. Dans ce cas, la pause est allongée.
  - La valeur (100 % - valeur de réglage calculée) est inférieure ou égale à la longueur de pause / PER_TM précédente. Un signal d'impulsion est alors affiché.

Voir aussi

Fonctionnement TCONT_CP (Page 440)
Mode de fonctionnement générateur d'impulsion (Page 449)

8.3.3 Mise en service de TCONT_CP

8.3.3.1 Optimisation TCONT_CP

Possibilités d'utilisation

L'optimisation du régulateur est applicable pour les processus de chauffage et de refroidissement purs des systèmes de type I. Mais vous pouvez également utiliser le module pour des systèmes d'ordre supérieur de type II ou III.
Les paramètres PI / PID sont automatiquement calculés et réglés. Le projet de régulateur est conçu pour afficher un comportement de perturbation optimal. Les paramètres "nets" en résultant entraînent, en cas de sauts de consigne, des oscillations parasites de 10 % à 40 % de la hauteur de l'échelon.

Phases d'optimisation du régulateur

L'optimisation du régulateur se fait en plusieurs phases, affichées au niveau du paramètre PHASE .

PHASE = 0

Il n'y a pas d'optimisation. TCONT_CP fonctionne en mode automatique ou manuel.
Pendant PHASE = 0, assurez-vous que le système réglé remplisse les conditions requises à l'optimisation.
Au terme de l'optimisation, TCONT_CP redevient PHASE = 0.

PHASE = 1

TCONT_CP est prêt à l'optimisation. PHASE = 1 peut uniquement être démarré lorsque les conditions requises pour l'optimisation sont remplies.
Pendant PHASE = 1, les valeurs suivantes sont calculées :

- Bruit de la mesure NOISE_PV
- Rampe ascendante de départ PVDT0
- Moyenne de la valeur de réglage
- Temps d'échantillonnage de l'algorithme PID CYCLE
- Temps d'échantillonnage du générateur d'impulsions CYCLE_P

PHASE = 2

En phase 2, à valeur de réglage constante, le point d'inflexion de la mesure est recherché. Le procédé empêche que le point d'inflexioin soit détecté trop tôt en raison du bruit de PV :
Pour le régulateur à impulsions, la valeur moyenne de PV est calculée sur N cycles d'impulsions et mise à la disposition de la partie régulateur. Dans la partie régulateur, une autre moyenne de PV est calculée : au début, cette moyenne est inactive, c'est-à-dire on ne calcule la moyenne que via 1 cycle. Dès que le bruit à atteint un certain seuil, le nombre de cycles est doublé.
La durée de période et l'amplitude du bruit sont calculées. Ce n'est que lorsque le gradient pendant la durée de période estimée est de plus en plus petit que la montée maximale, la recherche du point d'inflexion est interrompue et la phase 2 quittée. TU et T_P_INF sont toutefois calculés au point d'inflexion réel.
L'optimisation n'est toutefois arrêtée que si les deux conditions suivantes sont remplies :

1. La mesure est plus éloignée que 2*NOISE_PV du point d'inflexion.
2. La mesure a dépassé le point d'inflexion de 20 %.

REMARQUE

En cas d'activation via l'échelon de consigne, l'optimisation est arrêtée plus tard lorsque la mesure a traversé 75 % de l'échelon de consigne (SP_INT-PV0).

PHASE = 3, 4, 5

Les phases 3, 4 et 5 durent chacune 1 cycle.
En phase 3, les paramètres PI / PID valides avant l'optimisation sont enregistrés et les paramètres de processus sont calculés.
En phase 4, les nouveaux paramètres PI / PID sont calculés.
En phase 5, la nouvelle valeur de réglage est calculée et appliquée au système réglé.

PHASE = 7

Le type de système est contrôlé en phase 7, car TCONT_CP repasse automatiquement en mode automatique après l'optimisation. Le mode automatique démarre avec LMN = LMN0 + 0.75*TUN_DLMN comme valeur de réglage. Le contrôle du type de système se fait **en mode automatique** avec les paramètres du régulateur nouvellement calculés et se termine au plus tard 0,35*TA (période transitoire) après le point d'inflexion. Si l'ordre du processus dévie fortement de la valeur estimée, les paramètres du régulateur sont recalculés et STATUS_D est augmenté de 1. Dans le cas contraire, les paramètres du régulateur restent inchangés.
Le mode d'optimisation est alors terminé et TCONT_CP est à nouveau en PHASE = 0. Le paramètre STATUS_H vous indique si l'optimisation s'est terminée avec succès.

Abandon prématuré de l'optimisation.

En phase 1, 2 ou 3, vous pouvez annuler l'optimisation en réglant TUN_ON = FALSE, avant que les nouveaux paramètres soient calculés. Le régulateur démarre en mode automatique avec LMN = LMN0 + TUN_DLMN. Si le régulateur était en mode manuel avant l'optimisation, l'ancienne valeur de réglage manuelle est affichée.
Si l'optimisation est annulée en phase 4, 5 ou 7 car TUN_ON = FALSE, les paramètres du régulateur calculés jusque là sont conservés.

8.3.3.2 Conditions requises à l'optimisation

Transitoire amorti

Le processus doit afficher un transitoire amorti stable, à temporisation et asymptotique.
Après un échelon de la grandeur réglante, la mesure doit passer à un état stationnaire. Les processus, qui affichent d'ores et déjà un comportement oscillatoire sans régulation, ainsi que les systèmes sans stabilisation (intégrateur au sein du système), sont donc exclus.

⚠ ATTENTION

Risque de décès, blessures corporelles graves ou dommages matériels importants.

Lors de l'optimisation, le paramètre MAN_ON est désactivé. Par suite, la valeur de réglage ou la mesure peut prendre des valeurs indésirées, voire extrêmes.

La valeur de réglage est spécifiée par l'optimisation. Pour annuler l'optimisation, vous devez tout d'abord régler TUN_ON = FALSE. MAN_ON est alors à nouveau activé.

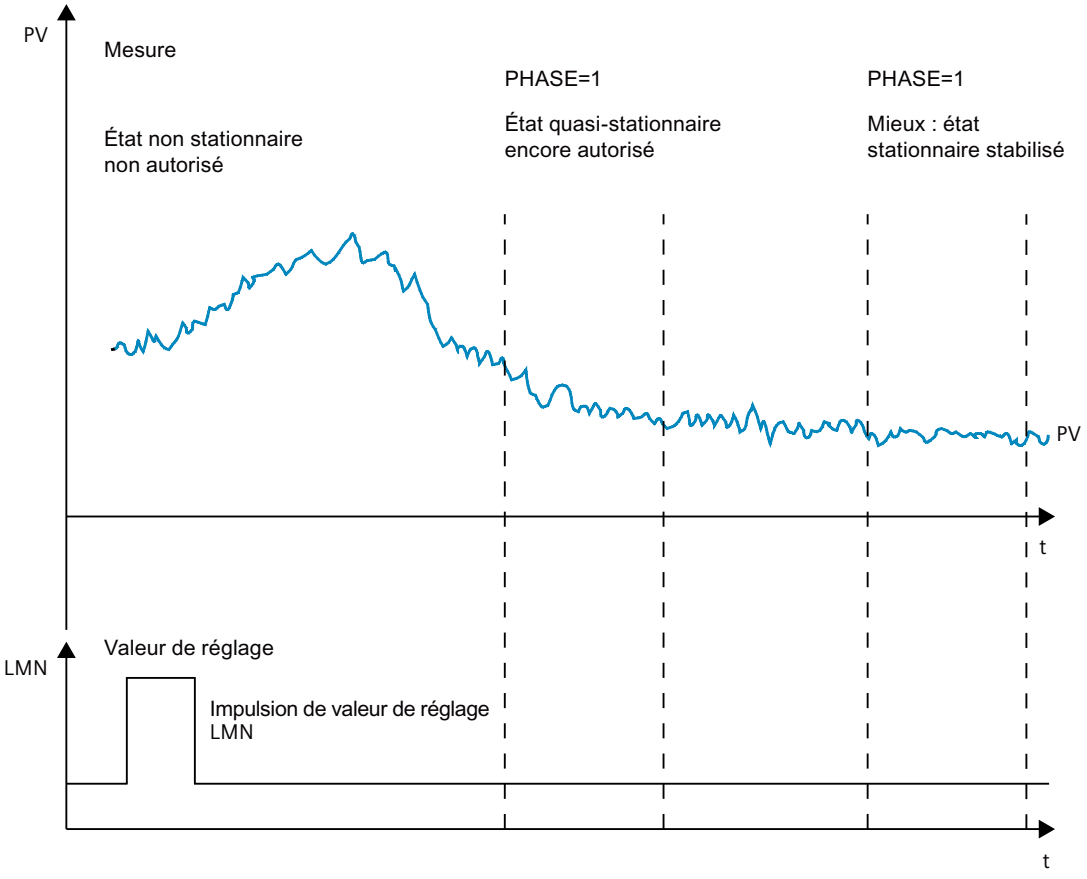
Garantir un état de démarrage quasi-stationnaire (Phase 0)

En cas d'oscillations à basse fréquence de la mesure, p. ex. en raison de paramètres erronés du régulateur, le régulateur doit être passé en mode manuel avant le début de l'optimisation et il convient d'attendre le décroissement de l'oscillation. Alternativement, il est possible de passer à un régulateur PI au réglage "doux" (petite amplification de cycle, long temps d'intégration).
Vous devez alors attendre que l'état stationnaire soit atteint, c.-à-d. que la mesure et la valeur de réglage soient en régime stationnaire. Un effet transitoire asymptotique ou une lente dérive de la mesure (état quasi-stationnaire, voir Fig. suivante) est admissible. La grandeur réglante doit être constante ou osciller d'une valeur moyenne constante.

REMARQUE

Évitez de modifier la grandeur réglante juste avant le démarrage de l'optimisation. Une modification de la grandeur réglante peut également être réalisée de manière involontaire par la mise en place des conditions opératoires d'essai (p. ex. fermeture d'une porte de four) ! Si c'est le cas, vous devez attendre au minimum jusqu'à ce que la mesure repasse de manière asymptotique à un état stationnaire. Vous obtiendrez de meilleurs paramètres de régulateur si vous attendez la fin du phénomène transitoire.

La figure suivante représente l'effet électrique transitoire en état stationnaire :



Linéarité et plage de travail

Le processus doit afficher un comportement linéaire au-dessus de la plage de travail. Un comportement non linéaire se produit par ex. lorsqu'une matière change d'état.
L'optimisation doit avoir lieu dans une partie linéaire de la plage de travail.
Cela signifie que les effets non linéaires au sein de cette plage de travail doivent être dérisoires tant pour l'optimisation que pour la régulation courante. Il est toutefois possible de corriger l'optimisation du processus au moment du changement du point de fonctionnement si elle est à nouveau effectuée au sein d'un petit environnement du nouveau point de fonctionnement et que, pendant cette opération, la non-linéarité n'est pas traversée.
Si certaines non-linéarités statiques (par ex. caractéristiques d'une vanne) sont connues, il est utile en tout cas de la compenser d'emblée par un tracé en polygone afin de linéariser le comportement du processus.

Effets perturbateurs dans les processus thermiques

Les effets perturbateurs, tels que le transfert de chaleur à des zones adjacentes, ne doivent pas compromettre l'ensemble du processus thermique. Il faut par ex. lors de l'optimisation de zones d'une extrudeuse, chauffer simultanément toutes les zones.

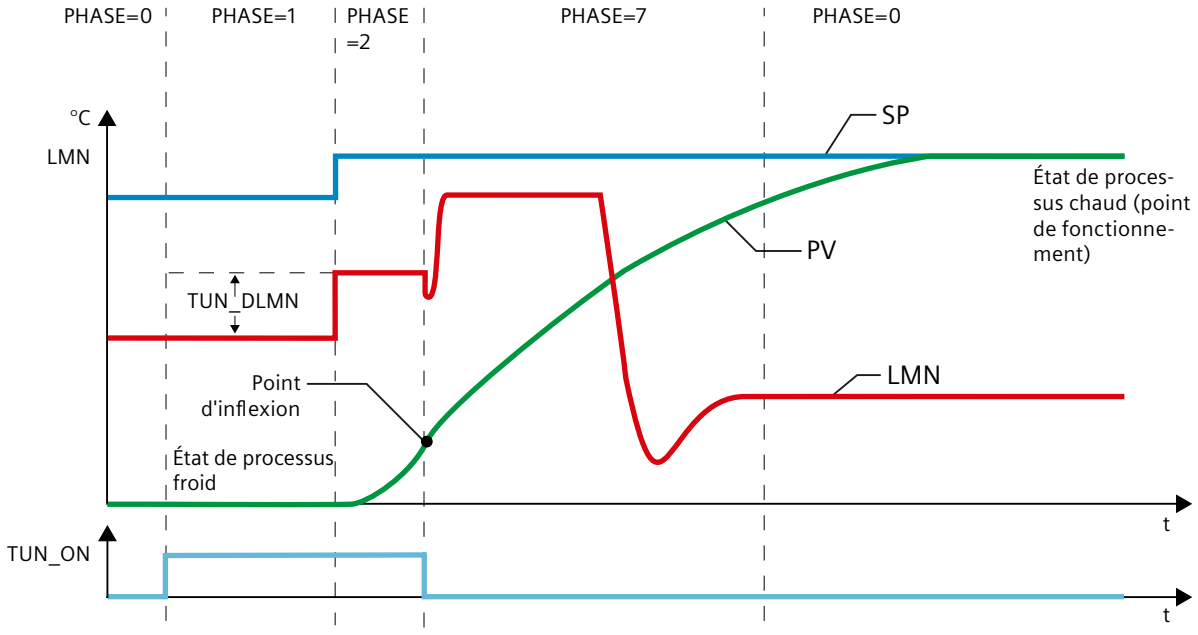
8.3.3.3 Possibilités d'optimisation

Les possibilités suivantes d'optimisation sont disponibles :

- Optimisation préalable
- Optimisation fine
- Optimisation fine manuelle en mode régulation

Optimisation préalable

Pendant ce processus d'optimisation, le point de fonctionnement est sorti de l'état froid par un échelon de valeur de consigne.
Avec TUN_ON = TRUE, réglez l'appareil en mode prêt à l'optimisation. Le régulateur passe de PHASE = 0 à PHASE = 1.



La valeur de réglage d'optimisation (LMN0 + TUN_DLMN) est appliquée en modifiant la consigne (transition phase 1 -> 2). La consigne n'est active qu'au moment d'atteindre le point d'inflexion (le mode automatique est alors activé).
Vous déterminez de votre propre chef la valeur différentielle du saut de valeur de réglage (TUN_DLMN) en fonction de la modification admissible de la mesure. Le signe précédant de TUN_DLMN doit être en fonction de la modification souhaitée de la mesure (tenir compte du sens de régulation).
Il faut que l'échelon de consigne et TUN_DLMN concordent l'un avec l'autre. Un TUN_DLMN trop grand peut être responsable du fait que le point d'inflexion ne sera pas trouvé avant d'avoir atteint 75 % de l'échelon de consigne.
Il faut toutefois que TUN_DLMN soit suffisamment grand pour que la mesure atteigne au moins 22 % de l'échelon de consigne. Dans le cas contraire, le procédé reste en optimisation (phase 2).

Solution : Diminuez la consigne pendant la recherche du poind d'inflexion.

REMARQUE

En ce qui concerne les processus à forte temporisation, il est conseillé de fixer la valeur de consigne cible légèrement en dessous du point de fonctionnement souhaité lors d'une optimisation et de bien surveiller les bits d'état et PV (risque de suroscillation).

Optimisation seulement en plage linéaire :

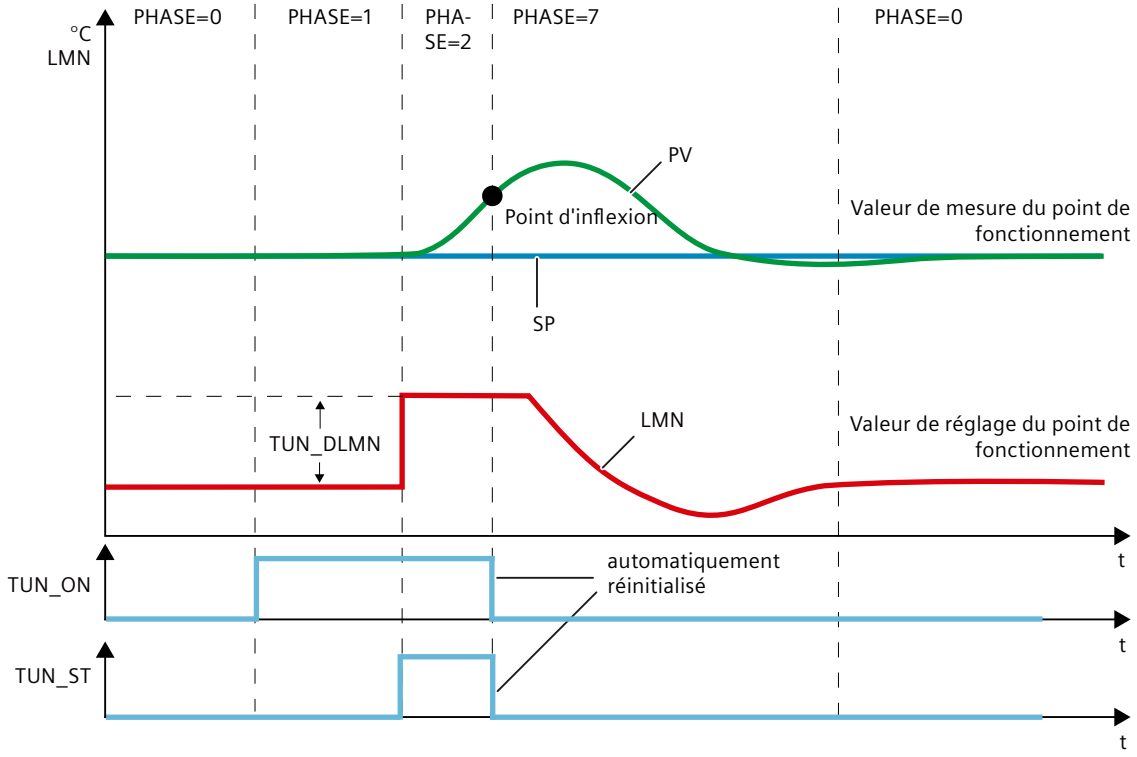
Certains systèmes réglés (par ex. creuset pour zinc ou magnésium) traversent une plage non linéaire peu de temps avant le point de fonctionnement (changement d'état de la matière).

Par un choix judicieux de l'échelon de consigne, l'optimisation peut être limitée à la plage linéaire. Lorsque la mesure a traversé 75 % de l'échelon de consigne (SP_INT-PV0), l'optimisation est arrêtée.

Parallèlement, il est conseillé de diminuer TUN_DLMN de telle manière que le point d'inflexion sera trouvé avant d'atteindre 75 % de l'échelon de consigne.

Optimisation fine

Pendant ce processus d'optimisation, le processus est excité à valeur de consigne constante par un échelon de valeur de consigne.



La valeur de réglage d'optimisation (LMN0 + TUN_DLMN) est appliquée en mettant le bit de départ TUN_ST (transition phase 1 -> 2). Si vous modifiez la consigne, elle ne sera active qu'au moment d'atteindre le point d'inflexion (le mode automatique sera alors activé).

Vous déterminez de votre propre chef la valeur différentielle du saut de valeur de réglage (TUN_DLMN) en fonction de la modification admissible de la mesure. Le signe précédant de

TUN_DLMN doit être en fonction de la modification souhaitée de la mesure (tenir compte du sens de régulation).

IMPORTANT
En cas d'activation via TUN_ST, il n'y a pas de coupure de sécurité à 75 %. L'optimisation est arrêtée quand le point d'inflexion est atteint. En cas de systèmes bruyants, le point d'inflexion peut être nettement dépassé.

Optimisation fine manuelle en mode régulation

Afin d'obtenir un comportement de consigne exempt de dépassement, vous pouvez prendre les mesures suivantes :

- Adapter la plage de régulation
- Optimiser le comportement de référence
- Atténuer les paramètres de régulation
- Modifier les paramètres de régulation

8.3.3.4 Résultat de l'optimisation

Le chiffre gauche de STATUS_H indique l'état d'optimisation

STATUS_H	Résultat
0	Valeur par défaut ou aucun paramètre trouvé ou pas encore de nouveaux paramètres de régulateur.
10000	Paramètres de régulation idoines trouvés
2xxxx	Paramètres de régulation trouvés via valeurs estimées ; vérifiez le comportement de réglage ou consultez le message du diagnostic STATUS_H et renouvelez l'optimisation du régulateur.
3xxxx	Une erreur de commande s'est produite ; consultez le message du diagnostic STATUS_H et renouvelez l'optimisation du régulateur.

Les temps d'échantillonnage CYCLE et CYCLE_P ont déjà été vérifiés en phase 1.
Les paramètres de réglage suivants sont actualisés au niveau de TCONT_CP :

- P (coefficient d'action proportionnelle GAIN)
- I (temps d'intégration TI)
- D (temps de dérivation TD)
- Pondération de l'action P PFAC_SP
- Coefficient DT1 (D_F)
- Plage de régulation marche/arrêt CONZ_ON
- Largeur de la plage de régulation CON_ZONE

La plage de régulation n'est activée que pour les systèmes idoines de types I et II et les régulateurs PID (CONZ_ON = TRUE).
En fonction de PID_ON, la régulation est effectuée avec un régulateur PI ou PID. Les anciens paramètres de régulation sont sauvegardés et peuvent être réactivés via UNDO_PAR. En outre, un jeu de paramètres PI et PID est sauvegardé dans les structures PI_CON et PID_CON.

LOAD_PID et une activation correspondante de PID_ON permettent également ultérieurement de permuter entre les paramètres PI ou PID optimisés.

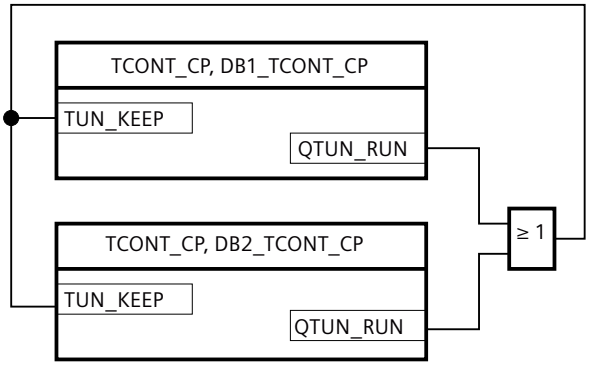
8.3.3.5 Optimisation parallèle des voies du régulateur

Zones adjacentes (couplage thermique élevé)

Si deux régulateurs ou plus règlent la température, par ex. sur une plaque (c.-à-d deux chauffages et deux mesures avec contact thermique important), procédez de la manière suivante :

1. Reliez par les deux sorties QTUN_RUN par un OU logique.
2. Interconnectez chacune dles deux entrées TUN_KEEP avec la sortie du OU logique.
3. Démarrez les deux régulateurs en spécifiant parallèlement un échelon de consigne ou en appliquant parallèlement TUN_ST.

La figure suivante représente l'optimisation parallèle des voies du régulateur :



Avantage :

Les deux régulateurs émettent LMN0 + TUN_DLMN jusqu'à ce qu'ils aient quitté la phase 2. On évite ainsi que le régulateur, qui quitte plus tôt l'optimisation, falsifie le résultat de l'optimisation de l'autre régulateur en modifiant sa valeur de réglage.

IMPORTANT
Après avoir atteint 75 % de l'échelon de consigne, la phase 2 est quittée et la sortie QTUN_RUN remise à son état initial. Cependant, le mode automatique ne commence que lorsque TUN_KEEP est également égal à 0.

Zones adjacentes (couplage thermique faible)

En général, l'optimisation doit correspondre à la régulation ultérieure. Si, en mode production, les zones sont traitées, conjointement et parallèlement, de telle manière que les écarts de température entre les zones restent identiques, on devrait par conséquent augmenter la température des zones adjacentes lors de l'optimisation.

Les différences de température au début de l'essai ne jouent aucun rôle car elles seront compensées par un chauffage initial correspondant (→ montée initiale = 0).

8.3.3.6 Erreurs et solutions

Compenser les erreurs de conduite

Erreur de conduite	STATUS et mesure à prendre	Commentaire
Mise à 1 de TUN_ON et échelon de consigne ou TUN_ST simultanément	Transition en phase 1, mais pas de démarrage de l'optimisation. • SP_INT = SP_{anc} ou • TUN_ST = FALSE	La modification de la consigne est supprimée. Ceci empêche que la régulateur valide la nouvelle consigne et quitte inutilement le point de fonctionnement stationnaire.
TUN_DLMN effectif < 5 % (fin de la phase 1)	STATUS_H = 30002 • Transition en phase 0 • TUN_ON = FALSE • SP = $SP_{ancienne}$	Abandon de l'optimisation. La modification de la consigne est supprimée. Ceci empêche que la régulateur valide la nouvelle consigne et quitte inutilement le point de fonctionnement stationnaire.

Point d'inflexion pas atteint (seulement en cas d'activation par échelon de consigne)

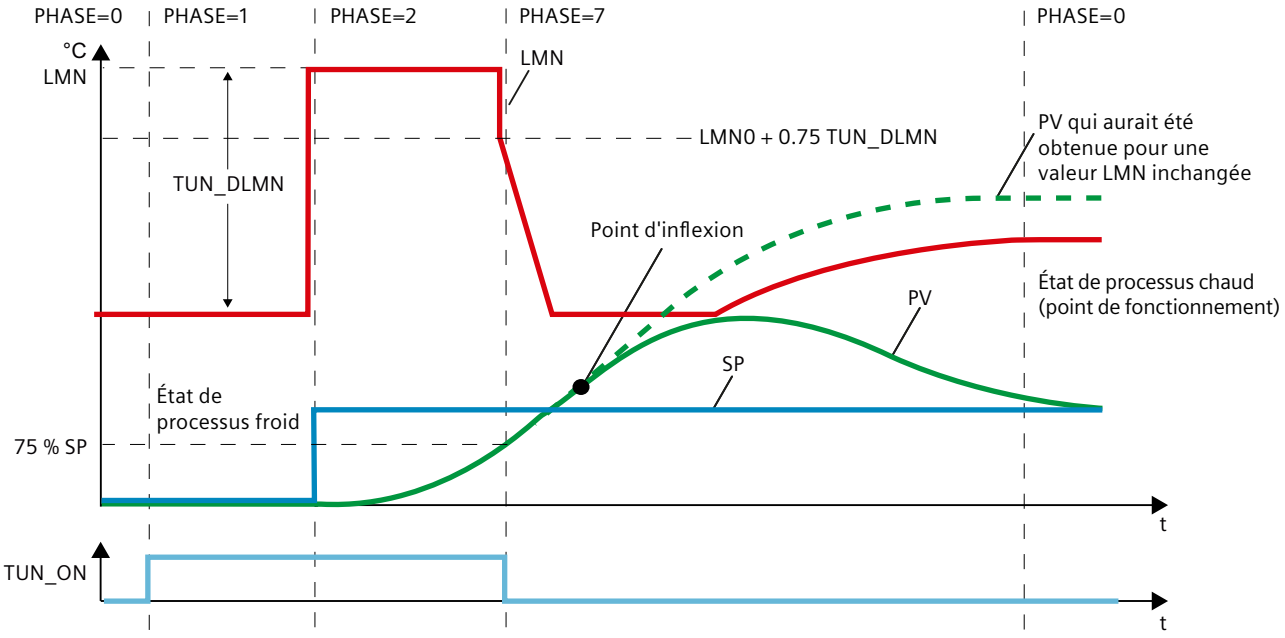
L'optimisation sera arrêtée dès que la mesure a traversé 75 % de l'échelon de consigne (SP_INT-PV0), ce qui sera signalé par "Point d'inflexion pas atteint" dans STATUS_H (2xx2x).

La consigne réglée momentanément s'applique toujours. En réduisant la consigne, il est donc possible d'entraîner ultérieurement la fin anticipée de l'optimisation.

Sur les systèmes de thermorégulation courants, l'abandon de l'optimisation à 75 % de l'échelon de consigne suffit généralement pour éviter un dépassement. Notamment en présence de systèmes à forte temporisation (TU/TA > 0.1, de type III), la prudence est toutefois de rigueur. En cas d'activation trop forte par rapport à l'échelon de consigne, la mesure peut subir un dépassement important (jusqu'au facteur 3).

Si, sur des systèmes de rang prioritaire, le point d'inflexion est encore très éloigné après avoir atteint 75 % de l'échelon de consigne, il y a un net dépassement. De plus, les paramères de régulation sont trop forts. Il faut donc les atténuer ou renouveler l'essai.

La figure suivante représente le dépassement de la mesure en cas d'activation trop forte (système de type III) :



Sur les systèmes de thermorégulation courants, une interruption peu avant d'atteindre le point d'inflexion ne pose aucun problème du point de vue des paramètres de régulation.
Si vous renouvelez l'essai, diminuez TUN_DLMN ou augmentez l'échelon de consigne.
Principe : La valeur de réglage d'optimisation doit concordé à l'échelon de consigne.

Erreur d'estimation en cas de temporisation ou ordre

Impossible de saisir conformément la temporisation (STATUS_H = 2x1xx ou 2x3xx) ou l'ordre (STATUS_H = 21xxx ou 22xxx). Le travail est poursuivi avec une valeur estimée susceptible d'engendrer des paramètres de régulation qui ne sont pas optimaux.
Renouvelez l'optimisation et veillez à ce que la mesure ne subisse aucune perturbation.

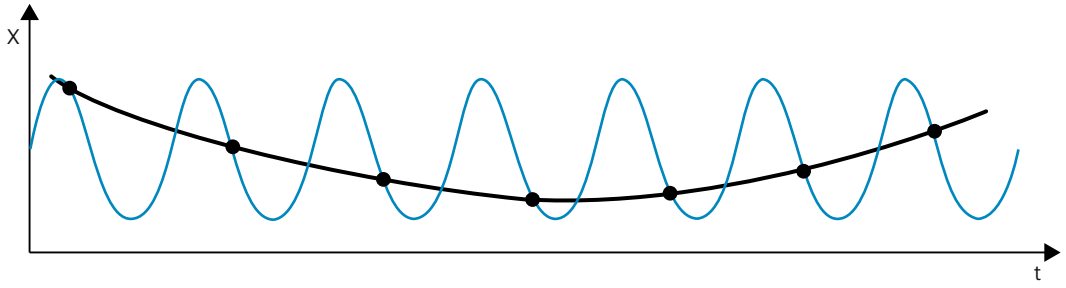
REMARQUE

Le cas particulier d'un système pur PT1 est également signalé par STATUS_H = 2x1xx (TU <= 3*CYCLE). Il n'est pas alors nécessaire de renouveler l'essai. Atténuez les paramètres de régulation si la régulation oscille.

Qualité des signaux de mesure (bruits de mesure, perturbations basse fréquence)

Les bruits de mesure ou des perturbations basse fréquence sont susceptibles de compromettre le résultat de l'optimisation. Tenez compte des points suivants :

- En cas de bruits de mesure, sélectionnez une fréquence d'échantillonnage plutôt élevée que basse. Veillez à ce que la mesure soit échantillonnée au moins deux fois pendant une période de bruit. En mode par impulsions, le filtrage de la valeur moyenne intégré est une aide précieuse. Il implique toutefois que la mesure PV soit transmise à l'instruction pendant le cycle d'impulsions rapide. Le niveau de bruit ne devrait pas être supérieur de 5 % à la modification du signal utile.
- Il est impossible de filtrer les perturbations haute fréquence à l'aide de TCONT_CP. Celles-ci auraient dû déjà être filtrées dans un transducteur afin d'éviter le crénelage.
  La figure suivante représente le crénelage en cas de période d'échantillonnage trop grande :



- En cas de perturbations basse fréquence, il est relativement facile d'assurer un taux de balayage suffisamment élevé. D'autre part, le TCONT_CP doit alors générer un signal de mesure régulier par un grand intervalle du filtrage de la valeur moyenne. Un filtrage de la valeur moyenne doit durer pendant au moins deux périodes de bruit. Il se produit ainsi au sein du bloc des périodes d'échantillonnage rapidement plus grandes de telle manière à compromettre la précision de l'optimisation. Une précision suffisamment grande doit être garantie pour au moins 40 périodes de bruit jusqu'au point d'inflexion.
  Mesure à prendre éventuellement en répétant l'essai :
  augmenter TUN_DLMN.

Dépassement

Les situations suivantes sont susceptibles de provoquer un dépassement :

Situation	Cause	Solution
Fin de l'optimisation	• Activation par une trop forte modification de la valeur de réglage par rapport à l'échelon de consigne (voir ci-dessus). • Régulateur PI activé par PID_ON = FALSE.	• Augrmenter l'échelon de consigne ou diminuer l'échelon de la valeur de réglage • Si le processus autorise un régulateur PID, lancez l'optimisation par PID_ON = TRUE.

Situation	Cause	Solution
Optimisation en phase 7	Tout d'abord, des paramètres de régulation plus doux ont été calculés (système de type III), susceptibles de provoquer un dépassement en phase 7.	-
Régulation	Régulateur PI et avec PFAC_SP = 1.0 pour système de type I.	Si le processus autorise un régulateur PID, lancez l'optimisation par PID_ON = TRUE.

8.3.3.7 Réaliser une optimisation préalable

Conditions

- L'instruction et l'objet technologique sont chargés sur la CPU.

Marche à suivre

Pour calculer les paramètres PID optimaux lors de la première mise en service, procédez de la manière suivante :

1. Cliquez sur l'icône "Start".
   Si aucune connexion en ligne n'existe encore, une connexion est établie. Les valeurs actuelles de la consigne, de la mesure et de la valeur de réglage sont enregistrées.
2. Dans la liste "Mode", sélectionnez l'entrée "Optimisation préalable".
   TCONT_CP est prêt pour l'optimisation.
3. Dans le champ "Echelon de valeur de réglage" saisissez la valeur de laquelle la valeur de réglage doit être augmentée.
4. Dans le champ "Consigne", saisissez une consigne. L'échelon de valeur de réglage n'est activé que par une autre consigne.
5. Cliquez sur l'icône "Démarrage de l'optimisation".
   L'optimisation préalable démarre. L'état de l'optimisation s'affiche.

8.3.3.8 Effectuer une optimisation fine

Conditions

- L'instruction et l'objet technologique sont chargés sur la CPU.

Marche à suivre

Pour calculer les paramètres PID optimaux au point de fonctionnement, procédez de la manière suivante :

1. Cliquez sur l'icône "Start".
   Si aucune connexion en ligne n'existe encore, une connexion est établie. Les valeurs actuelles de la consigne, de la mesure et de la valeur de réglage sont enregistrées.
2. Dans la liste "Mode", sélectionnez l'entrée "Optimisation fine".
   TCONT_CP est prêt pour l'optimisation.
3. Dans le champ "Echelon de valeur de réglage" saisissez la valeur de laquelle la valeur de réglage doit être augmentée.
4. Cliquez sur l'icône "Démarrage de l'optimisation".
   L'optimisation fine démarre. L'état de l'optimisation s'affiche.

8.3.3.9 Annulation de l'optimisation préalable ou de l'optimisation fine

Pour annuler une optimisation préalable ou une optimisation fine, cliquez sur l'icône "Arrêt de l'autoréglage".
Si les paramètres PID n'ont pas encore été calculés et enregistrés TCONT_CP démarre en mode automatique avec LMN = LMN0 + TUN_DLMN. Si le régulateur était en mode manuel avant l'optimisation, l'ancienne valeur de réglage manuelle est affichée.
Si les paramètres PID calculés sont déjà enregistrés, TCONT_CP démarre en mode automatique et travaille avec les paramètres PID déterminés jusque là.

8.3.3.10 Optimisation fine manuelle en mode régulation

Afin d'obtenir un comportement de consigne exempt de dépassement, vous pouvez prendre les mesures suivantes :

Adapter la plage de régulation

En cas d'optimisation, une plage de régulation CON_ZONE du TCONT_CP est calculée et activée pour les systèmes idoines de types I et II et le régulateur PID (CONZ_ON = TRUE). Vous pouvez modifier la plage de régulation en mode régulation ou la désactiver (avec CONZ_ON = FALSE).

REMARQUE

Activer la plage de régulation sur les systèmes de rang prioritaire (de type III) n'apporte généralement pas l'effet escompté car la plage de régulation est alors plus grande que celle pouvant être atteinte avec la valeur de réglage de 100 %. Une activation de la plage de régulation pour un régulateur PI n'apporte rien non plus.

Avant d'activer manuellement la plage de régulation, assurez-vous que sa largeur n'est pas trop petite. Si tel est le cas, la grandeur réglante et la mesure oscilleront.

Atténuer en permance le comportement aux modifications avec PFAC_SP

Le paramètre PFAC_SP vous permet d'atténuer le comportement aux modifications. Ce paramètre détermine à quelle action l'action proportionnelle P sera active en cas d'échelons de consigne.

Quelque soit le type de système, PFAC_SP est spécifié par l'optimisation à 0.8, c'est à vous de modifier cette valeur. Afin de limiter le dépassement en cas d'échelons de consigne (et si tous les autres paramètres de régulation sont conformes) à environ 2 %, les valeurs suivantes pour PFAC_SP sont suffisantes :

	Systèmes de type I	**Systèmes de type II**	**Systèmes de type III**
	Thermorégulation courante	Plage de transition	Thermorégulation de rang prioritaire
PI	0.8	0.82	0.8
PID	0.6	0.75	0.96

Corrigez le réglage par défaut (0.8) notamment dans les cas suivants :

- Système de type I avec PID (0.8 -> 0.6) : les échelons de consigne au sein de la plage de régulation engendrent avec PFAC_SP = 0.8 encore env. un dépassement de 18 %.
- Système de type III avec PID (0.8 -> 0.96) : les échelons de consigne avec PFAC_SP = 0.8 sont trop fortement atténués. On gaspille nettement de la durée de réponse.

Atténuer les paramètres de régulation

Si des oscillations se produisent dans la boucle de régulation ou des dépassements après des échelons de consigne, vous pouvez diminuer le gain du régulateur (par ex. à 80 % de la valeur initiale) et augmenter le temps d'intégration TI (par ex. à 150 % de la valeur initiale). Si la valeur de réglage analogique du régulateur continu avec un formateur d'impulsions est convertie en signaux de réglage binaires, il peut se produire des oscillations continues de faible amplitude dues à la quantification. Vous pouvez les éliminer en augmentant la zone morte du régulateur DEADB_W.

Modifer les paramètres de régulation

Si vous souhaitez modifier les paramètres de régulation, procédez de la manière suivante :

1. Sauvegardez les paramètres actuels avec SAVE_PAR.
2. Modifiez les paramètres.
3. Testez le comportement de réglage.

Si les nouveaux paramètres n'apportent aucune amélioration, UNDO_PAR vous permettent de réactiver les anciens paramètres.

8.3.3.11 Effectuer une optimisation fine manuelle

Conditions

- L'instruction et l'objet technologique sont chargés sur la CPU.

Marche à suivre

Pour déterminer manuellement les paramètres PID optimaux, procédez de la manière suivante :

1. Cliquez sur l'icône "Start".
   Si aucune connexion en ligne n'existe encore, une connexion est établie. Les valeurs actuelles de la consigne, de la mesure et de la valeur de réglage sont enregistrées.
2. Dans la liste "Mode", sélectionnez l'entrée "Manuel".
3. Saisissez les nouveaux paramètres PID.
4. Dans le groupe "Optimisation", cliquez sur l'icône "Envoyer les paramètres à la CPU".
5. Dans le groupe "Valeurs actuelles", cochez la case "Spécifier la consigne".
6. Saisissez une nouvelle consigne et cliquez sur l'icône, dans le groupe "Valeurs actuelles".
7. Le cas échéant, décochez la case "Mode manuel".
   Le régulateur fonctionne avec les nouveaux paramètres PID et régule en fonction de la nouvelle consigne.
8. Contrôlez la qualité des paramètres PID en observant l'allure des courbes.
9. Répétez les étapes 3 à 8 jusqu'à ce que vous soyez satisfait du résultat du régulateur.

8.4 TCONT_S

8.4.1 Objet technologique TCONT_S

L'objet technologique TCONT_S fournit un régulateur pas à pas pour un actionneur à comportement intégral et sert à régler les processus thermiques techniques avec des signaux de sortie TOR de la valeur de réglage. L'objet technologique correspond au DB de données d'instance de l'instruction TCONT_S. Son fonctionnement est basé sur l'algorithme de régulation PI du régulateur à échantillonnage. Ce régulateur pas à pas fonctionne sans signalisation de position. Les modes de fonctionnement manuel et automatique sont possibles.

S7-1500

Toutes les variables et tous les paramètres de l'objet technologique sont rémanents et peuvent être modifiés uniquement lors du chargement dans l'appareil si vous avez chargé complètement TCONT_S.

Voir aussi

Présentation des régulateurs de logiciel (Page 43)

Ajouter des objets technologiques (Page 45)

Configurer les objets technologiques (Page 46)

Charger des objets technologiques dans l'appareil (Page 48)

TCONT_S (Page 461)

8.4.2 Configurer le signal d'écart TCONT_S

Utiliser la mesure de périphérie

Pour utiliser le paramètre d'entrée PV_PER, procédez de la manière suivante :

1. Sélectionnez l'entrée "Périphérie" dans la liste "Source".
2. Sélectionnez le "Type de capteur".
   Suivant le type de capteur, la mesure est normalisée selon différentes formules.
   - Standard
     Thermocouples ; PT100/NI100
     PV = 0.1 × PV_PER × PV_FAC + PV_OFFS
   - Climatique ;
     PT100/NI100
     PV = 0.01 × PV_PER × PV_FAC + PV_OFFS
   - Courant/tension
     PV = 100/27648 × PV_PER × PV_FAC + PV_OFFS
3. Enregistrez le facteur et le décalage pour la normalisation de la mesure de périphérie.

Utiliser la mesure interne

Pour utiliser le paramètre d'entrée PV_IN, procédez de la manière suivante :

1. Sélectionnez l'entrée "Interne" dans la liste "Source".

Signal d'écart

Dans les conditions suivantes, vous réglez une largeur de zone morte :

- Le signal de la mesure est brouillé.
- Le gain du régulateur est élevé.
- L'action D est activée.

La part de bruit de la mesure occasionne dans ce cas de fortes variations de la valeur de réglage. La zone morte réduit la part de bruit lorsque le régulateur est à l'état stationnaire. La largeur de la zone morte indique la taille de la zone morte. Lorsque la largeur de la zone morte est 0.0, la zone morte est désactivée.

Voir aussi

Fonctionnement TCONT_S (Page 462)

8.4.3 Configurer l'algorithme de régulation TCONT_S

Généralités

1. Saisissez le "Temps d'échantillonnage de l'algorithme PID".
   Le temps d'échantillonnage du régulateur ne doit pas dépasser 10 % du temps d'intégration du régulateur (TI).
2. Lorsque la structure du régulateur contient une action P, saisissez le "Gain proportionnel".
   Un gain proportionnel négatif inverse le sens de régulation.

Action P

En cas de modification de la consigne, des suroscillations peuvent se produire au niveau de l'action P. La pondération de l'action P permet de choisir le degré de l'effet de l'action P en cas de modifications de consigne. L'atténuation de l'action P s'obtient par compensation de l'action I.

1. Pour affaiblir l'action P lors de la modification des valeurs de consigne, vous entrez la "pondération de l'action P".
   - 1.0: Action P totalement opérante si modification de la consigne
   - 0.0: Action P non opérante si modification de la consigne

Action I

1. Lorsque la structure du régulateur contient une action I, saisissez le "Temps d'intégration".
   Pour un temps d'intégration de 0.0, l'action I est désactivée.

Voir aussi

Fonctionnement TCONT_S (Page 462)

8.4.4 Configurer la valeur de réglage TCONT_S

Générateur d'impulsions

1. Saisissez la durée minimale d'impulsion et la durée minimale de pause.
   Les valeurs doivent être supérieures ou égales au temps de cycle du paramètre d'entrée CYCLE. La fréquence de commutation en est réduite.
2. Saisissez le temps de positionnement du moteur.
   La valeur doit être supérieure ou égale au temps de cycle du paramètre d'entrée CYCLE.

Voir aussi

Fonctionnement TCONT_S (Page 462)

8.4.5 Mise en service de TCONT_S

Conditions

- L'instruction et l'objet technologique sont chargés sur la CPU.

Marche à suivre

Pour déterminer manuellement les paramètres PID optimaux, procédez de la manière suivante :

1. Cliquez sur l'icône "Start".
   Si aucune connexion en ligne n'existe encore, une connexion est établie. Les valeurs actuelles de la consigne, de la mesure et de la valeur de réglage sont enregistrées.
2. Saisissez les nouveaux paramètres PI dans les champs "P", "I", et Pondération de l'action P.
3. Dans le groupe "Optimisation", cliquez sur l'icône "Envoyer les paramètres à la CPU".
4. Dans le groupe "Valeurs actuelles", cochez la case "Spécifier la consigne".
5. Saisissez une nouvelle consigne et cliquez sur l'icône, dans le groupe "Valeurs actuelles".
6. Le cas échéant, décochez la case "Mode manuel".
   Le régulateur fonctionne avec les nouveaux paramètres et régule en fonction de la nouvelle consigne.
7. Contrôlez la qualité des paramètres PID en observant l'allure des courbes.
8. Répétez les étapes 2 à 6 jusqu'à ce que vous soyez satisfait du résultat du régulateur.

Fonctions auxiliaires

9

9.1 Polyline

Polyligne

L'instruction polyligne propose une fonction avec la courbe caractéristique du tracé polygonal dont les nœuds d'interpolation permettent par ex. de linéariser le comportement de capteurs non linéaires.
L'instruction Polyline est utilisable avec la CPU S7-1500 à partir de la version de firmware 2.0 et avec la CPU S7-1200 à partir de la version de firmware 4.2.

Informations complémentaires

Description de polyligne (Page 472)
Fonctionnement de la polyligne (Page 475)
Paramètres d'entrée de la polyligne (Page 478)
Paramètres de réglage de la polyligne (Page 479)
Variables statiques de la polyligne (Page 479)
Paramètre ErrorBits (Page 481)

9.2 SplitRange

SplitRange

L'instruction SplitRange divise la plage des valeurs de réglage du régulateur PID en plusieurs sous-intervalles. Ces sous-intervalles permettent la régulation d'un processus influencé par plusieurs actionneurs.
L'instruction SplitRange est utilisable avec la CPU S7-1500 à partir de la version de firmware 2.0 et avec la CPU S7-1200 à partir de la version de firmware 4.2.

Informations complémentaires

Description de SplitRange (Page 484)
Paramètre d'entrée SplitRange (Page 487)
Variables statiques SplitRange (Page 488)
Paramètre de sortie SplitRange (Page 487)
Paramètre ErrorBits (Page 488)

9.3 RampFunction

RampFunction

L'instruction RampFunction limite la vitesse de modification d'un signal. Un saut de signal à l'entrée est émis comme fonction de rampe de la valeur de réglage pour obtenir un comportement de guidage plus doux, sans influer sur le comportement de perturbation.

L'instruction RampFunction est utilisable avec la CPU S7-1500 à partir de la version de firmware 2.0 et avec la CPU S7-1200 à partir de la version de firmware 4.2.

Informations complémentaires

Description de RampFunction (Page 490)
Fonctionnement de RampFunction (Page 495)
Paramètre d'entrée RampFunction (Page 498)
Paramètre de sortie RampFunction (Page 498)
Variables statiques RampFunction (Page 499)
Paramètre ErrorBits (Page 500)

9.4 RampSoak

RampSoak

Avec l'instruction, vous générez une valeur de réglage qui suit un profil programmable dans le temps. Chaque point de ce profil a une valeur cible et une valeur de temps. Lorsque le profil s'exécute, la valeur cible du point actuel est atteinte dans la limite de la valeur de temps. L'instruction peut être utilisée par exemple pour fournir un profil de consignes en vue de réaliser un régulateur industriel de température Rampe/palier.

L'instruction RampSoak peut être utiisée sur une CPU S7-1500 à partir de la version de firmware 2.0 et sur une CPU S7-1200 à partir de la version de firmware 4.2.

Pour plus d'informations

Description RampSoak (Page 504)
Fonctionnement RampSoak (Page 506)
Paramètre d'entrée RampSoak (Page 520)
Paramètre de sortie RampSoak (Page 520)
Paramètre d'entrée/sortie RampSoak (Page 521)
Variables statiques RampSoak (Page 521)
Paramètre ErrorBits (Page 523)

9.5 Filter_PT1

Filter_PT1

L'instruction Filter_PT1 est un élément de transmission proportionnel avec un retard de premier ordre. Vous pouvez utiliser Filter_PT1 comme filtre passe-bas, élément de temporisation ou bloc de simulation de processus.

L'instruction Filter_PT1 est utilisable avec la CPU S7-1500 à partir de la version de firmware 2.0 et avec la CPU S7-1200 à partir de la version de firmware 4.2.

Pour plus d'informations

Description Filter_PT1 (Page 527)
Fonctionnement Filter_PT1 (Page 533)
Paramètres d'entrée Filter_PT1 (Page 535)
Paramètres de sortie Filter_PT1 (Page 535)
Variables statiques Filter_PT1 (Page 535)
Paramètre ErrorBits (Page 536)

9.6 Filter_PT2

Filter_PT2

L'instruction Filter_PT2 est une fonction de transfert proportionnelle avec retard du second ordre. Vous pouvez utiliser l'instruction Filter_PT2 comme filtre passe-bas, élément de retard ou bloc de simulation de processus.

L'instruction Filter_PT2 peut être utilisée sur une CPU S7-1500 à partir de la version de firmware 2.0 et sur une CPU S7-1200 à partir de la version de firmware 4.2.

Pour plus d'informations

Description Filter_PT2 (Page 539)
Fonctionnement Filter_PT2 (Page 546)
Paramètres d'entrée Filter_PT2 (Page 548)
Paramètres de sortie Filter_PT2 (Page 548)
Variables statiques Filter_PT2 (Page 549)
Paramètre ErrorBits (Page 550)

9.7 Filter_DT1

Filter_DT1

L'instruction Filter_DT1 est une action proportionnelle dérivée avec retard du premier ordre. Vous pouvez utiliser l'instruction Filter_DT1 comme filtre passe-haut, action proportionnelle dérivée ou action anticipatrice.

L'instruction Filter_DT1 peut être utilisée sur une CPU S7-1500 à partir de la version de firmware 2.0 et sur une CPU S7-1200 à partir de la version de firmware 4.2.

Pour plus d'informations

Description Filter_DT1 (Page 553)
Fonctionnement Filter_DT1 (Page 560)
Paramètres d'entrée Filter_DT1 (Page 561)
Paramètres de sortie Filter_DT1 (Page 562)
Variables statiques Filter_DT2 (Page 562)
Paramètre ErrorBits (Page 563)

9.8 Filter_Universal

Filter_Universal

L'instruction Filter_Universal est un filtre numérique configurable dans l'ordre 1 à 10. Vous pouvez utiliser Filter_Universal comme filtre passe-haut, passe-bas, passe-bande ou coupe-bande.

L'instruction Filter_Universal peut être utilisée sur une CPU S7-1500 à partir de la version de firmware 2.0.

Pour plus d'informations

Description Filter_Universal
Mode opératoire Filter_Universal
Paramètres d'entrée Filter_Universal
Paramètres de sortie Filter_Universal
Variable statique Filter_Universal
Paramètre ErrorBits

10

Instructions

10.1 PID_Compact

10.1.1 Nouveautés PID_Compact

PID_Compact V3.0

- **Zone morte**
  Le réglage manuel d'une largeur de bande morte peut réduire des fluctuations inutiles de la valeur de sortie si la mesure est bruitée.
- **Limitation active de l'action intégrale et modification des limites de valeur de sortie en mode automatique**
  En plus de l'arrêt en fonction du sens de l'action I, celle-ci est désormais limitée activement.
  Ceci permet de modifier les limites de valeur de sortie en mode automatique.
- **Nouvelles variables Config.OutputSelect et Retain.CtrlParams.SetByUser**
  Les variables Config.OutputSelect et Retain.CtrlParams.SetByUser remplacent les anciennes variables _Config.OutputSelect et _Retain.CtrlParams.SetByUser qui ne sont disponibles que via Openness API.
  Les variables Config.OutputSelect et Retain.CtrlParams.SetByUser sont présentes dans le bloc de données d'instance et disponibles via Openness API.
  Les valeurs existantes des nouvelles variables Config.OutputSelect et Retain.CtrlParams.SetByUser sont reprises dans les différentes instances après la bascule vers V3. Dans les multi-instances, les nouvelles variables prennent la valeur par défaut après le passage à V3. Réglez à nouveau manuellement les paramètres correspondants.
  Si vous utilisiez jusqu'à présent dans l'application Openness _Config.OutputSelect ou _Retain.CtrlParams.SetByUser, remplacez-les lors de la bascule vers PID_Compact V3 par les variables Config.OutputSelect et Retain.CtrlParams.SetByUser.

PID_Compact V2.4

- **Initialisation de l'action intégrale**
  PID_Compact initialise correctement l'action intégrale uniquement si vous utilisez OverwriteInitialOutputValue avec un sens de régulation inversé.
  Si vous utilisiez jusqu'à présent OverwriteInitialOutputValue avec un sens de régulation inversé, notez que le signe de la valeur de sortie change avec PID_Compact V2.4.

PID_Compact V2.3

- **Réaction de la valeur de sortie lorsque le mode de fonctionnement passe de "Inactif" à "Mode automatique"**
  La nouvelle option IntegralResetMode = 4 a été ajoutée et définie comme valeur par défaut. Avec IntegralResetMode = 4, l'action I est préréglée automatiquement de manière à ce que, lorsque le mode de fonctionnement passe de "Inactif" à "Mode automatique", le signal d'écart entraîne un saut de même signe de la valeur de sortie.
- **Initialisation de l'action intégrale en mode automatique**
  L'action intégrale peut être initialisée en mode automatique à l'aide des variables OverwriteInitialOutputValue et PIDCtrl.PIDInit. Cela facilite l'utilisation de PID_Compact dans des régulations en mode alternatif.

PID_Compact V2.2

- **Utilisation avec S7-1200**
  A partir de PID_Compact V2.2, il est possible d'utiliser l'instruction avec une fonctionnalité V2, y compris sur une S7-1200 à partir de la version de firmware 4.0.

PID_Compact V2.0

- **Comportement en cas d'erreur**
  Le comportement en cas d'erreur a été revu intégralement. PID_Compact est maintenant plus tolérant aux erreurs dans le réglage par défaut. Ce comportement est paramétré lors de la copie de PID_Compact V1.X depuis une CPU S7-1200 vers une CPU S7-1500.

IMPORTANT
Votre installation peut être endommagée. Quand vous utilisez le réglage par défaut, PID_Compact reste en mode automatique en cas de dépassement des limites de la mesure. Cela peut endommager votre installation. Configurez un comportement en cas d'erreur pour votre système réglé, qui protège votre installation de tout endommagement.

  Le paramètre Error indique si une erreur est actuellement présente. Quand l'erreur a disparu, Error = FALSE. Le paramètre ErrorBits indique les erreurs survenues. Utilisez ErrorAck pour acquitter les erreurs et avertissements sans redémarrer le régulateur ni supprimer l'action I. Les erreurs qui ne sont plus présentes ne sont plus supprimées par un changement du mode de fonctionnement.
  Le comportement en cas d'erreur peut être configuré avec SetSubstituteOutput et ActivateRecoverMode.
- **Valeur de sortie de remplacement**
  Vous pouvez configurer une valeur de sortie de remplacement qui sera utilisée en cas d'erreur.
- **Changement de mode de fonctionnement**
  Le mode de fonctionnement est défini au paramètre d'entrée/sortie Mode et est démarré via un front montant à ModeActivate. La variable sRet.i_Mode est supprimée.
- **Fonctionnalité multiinstance**
  Vous pouvez appeler PID_Compact en tant que DB multiinstance.

- **Comportement au démarrage**
  Le mode de fonctionnement défini à Mode est également démarré en cas de front descendant à Reset et en cas de démarrage à froid de la CPU, si RunModeByStartup = TRUE.
- **Comportement ENO**
  ENO est défini en fonction du mode de fonctionnement.
  Si State = 0, alors ENO = FALSE.
  Si State ≠ 0, alors ENO = TRUE.
- **Définition de la valeur de la consigne pendant l'optimisation**
  Les fluctuations autorisées de la valeur de consigne pendant l'optimisation sont configurées à la variable CancelTuningLevel.
- **Plage de valeurs pour les limites de valeur de sortie**
  La valeur 0.0 ne doit plus être dans les limites de valeur de sortie.
- **Prérégler l'action I**
  Définissez avec les variables IntegralResetMode et OverwriteInitialOutputValue le préréglage de l'action I lors du passage du mode de fonctionnement de "Inactif" à "Mode automatique".
- **Application d'une grandeur perturbatrice**
  Vous pouvez appliquer une valeur perturbatrice au paramètre Disturbance.
- **Valeurs par défaut des paramètres PID**
  Les valeurs par défaut suivantes ont été modifiées :
  – Pondération de l'action P (PWeighting) de 0.0 à 1.0
  – Pondération de l'action D (DWeighting) de 0.0 à 1.0
  – Coefficient de l''action par dérivation (TdFiltRatio) de 0.0 à 0.2
- **Renommer les variables**
  Les variables statiques ont été renommées et portent désormais des noms compatibles avec PID_3Step.

PID_Compact V1.2

- **Fonctionnement manuel à la mise en route de la CPU**
  Lorsque ManualEnable = TRUE au démarrage de la CPU, PID_Compact démarre en mode manuel. Un front montant ManualEnable n'est pas nécessaire.
- **Optimisation préalable**
  Si la CPU est désactivée pendant l'optimisation préalable, celle-ci est à nouveau démarrée lors de l'activation de la CPU.

PID_Compact V1.1

- **Fonctionnement manuel à la mise en route de la CPU**
  Au démarrage de la CPU, PID_Compact passe en manuel uniquement s'il y a un front montant à ManualEnable. En l'absence de ce front montant, PID_Compact démarre dans le dernier mode de fonctionnement pour lequel ManualEnable était FALSE.
- **Comportement avec Reset**
  Un front montant sur Reset remet à zéro les erreurs et les avertissements et supprime l'action I. Un front descendant sur Reset fait passer au dernier mode de fonctionnement actif.
- **Valeur par défaut de la limite supérieure de la mesure**
  Par défaut, la valeur de r_Pv_Hlm est maintenant 120.0.
- **Surveillance du temps d'échantillonnage**
  - Plus aucune erreur n'est affichée pour une période actuelle d'échantillonnage ≥ 1,5 x la valeur moyenne actuelle ou pour une période actuelle d'échantillonnage ≤ 0,5 x la valeur moyenne actuelle. En mode automatique, le temps d'échantillonnage peut varier de manière plus importante.
  - PID_Compact est compatible avec FW à partir de V2.0.
- **Accès aux variables**
  Les variables suivantes peuvent désormais être utilisées dans le programme utilisateur.
  - i_Event_SUT
  - i_Event_TIR
  - r_Ctrl_Ioutv
- **Correction d'erreur**
  PID_Compact affiche désormais des impulsions correctes quand le plus petit temps ON est différent du plus petit temps OFF.

10.1.2 Compatibilité avec CPU et FW

Le tableau suivant montre les CPU et les versions de PID_Compact compatibles.

CPU	FW	PID_Compact
S7-1200	à partir de V4.2	V2.3 V2.2 V1.2
	V4.0 à V4.1	V2.2 V1.2
	V3.x	V1.2 V1.1
	V2.x	V1.2 V1.1
	V1.x	V1.0
S7-1500	À partir de V3.1	V3.0 V2.4
	V3.0	V2.4

CPU	FW	PID_Compact
S7-1500	V2.5 à V2.9	V2.4 V2.3 V2.2 V2.1 V2.0
	V2.0 et V2.1	V2.3 V2.2 V2.1 V2.0
	V1.5 à V1.8	V2.2 V2.1 V2.0
	V1.1	V2.1 V2.0
	V1.0	V2.0

10.1.3 Temps de traitement de la CPU et espace mémoire requis PID_Compact à partir de V2

Temps de traitement de la CPU

Temps de traitement de CPU typiques de l'objet technologique PID_Compact à partir de la version V2.0 en fonction du type de CPU et mode de fonctionnement pour CPU standard, F, T et TF.

CPU	Firm-ware	Temps de traitement de CPU typ. Mode automatique	Temps de traitement de CPU typ. Optimisation préalable et optimisation fine
CPU 1211	≥ V4.0	190 µs	270 µs
CPU 1212			
CPU 1214			
CPU 1215			
CPU 1217			
CPU 1510SP	≤ V2.9	65 µs	80 µs
CPU 1511			
CPU 1511C			
CPU 1512C			
CPU 1512SP			
CPU 1513			
CPU 1515		50 µs	65 µs
CPU 1516			
CPU 1517	Chaque	8 µs	12 µs
CPU 1518		4 µs	6 µs
CPU 1510SP	≥ V3.0	55 µs	70 µs
CPU 1511			
CPU 1511C			
CPU 1512C			
CPU 1512SP			
CPU 1513			
CPU 1514SP			
CPU 1515		40 µs	55 µs
CPU 1516			

Temps de traitement de CPU typiques de l'objet technologique PID_Compact à partir de la version V2.0 en fonction du type de CPU et mode de fonctionnement pour des CPU R dans l'état système RUN-Redundant.

CPU	Firm-ware	Temps de traitement de CPU typ. Mode automatique	Temps de traitement de CPU typ. Optimisation préalable et optimisation fine
CPU 1513R	≥ V3.0	90 µs	140 µs
CPU 1515R		70 µs	90 µs

Espace mémoire requis

Espace mémoire requis d'un DB d'instance de l'objet technologique PID_Compact à partir de la version V2.0.

Espace mémoire requis	Espace mémoire requis du DB d'instance de PID_Compact V2.x	Espace mémoire requis du DB d'instance de PID_Compact V3.x
Taille de mémoire de chargement requise	env. 3700 octets	env. 3750 octets
Taille totale de la mémoire de travail requise	788 octets	802 octets
Taille de la mémoire rémanente requise	44 octets	52 octets

10.1.4 PID_Compact à partir de V2

10.1.4.1 Description PID_Compact V3

Description

L'instruction PID_Compact met à disposition un régulateur PID avec optimisation intégrée pour actionneurs proportionnels.
Les modes suivants sont disponibles :

- Inactif
- Optimisation préalable
- Optimisation fine
- Mode automatique
- Mode manuel
- Valeur de sortie de remplacement avec surveillance des erreurs

Les modes de fonctionnement sont décrits en détail dans le paramètre State.

Algorithme PID

PID_Compact est un régulateur PIDT1 avec anti-saturation et pondération de l'action P et de l'action D. L'algorithme PID fonctionne selon la formule suivante (zone morte désactivée) :

$$y = K_p \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

Symbole	Description	Paramètres correspondants de l'instruction PID_Compact
y	Valeur de sortie de l'algorithme PID	-
K_p	Gain proportionnel	Retain.CtrlParams.Gain
s	Opérateur de Laplace	-
b	Pondération de l'action P	Retain.CtrlParams.PWeighting
w	Consigne	CurrentSetpoint
x	Mesure	ScaledInput
T_I	Temps d'intégration	Retain.CtrlParams.Ti
T_D	Temps de dérivation	Retain.CtrlParams.Td
a	Coefficient pour l'action par dérivation (action par dérivation T1 = a × T_D)	Retain.CtrlParams.TdFiltRatio
c	Pondération de l'action D	Retain.CtrlParams.DWeighting
DeadZone	Largeur de zone morte	Retain.CtrlParams.DeadZone

Schéma fonctionnel PID_Compact

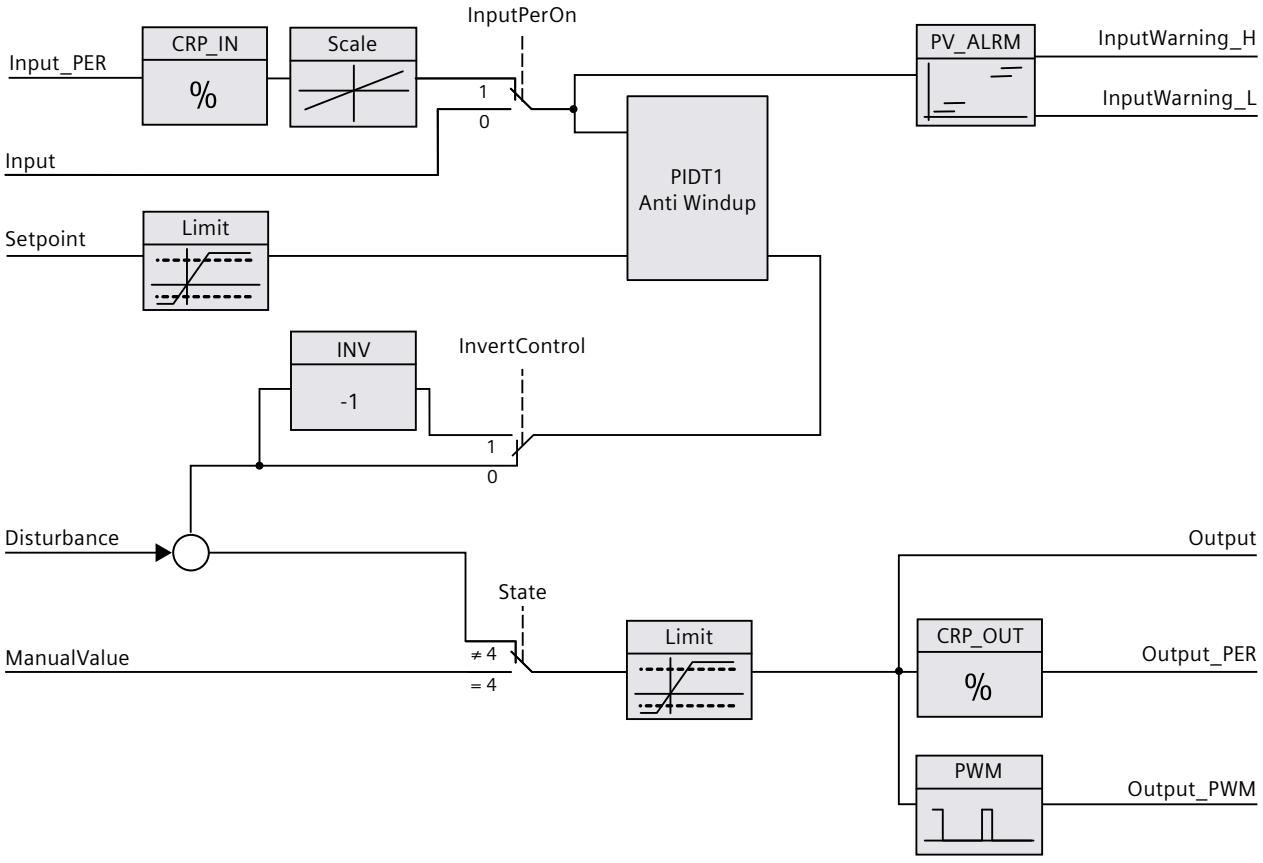
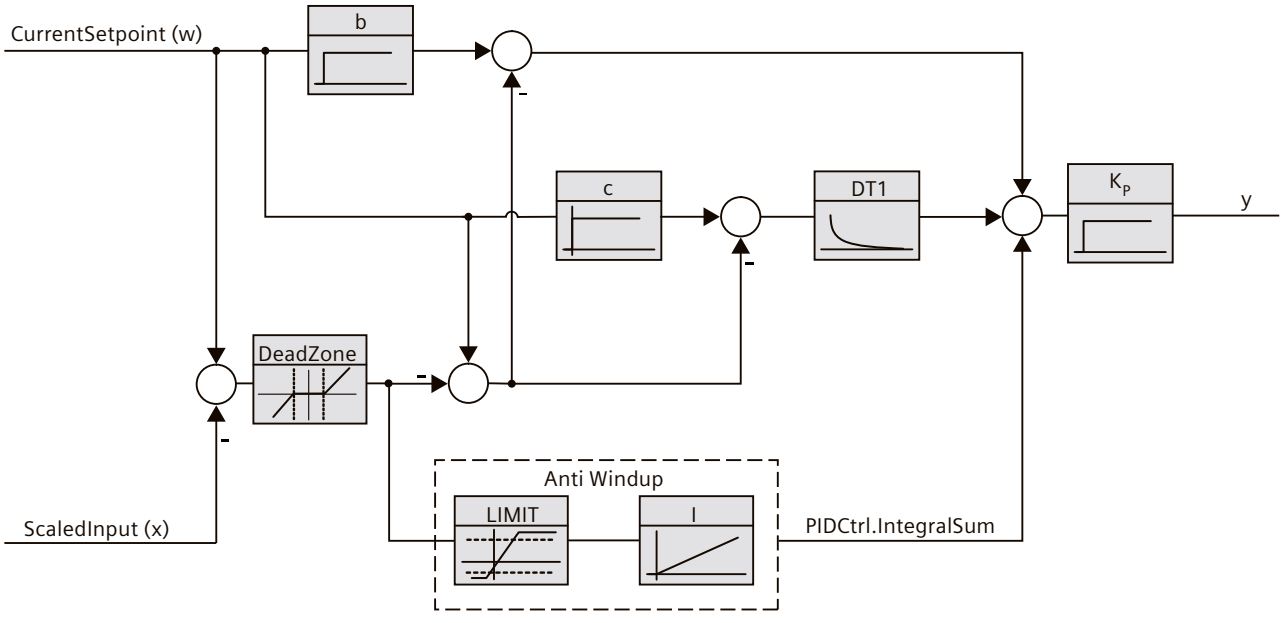


Schéma fonctionnel PIDT1 avec anti-saturation



Appel

PID_Compact est appelé à intervalle de temps constant d'une alarme cyclique de l'OB.

Chargement dans l'appareil

Les valeurs effectives de variables rémanentes ne sont actualisées que si vous chargez entièrement PID_Compact.
Charger un objet technologique dans l'appareil

Démarrage

Au démarrage de la CPU, PID_Compact démarre dans le mode de fonctionnement enregistré aux paramètres d'entrée/sortie Mode. Réglez RunModeByStartup = FALSE pour passer en mode de fonctionnement "Inactif" au démarrage.

Comportement en cas d'erreur

Le comportement en cas d'erreur dépend des variables SetSubstituteOutput et ActivateRecoverMode. Si ActivateRecoverMode = TRUE, le comportement dépend en outre de la nature de l'erreur.

SetSubstitu-teOutput	**ActivateReco-verMode**	**Éditeur de configuration > Valeur de sortie > Régler Output**	**Comportement**
non signif.	FALSE	Zéro (inactif)	Passage au mode de fonctionnement "Inactif" (State = 0) La valeur.0 0 est transmise à l'actionneur.
FALSE	TRUE	Valeur de sortie actuelle pour la durée de l'erreur	Passage au mode de fonctionnement "Valeur de sortie de remplacement avec surveillance des erreurs" (State = 5) La valeur de sortie actuelle est transmise à l'actionneur pour la durée de l'erreur.
TRUE	TRUE	Valeur de sortie de remplacement pour la durée de l'erreur	Passage au mode de fonctionnement "Valeur de sortie de remplacement avec surveillance des erreurs" (State = 5) La valeur à SubstituteOutput est transmise à l'actionneur pour la durée de l'erreur.

PID_Compact utilise ManualValue comme valeur de sortie en mode manuel, sauf si ManualValue est invalide. Si ManualValue est invalide, SubstituteOutput est utilisé. Si ManualValue et SubstituteOutput sont invalides, Config.OutputLowerLimit est utilisé.
Le paramètre Error indique si une erreur est actuellement présente. Quand l'erreur a disparu, Error = FALSE. Le paramètre ErrorBits indique les erreurs qui se sont produites. ErrorBits est réinitialisé par un front montant à Reset ou ErrorAck.

10.1.4.2 Description PID_Compact V2

Description

L'instruction PID_Compact met à disposition un régulateur PID avec optimisation intégrée pour actionneurs proportionnels.

Les modes suivants sont disponibles :

- Inactif
- Optimisation préalable
- Optimisation fine
- Mode automatique
- Mode manuel
- Valeur de sortie de remplacement avec surveillance des erreurs

Les modes de fonctionnement sont décrits en détail dans le paramètre State.

Algorithme PID

PID_Compact est un régulateur PIDT1 avec anti-saturation et pondération de l'action P et de l'action D. L'algorithme PID fonctionne selon la formule suivante :

$$y = K_p \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

Symbole	Description	Paramètres correspondants de l'instruction PID_Compact
y	Valeur de sortie de l'algorithme PID	-
K_p	Gain proportionnel	Retain.CtrlParams.Gain
s	Opérateur de Laplace	-
b	Pondération de l'action P	Retain.CtrlParams.PWeighting
w	Consigne	CurrentSetpoint
x	Mesure	ScaledInput
T_I	Temps d'intégration	Retain.CtrlParams.Ti
T_D	Temps de dérivation	Retain.CtrlParams.Td
a	Coefficient pour l'action par dérivation (action par dérivation T1 = a × T_D)	Retain.CtrlParams.TdFiltRatio
c	Pondération de l'action D	Retain.CtrlParams.DWeighting

Schéma fonctionnel PID_Compact

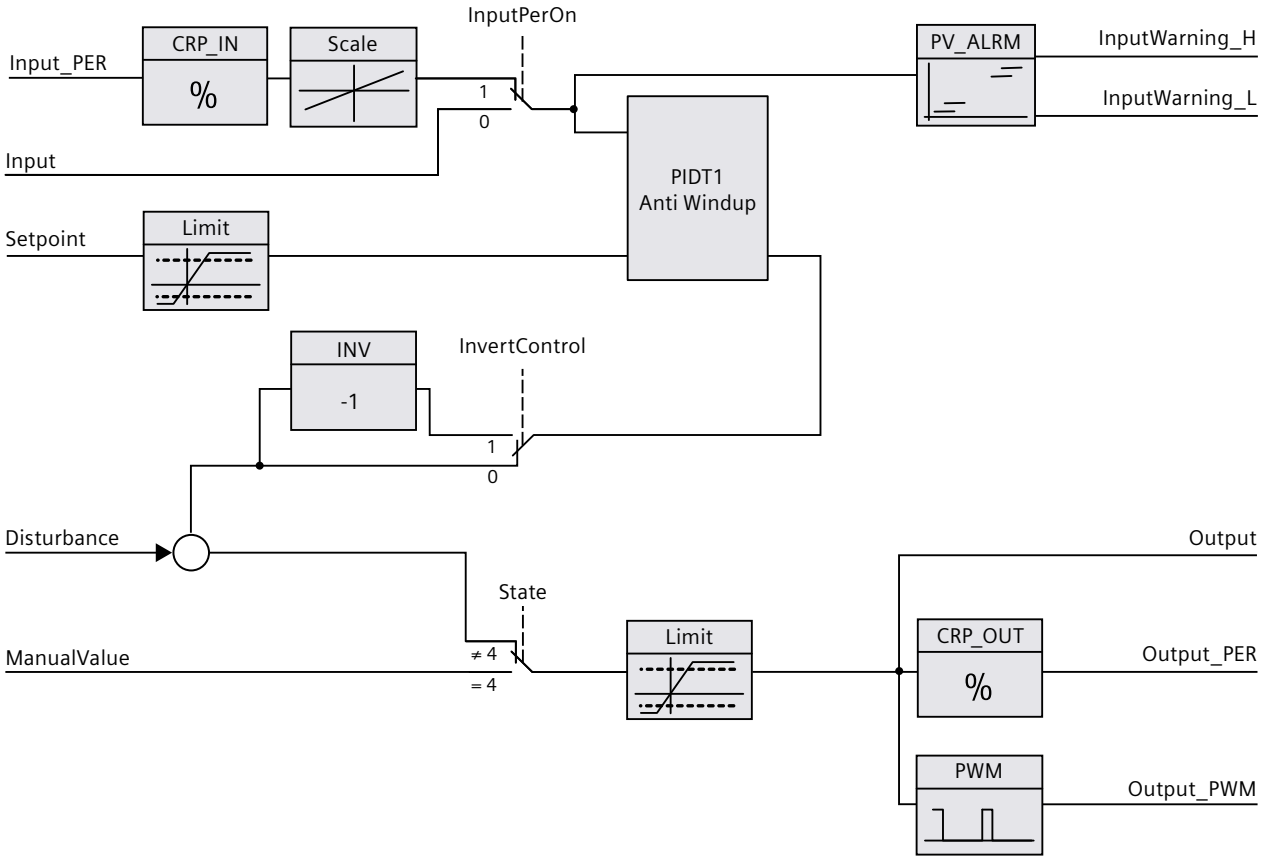
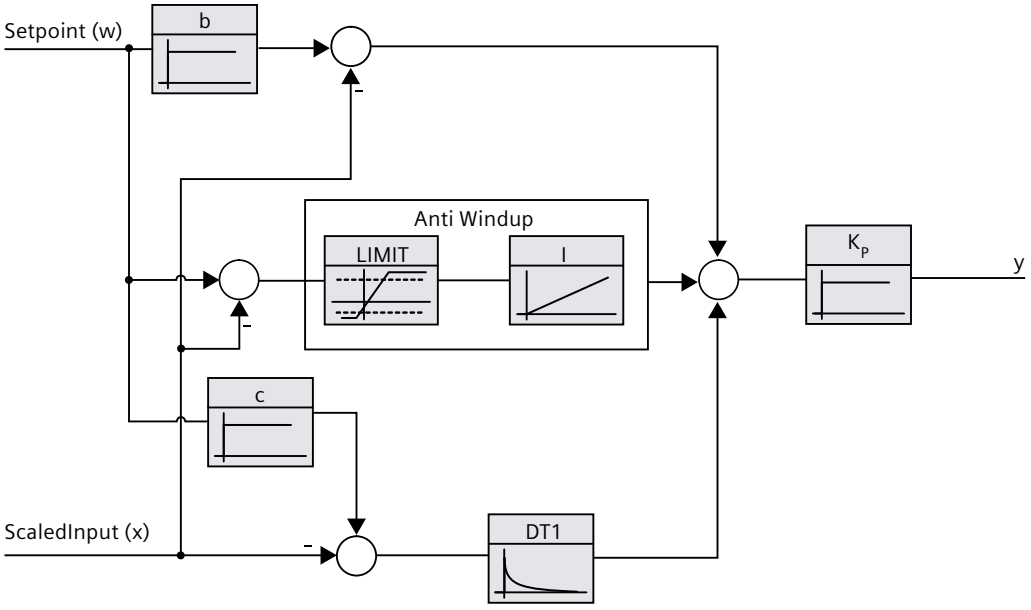


Schéma fonctionnel PIDT1 avec anti-saturation



Appel

PID_Compact est appelé à intervalle de temps constant d'une alarme cyclique de l'OB.

Chargement dans l'appareil

Les valeurs effectives de variables rémanentes ne sont actualisées que si vous chargez entièrement PID_Compact.
Charger des objets technologiques dans l'appareil (Page 48)

Démarrage

Au démarrage de la CPU, PID_Compact démarre dans le mode de fonctionnement enregistré aux paramètres d'entrée/sortie Mode. Réglez RunModeByStartup = FALSE pour passer en mode de fonctionnement "Inactif" au démarrage.

Comportement en cas d'erreur

Le comportement en cas d'erreur dépend des variables SetSubstituteOutput et ActivateRecoverMode. Si ActivateRecoverMode = TRUE, le comportement dépend en outre de la nature de l'erreur.

SetSubstituteOutput	**ActivateRecoverMode**	**Éditeur de configuration > Valeur de sortie > Régler Output**	**Comportement**
non signif.	FALSE	Zéro (inactif)	Passage au mode de fonctionnement "Inactif" (State = 0) La valeur.0 0 est transmise à l'actionneur.
FALSE	TRUE	Valeur de sortie actuelle pour la durée de l'erreur	Passage au mode de fonctionnement "Valeur de sortie de remplacement avec surveillance des erreurs" (State = 5) La valeur de sortie actuelle est transmise à l'actionneur pour la durée de l'erreur.
TRUE	TRUE	Valeur de sortie de remplacement pour la durée de l'erreur	Passage au mode de fonctionnement "Valeur de sortie de remplacement avec surveillance des erreurs" (State = 5) La valeur à SubstituteOutput est transmise à l'actionneur pour la durée de l'erreur.

PID_Compact utilise ManualValue comme valeur de sortie en mode manuel, sauf si ManualValue est invalide. Si ManualValue est invalide, SubstituteOutput est utilisé. Si ManualValue et SubstituteOutput sont invalides, Config.OutputLowerLimit est utilisé.
Le paramètre Error indique si une erreur est actuellement présente. Quand l'erreur a disparu, Error = FALSE. Le paramètre ErrorBits indique les erreurs qui se sont produites. ErrorBits est réinitialisé par un front montant à Reset ou ErrorAck.

10.1.4.3 Mode opératoire PID_Compact à partir de V2

Surveiller les limites de mesure

Vous définissez une limite supérieure et une limite inférieure de la mesure dans les variables Config.InputUpperLimit et Config.InputLowerLimit. Si la mesure se trouve en dehors de ces limites, une erreur survient (ErrorBits = 0001h).

Vous définissez une limite d'alerte supérieure et une limite d'alerte inférieure de la mesure dans les variables Config.InputUpperWarning et Config.InputLowerWarning. Si la mesure se trouve en dehors de ces limites d'alerte, une alerte survient (Warning = 0040h) et le paramètre de sortie InputWarning_H ou InputWarning_L passe à TRUE.

Limiter consigne

Vous définissez une limite supérieure et inférieure de la consigne dans les variables Config.SetpointUpperLimit et Config.SetpointLowerLimit. PID_Compact limite automatiquement la consigne aux limites de la mesure. Vous pouvez limiter la consigne à une plage inférieure. PID_Compact contrôle si cette plage se trouve dans les limites de la mesure. Si la consigne se trouve hors de ces limites, les limites inférieure et supérieure sont utilisées comme consigne et le paramètre de sortie SetpointLimit_H ou SetpointLimit_L passe à TRUE.

La consigne est limitée dans tous les modes de fonctionnement.

Limiter la valeur de réglage

Vous déterminez une limite supérieure et une limite inférieure de la valeur de réglage dans les variables Config.OutputUpperLimit et Config.OutputLowerLimit. Output, ManualValue et SubstituteOutput sont limités à ces valeurs. Les limites de valeur de réglage doivent être dans le sens de régulation.

Les valeurs admissibles pour les limites de la valeur de réglage dépendent de Output utilisé.

Output	de -100.0 à 100.0 %
Output_PER	de -100.0 à 100.0 %
Output_PWM	de 0.0 à 100.0 %

Règle à appliquer :
OutputUpperLimit > OutputLowerLimit

REMARQUE

Utilisation avec deux actionneurs ou plus

PID_Compact ne convient pas pour l'utilisation avec deux actionneurs ou plus (p. ex. dans des applications de chauffage ou de refroidissement), car des actionneurs différents ont besoin de paramètres PID différents pour obtenir un bon comportement de régulation. Pour les applications faisant appel à deux actionneurs d'actions opposées, utilisez PID_Temp.

Valeur de réglage de remplacement

En cas d'erreur, PID_Compact peut fournir une valeur de réglage de remplacement que vous définissez au niveau de la variable SubstituteOutput. La valeur de réglage de remplacement doit être dans les limites de la valeur de réglage.

Surveiller la validité des signaux

En cas d'utilisation, la validité des valeurs des paramètres suivants est surveillée :

- Setpoint
- Input
- Input_PER
- Disturbance
- ManualValue
- SubstituteOutput
- Output
- Output_PER
- Output_PWM

Surveillance du temps d'échantillonnage PID_Compact

Le temps d'échantillonnage correspond idéalement au temps de cycle de l'OB appelant. L'instruction PID_Compact permet de mesurer l'intervalle de temps entre deux appels respectifs. Le résultat est le temps d'échantillonnage actuel. Lors de chaque changement du mode de fonctionnement et à la première mise en route, une moyenne est calculée à partir des 10 premiers temps d'échantillonnage. Si la période d'échantillonnage actuelle diverge trop de cette valeur moyenne, une erreur survient (Error = 0800h).

Une erreur survient en cours d'optimisation si :

- Nouvelle valeur moyenne >= 1,1 x ancienne valeur moyenne
- Nouvelle valeur moyenne <= 0,9 x ancienne valeur moyenne

Une erreur survient en mode automatique si :

- Nouvelle valeur moyenne >= 1,5 x ancienne valeur moyenne
- Nouvelle valeur moyenne <= 0,5 x ancienne valeur moyenne

Si la surveillance du temps d'échantillonnage est désactivée (CycleTime.EnMonitoring = FALSE), vous pouvez aussi appeler PID_Compact dans OB1. Dans ce cas, vous devez accepter une moindre qualité de régulation du fait de la fluctuation du temps d'échantillonnage.

Temps d'échantillonnage de l'algorithme PID

Comme le système réglé nécessite un certain temps pour réagir à une modification de la valeur de réglage, il est judicieux de ne pas calculer cette valeur à chaque cycle. Le temps d'échantillonnage de l'algorithme PID est le temps entre deux calculs de valeurs de réglage. Il est déterminé pendant l'optimisation et arrondi à un multiple du temps de cycle. Toutes les autres fonctions de PID_Compact sont exécutées lors de chaque appel.

Si vous utilisez Output_PWM, la période d'échantillonnage de l'algorithme PID est utilisée comme durée de la période de la modulation de largeur d'impulsions. La précision du signal de sortie est déterminée par le rapport entre la période d'échantillonnage de l'algorithme PID et le temps de cycle de l'OB. Pour cette raison, il est recommandé que le temps de cycle ne dépasse pas un dixième de la période d'échantillonnage de l'algorithme PID.

Sens de régulation

La plupart du temps, une augmentation de la mesure doit être atteinte avec une augmentation de la valeur de réglage. Dans ce cas, on parle d'un sens de régulation normal. Il peut être nécessaire d'inverser le sens de régulation pour les refroidissements et les régulations d'écoulement. PID_Compact ne fonctionne pas avec un gain proportionnel négatif. Si InvertControl = TRUE, un signal d'écart croissant provoque une diminution de la valeur de réglage. Le sens de régulation est pris en compte aussi pendant l'optimisation préalable et l'optimisation fine.

10.1.4.4 Paramètres d'entrée PID_Compact à partir de V2

Les noms des paramètres suivants sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Tableau 10-1

Paramètre	Type de données	Valeur par défaut	Description
Setpoint	REAL	0.0	Consigne du régulateur PID en mode automatique
Input	REAL	0.0	Une variable du programme utilisateur est utilisée comme source de la mesure. Si vous utilisez le paramètre Input, il faut que Config.InputPerOn = FALSE.
Input_PER	INT	0	Une entrée analogique est utilisée comme source de la mesure. Si vous utilisez le paramètre Input_PER, il faut que Config.InputPerOn = TRUE.
Disturbance	REAL	0.0	Grandeur perturbatrice ou valeur de commande anticipatrice
ManualEnable	BOOL	FALSE	• Le front FALSE -> TRUE active le mode de fonctionnement "Mode manuel", State = 4, Mode ne change pas. Tant que ManualEnable = TRUE, vous ne pouvez pas changer le mode de fonctionnement via un front montant à ModeActivate ni utilisez la boîte de dialogue de mise en service. • Le front FALSE -> TRUE active le mode de fonctionnement prédéfini à Mode.

Paramètre	Type de données	Valeur par défaut	Description
			Il est recommandé de modifier le mode de fonctionnement uniquement via ModeActivate.
ManualValue	REAL	0.0	Valeur manuelle Cette valeur est utilisée comme valeur de réglage en mode manuel. Les valeurs autorisées sont comprises entre Config.OutputLowerLimit et Config.OutputUpperLimit.
ErrorAck	BOOL	FALSE	• Front FALSE -> TRUE ErrorBits et Warning sont remis à zéro.
Reset	BOOL	FALSE	Effectue un redémarrage du régulateur. • Front FALSE -> TRUE – Passage en mode de fonctionnement "Inactif" – ErrorBits et Warnings sont remis à zéro. • Tant que Reset = TRUE, – PID_Compact reste en mode de fonctionnement "Inactif" (State = 0) – vous ne pouvez pas changer de mode de fonctionnement avec Mode et ModeActivate ou ManualEnable – vous ne pouvez pas utiliser le dialogue de mise en service. • Front TRUE -> FALSE – Si ManualEnable = FALSE, PID_Compact passe au mode de fonctionnement qui est enregistré dans Mode. – Si Mode = 3, l'action I est traitée comme ce qui est configuré avec la variable IntegralResetMode.
ModeActivate	BOOL	FALSE	• Front FALSE -> TRUE PID_Compact passe au mode de fonctionnement qui est enregistré dans Mode".

10.1.4.5 Paramètres de sortie PID_Compact à partir de V2

Les noms des paramètres suivants sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Tableau 10-2

Parameter	Type de données	Valeur par défaut	Description
ScaledInput	REAL	0.0	Mesure mise à l'échelle
Les sorties "Output", "Output_PER" et "Output_PWM" peuvent être utilisées en parallèle.			
Output	REAL	0.0	Valeur de réglage au format REAL
Output_PER	INT	0	Valeur de réglage analogique
Output_PWM	BOOL	FALSE	Valeur de réglage modulée en largeur d'impulsion La valeur de réglage est calculée au moyen de temps d'activation et de désactivation variables.

Parameter	Type de données	Valeur par défaut	Description
SetpointLimit_H	BOOL	FALSE	Quand SetpointLimit_H = TRUE, la limite supérieure absolue de la consigne est atteinte (Setpoint ≥ Config.SetpointUpperLimit). La consigne est limitée à Config.SetpointUpperLimit .
SetpointLimit_L	BOOL	FALSE	Quand SetpointLimit_L = TRUE, la limite inférieure absolue de la consigne est atteinte (Setpoint ≤ Config.SetpointLowerLimit). La consigne est limitée à Config.SetpointLowerLimit .
InputWarning_H	BOOL	FALSE	Si InputWarning_H = TRUE, la limite d'alerte supérieure de la mesure est atteinte ou dépassée.
InputWarning_L	BOOL	FALSE	Si InputWarning_L = TRUE, la limite d'alerte inférieure de la mesure est atteinte ou dépassée.
State	INT	0	Le paramètre State (Page 264) affiche le mode de fonctionnement actuel du régulateur PID. Le mode de fonctionnement peut être modifié avec le paramètre d'entrée Mode et un front montant à ModeActivate. • State = 0 : Inactif • State = 1 : Optimisation préalable • State = 2 : Optimisation fine • State = 3 : Mode automatique • State = 4 : Mode manuel • State = 5 : Valeur de réglage de remplacement avec surveillance des erreurs
Error	BOOL	FALSE	Si Error = TRUE, un message d'erreur au moins existe dans ce cycle.
ErrorBits	DWORD	DW#16#0	Le paramètre ErrorBits (Page 268) signale la présence de messages d'erreur. ErrorBits est rémanent et réinitialisé à Reset ou ErrorAck en cas de front montant.

10.1.4.6 Paramètre d'entrée/sortie PID_Compact à partir de V2

Les noms des paramètres suivants sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Tableau 10-3

Parameter	Type de données	Valeur par défaut	Description
Mode	INT	4	Définissez le mode de fonctionnement à Mode dans lequel PID_Compact doit passer. Sont possibles : • Mode = 0 : Inactif • Mode = 1 : Optimisation préalable • Mode = 2 : Optimisation fine • Mode = 3 : Mode automatique • Mode = 4 : Mode manuel Le mode de fonctionnement est activé par : • Front montant à ModeActivate • Front descendant à Reset • Front descendant à ManualEnable • Démarrage à froid de la CPU, si RunModeByStartup = TRUE

Parameter	Type de données	Valeur par défaut	Description
			Mode est rémanent. Pour une description détaillée des modes de fonctionnement, voir Paramètres State et Mode à partir de V2 (Page 264).

Voir aussi

Paramètres State et Mode à partir de V2 (Page 264)

10.1.4.7 Variables statiques PID_Compact à partir de V2

REMARQUE

Faites passer les variables identifiées par (1) seulement en mode de fonctionnement "Inactif" pour éviter un comportement erroné du régulateur PID.

Sauf indications contraires, les noms des variables suivantes sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Variable	Type de données	Valeur par défaut	Description
IntegralResetMode	INT	jusqu'à V2.2 : 1, à partir de V2.3 : 4	La Variable IntegralResetMode à partir de V2 (Page 272) définit comment l'action I PIDCtrl.IntegralSum est préréglée lorsque le mode de fonctionnement passe de "Inactif" à "Mode automatique". Ce réglage s'applique uniquement pour un cycle. Sont possibles : • IntegralResetMode = 0 : lissage • IntegralResetMode = 1 : supprimer • IntegralResetMode = 2 : conserver • IntegralResetMode = 3 : valeur par défaut • IntegralResetMode = 4 : comme variation de la consigne (uniquement pour PID_Compact en version ≥ 2.3)
OverwriteInitialOutputValue	REAL	0.0	Si l'une des conditions suivantes est remplie, l'action par intégration PIDCtrl.IntegralSum est préréglée automatiquement comme si on avait eu Output = OverwriteInitialOutputValue dans le cycle précédent : • IntegralResetMode = 3 lorsque le mode de fonctionnement passe de "Inactif" à "Mode automatique" • IntegralResetMode = 3, Front TRUE -> FALSE sur le paramètre Reset et le paramètre Mode = 3 • PIDCtrl.PIDInit = TRUE en “mode automatique” (disponible à partir de PID_Compact version 2.3)
RunModeByStartup	BOOL	TRUE	Activer le mode de fonctionnement à Mode après le démarrage de la CPU Si RunModeByStartup = TRUE, au démarrage de la CPU, PID_Compact démarre dans le mode de fonctionnement enregistré à Mode. Si RunModeByStartup = FALSE, PID_Compact reste en mode "Inactif" après démarrage de la CPU.

Variable	Type de données	Valeur par défaut	Description
LoadBackUp	BOOL	FALSE	Si LoadBackUp = TRUE, le dernier jeu de paramètres PID est rechargé depuis la structure CtrlParamsBackUp. Le jeu a été enregistré avant la dernière optimisation. LoadBackUp est remis automatiquement à FALSE.
PhysicalUnit	INT	0	Unité physique de la mesure et de la consigne, par ex. °C ou °F. PhysicalUnit sert à afficher les valeurs dans les éditeurs et n'a aucune influence sur le comportement de l'algorithme de régulation dans la CPU. En cas d'importation de PID_Compact jusqu'à la version 2.4 via l'API Openness, PhysicalUnit est remis à la valeur par défaut. À partir de la version 3.0, la valeur de PhysicalUnit est conservée en cas d'importation.
PhysicalQuantity	INT	0	Grandeur physique de la mesure et de la consigne, par ex. température. PhysicalQuantity sert à afficher les valeurs dans les éditeurs et n'a aucune influence sur le comportement de l'algorithme de régulation dans la CPU. En cas d'importation de PID_Compact jusqu'à V2.4 via l'API Openness, la valeur par défaut de PhysicalQuantity est rétablie. À partir de la version 3.0, la valeur de PhysicalQuantity est conservée lors de l'importation.
ActivateRecoverMode	BOOL	TRUE	La Variable ActivateRecoverMode à partir de V2 (Page 270) détermine le comportement en cas d'erreur.
Warning	DWORD	0	La Variable Warning à partir de V2 (Page 271) affiche les avertissements depuis Reset = TRUE ou ErrorAck =TRUE. Warning est rémanent.
Progress	REAL	0.0	Progression de la phase actuelle de l'optimisation en % (0.0 à 100.0)
CurrentSetpoint	REAL	0.0	CurrentSetpoint indique toujours la consigne actuellement opérante. Cette valeur est gelée pendant l'optimisation.
CancelTuningLevel	REAL	10.0	Fluctuations admissibles de la valeur de consigne pendant l'optimisation. Une optimisation est interrompue si : • Setpoint > CurrentSetpoint + CancelTuningLevel ou • Setpoint < CurrentSetpoint - CancelTuningLevel
SubstituteOutput	REAL	0.0	Valeur de sortie de remplacement Lorsque les conditions suivantes sont remplies, la valeur de sortie de remplacement est utilisée comme valeur de sortie : • Il y a une ou plusieurs erreurs en mode automatique pour lesquelles ActivateRecoverMode est opérant • SetSubstituteOutput = TRUE • ActivateRecoverMode = TRUE Config.OutputUpperLimit ≥ SubstituteOutput ≥ Config.OutputLowerLimit

Variable	Type de données	Valeur par défaut	Description
SetSubstituteOutput	BOOL	TRUE	Choix de la valeur de sortie tant qu'une erreur est présente (State = 5) • Si SetSubstituteOutput = TRUE et ActivateRecoverMode = TRUE, la valeur de sortie de remplacement configurée SubstituteOutput s'affiche tant qu'une erreur est présente. • Si SetSubstituteOutput = FALSE et ActivateRecoverMode = TRUE, l'actionneur reste à la valeur de sortie actuelle pendant la durée de l'erreur. • Si ActivateRecoverMode = FALSE, alors SetSubstituteOutput ne s'applique pas. • Si SubstituteOutput est invalide (ErrorBits = 16#0002_0000), la valeur de sortie de remplacement ne peut pas être affichée. Dans ce cas, c'est la limite inférieure de la valeur de sortie (Config.OutputLowerLimit) qui est utilisée comme valeur de sortie.
Config.InputPerOn(1)	BOOL	TRUE	Si InputPerOn = TRUE, c'est le paramètre Input_PER qui est utilisé pour l'acquisition de la mesure. Si InputPerOn = FALSE, c'est le paramètre Input qui est utilisé.
Config.InvertControl(1)	BOOL	FALSE	Inversion du sens de régulation Si InvertControl = TRUE, un signal d'écart croissant provoque une diminution de la valeur de sortie.
Config.OutputSelect	INT	0	Sélection de la valeur de sortie (à partir de la version 3.0) : • OutputSelect = 0: Output_PER (analog) • OutputSelect = 1: Output • OutputSelect = 2: Output_PWM Config.OutputSelect sert à configurer le régulateur dans TIA Portal et n'a aucune influence sur le calcul des valeurs de sortie dans la CPU.
_Config.OutputSelect	INT	0	Sélection de la valeur de sortie (jusqu'à version 2.4) : • OutputSelect = 0: Output_PER (analogique) • OutputSelect = 1: Output • OutputSelect = 2: Output_PWM _Config.OutputSelect sert à configurer le régulateur dans TIA Portal et n'a aucune influence sur le calcul des valeurs de sortie dans la CPU. _Config.OutputSelect n'est pas disponible dans le bloc de données et ne peut être configuré que dans l'éditeur de configuration ou via API Openness. Lors de l'importation de PID_Compact via API Openness, la valeur par défaut de _Config.OutputSelect est rétablie.
Config.InputUpperLimit(1)	REAL	120.0	Limite supérieure de la mesure L'observation de cette limite est surveillée pour Input et Input_PER. Quand la limite est dépassée, une erreur est générée et la réaction dépend de ActivateRecoverMode. A l'entrée de périphérie, la mesure peut dépasser de 18 % au plus la plage normée (dépassement haut). La limite ne peut donc pas se trouver dépassée si vous utilisez

Variable	Type de données	Valeur par défaut	Description
			l'entrée de périphérie avec la valeur par défaut de la limite supérieure et la mise à l'échelle de la mesure. Au démarrage d'une optimisation préalable, le système contrôle, au moyen de la différence entre les limites supérieure et inférieure de la mesure, si l'écart entre consigne et mesure répond aux conditions nécessaires. InputUpperLimit > InputLowerLimit
Config.InputLowerLimit(1)	REAL	0.0	Limite inférieure de la mesure L'observation de cette limite est surveillée pour Input et Input_PER. Quand la limite est dépassée par le bas, une erreur est générée et la réaction dépend de ActivateRecoverMode. InputLowerLimit < InputUpperLimit
Config.InputUpperWarning(1)	REAL	3.402822e+38	Limite d'alerte supérieure de la mesure L'observation de cette limite est surveillée pour Input et Input_PER. Quand la limite est dépassée, un avertissement est généré au paramètre. Si vous configurez InputUpperWarning en dehors des limites de la mesure, la limite supérieure absolue configurée pour la mesure est utilisée comme limite d'alerte supérieure. Si vous configurez InputUpperWarning dans les limites de la mesure, cette valeur sera utilisée comme limite d'alerte supérieure. InputUpperWarning > InputLowerWarning
Config.InputLowerWarning(1)	REAL	-3.402822e+38	Limite d'alerte inférieure de la mesure L'observation de cette limite est surveillée pour Input et Input_PER. Quand la limite est dépassée par le bas, un avertissement est généré au paramètre Warning. Si vous configurez InputLowerWarning en dehors des limites de la mesure, la limite inférieure absolue configurée pour la mesure est utilisée comme limite d'alerte inférieure. Si vous configurez InputLowerWarning dans les limites de la mesure, cette valeur sera utilisée comme limite d'alerte inférieure. InputLowerWarning < InputUpperWarning
Config.OutputUpperLimit	REAL	100.0	Limite supérieure de la valeur de sortie Pour plus de détails, voir OutputLowerLimit 100.0 ≥ OutputUpperLimit > OutputLowerLimit
Config.OutputLowerLimit	REAL	0.0	Limite inférieure de la valeur de sortie Si Output ou Output_PER, la plage de valeurs de -100.0 à +100.0 s'applique, y compris le zéro. Pour -100.0 : Output_PER = -27648 ; pour +100.0 : Output_PER = 27648. Si Output_PWM, la plage de valeurs de 0.0 à +100.0 s'applique. Les limites de valeur de sortie doivent être dans le sens de régulation. À partir de la version 3.0, PID_Compact prend en charge la modification des limites de valeur de sortie en mode automatique. Jusqu'à la version 2.4, elles ne sont modifiables qu'en mode inactif ou en mode manuel. OutputLowerLimit < OutputUpperLimit

Variable	Type de données	Valeur par défaut	Description
Config.SetpointUpperLimit(1)	REAL	3.402822e+38	Limite supérieure de la consigne L'observation de cette limite est surveillée pour Setpoint. Quand la limite est dépassée, un avertissement est généré au paramètre Warning. Quand vous configurez SetpointUpperLimit en dehors des limites de la mesure, la limite supérieure absolue configurée pour la mesure est utilisée comme limite supérieure de la consigne. Quand vous configurez SetpointUpperLimit dans les limites de la mesure, cette valeur est utilisée comme limite supérieure de la consigne. SetpointUpperLimit > SetpointLowerLimit
Config.SetpointLowerLimit(1)	REAL	-3.402822e+38	Limite inférieure de la consigne L'observation de cette limite est surveillée pour Setpoint. Quand la limite est dépassée par le bas, un avertissement est généré au paramètre Warning. Quand vous configurez SetpointLowerLimit en dehors des limites de la mesure, la limite inférieure absolue configurée pour la mesure est utilisée comme limite inférieure de la consigne. Quand vous configurez SetpointLowerLimit dans les limites de la mesure, cette valeur est utilisée comme limite inférieure de la consigne. SetpointLowerLimit < SetpointUpperLimit
Config.MinimumOnTime(1)	REAL	0.0	Plus petit temps ON en secondes de la modulation de largeur d'impulsions, arrondi à MinimumOnTime = n×CycleTime.Value 100000.0 ≥ MinimumOnTime ≥ 0.0
Config.MinimumOffTime(1)	REAL	0.0	Plus petit temps OFF en secondes de la modulation de largeur d'impulsions, arrondi à MinimumOffTime = n×CycleTime.Value 100000.0 ≥ MinimumOffTime ≥ 0.0
Config.InputScaling.UpperPointIn(1)	REAL	27648.0	Mise à l'échelle Input_PER Haut UpperPointOut est mis à l'échelle au moyen des deux paires de valeurs UpperPointIn, LowerPointOut et LowerPointIn, ainsi que Input_PER. Opérant seulement quand Input_PER est utilisé pour l'acquisition de la mesure ((Config.InputPerOn = TRUE)). UpperPointIn > LowerPointIn
Config.InputScaling.LowerPointIn(1)	REAL	0.0	Mise à l'échelle Input_PER Bas UpperPointOut est mis à l'échelle au moyen des deux paires de valeurs UpperPointIn, LowerPointOut et LowerPointIn, ainsi que Input_PER. Opérant seulement quand Input_PER est utilisé pour l'acquisition de la mesure (Config.InputPerOn = TRUE). LowerPointIn < UpperPointIn
Config.InputScaling.UpperPointOut(1)	REAL	100.0	Mesure supérieure à l'échelle UpperPointOut est mis à l'échelle au moyen des deux paires de valeurs UpperPointIn, LowerPointOut et LowerPointIn, ainsi que Input_PER. Opérant seulement quand Input_PER est utilisé pour l'acquisition de la mesure (Config.InputPerOn = TRUE). UpperPointOut > LowerPointOut

Variable	Type de données	Valeur par défaut	Description
Config.InputScaling.LowerPointOut(1)	REAL	0.0	Mesure inférieure à l'échelle UpperPointOut est mis à l'échelle au moyen des deux paires de valeurs UpperPointIn, LowerPointOut et LowerPointIn, ainsi que Input_PER. Opérant seulement quand Input_PER est utilisé pour l'acquisition de la mesure (Config.InputPerOn = TRUE). LowerPointOut < UpperPointOut
CycleTime.StartEstimation	BOOL	TRUE	Si CycleTime.EnEstimation = TRUE, CycleTime.StartEstimation = TRUE démarre le calcul automatique de la période d'échantillonnage PID_Compact (temps de cycle de l'OB appelant). Après la fin de la mesure, on a CycleTime.StartEstimation = FALSE gesetzt.
CycleTime.EnEstimation	BOOL	TRUE	Si CycleTime.EnEstimation = TRUE, le temps d'échantillonnage PID_Compact est calculé automatiquement. Si CycleTime.EnEstimation = FALSE, la période d'échantillonnage PID_Compact n'est pas calculée automatiquement et vous devez configurer manuellement la CycleTime.Value correcte.
CycleTime.EnMonitoring	BOOL	TRUE	Si CycleTime.EnMonitoring = FALSE, le temps d'échantillonnage PID_Compact n'est pas surveillé. Si PID_Compact ne peut pas être exécuté durant la période d'échantillonnage, aucune erreur (ErrorBits=16#0000_0800) n'est émise et PID_Compact ne réagit pas comme vous l'avez configuré avec ActivateRecoverMode.
CycleTime.Value(1)	REAL	0.1	Période d'échantillonnage de PID_Compact (temps de cycle de l'OB appelant) en secondes CycleTime.Value est automatiquement déterminée et correspond normalement au temps de cycle de l'OB appelant.
Il est possible de charger les valeurs de la structure CtrlParamsBackUp avec LoadBackUp = TRUE.			
CtrlParamsBackUp.SetByUser	BOOL	FALSE	Valeur enregistrée de Retain.CtrlParams.SetByUser (à partir de la version 3.0)
CtrlParamsBackUp.Gain	REAL	1.0	Gain proportionnel enregistré
CtrlParamsBackUp.Ti	REAL	20.0	Période d'intégration enregistrée en secondes
CtrlParamsBackUp.Td	REAL	0.0	Temps de dérivation enregistré en secondes
CtrlParamsBackUp.TdFiltRatio	REAL	0.2	Coefficient de l'action par dérivation enregistré
CtrlParamsBackUp.PWeighting	REAL	1.0	Facteur de pondération de l'action P enregistré
CtrlParamsBackUp.DWeighting	REAL	1.0	Facteur de pondération de l'action D enregistré
CtrlParamsBackUp.Cycle	REAL	1.0	Période d'échantillonnage de l'algorithme PID enregistrée en secondes
CtrlParamsBackUp.DeadZone	REAL	0.0	Largeur de bande morte enregistrée (à partir de la version 3.0)
PIDSelfTune.SUT.CalculateParams	BOOL	FALSE	Les propriétés du système réglé sont enregistrées lors de l'optimisation. Si SUT.CalculateParams = TRUE, les paramètres pour l'optimisation préalable sont recalculés selon ces propriétés. Cela permet de modifier la méthode de calcul de paramètres sans répéter l'optimisation. SUT.CalculateParams est mis sur FALSE après le calcul.

Variable	Type de données	Valeur par défaut	Description
PIDSelfTune.SUT.TuneRule	INT	0	Calculer les paramètres pendant l'optimisation préalable selon la méthode : • SUT.TuneRule = 0 : PID selon Chien, Hrones et Reswick • SUT.TuneRule = 1 : PI selon Chien, Hrones et Reswick
PIDSelfTune.SUT.State	INT	0	La variable SUT.State indique la phase actuelle de l'optimisation préalable : • State = 0 : Initialiser l'optimisation préalable • State = 100 : Calculer l'écart type • State = 200 : Déterminer le point d'inflexion • State = 9900 : Optimisation préalable réussie • State = 1 : Optimisation préalable échouée
PIDSelfTune.TIR.RunIn	BOOL	FALSE	La variable RunIn permet de définir l'exécution d'une optimisation fine aussi sans optimisation préalable. • RunIn = FALSE Si l'optimisation fine est démarrée depuis le mode inactif ou manuel, une optimisation préalable est lancée. Si les conditions d'une optimisation préalable ne sont pas réunies, PID_Compact a le même comportement que lorsque RunIn = TRUE. Si l'optimisation fine est démarrée depuis le mode automatique, les paramètres PID existants sont utilisés pour un réglage sur la consigne. C'est seulement après cela que l'optimisation fine commence. Si l'optimisation préalable n'est pas possible, PID_Compact passe dans le mode de fonctionnement à partir duquel l'optimisation a été lancée. • RunIn = TRUE L'optimisation préalable est sautée. PID_Compact essaie d'atteindre la consigne avec la valeur de sortie mini ou maxi. Cela peut entraîner une suroscillation élevée. L'optimisation fine démarre ensuite automatiquement. RunIn est mis sur FALSE après l'optimisation fine.
PIDSelfTune.TIR.CalculateParams	BOOL	FALSE	Les propriétés du système réglé sont enregistrées lors de l'optimisation. Si TIR.CalculateParams = TRUE, les paramètres pour l'optimisation fine sont recalculés selon ces propriétés. Cela permet de modifier la méthode de calcul de paramètres sans répéter l'optimisation. TIR.CalculateParams est mis sur FALSE après le calcul.
PIDSelfTune.TIR.TuneRule	INT	0	Calculer les paramètres pendant l'optimisation fine selon la méthode : • TIR.TuneRule = 0 : PID automatique • TIR.TuneRule = 1 : PID rapide (comportement de régulation plus rapide avec des amplitudes plus fortes de la valeur de sortie qu'avec TIR.TuneRule = 2) • TIR.TuneRule = 2 : PID lent (comportement de régulation plus lent avec des amplitudes plus faibles de la valeur de sortie qu'avec TIR.TuneRule = 1) • TIR.TuneRule = 3 : PID Ziegler-Nichols • TIR.TuneRule = 4 : PI Ziegler-Nichols • TIR.TuneRule = 5 : P Ziegler-Nichols

Variable	Type de données	Valeur par défaut	Description
			Pour pouvoir répéter le calcul des paramètres PID avec TIR.CalculateParams et TIR.TuneRule = 0, 1 ou 2, il faut avoir exécuté l'optimisation fine précédente également avec TIR.TuneRule = 0, 1 ou 2. Si ce n'est pas le cas, TIR.TuneRule = 3 sera utilisé. Il est toujours possible de recalculer les paramètres PID avec TIR.CalculateParams et TIR.TuneRule = 3, 4 ou 5.
PIDSelfTune.TIR.State	INT	0	La variable TIR.State indique la phase actuelle de l'"optimisation fine" : • State = -100 L'optimisation fine n'est pas possible. Une optimisation préalable est d'abord réalisée. • State = 0 : Initialiser l'optimisation fine • State = 200 : Calculer l'écart type • State = 300 : Tentative d'atteindre la consigne • State = 400 : Essayer d'atteindre la consigne avec les paramètres PID existants (si optimisation préalable réussie) • State = 500 : Déterminer l'oscillation et calculer les paramètres • State = 9900 : Optimisation fine réussie • State = 1 : Optimisation fine échouée
PIDCtrl.IntegralSum(1)	REAL	0.0	Action I actuelle
PIDCtrl.PIDInit	BOOL	FALSE	PIDCtrl.PIDInit est disponible à partir de PID_Compact version 2.3. Si en "mode automatique" PIDCtrl.PIDInit = TRUE, l'action par intégration PIDCtrl.IntegralSum est préréglée automatiquement comme si on avait eu Output = OverwriteInitialOutputValue dans le cycle précédent. Cela est utilisable pour une Régulation en mode alternatif avec PID_Compact à partir de V2 (Page 96).
Retain.CtrlParams.SetByUser(1)	BOOL	FALSE	Activer la saisie manuelle des paramètres PID (à partir de la version 3.0) Quand Retain.CtrlParams.SetByUser = TRUE, les paramètres PID sont éditables. Retain.CtrlParams.SetByUser sert à configurer le régulateur dans TIA Portal et n'a aucune influence sur le comportement de l'algorithme de régulation dans la CPU. SetByUser est rémanent.
_Retain.CtrlParams.SetByUser(1)	BOOL	FALSE	Activer la saisie manuelle des paramètres PID (jusqu'à la version 2.4) Quand _Retain.CtrlParams.SetByUser = TRUE, les paramètres PID sont éditables. _Retain.CtrlParams.SetByUser sert à configurer le régulateur dans TIA Portal et n'a aucune influence sur le comportement de l'algorithme de régulation dans la CPU. _Retain.CtrlParams.SetByUser n'est pas disponible dans le bloc de données et ne peut être configuré que dans l'éditeur de configuration ou via API Openness. Lors de l'importation de PID_Compact via API Openness, la valeur par défaut de _Retain.CtrlParams.SetByUser est rétablie.

Variable	Type de données	Valeur par défaut	Description
Retain.CtrlParams.Gain(1)	REAL	1.0	Gain proportionnel actif PID_Compact ne fonctionne pas avec un gain proportionnel négatif. Utilisez la variable Config.InvertControl pour inverser le sens de régulation. Gain est rémanent. Gain ≥ 0.0
Retain.CtrlParams.Ti(1)	REAL	20.0	• CtrlParams.Ti > 0.0 : Temps d'intégration actif en secondes • CtrlParams.Ti = 0.0 : L'action I est désactivée Ti est rémanent. 100000.0 ≥ Ti ≥ 0.0
Retain.CtrlParams.Td(1)	REAL	0.0	• CtrlParams.Td > 0.0 : Temps de dérivation actif en secondes • CtrlParams.Td = 0.0 : l'action D est désactivée Td est rémanent. 100000.0 ≥ Td ≥ 0.0
Retain.CtrlParams.TdFiltRatio(1)	REAL	0.2	Coefficient actif de l'action par dérivation L'effet de l'action D est retardé par le coefficient de l'action par dérivation. Action par dérivation = Temps de dérivation x Coefficient de l'action par dérivation • 0.0 : L'action D n'est active que pour un seul cycle et est donc quasiment inactive. • 0.5 : Cette valeur a fait ses preuves dans la pratique pour les systèmes réglés avec **une** constante de temps dominante. • > 1.0 : Plus le coefficient est grand, plus l'effet de l'action D est retardé. TdFiltRatio est rémanent. TdFiltRatio ≥ 0.0
Retain.CtrlParams.PWeighting(1)	REAL	1.0	Pondération active de l'action P En cas de modification de consigne, vous pouvez réduire l'action P. Les valeurs judicieuses sont comprises entre 0.0 et 1.0. • 1.0 : Action P totalement opérante si modification de la consigne • 0.0 : Action P non opérante si modification de la consigne En cas de variation de la mesure, l'action P est toujours totalement opérante. PWeighting est rémanent. 1.0 ≥ PWeighting ≥ 0.0
Retain.CtrlParams.DWeighting(1)	REAL	1.0	Pondération active de l'action D En cas de modification de consigne, vous pouvez réduire l'action D. Les valeurs judicieuses sont comprises entre 0.0 et 1.0. • 1.0 : En cas de modification de la consigne, l'action D est totalement opérante • 0.0 : En cas de modification de la consigne, l'action D n'est pas opérante En cas de variation de la mesure, l'action D est toujours totalement opérante. DWeighting est rémanent.

Variable	Type de données	Valeur par défaut	Description
			1.0 ≥ DWeighting ≥ 0.0
Retain.CtrlParams.Cycle(1)	REAL	1.0	Temps d'échantillonnage actif de l'algorithme PID en secondes CtrlParams.Cycle est déterminé pendant l'optimisation et arrondi à un multiple entier de CycleTime.Value. CtrlParams.Cycle est utilisé comme période de la modulation de largeur d'impulsion. Cycle est rémanent. 100000.0 ≥ Cycle > 0.0
Retain.CtrlParams.DeadZone(1)	REAL	0.0	Largeur de bande morte active CtrlParams.DeadZone est disponible à partir de PID_Compact version 3.0. Avec CtrlParams.DeadZone = 0.0, la zone morte est désactivée. CtrlParams.DeadZone n'est pas réglé automatiquement ni adapté pendant l'optimisation. Vous devez configurer correctement CtrlParams.DeadZone manuellement. Lorsque la zone morte est activée, un signal d'écart (écart entre consigne et mesure) permanent peut s'établir. Cela peut avoir un effet négatif sur l'exécution d'une optimisation fine. DeadZone est rémanent. DeadZone ≥ 0.0

Voir aussi

Variable ActivateRecoverMode à partir de V2 (Page 270)
Variable Warning à partir de V2 (Page 271)
Charger des objets technologiques dans l'appareil (Page 48)

10.1.4.8 Modifications de l'interface PID_Compact à partir de V2

Le tableau suivant indique ce qui a changé sur l'interface de l'instruction PID_Compact.

PID_Compact V1	PID_Compact à partir de V2	Modification
Input_PER	Input_PER	Type de données de Word à Int
	Disturbance	Nouveau
	ErrorAck	Nouveau
	ModeActivate	Nouveau
Output_PER	Output_PER	Type de données de Word à Int
Error	ErrorBits	Renommé
	Error	Nouveau
	Mode	Nouveau
sb_RunModeByStartup	RunModeByStartup	Fonction
	IntegralResetMode	
	OverwriteInitialOutputValue	Nouveau

PID_Compact V1	PID_Compact à partir de V2	Modification
	SetSubstituteOutput	Nouveau
	CancelTuningLevel	Nouveau
	SubstituteOutput	Nouveau

Le tableau suivant indique quelles variables ont été renommées.

PID_Compact V1.x	PID_Compact à partir de V2
sb_GetCycleTime	CycleTime.StartEstimation
sb_EnCyclEstimation	CycleTime.EnEstimation
sb_EnCyclMonitoring	CycleTime.EnMonitoring
sb_RunModeByStartup	RunModeByStartup
si_Unit	PhysicalUnit
si_Type	PhysicalQuantity
sd_Warning	Warning
sBackUp.r_Gain	CtrlParamsBackUp.Gain
sBackUp.r_Ti	CtrlParamsBackUp.Ti
sBackUp.r_Td	CtrlParamsBackUp.Td
sBackUp.r_A	CtrlParamsBackUp.TdFiltRatio
sBackUp.r_B	CtrlParamsBackUp.PWeighting
sBackUp.r_C	CtrlParamsBackUp.DWeighting
sBackUp.r_Cycle	CtrlParamsBackUp.Cycle
sPid_Calc.r_Cycle	CycleTime.Value
sPid_Calc.b_RunIn	PIDSelfTune.TIR.RunIn
sPid_Calc.b_CalcParamSUT	PIDSelfTune.SUT.CalculateParams
sPid_Calc.b_CalcParamTIR	PIDSelfTune.TIR.CalculateParams
sPid_Calc.i_CtrlTypeSUT	PIDSelfTune.SUT.TuneRule
sPid_Calc.i_CtrlTypeTIR	PIDSelfTune.TIR.TuneRule
sPid_Calc.r_Progress	Progress
sPid_Cmpt.r_Sp_Hlm	Config.SetpointUpperLimit
sPid_Cmpt.r_Sp_Llm	Config.SetpointLowerLimit
sPid_Cmpt.r_Pv_Norm_IN_1	Config.InputScaling.LowerPointIn
sPid_Cmpt.r_Pv_Norm_IN_2	Config.InputScaling.UpperPointIn
sPid_Cmpt.r_Pv_Norm_OUT_1	Config.InputScaling.LowerPointOut
sPid_Cmpt.r_Pv_Norm_OUT_2	Config.InputScaling.UpperPointOut
sPid_Cmpt.r_Lmn_Hlm	Config.OutputUpperLimit
sPid_Cmpt.r_Lmn_Llm	Config.OutputLowerLimit
sPid_Cmpt.b_Input_PER_On	Config.InputPerOn
sPid_Cmpt.b_LoadBackUp	LoadBackUp
sPid_Cmpt.b_InvCtrl	Config.InvertControl
sPid_Cmpt.r_Lmn_Pwm_PPTm	Config.MinimumOnTime

PID_Compact V1.x	PID_Compact à partir de V2
sPid_Cmpt.r_Lmn_Pwm_PBTm	Config.MinimumOffTime
sPid_Cmpt.r_Pv_Hlm	Config.InputUpperLimit
sPid_Cmpt.r_Pv_Llm	Config.InputLowerLimit
sPid_Cmpt.r_Pv_HWrn	Config.InputUpperWarning
sPid_Cmpt.r_Pv_LWrn	Config.InputLowerWarning
sParamCalc.i_Event_SUT	PIDSelfTune.SUT.State
sParamCalc.i_Event_TIR	PIDSelfTune.TIR.State
sRet.i_Mode	sRet.i_Mode est supprimé. Le mode de fonctionnement est modifié par Mode et ModeActivate.
sRet.r_Ctrl_Gain	Retain.CtrlParams.Gain
sRet.r_Ctrl_Ti	Retain.CtrlParams.Ti
sRet.r_Ctrl_Td	Retain.CtrlParams.Td
sRet.r_Ctrl_A	Retain.CtrlParams.TdFiltRatio
sRet.r_Ctrl_B	Retain.CtrlParams.PWeighting
sRet.r_Ctrl_C	Retain.CtrlParams.DWeighting
sRet.r_Ctrl_Cycle	Retain.CtrlParams.Cycle

10.1.4.9 Paramètres State et Mode à partir de V2

Corrélation entre les paramètres

Le paramètre State affiche le mode de fonctionnement actuel du régulateur PID. Vous ne pouvez pas modifier le paramètre State.

Avec un front montant à ModeActivate, PID_Compact passe en mode de fonctionnement enregistré au paramètre d'entrée/sortie Mode.

Quand la CPU est mise en route ou passe de STOP à RUN, PID_Compact démarre dans le mode de fonctionnement enregistré à Mode. Pour laisser PID_Compact en mode "Inactif", mettez RunModeByStartup = FALSE.

Signification des valeurs

State/Mode	Description du mode de fonctionnement
0	Inactif En mode de fonctionnement "Inactif", la valeur de réglage 0.0 est toujours affichée, indépendamment de Config.OutputUpperLimit et Config.OutputLowerLimit. La modulation de largeur d'impulsions est désactivée.
1	Optimisation préalable L'optimisation préalable détermine la réponse du processus à un échelon de la valeur de réglage et recherche le point d'inflexion. Les paramètres PID sont calculés à partir de l'incrémentation maximale et du temps mort du système réglé. Les meilleurs paramètres PID sont obtenus pendant l'exécution d'une optimisation préalable et d'une optimisation fine.

State/Mode	Description du mode de fonctionnement
	Conditions pour une optimisation préalable : • Mode inactif (State = 0), manuel (State = 4) ou automatique (State = 3) • ManualEnable = FALSE • Reset = FALSE • La mesure ne doit pas être trop proche de la consigne. |Setpoint - Input| > 0.3 * | Config.InputUpperLimit - Config.InputLowerLimit| et |Setpoint - Input| > 0.5 * |Setpoint| • La consigne et la mesure se trouvent dans les limites configurées. Plus la mesure est stable, plus il sera facile de déterminer des paramètres PID précis. Un bruit de la mesure est acceptable tant que la croissance de la mesure est nettement supérieure au bruit. La consigne est gelée dans la variable CurrentSetpoint. Une optimisation est interrompue si : • Setpoint > CurrentSetpoint + CancelTuningLevel ou • Setpoint < CurrentSetpoint - CancelTuningLevel Avant que les paramètres PID soient recalculés, ils sont sauvegardés et peuvent être réactivés avec LoadBackUp. Après une optimisation préalable réussie, le mode de fonctionnement passe en automatique ; si l'optimisation préalable échoue, le mode de fonctionnement change en fonction de ActivateRecoverMode. La phase d'optimisation préalable est indiquée par PIDSelfTune.SUT.State. Pour démarrer l'optimisation préalable à partir du mode automatique, il est conseillé d'effectuer la modification nécessaire de la valeur de consigne en même temps que le front montant sur ModeActivate. Si la valeur de consigne est modifiée en premier et que l'optimisation préalable est démarrée ultérieurement, la valeur de réglage est adaptée en conséquence en mode automatique et entraîne une modification de la mesure. Cela peut avoir un effet négatif sur l'optimisation préalable ultérieure ou l'empêcher de démarrer.
2	Optimisation fine L'optimisation fine génère une oscillation constante limitée de la mesure. Les paramètres PID sont recalculés à partir de l'amplitude et de la fréquence de cette oscillation. Les paramètres PID de l'optimisation fine montrent généralement un meilleur comportement de référence et de perturbation que les paramètres PID de l'optimisation préalable. Les meilleurs paramètres PID sont obtenus pendant l'exécution d'une optimisation préalable et d'une optimisation fine. PID_Compact essaie automatiquement de créer une oscillation supérieure au bruit de la mesure. La stabilité de la mesure n'influence l'optimisation fine que de manière insignifiante. La consigne est gelée dans la variable CurrentSetpoint. Une optimisation est interrompue si : • Setpoint > CurrentSetpoint + CancelTuningLevel ou • Setpoint < CurrentSetpoint - CancelTuningLevel Avant que les paramètres PID soient recalculés, ils sont sauvegardés et peuvent être réactivés avec LoadBackUp. Conditions pour une optimisation fine : • Aucune perturbation n'est attendue. • La consigne et la mesure sont dans les limites configurées • ManualEnable = FALSE • Reset = FALSE • Mode automatique (State = 3), inactif (State = 0) ou mode manuel (State = 4) L'optimisation fine se déroule de la manière suivante au démarrage : • Mode automatique (State = 3) Si vous souhaitez améliorer les paramètres PID existants à l'aide de l'optimisation, démarrez l'optimisation fine à partir du mode automatique. PID_Compact régule avec les paramètres PID existants jusqu'à ce que la boucle de régulation soit en régime établi et que les conditions pour une optimisation fine soient remplies. C'est seulement après cela que l'optimisation fine commence. • Inactif (State = 0) ou mode manuel (State = 4) Une optimisation préalable est lancée lorsque les conditions correspondantes sont réunies. Une régulation est effectuée avec les paramètres PID déterminés jusqu'à ce que la boucle de régulation soit en régime établi et que les conditions pour une optimisation fine soient remplies.

State/Mode	Description du mode de fonctionnement
	Quand la mesure est déjà trop proche de la consigne pour une optimisation préalable ou que PIDSelfTune.TIR.RunIn = TRUE, le système essaie d'atteindre la consigne avec la valeur de réglage mini ou maxi. Cela peut entraîner une suroscillation élevée. C'est seulement après cela que l'optimisation fine commence. Après une optimisation fine réussie, le mode de fonctionnement passe en automatique ; si l'optimisation fine échoue, le mode de fonctionnement change de mode en fonction de ActivateRecoverMode. La phase d'"optimisation fine" est affichée avec la PIDSelfTune.TIR.State.
3	Mode automatique En mode automatique, PID_Compact régule le système réglé en fonction des paramètres prédéfinis. Si l'une des conditions préalables suivantes est remplie, le système passe en mode automatique : • Optimisation préalable réussie • Optimisation fine réussie • Modification du paramètre d'entrée/sortie Mode à la valeur 3 et un front montant à ModeActivate. Le passage du mode automatique en mode manuel s'effectue sans à-coups uniquement dans l'éditeur de mise en service. Le mode automatique tient compte de la variable ActivateRecoverMode.
4	Mode manuel En mode manuel, vous spécifiez une valeur de réglage manuelle au paramètre ManualValue. Ce mode est également activable via ManualEnable = TRUE. Il est recommandé de changer de mode de fonctionnement uniquement via Mode et ModeActivate. Le passage du mode manuel au mode automatique s'effectue sans à-coups. En cas d'erreur, le mode manuel est également possible.
5	Valeur de réglage de remplacement avec surveillance des erreurs L'algorithme de régulation est arrêté. La variable SetSubstituteOutput détermine la valeur de réglage à utiliser dans ce mode de fonctionnement. • SetSubstituteOutput = FALSE : dernière valeur de réglage valide • SetSubstituteOutput = TRUE : Valeur de réglage de remplacement Ce mode de fonctionnement ne peut pas être activé avec Mode = 5. Il est activé en cas d'erreur au lieu du mode de fonctionnement "Inactif" si toutes les conditions suivantes sont remplies : • Mode automatique (Mode = 3) • ActivateRecoverMode = TRUE • Une ou plusieurs erreurs sont apparues pour lesquelles ActivateRecoverMode s'applique. Dès que les erreurs ont disparu, PID_Compact repasse en mode automatique.

Comportement ENO

Si State = 0, alors ENO = FALSE.
Si State ≠ 0, alors ENO = TRUE.

Changement de mode de fonctionnement automatique pendant la mise en route

Après une optimisation préalable ou fine réussie, le mode automatique est activé. Le tableau suivant indique comment Mode et State évoluent pendant une optimisation préalable réussie.

Numéro de cycle	Mode	State	Action
0	4	4	Mise à 1 de Mode = 1
1	1	4	Mise à 1 de ModeActivate = TRUE
1	4	1	La valeur de State est enregistrée à Mode L'optimisation préalable est lancée
n	4	1	Optimisation préalable terminée avec succès
n	3	3	Le mode automatique est lancé

En cas d'erreur, PID_Compact change automatiquement de mode de fonctionnement. Le tableau suivant indique comment Mode et State évoluent pendant une optimisation préalable erronée.

Numéro de cycle	Mode	State	Action
0	4	4	Mise à 1 de Mode = 1
1	1	4	Mise à 1 de ModeActivate = TRUE
1	4	1	La valeur de State est enregistrée à Mode L'optimisation préalable est lancée
n	4	1	Optimisation préalable interrompue
n	4	4	Le mode manuel est démarré

Si ActivateRecoverMode = TRUE, le mode de fonctionnement qui est enregistré dans Mode est activé. Au démarrage de l'optimisation préalable ou fine, PID_Compact a enregistré la valeur de State au paramètre d'entrée/sortie Mode. PID_Compact passe donc dans le mode de fonctionnement à partir duquel l'optimisation a été lancée.
Si ActivateRecoverMode = FALSE, le système passe en mode de fonctionnement "Inactif".

Voir aussi

Paramètres de sortie PID_Compact à partir de V2 (Page 251)

10.1.4.10 Paramètre ErrorBits à partir de V2

En présence de plusieurs erreurs simultanées, les valeurs des ErrorBits s'affichent additionnées en binaire. L'affichage ErrorBits = 16#0000_0003, p. ex., indique la présence simultanée des erreurs 16#0000_0001 et 16#0000_0002.

PID_Compact utilise ManualValue comme valeur de sortie en mode manuel. Errorbits = 16#0001_0000 est l'exception.

ErrorBits (DW#16#...)	Description
0000_0000	Aucune erreur.
0000_0001	Le paramètre "Input" se trouve hors des limites de la mesure. • Input > Config.InputUpperLimit ou • Input < Config.InputLowerLimit Si le mode automatique était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Compact reste en mode automatique. Si le mode Optimisation préalable ou Optimisation fine était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Compact passe dans le mode de fonctionnement enregistré dans Mode.
0000_0002	Valeur invalide au paramètre "Input_PER". Vérifiez s'il y a une erreur à l'entrée analogique. Si le mode automatique était activé avant l'apparition de l'erreur et si ActivateRecoverMode = TRUE, PID_Compact sort la valeur de sortie de remplacement configurée. Dès que l'erreur a disparu, PID_Compact repasse en mode automatique. Si le mode Optimisation préalable ou Optimisation fine était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Compact passe dans le mode de fonctionnement enregistré dans Mode.
0000_0004	Erreur pendant l'optimisation fine. L'oscillation des températures n'a pas pu être maintenue. Si avant l'apparition de l'erreur ActivateRecoverMode = TRUE, PID_Compact interrompt l'optimisation et passe dans le mode de fonctionnement enregistré dans Mode.
0000_0008	Erreur au démarrage de l'optimisation préalable. La mesure est trop proche de la consigne. Démarrez l'optimisation fine. Si avant l'apparition de l'erreur ActivateRecoverMode = TRUE, PID_Compact interrompt l'optimisation et passe dans le mode de fonctionnement enregistré dans Mode.
0000_0010	La consigne a été modifiée durant l'optimisation. Les fluctuations admissibles de la consigne peuvent être réglées à la variable CancelTuningLevel. Si avant l'apparition de l'erreur ActivateRecoverMode = TRUE, PID_Compact interrompt l'optimisation et passe dans le mode de fonctionnement enregistré dans Mode.
0000_0020	L'optimisation préalable n'est pas autorisée pendant l'optimisation fine. Si avant l'apparition de l'erreur ActivateRecoverMode = TRUE, PID_Compact reste en mode de fonctionnement Optimisation fine.
0000_0080	Erreur lors de l'optimisation préalable. Les limites de la valeur de sortie ne sont pas configurées correctement ou la mesure ne réagit pas comme prévu. Vérifiez les points suivants : • Les limites de la valeur de sortie sont configurées correctement et conviennent au sens de la régulation. • Il est possible de modifier la valeur de sorte que la mesure se rapproche de la consigne. La valeur de sortie n'est pas limitée par la limite de valeur de sortie correspondante avant le démarrage de l'optimisation préalable. Exemple : Dans le sens normal de régulation et si la mesure est inférieure à la consigne, la valeur de sortie ne doit pas atteindre la limite supérieure avant le démarrage de l'optimisation préalable. • Il n'apparaît pas d'oscillations importantes de la mesure avant le démarrage de l'optimisation préalable. Pour démarrer une optimisation préalable à partir du mode automatique, il est recommandé de modifier la consigne sur un front montant de ModeActivate. Cela permet d'éviter que la valeur de sortie atteigne la

ErrorBits (DW#16#...)	Description
	limite entre la modification de la consigne et le démarrage de l'optimisation préalable. Il est également possible d'y parvenir en démarrant l'optimisation préalable à partir du mode manuel ou du mode "Inactif". Si ActivateRecoverMode = TRUE avant l'apparition de l'erreur, PID_Compact interrompt l'optimisation et passe dans le mode de fonctionnement enregistré dans Mode.
0000_0100	Une erreur durant l'optimisation fine a entraîné des paramètres non valides. Si avant l'apparition de l'erreur ActivateRecoverMode = TRUE, PID_Compact interrompt l'optimisation et passe dans le mode de fonctionnement enregistré dans Mode.
0000_0200	Valeur invalide au paramètre "Input" : Le format numérique de la valeur est invalide. Si le mode automatique était activé avant l'apparition de l'erreur et si ActivateRecoverMode = TRUE, PID_Compact sort la valeur de sortie de remplacement configurée. Dès que l'erreur a disparu, PID_Compact repasse en mode automatique. Si le mode Optimisation préalable ou Optimisation fine était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Compact passe dans le mode de fonctionnement enregistré dans Mode.
0000_0400	Le calcul de la valeur de sortie a échoué. Vérifiez les paramètres PID. Si le mode automatique était activé avant l'apparition de l'erreur et si ActivateRecoverMode = TRUE, PID_Compact sort la valeur de sortie de remplacement configurée. Dès que l'erreur a disparu, PID_Compact repasse en mode automatique. Si le mode Optimisation préalable ou Optimisation fine était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Compact passe dans le mode de fonctionnement enregistré dans Mode.
0000_0800	Erreur de période d'échantillonnage : PID_Compact n'est pas appelé pendant la période d'échantillonnage de l'OB d'alarme cyclique. Il est recommandé d'appeler PID_Compact dans un OB d'alarme cyclique sans conditions et de l'activer ou le désactiver via le mode de fonctionnement au paramètre Mode. Les appels conditionnels ou l'appel dans l'OB1 peuvent avoir une influence négative sur la qualité de la régulation. La surveillance de la période d'échantillonnage peut être désactivée avec CycleTime.EnMonitoring = FALSE Si le mode automatique était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Compact reste en mode automatique. Si le mode Optimisation préalable ou Optimisation fine était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Compact passe dans le mode de fonctionnement enregistré dans Mode. Si cette erreur s'est produite lors de la simulation avec PLCSIM, tenez compte des indications de la rubrique Simuler PID_Compact à partir de V2 avec PLCSIM (Page 100).
0000_1000	Valeur invalide au paramètre "Setpoint" : Le format numérique de la valeur est invalide. Si le mode automatique était activé avant l'apparition de l'erreur et si ActivateRecoverMode = TRUE, PID_Compact utilise la valeur de sortie de remplacement configurée. Dès que l'erreur a disparu, PID_Compact repasse en mode automatique. Si le mode Optimisation préalable ou Optimisation fine était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Compact passe dans le mode de fonctionnement enregistré dans Mode.
0001_0000	Valeur invalide au paramètre ManualValue. Le format numérique de la valeur est invalide. Si ActivateRecoverMode = TRUE avant l'apparition de l'erreur, PID_Compact utilise SubstituteOutput comme valeur de sortie. Dès qu'une valeur valide est spécifiée à ManualValue, PID_Compact l'utilise comme valeur de sortie.
0002_0000	Valeur invalide à la variable SubstituteOutput. Le format numérique de la valeur est invalide. PID_Compact utilise la limite inférieure comme valeur de sortie. Si le mode automatique était activé avant l'apparition de l'erreur, que ActivateRecoverMode = TRUE et que l'erreur a disparu, PID_Compact repasse en mode automatique.

ErrorBits (DW#16#...)	Description
0004_0000	Valeur invalide au paramètre Disturbance. Le format numérique de la valeur est invalide. Si le mode automatique était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, Disturbance est remis à zéro. PID_Compact reste en mode automatique. Si le mode Optimisation préalable ou Optimisation fine était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Compact passe dans le mode de fonctionnement enregistré dans Mode. Si Disturbance n'influe pas sur la valeur de sortie dans la phase actuelle, l'optimisation n'est pas interrompue.

10.1.4.11 Variable ActivateRecoverMode à partir de V2

La variable ActivateRecoverMode détermine le comportement en cas d'erreur. Le paramètre Error indique si une erreur est actuellement présente. Quand l'erreur a disparu, Error = FALSE. Le paramètre ErrorBits indique les erreurs survenues.

Mode automatique

IMPORTANT
Votre installation peut être endommagée. Si ActivateRecoverMode = TRUE, PID_Compact reste en cas d'erreur en mode automatique, y compris en cas de dépassement des limites de la mesure. Cela peut endommager votre installation. Configurez un comportement en cas d'erreur pour votre système réglé, qui protège votre installation de tout endommagement.

ActivateRecover-Mode	Description
FALSE	En cas d'erreur, PID_Compact passe en mode "Inactif". Le régulateur n'est activé que par un front descendant à Reset ou un front montant à ModeActivate.
TRUE	Si des erreurs apparaissent fréquemment en mode automatique, ce réglage détériore le comportement de régulation car PID_Compact alterne à chaque erreur entre la valeur de réglage calculée et la valeur de réglage de remplacement. Vérifiez alors le paramètre ErrorBits et éliminez la cause d'erreur. Quand l'une ou plusieurs des erreurs suivantes apparaissent, PID_Compact reste en mode automatique : • 0001h : Le paramètre "Input" se trouve en dehors des limites de la mesure. • 0800h : Erreur de temps d'échantillonnage • 40000h : Valeur invalide au paramètre Disturbance. Si l'une ou plusieurs des erreurs suivantes apparaissent, PID_Compact passe en mode de fonctionnement "Valeur de réglage de remplacement avec surveillance des erreurs" : • 0002h : Valeur invalide au paramètre Input_PER. • 0200h : Valeur invalide au paramètre Input. • 0400h : Le calcul de la valeur de réglage a échoué. • 1000h : Valeur invalide au paramètre Setpoint. Si l'erreur suivante apparaît, PID_Compact passe en mode de fonctionnement "Valeur de réglage de remplacement avec surveillance des erreurs" et l'actionneur se place sur Config.OutputLowerLimit : • 20000h : Valeur invalide à la variable SubstituteOutput. Le format numérique de la valeur est invalide. Ce comportement est indépendant de SetSubstituteOutput. Dès que les erreurs ont disparu, PID_Compact repasse en mode automatique.

Optimisation préalable et optimisation fine

ActivateRecover-Mode	Description
FALSE	En cas d'erreur, PID_Compact passe en mode "Inactif". Le régulateur n'est activé que par un front descendant à Reset ou un front montant à ModeActivate.
TRUE	Si l'erreur suivante se produit, PID_Compact reste en mode actif : • 0020h : L'optimisation préalable n'est pas autorisée pendant l'optimisation fine. Les erreurs suivantes sont ignorées : • 10000h : Valeur invalide au paramètre ManualValue. • 20000h : Valeur invalide à la variable SubstituteOutput. Pour toutes les autres erreurs, PID_Compact interrompt l'optimisation et passe dans le mode de fonctionnement à partir duquel l'optimisation a été lancée.

Mode manuel

En mode manuel, ActivateRecoverMode n'a aucun effet.

10.1.4.12 Variable Warning à partir de V2

En présence simultanée de plusieurs alertes, les valeurs des variables Warning sont affichées sous forme d'addition binaire. Si, p. ex. l'avertissement affiche 16#0000_0003, cela indique la présence simultanée des avertissements 0000_0001 et 0000_0002.

Warning (DW#16#....)	Description
0000_0000	Aucune alerte n'est présente.
0000_0001	Le point d'inflexion n'a pas été trouvé pendant l'optimisation préalable.
0000_0004	La consigne a été limitée à des limites paramétrées.
0000_0008	Toutes les propriétés nécessaires du système réglé n'ont pas été déterminées pour la méthode de calcul choisie. Les paramètres PID ont été calculés avec la méthode TIR.TuneRule = 3 à titre de remplacement.
0000_0010	Impossible de modifier le mode de fonctionnement car Reset = TRUE ou ManualEnable = TRUE
0000_0020	Le temps d'échantillonnage de l'algorithme PID est limité par le temps de cycle de l'OB appelant. Afin d'obtenir de meilleurs résultats, utilisez des temps de cycle de l'OB plus courts.
0000_0040	La mesure a dépassé l'une de ses limites d'alerte.
0000_0080	Valeur invalide à Mode. Le changement de mode de fonctionnement n'est pas effectué.
0000_0100	La valeur manuelle a été limitée aux limites de la sortie du régulateur.
0000_0200	La règle mentionnée pour l'optimisation n'est pas prise en charge. Aucun paramètre PID n'est calculé.
0000_1000	La valeur de sortie de remplacement ne peut pas être atteinte, car elle est en dehors des limites de la valeur de sortie.

Les alarmes suivantes sont supprimées dès que la cause est éliminée :

- 16#0000_0001
- 16#0000_0004
- 16#0000_0008
- 16#0000_0040
- 16#0000_0100

Toutes les autres alertes sont supprimées avec un front montant à Reset ou ErrorAck.

10.1.4.13 Variable IntegralResetMode à partir de V2

La variable IntegralResetMode détermine la valeur par défaut de l'action I PIDCtrl.IntegralSum :

- lorsque le mode de fonctionnement passe de "Inactif" à "Mode automatique"
- en cas de front TRUE -> FALSE sur le paramètre Reset et paramètre Mode = 3

Ce réglage n'agit que pendant un cycle et n'est effectif que si l'action I est activée (variable Retain.CtrlParams.Ti > 0.0).

IntegralReset-Mode	Description
0	Lissage La valeur de PIDCtrl.IntegralSum est préréglée de manière à ce que la commutation soit sans à-coup, c'est-à-dire que le "mode automatique" démarre à partir de la valeur de réglage = 0.0 (paramètre Output) et qu'il n'y ait pas de saut de la valeur de réglage indépendamment du signal d'écart (consigne – mesure).
1	Supprimer Il est recommandé de mettre la pondération de l'action P (Retain.CtrlParams.PWeighting) à 1.0 dans le cas où cette option est utilisée. La valeur de PIDCtrl.IntegralSum est supprimée. Si un signal d'écart est disponible, cela revient à un saut de la valeur de réglage. Le sens du saut de la valeur de réglage dépend de la pondération de l'action P configurée (variable Retain.CtrlParams.PWeighting) et du signal d'écart : • Pondération de l'action P = 1.0 : le saut de la valeur de réglage et le signal d'écart ont le même signe. Exemple : Si la mesure est inférieure à la consigne (signal d'écart positif), la valeur de réglage saute à une valeur positive. • Pondération de l'action P < 1.0 : pour de grands signaux d'écart, le saut de la valeur de réglage et le signal d'écart ont le même signe. Exemple : Si la mesure est largement inférieure à la consigne (signal d'écart positif), la valeur de réglage saute à une valeur positive. Pour de petits signaux d'écart, le saut de la valeur de réglage et le signal d'écart ont des signes différents. Exemple : Si la mesure est légèrement inférieure à la consigne (signal d'écart positif), la valeur de réglage saute à une valeur négative. Cela n'est généralement pas souhaitable, car cela conduit à une augmentation temporaire du signal d'écart. Le signal d'écart doit être d'autant plus grand, pour obtenir un saut de valeur de réglage de même signe, que la pondération de l'action P configurée est faible. Il est recommandé de mettre la pondération de l'action P (Retain.CtrlParams.PWeighting) à 1.0 dans le cas où cette option est utilisée. Dans le cas contraire, cela peut entraîner le comportement indésirable décrit, si le signal d'écart est faible. Une autre solution consiste à utiliser IntegralResetMode = 4. Cette option garantit des signes identiques pour le saut de la valeur de réglage et le signal d'écart, indépendamment de la pondération de l'action P configurée et du signal d'écart.
2	Conserver La valeur de PIDCtrl.IntegralSum n'est pas modifiée. Vous pouvez spécifier une nouvelle valeur via le programme utilisateur.
3	Valeur par défaut La valeur de PIDCtrl.IntegralSum est automatiquement préréglée comme si on avait eu Output = OverwriteInitialOutputValue dans le dernier cycle.

IntegralReset-Mode	Description
4	Comme variation de la consigne (uniquement pour PID_Compact en version ≥ 2.3) La valeur de PIDCtrl.IntegralSum est automatiquement préréglée de manière à ce que se produise un saut de valeur de réglage analogue à celui d'un régulateur PI en mode automatique lorsque la consigne varie de la valeur de mesure actuelle à la valeur de consigne actuelle. En présence d'un signal d'écart, il en résulte un saut de la valeur de réglage. Le saut de la valeur de réglage et le signal d'écart ont le même signe. Exemple : Si la mesure est inférieure à la consigne (signal d'écart positif), la valeur de réglage saute à une valeur positive. Ce comportement est indépendant de la pondération de l'action P configurée et du signal d'écart.

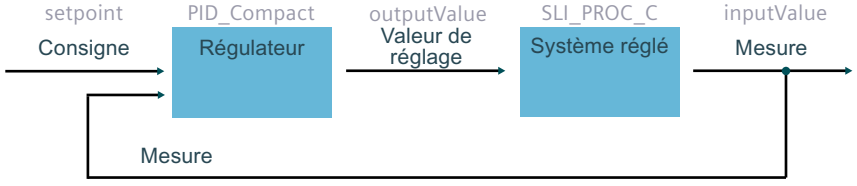
Si une valeur extérieure à la plage admissible est affectée à IntegralResetMode, PID_Compact se comporte comme avec la valeur par défaut de IntegralResetMode :

- PID_Compact jusqu'à V2.2 : IntegralResetMode = 1
- PID_Compact a partir de V2.3 : IntegralResetMode = 4

Toutes les mentions ci-dessus concernant le signe du saut de valeur de réglage s'appliquent dans le cas d'un sens de régulation normal (variable Config.InvertControl = FALSE). Si le sens de régulation est inversé (Config.InvertControl = TRUE), le signe du saut de valeur de réglage est inversé.

10.1.4.14 Exemple de programme pour PID_Compact V2

Dans l'exemple suivant, vous contrôlez des valeurs de température avec l'objet technologique de l'instruction "PID_Compact". Les valeurs de température sont simulées au moyen d'un bloc qui reproduit un opérateur à retard de troisième ordre (opérateur PT3). Vous utilisez l'optimisation préalable pour donner automatiquement des valeurs aux paramètres PID de l'objet technologique.



Stockage des données

Vous créez sept variables dans un bloc de données global pour stocker les données de connexion.

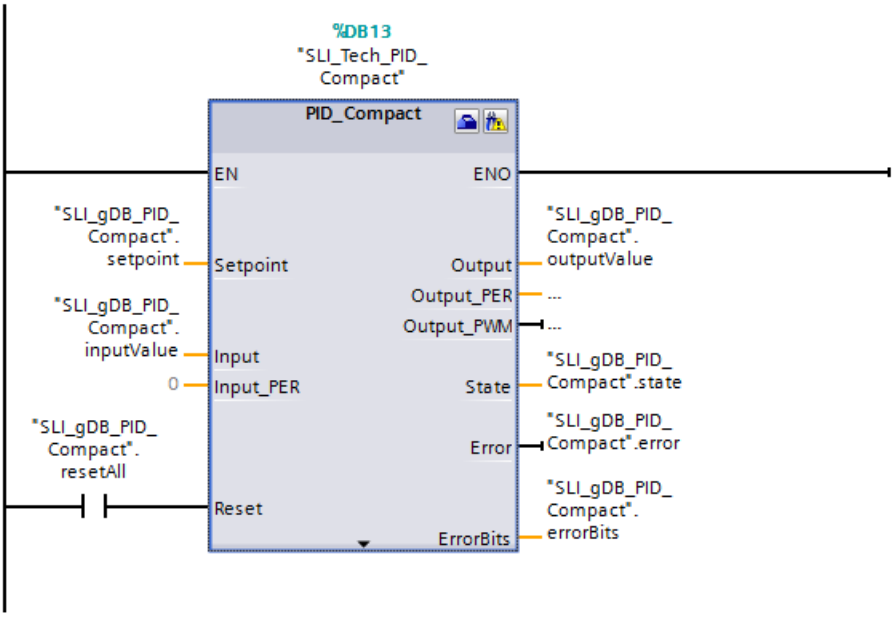
SLI_gDB_PID_Compact

		Name	Data type	Start value
1		Static		
2		setpoint	Real	75.0
3		inputValue	Real	0.0
4		outputValue	Real	0.0
5		state	Int	0
6		error	Bool	false
7		errorBits	DWord	16#0
8		resetAll	Bool	false

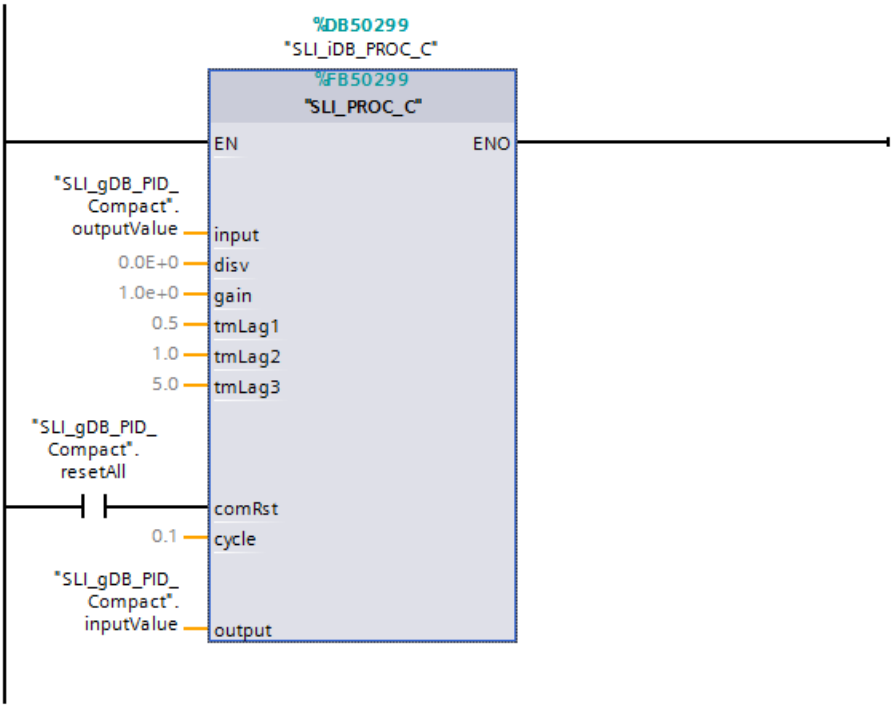
Connexion des paramètres

Vous appelez les connexions suivantes dans un OB d'alarme cyclique.

Réseau 1 : Vous connectez les paramètres de l'instruction "PID_Compact" de la manière suivante.

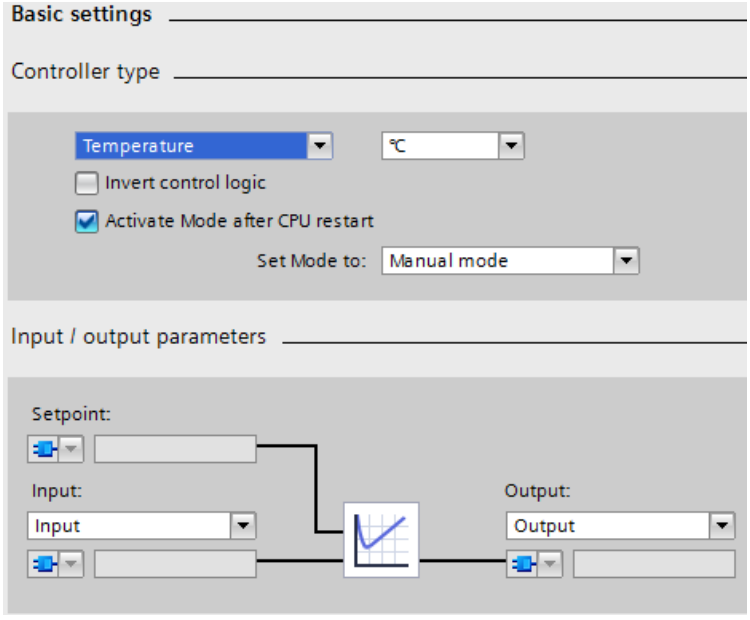


Réseau 2 : Vous connectez les paramètres du bloc simulant les valeurs de température "SLI_PROC_C" de la manière suivante.



Objet technologique

L'objet technologique peut être configuré via les propriétés de l'instruction "PID_Compact" ou via le chemin Objet technologique > Configuration. Pour l'exemple, le type de régulation et les paramètres d'entrée et de sortie sont significatifs. Le type de régulation permet de définir une présélection de l'unité de la valeur réglée. Dans cet exemple, le type de régulation utilisé est "Température" avec l'unité "°C". Les paramètres du "PID_Compact" sont déjà connectés avec les variables globales. Il suffit par conséquent d'indiquer l'utilisation des paramètres Input et Output.



Marche à suivre pour démarrer la régulation

Une fois chargé dans la CPU, PID_Compact est en mode manuel avec la valeur manuelle 0.0. Pour débuter la régulation, procédez comme suit :

1. Ouvrez la mise en service de l'objet technologique "SLI_Tech_PID_Compact".
2. Dans la zone "Mesure", cliquez sur le bouton "Démarrer".

La mesure démarre et PID_Compact peut être activé.

3. L'optimisation préalable est sélectionnée.
   Dans la zone "Type d'optimisation", cliquez sur le bouton "Démarrer".
   Une optimisation préalable est effectuée. Les paramètres PID sont alors adaptés automatiquement au système réglé. Une fois terminée l'optimisation préalable, PID_Compact passe en mode automatique.

REMARQUE

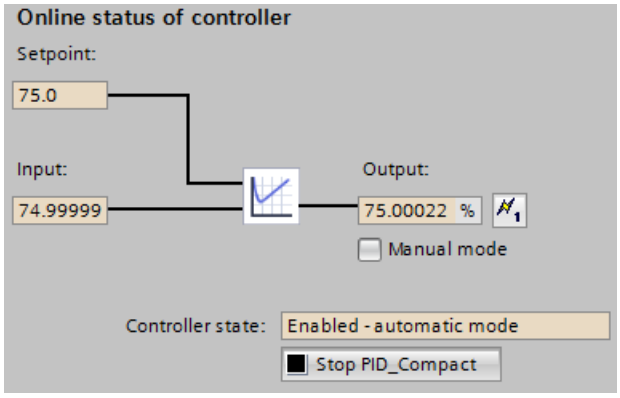
Autre façon de démarrer PID_Compact

Vous pouvez aussi mettre PID_Compact en mode automatique, sans optimisation préalable, dans la zone "État en ligne du régulateur" au moyen du bouton "Arrêt PID_Compact" / "Marche PID_Compact". Dans ce cas, le régulateur utilise des valeurs par défaut pour les paramètres PID et son comportement de régulation est de moindre qualité pour le cas d'application.

Marche à suivre pour stopper la régulation

Pour stopper et quitter PID_Compact et le programme, procédez de la manière suivante :

1. Dans l'objet technologique "SLI_Tech_PID_Compact", zone "Etat en ligne du régulateur", cliquez sur le bouton "Arrêt PID_Compact".



   L'instruction "PID_Compact" arrête la régulation et fournit la valeur "0.0" comme valeur de réglage.
2. Dans la zone "Mesure", cliquez sur le bouton "Arrêt".
3. Pour mettre la mesure immédiatement à la valeur "0.0", procédez comme suit :
   Dans le bloc "SLI_OB_PID_Compact", mettez la variable "resetAll" à la valeur "TRUE", puis à la valeur "FALSE".

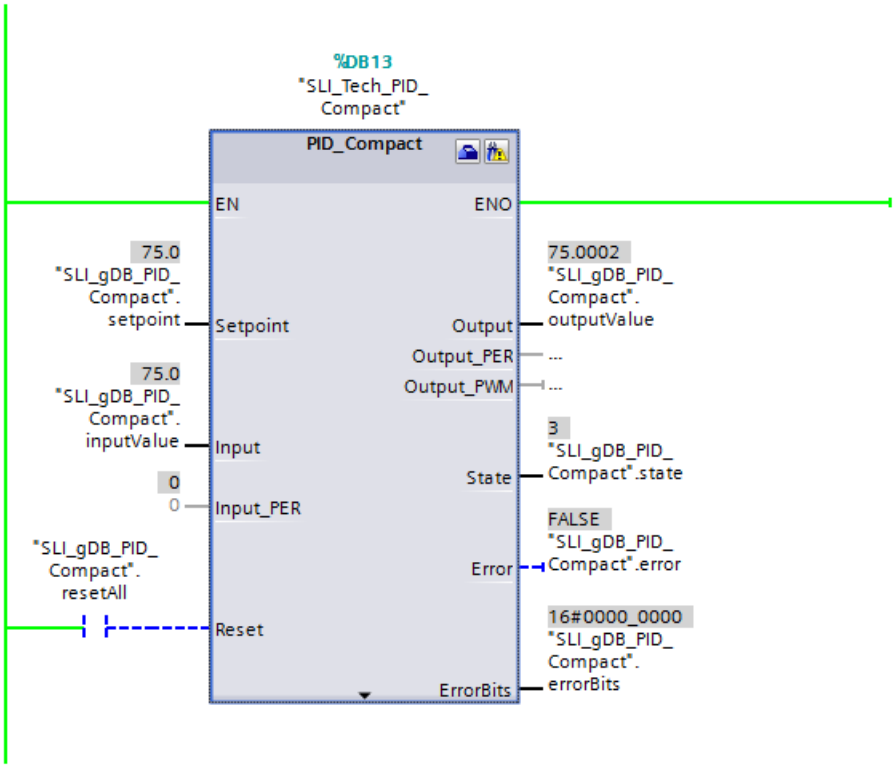
Instruction "PID_Compact"

La valeur de consigne pour la température à régler est transmise sur le paramètre Setpoint ("setpoint"). Quand l'instruction "PID_Compact" a été lancée au moyen de l'objet technologique, la régulation démarre. L'instruction "PID_Compact" fournit une valeur de réglage dans le paramètre de sortie Output ("outputValue"). Le paramètre d'entrée Input ("inputValue") sert à fournir la valeur de mesure de la température à l'instruction "PID_Compact".

Selon l'évolution de la différence entre consigne ("setpoint") et mesure ("inputValue"), l'instruction "PID_Compact" adapte la valeur de réglage ("outputValue"). Cette opération se répète, si bien que la mesure ("inputValue") se rapproche de la consigne ("setpoint") sous l'influence de la valeur de réglage ("outputValue").

Le paramètre de sortie State ("state") indique le mode de fonctionnement actuel de l'instruction "PID_Compact". Après une optimisation préalable (la valeur de "state" est "1"), le PID_Compact passe en mode automatique (la valeur est "3").

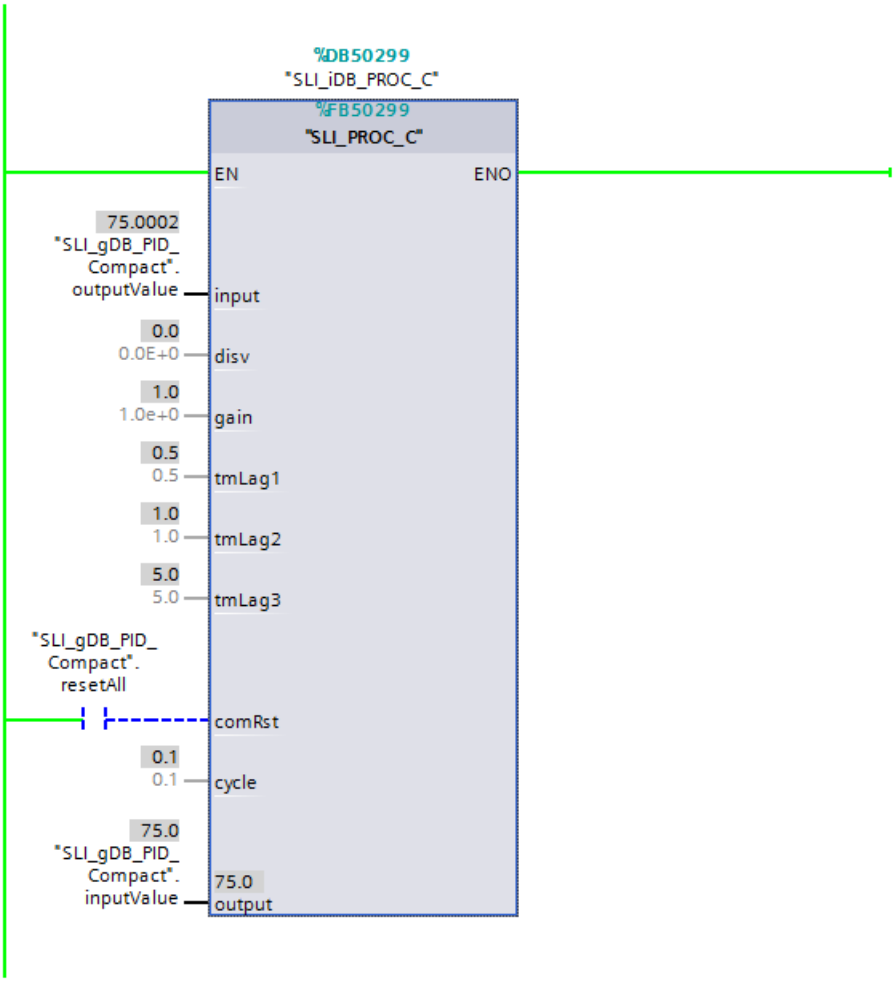
Le paramètre de sortie Error ("error") indique qu'il n'y a pas d'erreur actuellement. En cas d'erreur, le paramètre de sortie ErrorBits ("errorBits") renseigne sur la nature de l'erreur. Quand une erreur apparaît, il est possible de l'acquitter avec le bouton "ErrorAck" dans la zone État d'optimisation de l'objet technologique.



Bloc "SLI_PROC_C"

Le bloc "SLI_PROC_C" simule la mesure ("inputValue") de la température en augmentation d'une installation. Le bloc "SLI_PROC_C" reçoit la valeur de réglage du régulateur ("outputValue") et simule le comportement de température du système réglé. Cette température est entrée dans le régulateur comme mesure ("inputValue").
Une modification des valeurs de la variable "resetAll" (du paramètre comRst) a les effets suivants :

Paramètre comRst ("resetAll")	L'instruction "PID_Compact" s'exécute	L'instruction "PID_Compact" a été arrêtée
comRst ("resetAll") reste à la valeur "FALSE"	Le bloc "SLI_PROC_C" fournit, sur la base d'une valeur de réglage ("outputValue"), une nouvelle valeur de mesure ("inputValue").	Le bloc "SLI_PROC_C" ne reçoit pas de valeur de réglage > "0.0", mais il continue à fournir une mesure > "0.0".
comRst ("resetAll") : Modification de "FALSE" à la valeur "TRUE"	La valeur de réglage ("outputValue") tout comme la mesure fournie ("inputValue") sont remises à "0.0".	La mesure fournie ("inputValue") / la température du bloc "SLI_PROC_C" est remise à "0.0".
comRst ("resetAll") : Modification de "TRUE" à la valeur "FALSE"	La régulation de température redémarre.	La mesure fournie / la température ("inputValue") reste "0.0".



Code programme

Pour plus d'informations sur le code de programme de l'exemple ci-dessus, voir le mot-clé "Sample Library for Instructions".

10.1.5 PID_Compact V1

10.1.5.1 Description PID_Compact V1

Description

L'instruction PID_Compact met à disposition un régulateur PID avec optimisation intégrée pour les modes automatique et manuel.

Appel

L'appel de l'instruction PID_Compact s'effectue durant l'intervalle de temps constant du temps de cycle de l'OB appelant (de préférence dans un OB d'alarme cyclique).

Chargement dans l'appareil

Les valeurs effectives de variables rémanentes ne sont actualisées que si vous chargez entièrement PID_Compact.
Charger des objets technologiques dans l'appareil (Page 48)

Démarrage

PID_Compact est démarrée dans le dernier mode de fonctionnement actif lors du démarrage de la CPU. Pour laisser PID_ Compact en mode "Inactif", mettez sb_RunModeByStartup = FALSE.

Surveillance du temps d'échantillonnage PID_Compact

Le temps d'échantillonnage correspond idéalement au temps de cycle de l'OB appelant. L'instruction PID_Compact permet de mesurer l'intervalle de temps entre deux appels respectifs. Le résultat est le temps d'échantillonnage actuel. Lors de chaque changement du mode de fonctionnement et à la première mise en route, une moyenne est calculée à partir des 10 premiers temps d'échantillonnage. Si le temps d'échantillonnage actuel diverge trop de cette moyenne, une erreur survient (Error = 0800 hex) et PID_Compact passe en mode de fonctionnement "Inactif".
Les conditions suivantes font passer, à partir de la version 1.1, PID_Compact en mode de fonctionnement "Inactif" pendant l'optimisation :

- Nouvelle valeur moyenne >= 1,1 x ancienne valeur moyenne
- Nouvelle valeur moyenne <= 0,9 x ancienne valeur moyenne

Les conditions suivantes font passer, à partir de la version 1.1, PID_Compact en mode de fonctionnement "Inactif" en cas de mode automatique :

- Nouvelle valeur moyenne >= 1,5 x ancienne valeur moyenne
- Nouvelle valeur moyenne <= 0,5 x ancienne valeur moyenne

Les conditions suivantes font passer PID_Compact 1.0 en mode de fonctionnement "Inactif" pendant l'optimisation et en mode automatique :

- Nouvelle valeur moyenne >= 1,1 x ancienne valeur moyenne
- Nouvelle valeur moyenne <= 0,9 x ancienne valeur moyenne
- Temps d'échantillonnage actuel >= 1,5 x valeur moyenne actuelle
- Temps d'échantillonnage actuel <= 0,5 x valeur moyenne actuelle

Temps d'échantillonnage de l'algorithme PID

Comme le système réglé nécessite un certain temps pour réagir à une modification de la valeur de réglage, il est judicieux de ne pas calculer cette valeur à chaque cycle. Le temps d'échantillonnage de l'algorithme PID est le temps entre deux calculs de valeurs de réglage. Il est déterminé pendant l'optimisation et arrondi à un multiple du temps de cycle. Toutes les autres fonctions de PID_Compact sont exécutées lors de chaque appel.

Algorithme PID

PID_Compact est un régulateur PIDT1 avec anti-saturation et pondération de l'action P et D. La valeur de réglage est calculée avec la formule suivante :

$$y = K_p \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

Symbole	Description
y	Valeur de réglage
K_p	Gain proportionnel
s	Opérateur de Laplace
b	Pondération de l'action P
w	Consigne
x	Mesure
T_I	Temps d'intégration
a	Coefficient pour l'action par dérivation ($T1 = a \times T_D$)
	Temps de dérivation
c	Pondération de l'action D

Schéma fonctionnel PID_Compact

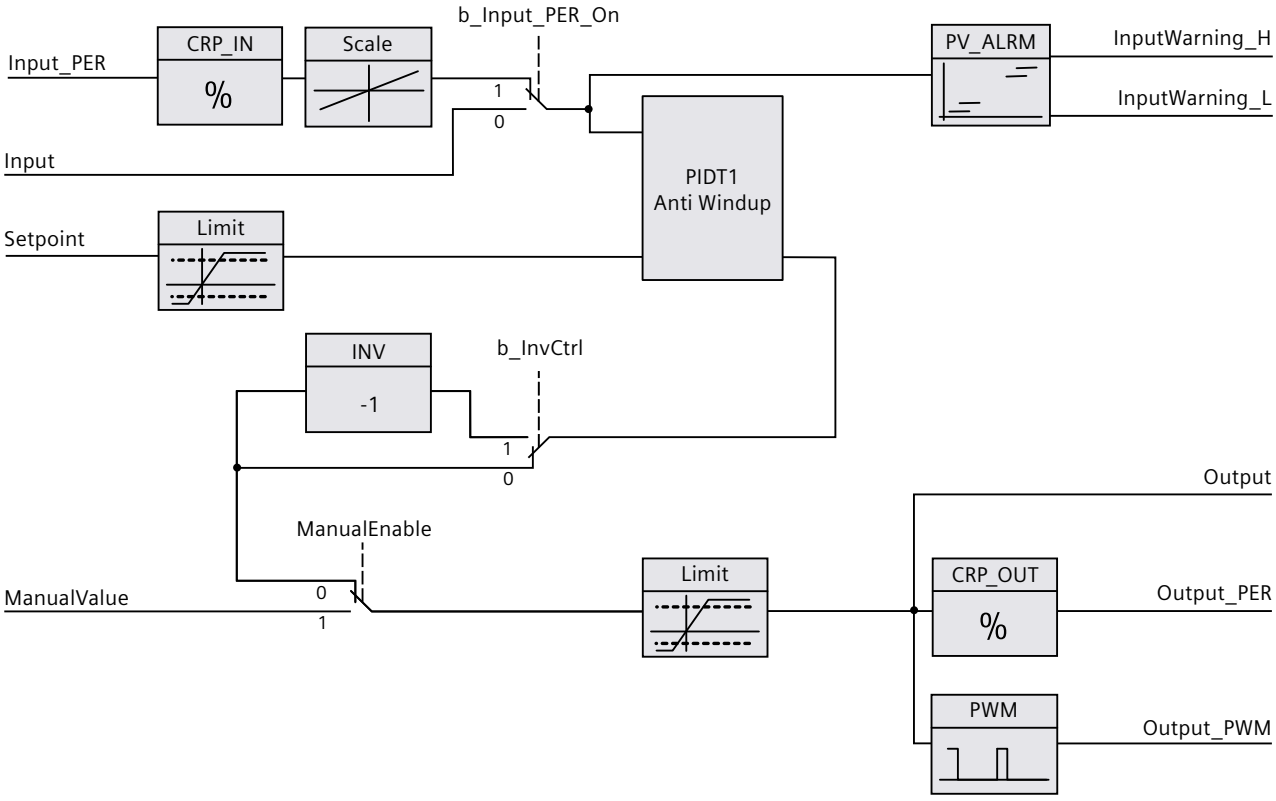
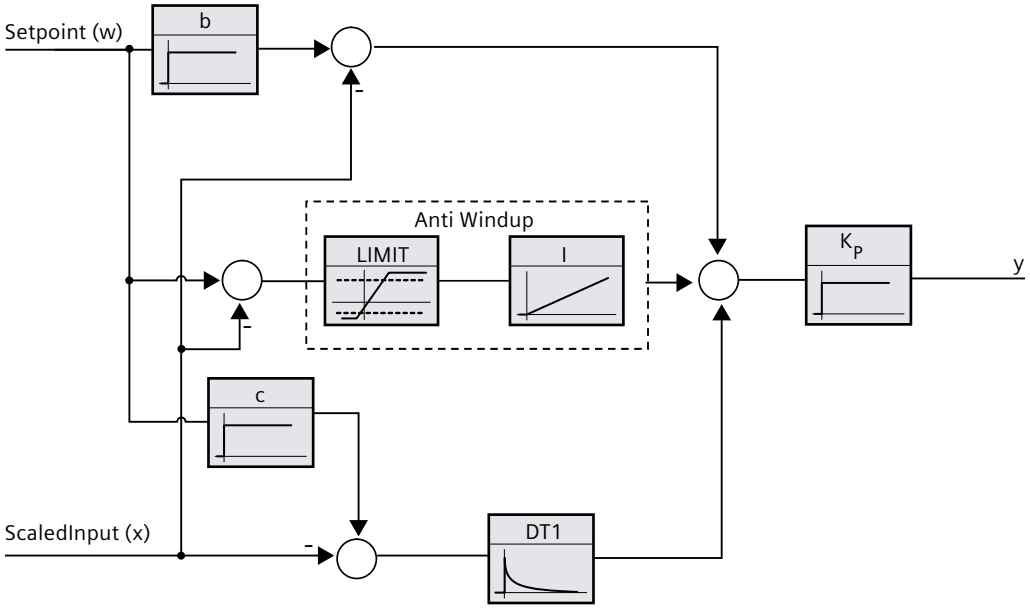


Schéma fonctionnel PIDT1 avec anti-saturation



Comportement en cas d'erreur

Si des erreurs surviennent, elles sont affichées au niveau du paramètre Error et PID_Compact passe au mode de fonctionnement "Inactif". Le paramètre Reset permet de remettre à 0 toutes les erreurs.

Sens de régulation

La plupart du temps, une augmentation de la mesure doit être atteinte avec une augmentation de la valeur de réglage. Dans ce cas, on parle d'un sens de régulation normal. Il peut être nécessaire d'inverser le sens de régulation pour les refroidissements et les régulations d'écoulement. PID_Compact ne fonctionne pas avec un gain proportionnel négatif. Si InvertControl = TRUE, un signal d'écart croissant provoque une diminution de la valeur de réglage. Le sens de régulation est pris en compte aussi pendant l'optimisation préalable et l'optimisation fine.

Voir aussi

Type de régulation V1 (Page 101)

10.1.5.2 Paramètres d'entrée PID_Compact V1

Tableau 10-4

Paramètre	Type de données	Valeur par défaut	Description
Setpoint	REAL	0.0	Consigne du régulateur PID en mode automatique
Input	REAL	0.0	Une variable du programme utilisateur est utilisée comme source de la mesure. Si vous utilisez le paramètre Input, il faut que sPid_Cmpt.b_Input_PER_On = FALSE.
Input_PER	WORD	W#16#0	Entrée analogique comme source de la mesure Si vous utilisez le paramètre Input_PER, il faut que sPid_Cmpt.b_Input_PER_On = TRUE.
ManualEnable	BOOL	FALSE	• Le front FALSE -> TRUE sélectionne le mode de fonctionnement "Mode manuel", State = 4, sRet.i_Mode ne change pas. • Le front TRUE -> FALSE sélectionne le dernier mode de fonctionnement actif, State =sRet.i_Mode Pendant ManualEnable = TRUE, une modification de sRet.i_Mode n'a pas d'effet. La modification de sRet.i_Mode est prise en compte seulement au front TRUE -> FALSE à ManualEnable . **PID_Compact V1.2 et PID_Compact V1.0** Lorsque ManualEnable = TRUE au démarrage de la CPU, PID_Compact démarre en mode manuel. Un front montant (FALSE > TRUE) de ManualEnable n'est pas nécessaire. **PID_Compact V1.1** PID_Compact ne passe en mode manuel au démarrage de la CPU que s'il y a un front montant (FALSE->TRUE) de ManualEnable . En l'absence de ce front montant, PID_Compact démarre dans le dernier mode pour lequel ManualEnable était FALSE.
ManualValue	REAL	0.0	Valeur manuelle Cette valeur est utilisée comme valeur de réglage en mode manuel.
Reset	BOOL	FALSE	Le paramètre Reset (Page 292) effectue un redémarrage du régulateur.

10.1.5.3 Paramètres de sortie PID_Compact V1

Tableau 10-5

Parameter	Type de données	Valeur par défaut	Description
ScaledInput	REAL	0.0	Sortie de la mesure mise à l'échelle
Les sorties "Output", "Output_PER" et "Output_PWM" peuvent être utilisées en parallèle.			
Output	REAL	0.0	Valeur de réglage au format REAL
Output_PER	WORD	W#16#0	Valeur de réglage analogique
Output_PWM	BOOL	FALSE	Valeur de réglage modulée en largeur d'impulsion La valeur de réglage est calculée au moyen de temps d'activation et de désactivation variables.
SetpointLimit_H	BOOL	FALSE	Quand SetpointLimit_H = TRUE, la limite supérieure absolue de la consigne est atteinte. Dans la CPU, la consigne est limitée à la limite supérieure absolue configurée pour la consigne. La limite supérieure par défaut de la consigne est la limite supérieure absolue configurée pour la mesure. Si vous affectez à sPid_Cmpt.r_Sp_Hlm une valeur dans les limites de la mesure, cette valeur sera utilisée comme limite supérieure de la consigne.
SetpointLimit_L	BOOL	FALSE	Quand SetpointLimit_L = TRUE, la limite inférieure absolue de la consigne est atteinte. Dans la CPU, la consigne est limitée à la limite inférieure absolue configurée pour la consigne. La limite inférieure par défaut de la consigne est la limite inférieure absolue configurée pour la mesure. Si vous affectez à sPid_Cmpt.r_Sp_Llm une valeur dans les limites de la mesure, cette valeur sera utilisée comme limite inférieure de la consigne.
InputWarning_H	BOOL	FALSE	Si InputWarning_H = TRUE, la limite d'alerte supérieure de la mesure est atteinte ou dépassée.
InputWarning_L	BOOL	FALSE	Si InputWarning_L = TRUE, la limite d'alerte inférieure de la mesure est atteinte ou dépassée.
State	INT	0	Le paramètre State (Page 289) affiche le mode de fonctionnement actuel du régulateur PID. La variable sRet.i_Mode vous permet de modifier le mode de fonctionnement. • State = 0 : inactif • State = 1 : optimisation préalable • State = 2 : optimisation fine • State = 3 : mode automatique • State = 4 : mode manuel
Error	DWORD	W#16#0	Le paramètre Error (Page 291) affiche les messages d'erreur. Error = 0000 : pas de présence d'erreur.

10.1.5.4 Variables statiques PID_Compact v1

REMARQUE

Les variables qui ne sont pas mentionnées ne doivent pas être modifiées. Elles ne sont utilisées qu'en interne.

Faites passer les variables identifiées par (1) seulement en mode de fonctionnement "Inactif" pour éviter un comportement erroné du régulateur PID. Vous forcez le mode de fonctionnement "Inactif" en mettant la variable "sRet.i_Mode" à "0".

Tableau 10-6

Variable	Type de données	Valeur par défaut	Description
sb_GetCycleTime	BOOL	TRUE	Si sb_GetCycleTime = TRUE, la détermination automatique du temps de cycle est lancée. Après la fin de la mesure, on a CycleTime.StartEstimation = FALSE.
sb_EnCyclEstimation	BOOL	TRUE	Si sb_EnCyclEstimation = TRUE, le temps d'échantillonnage PID_Compact est calculé.
sb_EnCyclMonitoring	BOOL	TRUE	Si sb_EnCyclMonitoring = FALSE, le temps d'échantillonnage PID_Compact n'est pas surveillé. Si PID_Compact n'est pas exécutée pendant le temps d'échantillonnage, aucune erreur 0800 n'est affichée et PID_Compact ne passe pas au mode de fonctionnement "Inactif".
sb_RunModeByStartup	BOOL	TRUE	Activer le dernier mode de fonctionnement après le démarrage de la CPU Si sb_RunModeByStartup = FALSE, le régulateur reste inactif après la mise en route de la CPU. Si sb_RunModeByStartup = TRUE, le régulateur repasse au dernier mode de fonctionnement après la mise en route de la CPU.
si_Unit	INT	0	Unité physique de la mesure et de la consigne, par ex. °C ou °F.
si_Type	INT	0	Grandeur physique de la mesure et de la consigne, par ex. température.
sd_Warning	DWORD	DW#16#0	La variable sd_warning (Page 293) affiche les alertes depuis la remise à 0 ou le dernier changement du mode de fonctionnement.
sBackUp.r_Gain	REAL	1.0	Gain proportionnel enregistré Il est possible de recharger les valeurs de la structure sBackUp avec sPid_Cmpt.b_LoadBackUp = TRUE.
sBackUp.r_Ti	REAL	20.0	Temps d'intégration enregistré [s]
sBackUp.r_Td	REAL	0.0	Temps de dérivation enregistré [s]
sBackUp.r_A	REAL	0.0	Coefficient de l'action par dérivation enregistré
sBackUp.r_B	REAL	0.0	Facteur de pondération de l'action P enregistré
sBackUp.r_C	REAL	0.0	Facteur de pondération de l'action D enregistré
sBackUp.r_Cycle	REAL	1.0	Temps d'échantillonnage enregistré de l'algorithme PID

Variable	Type de données	Valeur par défaut	Description
sPid_Calc.r_Cycle(1)	REAL	0.1	Temps d'échantillonnage de l'instruction PID_Compact r_Cycle est automatiquement déterminée et correspond normalement au temps de cycle de l'OB appelant.
sPid_Calc.b_RunIn	BOOL	FALSE	• b_RunIn = FALSE Si l'optimisation fine est démarrée depuis le mode inactif ou manuel, une optimisation préalable est lancée. Si les conditions d'une optimisation préalable ne sont pas réunies, PID_Compact a le même comportement que lorsque b_RunIn = TRUE. Si l'optimisation fine est démarrée depuis le mode automatique, les paramètres PID existants sont utilisés pour un réglage sur la consigne. C'est seulement après cela que l'optimisation fine commence. Si l'optimisation préalable n'est pas possible, PID_Compact passe en mode de fonctionnement "Inactif". • b_RunIn = TRUE L'optimisation préalable est sautée. PID_3Compact essaie d'atteindre la consigne avec la valeur de réglage mini ou maxi. Cela peut entraîner une suroscillation élevée. L'optimisation fine démarre ensuite automatiquement. b_RunIn est mis sur FALSE après l'optimisation fine.
sPid_Calc.b_CalcParamSUT	BOOL	FALSE	Si b_CalcParamSUT = TRUE, les paramètres pour l'optimisation préalable sont recalculés. Cela permet de modifier la méthode de calcul de paramètres sans répéter l'optimisation. b_CalcParamSUT est mis sur FALSE après le calcul.
sPid_Calc.b_CalcParamTIR	BOOL	FALSE	Si b_CalcParamTIR = TRUE, les paramètres pour l'optimisation fine sont recalculés. Cela permet de modifier la méthode de calcul de paramètres sans répéter l'optimisation. b_CalcParamTIR est mis sur FALSE après le calcul.
sPid_Calc.i_CtrlTypeSUT	INT	0	Calculer les paramètres pendant l'optimisation préalable selon la méthode : • i_CtrlTypeSUT = 0 : PID selon Chien, Hrones et Reswick • i_CtrlTypeSUT = 1 : PI selon Chien, Hrones et Reswick
sPid_Calc.i_CtrlTypeTIR	INT	0	Calculer les paramètres pendant l'optimisation fine selon la méthode : • i_CtrlTypeTIR = 0 : PID automatique • i_CtrlTypeTIR = 1 : PID rapide (comportement de régulation plus rapide avec des amplitudes plus fortes de la valeur de réglage qu'avec i_CtrlTypeTIR = 2) • i_CtrlTypeTIR = 2 : PID lent (comportement de régulation plus lent avec des amplitudes plus faibles de la valeur de réglage qu'avec i_CtrlTypiTIR = 1) • i_CtrlTypeTIR = 3 : PID Ziegler-Nichols • i_CtrlTypeTIR = 4 : PI Ziegler-Nichols • i_CtrlTypeTIR = 5 : P Ziegler-Nichols

Variable	Type de données	Valeur par défaut	Description
			Pour pouvoir répéter le calcul des paramètres PID avec b_CalcParamTIR et i_CtrlTypeTIR = 0, 1 ou 2, il faut avoir exécuté l'optimisation fine précédente également avec i_CtrlTypeTIR = 0, 1 ou 2. Si ce n'est pas le cas, i_CtrlTypeTIR = 3 sera utilisé. Il est toujours possible de recalculer les paramètres PID avec b_CalcParamTIR et i_CtrlTypeTIR = 3, 4 ou 5.
sPid_Calc.r_Progress	REAL	0.0	Progrès de l'optimisation en % (0.0 à 100.0)
sPid_Cmpt.r_Sp_Hlm(1)	REAL	+3.402822e+-38	Limite supérieure de la consigne Quand vous configurez sPid_Cmpt.r_Sp_Hlm en dehors des limites de la mesure, la limite supérieure absolue configurée pour la mesure est utilisée comme limite supérieure de la consigne. Si vous configurez sPid_Cmpt.r_Sp_Hlm dans les limites de la mesure, cette valeur sera utilisée comme limite supérieure de la consigne.
sPid_Cmpt.r_Sp_Llm(1)	REAL	-3.402822e+3-8	Limite inférieure de la consigne Quand vous configurez sPid_Cmpt.r_Sp_Llm en dehors des limites de la mesure, la limite inférieure absolue configurée pour la mesure est utilisée comme limite inférieure de la consigne. Si vous configurez sPid_Cmpt.r_Sp_Llm dans les limites de la mesure, cette valeur sera utilisée comme limite inférieure de la consigne.
sPid_Cmpt.r_Pv_Norm_IN_1(1)	REAL	0.0	Mise à l'échelle Input_PER Bas Input_PER est converti en pourcentage à l'aide des deux couples de valeurs r_Pv_Norm_OUT_1, r_Pv_Norm_IN_1 et r_Pv_Norm_OUT_2, r_Pv_Norm_IN_2 de la structure sPid_Cmpt.
sPid_Cmpt.r_Pv_Norm_IN_2(1)	REAL	27648.0	Mise à l'échelle Input_PER Haut Input_PER est converti en pourcentage à l'aide des deux couples de valeurs r_Pv_Norm_OUT_1, r_Pv_Norm_IN_1 et r_Pv_Norm_OUT_2, r_Pv_Norm_IN_2 de la structure sPid_Cmpt.
sPid_Cmpt.r_Pv_Norm_OUT_1(1)	REAL	0.0	Mesure inférieure à l'échelle Input_PER est converti en pourcentage à l'aide des deux couples de valeurs r_Pv_Norm_OUT_1, r_Pv_Norm_IN_1 et r_Pv_Norm_OUT_2, r_Pv_Norm_IN_2 de la structure sPid_Cmpt.
sPid_Cmpt.r_Pv_Norm_OUT_2(1)	REAL	100.0	Mesure supérieure à l'échelle Input_PER est converti en pourcentage à l'aide des deux couples de valeurs r_Pv_Norm_OUT_1, r_Pv_Norm_IN_1 et r_Pv_Norm_OUT_2, r_Pv_Norm_IN_2 de la structure sPid_Cmpt.
sPid_Cmpt.r_Lmn_Hlm(1)	REAL	100.0	Limite supérieure de la valeur de réglage pour le paramètre de sortie "Output"
sPid_Cmpt.r_Lmn_Llm(1)	REAL	0.0	Limite inférieure de la valeur de réglage pour le paramètre de sortie "Output"
sPid_Cmpt.b_Input_PER_On(1)	BOOL	TRUE	Si b_Input_PER_On = TRUE, c'est le paramètre Input_PER qui est utilisé. Si b_Input_PER_On = FALSE, c'est le paramètre Input qui est utilisé.

Variable	Type de données	Valeur par défaut	Description
sPid_Cmpt.b_LoadBackUp	BOOL	FALSE	Activation du jeu de paramètres de sauvegarde. En cas d'échec d'une optimisation, la mise à 1 de ce bit permet de réactiver les paramètres PID précédents.
sPid_Cmpt.b_InvCtrl(1)	BOOL	FALSE	Inversion du sens de régulation Si b_InvCtrl = TRUE, un signal d'écart croissant provoque une diminution de la valeur de réglage.
sPid_Cmpt.r_Lmn_Pwm_PPTm(1)	REAL	0.0	Plus petit temps ON en secondes de la modulation de largeur d'impulsions, arrondi à r_Lmn_Pwm_PPTm = r_Cycle ou r_Lmn_Pwm_PPTm = n*r_Cycle
sPid_Cmpt.r_Lmn_Pwm_PBTm(1)	REAL	0.0	Plus petit temps OFF en secondes de la modulation de largeur d'impulsions, arrondi à r_Lmn_Pwm_PBTm = r_Cycle ou r_Lmn_Pwm_PBTm = n*r_Cycle
sPid_Cmpt.r_Pv_Hlm(1)	REAL	120.0	Limite supérieure de la mesure A l'entrée de périphérie, la mesure peut dépasser de 18 % au plus la plage normée (dépassement haut). Un dépassement de la "limite supérieure de la mesure" ne provoque plus le signalement d'une erreur. Seuls la rupture de fil et le court-circuit sont détectés et PID_Compact passe en mode "Inactif". r_Pv_Hlm > r_Pv_Llm
sPid_Cmpt.r_Pv_Llm(1)	REAL	0.0	Limite inférieure de la mesure r_Pv_Llm < r_Pv_Hlm
sPid_Cmpt.r_Pv_HWrn(1)	REAL	+3.402822e+-38	Limite d'alerte supérieure de la mesure Quand vous configurez r_Pv_HWrn en dehors des limites de la mesure, la limite supérieure absolue configurée pour la mesure est utilisée comme limite d'alerte supérieure. Si vous configurez r_Pv_HWrn dans les limites de la mesure, cette valeur sera utilisée comme limite d'alerte supérieure. r_Pv_HWrn > r_Pv_LWrn r_Pv_HWrn ≤ r_Pv_Hlm
sPid_Cmpt.r_Pv_LWrn(1)	REAL	-3.402822e+3-8	Limite d'alerte inférieure de la mesure Quand vous configurez r_Pv_LWrn en dehors des limites de la mesure, la limite inférieure absolue configurée pour la mesure est utilisée comme limite d'alerte inférieure. Si vous configurez r_Pv_LWrn dans les limites de la mesure, cette valeur sera utilisée comme limite d'alerte inférieure. r_Pv_LWrn < r_Pv_HWrn r_Pv_LWrn ≥ r_Pv_LWrn
sPidCalc.i_Ctrl_IOutv(1)	REAL	0.0	Action I actuelle
sParamCalc.i_Event_SUT	INT	0	La variable i_Event_SUT (Page 294) affiche la phase actuelle de "l'optimisation préalable" :
sParamCalc.i_Event_TIR	INT	0	La variable i_Event_TIR (Page 294) affiche la phase actuelle de "l'optimisation fine" :

Variable	Type de données	Valeur par défaut	Description
sRet.i_Mode	INT	0	Le changement de mode de fonctionnement est commandé par le front. Le mode de fonctionnement suivant est activé lors du passage à • i_Mode = 0 : mode de fonctionnement "Inactif" (arrêt du régulateur) • i_Mode = 1 : mode de fonctionnement "Optimisation préalable" • i_Mode = 2 : mode de fonctionnement "Optimisation fine" • i_Mode = 3 : mode de fonctionnement "Mode automatique" • i_Mode = 4 : mode de fonctionnement "Mode manuel" i_Mode est rémanent.
sRet.r_Ctrl_Gain(1)	REAL	1.0	Gain proportionnel actif Gain est rémanent.
sRet.r_Ctrl_Ti(1)	REAL	20.0	• r_Ctrl_Ti > 0.0 : Temps d'intégration actif • r_Ctrl_Ti = 0.0 : L'action I est désactivée r_Ctrl_Ti est rémanent.
sRet.r_Ctrl_Td(1)	REAL	0.0	• r_Ctrl_Td > 0.0 : Temps de dérivation actif • r_Ctrl_Td = 0.0 : L'action D est désactivée r_Ctrl_Td est rémanent.
sRet.r_Ctrl_A(1)	REAL	0.0	Coefficient actif de l'action par dérivation r_Ctrl_A est rémanent.
sRet.r_Ctrl_B(1)	REAL	0.0	Pondération active de l'action P r_Ctrl_B est rémanent.
sRet.r_Ctrl_C(1)	REAL	0.0	Pondération active de l'action D r_Ctrl_C est rémanent.
sRet.r_Ctrl_Cycle(1)	REAL	1.0	Temps d'échantillonnage actif de l'algorithme PID r_Ctrl_Cycle est déterminé pendant l'optimisation et arrondi à un multiple entier de r_Cycle. r_Ctrl_Cycle est utilisé comme période de la modulation de largeur d'impulsion. r_Ctrl_Cycle est rémanent.

Voir aussi

Charger des objets technologiques dans l'appareil (Page 48)

10.1.5.5 Paramètres State et sRet.i_Mode V1

Corrélation entre les paramètres

Le paramètre State affiche le mode de fonctionnement actuel du régulateur PID. Vous ne pouvez pas modifier le paramètre State.

Pour changer de mode de fonctionnement, vous devez modifier la variable sRet.i_Mode. Ceci est valable également lorsque la valeur pour le nouveau mode de fonctionnement figure déjà dans sRet.i_Mode. Dans ce cas, réglez d'abord sRet.i_Mode = 0 puis ensuite sRet.i_Mode = 3. Si le mode de fonctionnement actuel autorise ce changement, State est mis sur la valeur de sRet.i_Mode.

Si PID_Compact change automatiquement le mode de fonctionnement, alors : State != sRet.i_Mode.

Exemples :

- Optimisation préalable réussie
  State = 3 et sRet.i_Mode = 1
- En cas d'erreur
  State = 0 et sRet.i_Mode reste à la valeur en cours, par exemple sRet.i_Mode = 3
- ManualEnalbe = TRUE
  State = 4 et sRet.i_Mode reste à la valeur en cours, par exemple sRet.i_Mode = 3

REMARQUE

Vous souhaitez par exemple répéter une optimisation fine réussie sans terminer le mode de fonctionnement automatique avec i_Mode = 0.

Si vous mettez sRet.i_Mode à une valeur non valide pour un cycle, par exemple 9999, cela n'a aucun effet sur State. Au cycle suivant, vous réglez Mode = 2. Vous pouvez ainsi obtenir une modification de sRet.i_Mode sans passer d'abord au mode de fonctionnement "Inactif".

Signification des valeurs

State / sRet.i_Mode	Description du mode de fonctionnement
0	Inactif Le régulateur est arrêté. Avant la réalisation d'une optimisation préalable, le régulateur est en mode de fonctionnement "Inactif". Durant le fonctionnement, le régulateur PID passe au mode de fonctionnement "Inactif" si une erreur survient ou si vous cliquez sur l'icône "Désactiver le régulateur" dans la fenêtre de mise en route.
1	Optimisation préalable L'optimisation préalable détermine la réponse du processus à un échelon de la valeur de réglage et recherche le point d'inflexion. Les paramètres PID optimisés sont calculés à partir de l'incrémentation maximale et du temps mort du système réglé. Conditions pour une optimisation préalable : • le régulateur se trouve en mode de fonctionnement Inactif ou Mode manuel. • ManualEnable = FALSE • La mesure ne doit pas être trop proche de la consigne. |Setpoint - Input| > 0.3 * |sPid_Cmpt.r_Pv_Hlm - sPid_Cmpt.r_Pv_Llm| et |Setpoint - Input| > 0.5 * |Setpoint| • La consigne ne doit pas être modifiée pendant l'optimisation préalable.

State / sRet.i_Mode	Description du mode de fonctionnement
	Plus la mesure est stable, plus il sera facile de déterminer des paramètres PID précis. Un bruit de la mesure est acceptable tant que la croissance de la mesure est nettement supérieure au bruit. Avant que les paramètres PID soient recalculés, ils sont sauvegardés et peuvent être réactivés avec sPid_Cmpt.b_LoadBackUp. Après une optimisation préalable réussie, le mode de fonctionnement passe en automatique ; si l'optimisation préalable échoue, le mode de fonctionnement passe au mode "Inactif". La phase d'optimisation préalable est indiquée par Variable i_Event_SUT V1 (Page 294).
2	Optimisation fine L'optimisation fine génère une oscillation constante limitée de la mesure. Les paramètres PID sont optimisés à partir de l'amplitude et de la fréquence de cette oscillation. Les différences entre la réponse du processus pendant l'optimisation préalable et l'optimisation fine sont analysées. Tous les paramètres PID sont recalculés à partir des résultats. Les paramètres PID de l'optimisation fine montrent généralement un meilleur comportement de référence et de perturbation que les paramètres PID de l'optimisation préalable. PID_Compact essaie automatiquement de créer une oscillation supérieure au bruit de la mesure. La stabilité de la mesure n'influence l'optimisation fine que de manière insignifiante. Avant que les paramètres PID soient recalculés, ils sont sauvegardés et peuvent être réactivés avec sPid_Cmpt.b_LoadBackUp. Conditions pour une optimisation fine : • Aucune perturbation n'est attendue. • La consigne et la mesure sont dans les limites configurées • La consigne ne doit pas être modifiée pendant l'optimisation fine. • ManualEnable = FALSE • Mode automatique (State = 3), inactif (State = 0) ou mode manuel (State = 4) L'optimisation fine se déroule de la manière suivante au démarrage : • Mode automatique (State = 3) Si vous souhaitez améliorer les paramètres PID existants à l'aide de l'optimisation, démarrez l'optimisation fine à partir du mode automatique. PID_Comact effectue un réglage avec les paramètres PID existants jusqu'à ce que la boucle de régulation soit en régime stationnaire et que les conditions pour une optimisation fine soient remplies. C'est seulement après cela que l'optimisation fine commence. • Inactif (State = 0) ou mode manuel (State = 4) Une optimisation préalable est lancée lorsque les conditions correspondantes sont réunies. Un réglage a lieu avec les paramètres PID déterminés jusqu'à ce que la boucle de régulation soit en régime stationnaire et que les conditions pour une optimisation fine soient remplies. C'est seulement après cela que l'optimisation fine commence. Si l'optimisation préalable n'est pas possible, PID_Compact passe en mode de fonctionnement "Inactif". Quand la mesure est déjà trop proche de la consigne pour une optimisation préalable ou que sPid_Calc.b_RunIn = TRUE, le système essaie d'atteindre la consigne avec la valeur de réglage mini ou maxi. Cela peut entraîner une suroscillation élevée. Après avoir effectué correctement une "optimisation fine", le régulateur passe au mode de fonctionnement "Mode automatique" ; en cas d'échec de "l'optimisation fine", il passe au mode "Inactif". La phase d'"optimisation fine" est affichée avec la Variable i_Event_TIR V1 (Page 294).
3	Mode automatique En mode automatique, PID_Compact régule le système réglé en fonction des paramètres prédéfinis. Si l'une des conditions préalables suivantes est remplie, le système passe au mode automatique : • Optimisation préalable réussie • Optimisation fine réussie • Modification de la variable sRet.i_Mode sur la valeur 3. Quand la CPU est mise en route ou passe de STOP à RUN, PID_Compact démarre dans le dernier mode de fonctionnement actif. Pour laisser PID_Compact en mode "Inactif", mettez sb_RunModeByStartup = FALSE.

State / sRet.i_Mode	Description du mode de fonctionnement
4	Mode manuel En mode manuel, vous spécifiez une valeur de réglage manuelle au paramètre ManualValue. Ce mode de fonctionnement est activé si sRet.i_Mode = 4 ou en cas de front montant sur ManualEnable. Si ManualEnable = TRUE, seul State est modifié. sRet.i_Mode reste sur la même valeur. En cas de front descendant sur ManualEnable, PID_Compact repasse au mode de fonctionnement précédent. Le passage au mode automatique s'effectue sans à-coups.

Voir aussi

Paramètres de sortie PID_Compact V1 (Page 283)
Optimisation préalable V1 (Page 112)
Optimisation fine V1 (Page 114)
Mode de fonctionnement "Mode manuel" V1 (Page %getreference)
Variable i_Event_SUT V1 (Page 294)
Variable i_Event_TIR V1 (Page 294)

10.1.5.6 Paramètre Error V1

En présence de plusieurs erreurs simultanées, les valeurs des codes d'erreur s'affichent comme addition binaire. L'affichage du code d'erreur 0003, par ex., indique la présence simultanée des erreurs 0001 et 0002.

Error (DW#16#...)	Description
0000	Pas de présence d'erreur.
0001	Le paramètre "Input" se trouve en dehors des limites de la mesure. • Input > sPid_Cmpt.r_Pv_Hlm ou • Input < sPid_Cmpt.r_Pv_Llm Vous ne pourrez déplacer à nouveau l'actionneur qu'après avoir éliminé l'erreur.
0002	Valeur invalide au paramètre "Input_PER". Vérifiez si une erreur est présente à l'entrée analogique.
0004	Erreur pendant l'optimisation fine. L'oscillation de la mesure n'a pas pu être maintenue.
0008	Erreur lors du démarrage de l'optimisation préalable. La mesure est trop proche de la consigne. Démarrez l'optimisation fine.
0010	La consigne a été modifiée durant l'optimisation.
0020	L'optimisation préalable n'est pas autorisée en mode automatique et pendant l'optimisation fine.
0080	Erreur lors de l'optimisation préalable. Les limites de la valeur de réglage ne sont pas configurées correctement ou la mesure ne réagit pas comme prévu. Vérifiez si les limites de la valeur de réglage sont configurées correctement et conviennent au sens de régulation. Assurez-vous en outre que la mesure n'oscille pas fortement avant le début de l'optimisation préalable.
0100	Une erreur durant l'optimisation a conduit à des paramètres invalides.
0200	Valeur invalide au paramètre "Input" : Le format numérique de la valeur est invalide.
0400	Le calcul de la valeur de réglage a échoué. Vérifiez les paramètres PID.
0800	Erreur de période d'échantillonnage : PID_Compact n'est pas appelé pendant la période d'échantillonnage de l'OB d'alarme cyclique. Si cette erreur s'est produite lors de la simulation avec PLCSIM, tenez compte des indications de la rubrique Simuler PID_Compact V1 avec PLCSIM (Page 117).

Error (DW#16#...)	Description
1000	Valeur invalide au paramètre "Setpoint" : Le format numérique de la valeur est invalide.

Voir aussi

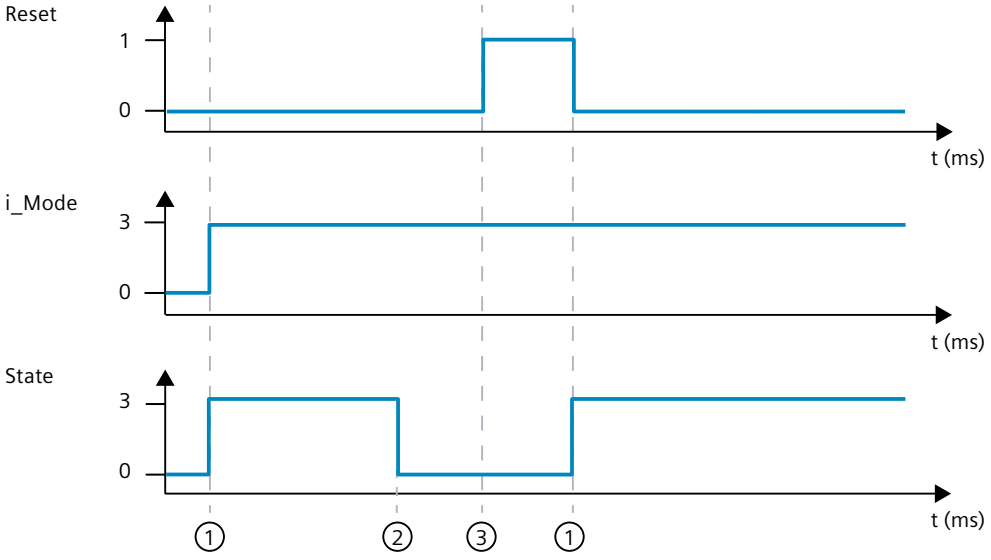
Paramètres de sortie PID_Compact V1 (Page 283)

10.1.5.7 Paramètre Reset V1

Le comportement si Reset = TRUE dépend de la version de l'instruction PID_Compact.

Comportement Reset PID_Compact à partir de V.1.1

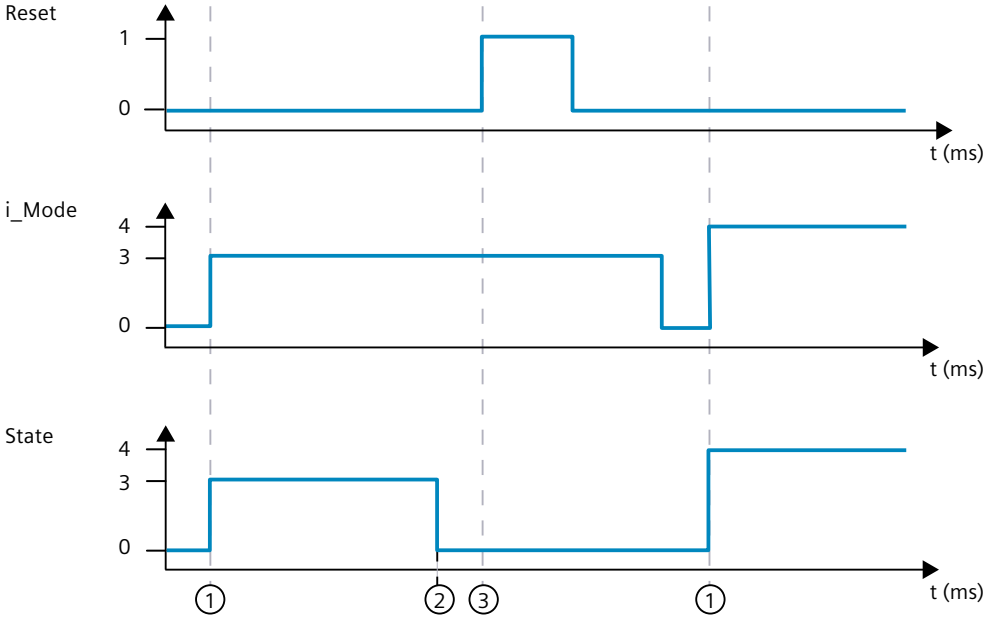
Un front montant sur Reset fait passer en mode de fonctionnement "inactif", remet à zéro les erreurs et les avertissements et supprime l'action I. Un front descendant sur Reset fait passer au dernier mode de fonctionnement actif. Si le mode automatique était actif précédemment, l'action I est préréglée de manière à ce que la commutation s'effectue sans à-coup.



① Activation
② Erreur
③ Reset

Comportement Reset PID_Compact V.1.0

Un front montant sur Reset fait passer en mode de fonctionnement "inactif", remet à zéro les erreurs et les avertissements et supprime l'action I. Le régulateur n'est réactivé que par un front sur i_Mode.



① Activation
② Erreur
③ Reset

10.1.5.8 Variable sd_warning V1

En présence simultanée de plusieurs alertes, les valeurs des variables sd_warning sont affichées sous forme d'addition binaire. Si l'alerte affiche 0003 p. ex., cela indique la présence simultanée des alertes 0001 et 0002.

sd_warning (DW#16#....)	Description
0000	Aucune alerte n'est présente.
0001	Le point d'inflexion n'a pas été trouvé pendant l'optimisation préalable.
0002	L'oscillation était renforcée pendant l'optimisation fine.
0004	La consigne se trouvait en dehors des limites paramétrées.
0008	Toutes les propriétés nécessaires du système réglé n'ont pas été déterminées pour la méthode de calcul choisie. Les paramètres PID ont été calculés avec la méthode "i_CtrlTypeTIR = 3" à titre de remplacement.
0010	Impossible de modifier le mode de fonctionnement car ManualEnable = TRUE
0020	Le temps d'échantillonnage de l'algorithme PID est limité par le temps de cycle de l'OB appelant. Afin d'obtenir de meilleurs résultats, utilisez des temps de cycle de l'OB plus courts.
0040	La mesure a dépassé l'une de ses limites d'alerte.

Les alarmes suivantes sont supprimées dès que la cause est éliminée :

- 0004
- 0020
- 0040

Toutes les autres alarmes sont supprimées avec un front montant sur Reset.

10.1.5.9 Variable i_Event_SUT V1

i_Event_SUT	Nom	Description
0	SUT_INIT	Initialiser l'optimisation préalable
100	SUT_STDABW	Calculer divergence standard
200	SUT_GET_POI	Déterminer point d'inflexion
9900	SUT_IO	Optimisation préalable réussie
1	SUT_NIO	Optimisation préalable échouée

Voir aussi

Variables statiques PID_Compact v1 (Page 284)
Paramètres State et sRet.i_Mode V1 (Page 289)

10.1.5.10 Variable i_Event_TIR V1

i_Event_TIR	Nom	Description
-100	TIR_FIRST_SUT	L'optimisation fine n'est pas possible. Une optimisation préalable est d'abord réalisée.
0	TIR_INIT	Initialiser l'optimisation fine
200	TIR_STDABW	Calculer divergence standard
300	TIR_RUN_IN	Tentative d'atteindre la consigne
400	TIR_CTRLN	Essayer d'atteindre la consigne avec les paramètres PID existants (si l'optimisation préalable a réussi)
500	TIR_OSZIL	Déterminer oscillation et calculer paramètres
9900	TIR_IO	Optimisation fine réussie
1	TIR_NIO	Optimisation fine échouée

Voir aussi

Variables statiques PID_Compact v1 (Page 284)
Paramètres State et sRet.i_Mode V1 (Page 289)

10.2 PID_3Step

10.2.1 Nouveautés PID_3Step

PID_3Step V2.3

- À partir de PID_3Step version 2.3, il est possible de désactiver la surveillance et la limitation du temps de course avec Config.VirtualActuatorLimit = 0.0.

PID_3Step V2.2

- **Utilisation avec S7-1200**
  A partir de PID_3Step V2.2, il est possible d'utiliser l'instruction avec une fonctionnalité V2, y compris sur une S7-1200 à partir de la version de firmware 4.0.

PID_3Step V2.0

- **Comportement en cas d'erreur**
  Le comportement avec ActivateRecoverMode = TRUE a été revu intégralement. PID_3Step est maintenant plus tolérant aux erreurs dans le réglage par défaut.

IMPORTANT
Votre installation peut être endommagée. Quand vous utilisez le réglage par défaut, PID_3Step reste en mode automatique aussi en cas de dépassement des limites de la mesure. Cela peut endommager votre installation. Configurez un comportement en cas d'erreur pour votre système réglé, qui protège votre installation de tout endommagement.

  Utilisez le paramètre d'entrée ErrorAck pour acquitter les erreurs et avertissements sans redémarrer le régulateur ni supprimer l'action I.
  Les erreurs qui ne sont plus présentes ne sont pas acquittées par un changement du mode de fonctionnement.
- **Changement de mode de fonctionnement**
  Le mode de fonctionnement est défini au paramètre d'entrée/sortie Mode et est démarré via un front montant à ModeActivate. La variable Retain.Mode est supprimée.
  La mesure du temps de positionnement ne peut plus être démarrée à l'aide de GetTransitTime.Start, mais uniquement avec Mode = 6 et un front montant sur ModeActivate.
- **Fonctionnalité multiinstance**
  Vous pouvez appeler PID_3Step en tant que DB multiinstance.
- **Comportement au démarrage**
  Le mode de fonctionnement défini à Mode est également démarré en cas de front descendant à Reset et en cas de démarrage à froid de la CPU, si RunModeByStartup = TRUE.

- **Comportement ENO**
  ENO est défini en fonction du mode de fonctionnement.
  Si State = 0, alors ENO = FALSE.
  Si State ≠ 0, alors ENO = TRUE.
- **Mode manuel**
  Les paramètres d'entrée Manual_UP et Manual_DN ne sont activés par les fronts. Le mode manuel activé par les fronts est toujours disponible via les variables ManualUpInternal et ManualDnInternal.
  En "mode manuel sans signaux de butée" (Mode = 10), les signaux de butée Actuator_H et Actuator_L sont ignorés bien qu'ils soient activés.
- **Valeurs par défaut des paramètres PID**
  Les valeurs par défaut suivantes ont été modifiées :
  - Pondération de l'action P (PWeighting) de 0.0 à 1.0
  - Pondération de l'action D (DWeighting) de 0.0 à 1.0
  - Coefficient de l'action par dérivation (TdFiltRatio) de 0.0 à 0.2
- **Limitation du temps de positionnement du moteur**
  A la variable Config.VirtualActuatorLimit, vous configurez de quel % du temps de positionnement du moteur l'actionneur peut se déplacer au maximum dans un sens.
- **Définition de la valeur de la consigne pendant l'optimisation**
  Les fluctuations autorisées de la valeur de consigne pendant l'optimisation sont configurées à la variable CancelTuningLevel.
- **Application d'une grandeur perturbatrice**
  Vous pouvez appliquer une valeur perturbatrice au paramètre Disturbance.
- **Correction d'erreur**
  Si les signaux de butée ne sont pas activés (ActuatorEndStopOn = FALSE), Actuator_H et Actuator_L ne sont plus pris en compte pour déterminer ScaledFeedback.

PID_3Step V1.1

- **Fonctionnement manuel à la mise en route de la CPU**
  Lorsque ManualEnable = TRUE au démarrage de la CPU, PID_3Step démarre en mode manuel. Un front montant ManualEnable n'est pas nécessaire.
- **Comportement en cas d'erreur**
  La variable ActivateRecoverMode ne s'applique plus en mode manuel.
- **Correction d'erreur**
  La variable Progress est réinitialisée après une optimisation réussie ou une mesure du temps de positionnement.

10.2.2 Compatibilité avec CPU et FW

Le tableau suivant montre les CPU et les versions de PID_3Step compatibles.

CPU	FW	PID_3Step
S7-1200	à partir de V4.2	V2.3 V2.2 V1.1
	V4.0 à V4.1	V2.2 V1.1
	V3.x	V1.1 V1.0
	V2.x	V1.1 V1.0
	V1.x	-
S7-1500	à partir de V3.0	V2.3
	V2.0 à V2.9	V2.3 V2.2 V2.1 V2.0
	V1.5 à V1.8	V2.2 V2.1 V2.0
	V1.1	V2.1 V2.0
	V1.0	V2.0

10.2.3 Temps de traitement de la CPU et espace mémoire requis PID_3Step V2.x

Temps de traitement de la CPU

Temps de traitement de CPU typiques de l'objet technologique PID_3Step à partir de la version V2.0 en fonction du type de CPU et mode de fonctionnement pour CPU standard, F, T et TF.

CPU	Firm-ware	Temps de traitement de CPU typ. Mode automatique	Temps de traitement de CPU typ. Optimisation préalable et optimisation fine
CPU 1211	≥ V4.0	260 µs	310 µs
CPU 1212			
CPU 1214			
CPU 1215			
CPU 1217			
CPU 1510SP	≤ V2.9	80 µs	95 µs
CPU 1511			
CPU 1511C			
CPU 1512C			
CPU 1512SP			
CPU 1513			
CPU 1515		70 µs	80 µs
CPU 1516			
CPU 1517	Chaque	11 µs	15 µs
CPU 1518		5 µs	7 µs
CPU 1510SP	≥ V3.0	65 µs	85 µs
CPU 1511			
CPU 1511C			
CPU 1512C			
CPU 1512SP			
CPU 1513			
CPU 1514SP			
CPU 1515		50 µs	70 µs
CPU 1516			

Temps de traitement de CPU typiques de l'objet technologique PID_3Step à partir de la version V2.0 en fonction du type de CPU et mode de fonctionnement pour des CPU R dans l'état système RUN-Redundant.

CPU	Firm-ware	Temps de traitement de CPU typ. mode automatique	Temps de traitement de CPU typ. Optimisation préalable et optimisation fine
CPU 1513R	≥ V3.0	110 µs	150 µs
CPU 1515R		85 µs	100 µs

Espace mémoire requis

Espace mémoire requis d'un DB d'instance de l'objet technologique PID_3Step à partir de la version V2.0.

Espace mémoire requis	Espace mémoire requis du DB d'instance de PID_3Step V2.x
Taille de mémoire de chargement requise	env. 4400 octets
Taille totale de la mémoire de travail requise	1040 octets
Taille de la mémoire rémanente requise	60 octets

10.2.4 PID_3Step V2

10.2.4.1 Description PID_3Step V2

Description

L'instruction PID_3Step permet de configurer un régulateur PID avec auto-optimisation pour les vannes ou actionneurs à comportement intégral.

Les modes suivants sont disponibles :

- Inactif
- Optimisation préalable
- Optimisation fine
- Mode automatique
- Mode manuel
- Accoster la valeur de réglage de remplacement
- Mesure du temps de positionnement
- Surveillance des erreurs
- Accoster la valeur de réglage de remplacement avec surveillance d'erreur
- Mode manuel sans signaux de butée

Les modes de fonctionnement sont décrits en détail dans le paramètre State.

Algorithme PID

PID_3Step est un régulateur PIDT1 avec anti-saturation et pondération de l'action P et D. L'algorithme PID fonctionne selon la formule suivante :

$$\Delta y = K_p \cdot s \cdot \left[(b \cdot w - x) + \frac{1}{T_I \cdot s}(w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1}(c \cdot w - x) \right]$$

Δy	Valeur de réglage de l'algorithme PID
K_p	Gain proportionnel
s	Opérateur de Laplace
b	Pondération de l'action P
w	Consigne
x	Mesure
T_I	Temps d'intégration
T_D	Temps de dérivation
a	Coefficient pour l'action par dérivation (action par dérivation T1 = a × T_D)
c	Pondération de l'action D

Schéma fonctionnel sans signalisation de position

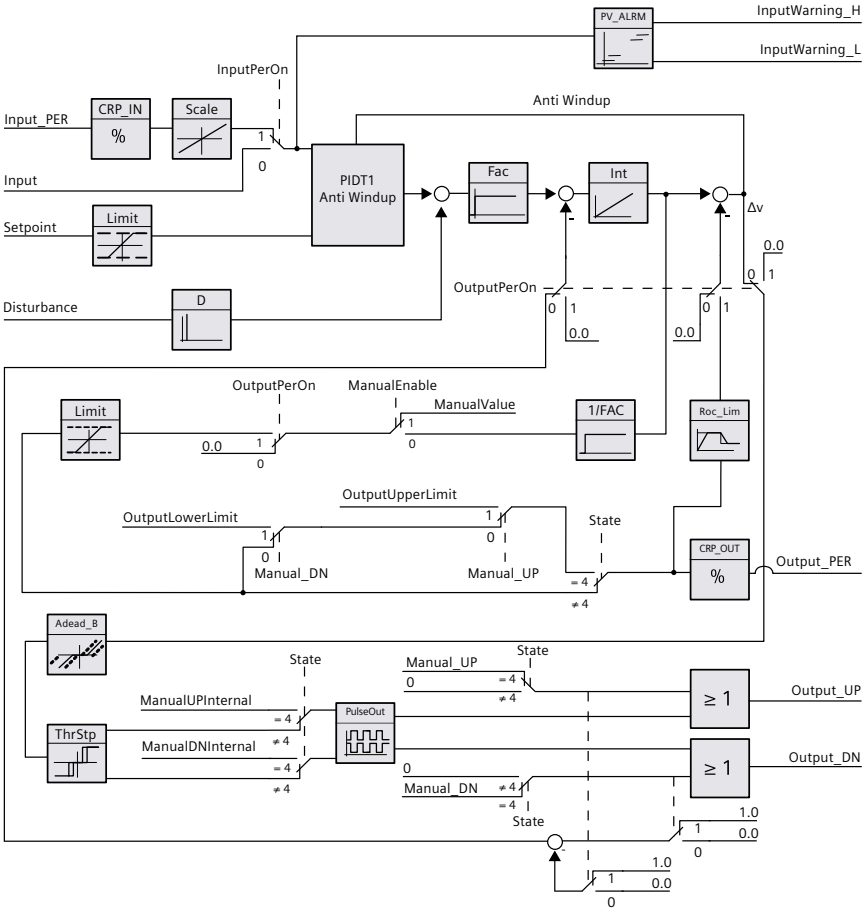


Schéma fonctionnel avec signalisation de position

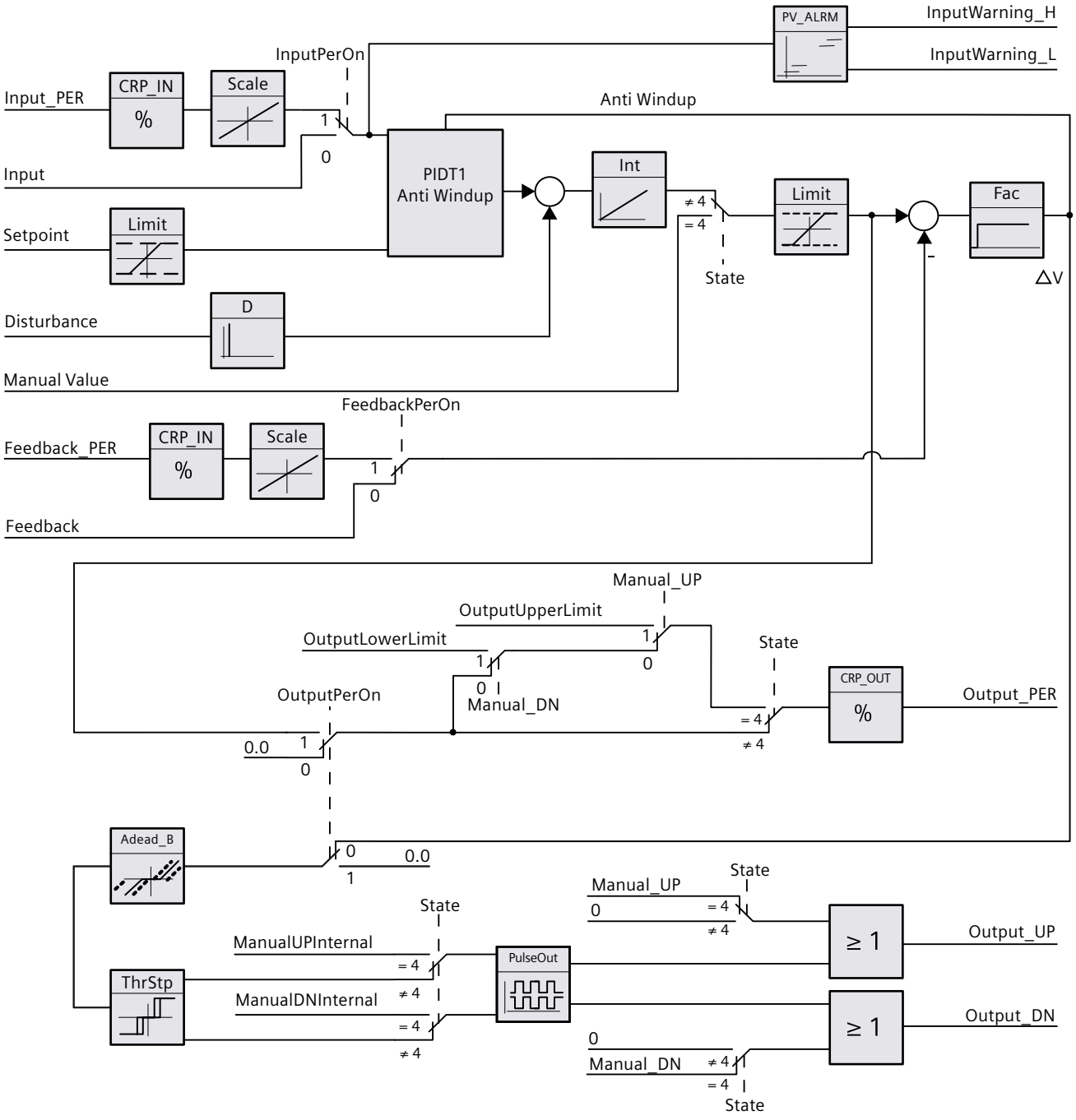
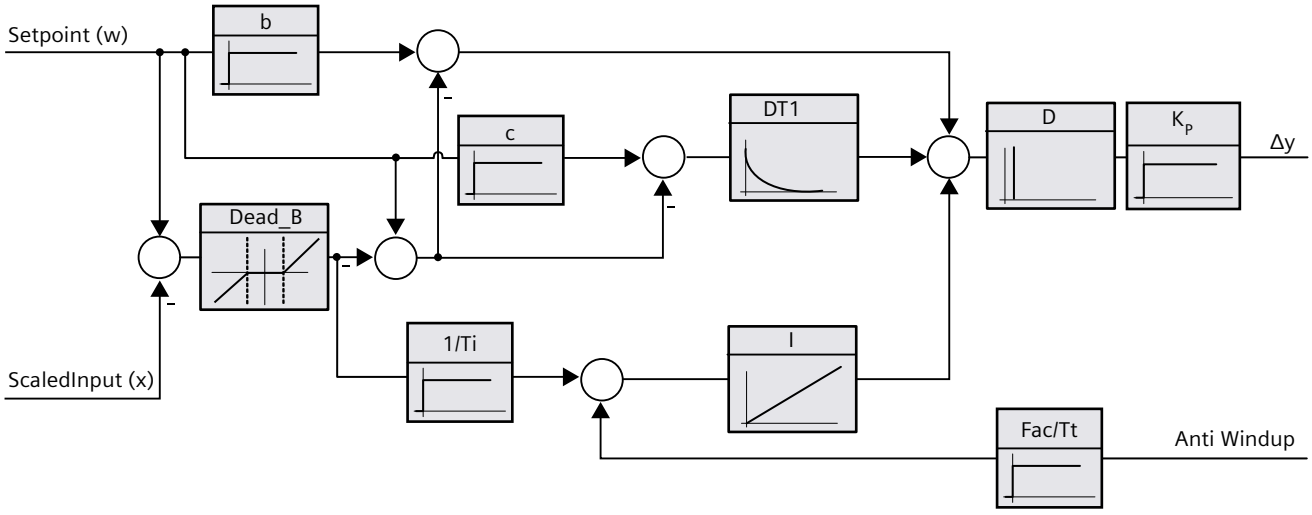


Schéma fonctionnel PIDT1 avec anti-saturation



Appel

PID_3Step est appelé à intervalle de temps constant d'une alarme cyclique de l'OB.

Chargement dans l'appareil

Les valeurs effectives de variables rémanentes ne sont actualisées que si vous chargez entièrement PID_3Step.
Charger des objets technologiques dans l'appareil (Page 48)

Démarrage

Au démarrage de la CPU, PID_3Step démarre dans le mode de fonctionnement enregistré aux paramètres d'entrée/sortie Mode. Pour laisser PID_3Step en mode "Inactif", mettez RunModeByStartup = FALSE.

Comportement en cas d'erreur

En mode automatique et pendant la mise en service, le comportement en cas d'erreur dépend des variables ErrorBehaviour et ActivateRecoverMode. En mode manuel, le comportement est indépendant de ErrorBehaviour et ActivateRecoverMode. Si ActivateRecoverMode = TRUE, le comportement dépend en outre de la nature de l'erreur.

ErrorBehaviour	**ActivateRecoverMode**	**Éditeur de configuration > Paramétrage de l'actionneur > Régler Output**	**Comportement**
FALSE	FALSE	Valeur de réglage actuelle	Passage au mode de fonctionnement "Inactif" (State = 0) L'actionneur reste dans la position actuelle.
FALSE	TRUE	Valeur de réglage actuelle pour la durée de l'erreur	Passage au mode de fonctionnement "Surveillance d'erreur" (State = 7) L'actionneur reste dans la position actuelle pour la durée de l'erreur.
TRUE	FALSE	Valeur de réglage de remplacement	Passage au mode de fonctionnement "Accoster la valeur de réglage de remplacement" (State = 5) L'actionneur est amené à la valeur de réglage de remplacement configurée. Passage au mode de fonctionnement "Inactif" (State = 0) L'actionneur reste dans la position actuelle.
TRUE	TRUE	Valeur de réglage de remplacement pour la durée de l'erreur	Passage au mode de fonctionnement "Accoster la valeur de réglage de remplacement avec surveillance d'erreur" (State = 8) L'actionneur est amené à la valeur de réglage de remplacement configurée. Passage au mode de fonctionnement "Surveillance d'erreur" (State = 7)

PID_3Step utilise ManualValue comme valeur de réglage en mode manuel, sauf pour les erreurs suivantes :

- 2000h : Valeur invalide au paramètre Feedback_PER.
- 4000h : Valeur invalide au paramètre Feedback.
- 8000h : Erreur dans la signalisation de position TOR.

Vous pouvez alors modifier la position de l'actionneur uniquement avec Manual_UP et Manual_DN, mais pas avec ManualValue :

Le paramètre Error signale l'apparition éventuelle d'erreur dans ce cycle. Le paramètre ErrorBits indique les erreurs qui se sont produites. ErrorBits est réinitialisé par un front montant àReset ou ErrorAck.

Voir aussi

Paramètres State et Mode V2 (Page 321)
Paramètre ErrorBits V2 (Page 326)
Configurer PID_3Step V2 (Page 119)

10.2.4.2 Mode opératoire PID_3Step V2

Surveiller les limites de mesure

Vous définissez une limite supérieure et une limite inférieure de la mesure dans les variables Config.InputUpperLimit et Config.InputLowerLimit. Si la mesure se trouve en dehors de ces limites, une erreur survient (ErrorBits = 0001h).

Vous définissez une limite d'alerte supérieure et une limite d'alerte inférieure de la mesure dans les variables Config.InputUpperWarning et Config.InputLowerWarning. Si la mesure se trouve en dehors de ces limites d'alerte, une alerte survient (Warning = 0040h) et le paramètre de sortie InputWarning_H ou InputWarning_L passe à TRUE.

Limiter consigne

Vous définissez une limite supérieure et inférieure de la consigne dans les variables Config.SetpointUpperLimit et Config.SetpointLowerLimit. PID_3Step limite automatiquement la consigne aux limites de la mesure. Vous pouvez limiter la consigne à une plage inférieure. PID_3Step contrôle si cette plage se trouve dans les limites de la mesure. Si la consigne se trouve hors de ces limites, les limites inférieure et supérieure sont utilisées comme consigne et le paramètre de sortie SetpointLimit_H ou SetpointLimit_L passe à TRUE.

La consigne est limitée dans tous les modes de fonctionnement.

Limiter la valeur de réglage

Vous déterminez une limite supérieure et une limite inférieure de la valeur de réglage dans les variables Config.OutputUpperLimit et Config.OutputLowerLimit. Les limites de la valeur de réglage doivent se trouver entre la "butée inférieure" et la "butée supérieure".

- Butée supérieure : Config.FeedbackScaling.UpperPointOut
- Butée inférieure : Config.FeedbackScaling.LowerPointOut

Règle à appliquer :

UpperPointOut ≥ OutputUpperLimit > OutputLowerLimit ≥ LowerPointOut

Les valeurs valables de la "Butée supérieure" et de la "Butée inférieure" dépendent de :

- FeedbackOn
- FeedbackPerOn
- OutputPerOn

OutputPerOn	FeedbackOn	FeedbackPerOn	LowerPointOut	UpperPointOut
FALSE	FALSE	FALSE	non réglable (0.0 %)	non réglable (100.0 %)
FALSE	TRUE	FALSE	-100.0 % ou 0.0 %	0.0 % ou +100.0 %
FALSE	TRUE	TRUE	-100.0 % ou 0.0 %	0.0 % ou +100.0 %

OutputPerOn	FeedbackOn	FeedbackPerOn	LowerPointOut	UpperPointOut
TRUE	FALSE	FALSE	non réglable (0.0 %)	non réglable (100.0 %)
TRUE	TRUE	FALSE	-100.0 % ou 0.0 %	0.0 % ou +100.0 %
TRUE	TRUE	TRUE	-100.0 % ou 0.0 %	0.0 % ou +100.0 %

Si OutputPerOn = FALSE et FeedbackOn = FALSE, vous ne pouvez pas limiter la valeur de réglage. Output_UP et Output_DN sont alors remis à 0 si Actuator_H = TRUE ou Actuator_L = TRUE. Si aucun signal de butée n'est disponible, Output_UP et Output_DN sont remis à 0 après un temps de course de Config.VirtualActuatorLimit × Retain.TransitTime/100. À partir de PID_3Step version 2.3, il est possible de désactiver la surveillance et la limitation du temps de course avec Config.VirtualActuatorLimit = 0.0.

La valeur de réglage est de 27648 pour 100 % et -27648 pour -100 %. PID_3Step doit fermer complètement la vanne.

REMARQUE

Utilisation avec deux actionneurs ou plus

PID_3 Step ne convient pas pour l'utilisation avec deux actionneurs ou plus (p. ex. dans des applications de chauffage ou de refroidissement), car des actionneurs différents ont besoin de paramètres PID différents pour obtenir un bon comportement de régulation.

Valeur de réglage de remplacement

En cas d'erreur, PID_3Step peut fournir une valeur de réglage de remplacement et placer l'actionneur dans une position sûre, que vous spécifiez au niveau de la variable SavePosition. La valeur de réglage de remplacement doit être dans les limites de la valeur de réglage.

Surveiller la validité des signaux

En cas d'utilisation, la validité des valeurs des paramètres suivants est surveillée :

- Setpoint
- Input
- Input_PER
- Input_PER
- Feedback
- Feedback_PER
- Disturbance
- ManualValue
- SavePosition
- Output_PER

Surveiller le temps d'échantillonnage PID_3Step

Le temps d'échantillonnage correspond idéalement au temps de cycle de l'OB appelant. L'instruction PID_3Step permet de mesurer l'intervalle de temps entre deux appels respectifs. Le résultat est le temps d'échantillonnage actuel. Lors de chaque changement du mode de fonctionnement et à la première mise en route, une moyenne est calculée à partir des 10 premiers temps d'échantillonnage. Si la période d'échantillonnage actuelle diverge trop de cette valeur moyenne, une erreur survient (ErrorBits = 0800h).

Une erreur survient en cours d'optimisation si :

- Nouvelle valeur moyenne >= 1,1 x ancienne valeur moyenne
- Nouvelle valeur moyenne <= 0,9 x ancienne valeur moyenne

Une erreur survient en mode automatique si :

- Nouvelle valeur moyenne >= 1,5 x ancienne valeur moyenne
- Nouvelle valeur moyenne <= 0,5 x ancienne valeur moyenne

Si la surveillance du temps d'échantillonnage est désactivée (CycleTime.EnMonitoring = FALSE), vous pouvez aussi appeler PID_3Step dans OB1. Dans ce cas, vous devez accepter une moindre qualité de régulation du fait de la fluctuation du temps d'échantillonnage.

Temps d'échantillonnage de l'algorithme PID

Comme le système réglé nécessite un certain temps pour réagir à une modification de la valeur de réglage, il est judicieux de ne pas calculer cette valeur à chaque cycle. Le temps d'échantillonnage de l'algorithme PID est le temps entre deux calculs de valeurs de réglage. Il est déterminé pendant l'optimisation et arrondi à un multiple du temps de cycle. Toutes les autres fonctions de PID_3Step sont exécutées lors de chaque appel.

Mesurer le temps de positionnement du moteur

Le temps de positionnement du moteur est le temps en secondes requis par le moteur pour faire passer l'actionneur de l'état fermé à l'état ouvert. L'actionneur est déplacé dans un sens au maximum Config.VirtualActuatorLimit × Retain.TransitTime/100. PID_3Step a besoin d'un temps de positionnement du moteur aussi exact que possible pour obtenir un bon résultat de régulation. Les indications dans la documentation de l'actionneur sont des valeurs moyennes pour ce type d'actionneur. La valeur peut être différente pour l'actionneur utilisé réellement. Vous pouvez mesurer le temps de positionnement du moteur pendant la mise en service. Les limites de valeur de réglage ne sont pas prises en compte lors de la mesure du temps de positionnement du moteur. Il est possible de déplacer l'actionneur jusqu'à la butée supérieure ou inférieure.

Le temps de positionnement du moteur est pris en compte dans le calcul de la valeur de réglage analogique ainsi que dans le calcul des valeurs de réglage TOR. Il est surtout nécessaire pour un fonctionnement correct lors de l'auto-optimisation et du comportement anti-windup. Configurez par conséquent le temps de positionnement du moteur à la valeur nécessaire au moteur pour faire passer l'actionneur de l'état fermé à l'état ouvert.

Si aucun temps de positionnement du moteur pertinent n'est actif (par ex. dans le cas d'électrovannes), si bien que la valeur de réglage est appliquée directement et entièrement au processus, utilisez à la place PID_Compact.

Sens de régulation

La plupart du temps, une augmentation de la mesure doit être atteinte avec une augmentation de la valeur de réglage. Dans ce cas, on parle d'un sens de régulation normal. Il peut être nécessaire d'inverser le sens de régulation pour les refroidissements et les régulations d'écoulement. PID_3Step ne fonctionne pas avec un gain proportionnel négatif. Si InvertControl = TRUE, un signal d'écart croissant provoque une diminution de la valeur de réglage. Le sens de régulation est pris en compte aussi pendant l'optimisation préalable et l'optimisation fine.

Voir aussi

Configurer PID_3Step V1 (Page 137)

10.2.4.3 Modifications de l'interface PID_3Step V2

Le tableau suivant indique ce qui a changé sur l'interface de l'instruction PID_3Step.

PID_3Step V1	PID_3Step V2	Modification
Input_PER	Input_PER	Type de données de Word à Int
Feedback_PER	Feedback_PER	Type de données de Word à Int
	Disturbance	Nouveau
Manual_UP	Manual_UP	Fonction
Manual_DN	Manual_DN	Fonction
	ErrorAck	Nouveau
	ModeActivate	Nouveau
Output_PER	Output_PER	Type de données de Word à Int
	ManualUPInternal	Nouveau
	ManualDNInternal	Nouveau
	CancelTuningLevel	Nouveau
	VirtualActuatorLImit	Nouveau
Config.Loadbackup	Loadbackup	Renommé
Config.TransitTime	Retain.TransitTime	Renommé et rémanence ajoutée
GetTransitTime.Start		Remplacé par Mode et ModeActivate.
SUT.CalculateSUTParams	SUT.CalculateParams	Renommé
SUT.TuneRuleSUT	SUT.TuneRule	Renommé
TIR.CalculateTIRParams	TIR.CalculateParams	Renommé
TIR.TuneRuleTIR	TIR.TuneRule	Renommé
Retain.Mode	Mode	Fonction Déclaration de la statique au paramètre d'entrée/sortie

10.2.4.4 Paramètres d'entrée PID_3Step V2

Les noms des paramètres suivants sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Tableau 10-7

Paramètre	Type de données	Valeur par défaut	Description
Setpoint	REAL	0.0	Consigne du régulateur PID en mode automatique
Input	REAL	0.0	Une variable du programme utilisateur est utilisée comme source de la mesure. Si vous utilisez le paramètre Input, il faut que Config.InputPerOn = FALSE.
Input_PER	INT	0	Une entrée analogique est utilisée comme source de la mesure. Si vous utilisez le paramètre Input_PER, il faut que Config.InputPerOn = TRUE.
Actuator_H	BOOL	FALSE	Signalisation de position TOR de la vanne pour la butée supérieure Si Actuator_H = TRUE, la position de la vanne est à la butée supérieure et la vanne n'est plus déplacée dans cette direction.
Actuator_L	BOOL	FALSE	Signalisation de position TOR de la vanne pour la butée inférieure Si Actuator_L = TRUE, la position de la vanne est à la butée inférieure et la vanne n'est plus déplacée dans cette direction.
Feedback	REAL	0.0	Signalisation de position de la vanne Si vous utilisez le paramètre Feedback, il faut que Config.FeedbackPerOn = FALSE.
Feedback_PER	INT	0	Signalisation de position analogique d'une vanne Si vous utilisez le paramètre Feedback_PER, il faut que Config.FeedbackPerOn = TRUE. Feedback_PER est mise à l'échelle avec les variables : • Config.FeedbackScaling.LowerPointIn • Config.FeedbackScaling.UpperPointIn • Config.FeedbackScaling.LowerPointOut • Config.FeedbackScaling.UpperPointOut
Disturbance	REAL	0.0	Grandeur perturbatrice ou valeur de commande anticipatrice
ManualEnable	BOOL	FALSE	• Le front FALSE -> TRUE active le mode de fonctionnement "Mode manuel", State = 4, Mode ne change pas. Tant que ManualEnable = TRUE, vous ne pouvez pas changer le mode de fonctionnement via un front montant à ModeActivate ni utilisez la boîte de dialogue de mise en service. • Le front FALSE -> TRUE active le mode de fonctionnement prédéfini à Mode. Il est recommandé de modifier le mode de fonctionnement uniquement via ModeActivate.
ManualValue	REAL	0.0	En mode manuel, la position absolue de la vanne est spécifiée. ManualValue n'est exploité que si vous utilisez Output_PER **ou** qu'une signalisation de position est disponible.

Paramètre	Type de données	Valeur par défaut	Description
Manual_UP	BOOL	FALSE	• Manual_UP = TRUE La vanne est ouverte même si vous utilisez Output_PER ou une signalisation de position. La vanne ne se déplace plus si la butée supérieure est atteinte. Voir aussi Config.VirtualActuatorLimit • Manual_UP = FALSE Si vous utilisez Output_PER ou une signalisation de position, la vanne est mise sur ManualValue. Sinon, la vanne n'est plus déplacée. Si Manual_UP et Manual_DN sont TRUE en même temps, la vanne n'est plus déplacée.
Manual_DN	BOOL	FALSE	• Manual_DN = TRUE La vanne est fermée même si vous utilisez Output_PER ou une signalisation de position. La vanne ne se déplace plus si la butée inférieure est atteinte. Voir aussi Config.VirtualActuatorLimit • Manual_DN = FALSE Si vous utilisez Output_PER ou une signalisation de position, la vanne est mise sur ManualValue. Sinon, la vanne n'est plus déplacée.
ErrorAck	BOOL	FALSE	• Front FALSE -> TRUE ErrorBits et Warning sont remis à zéro.
Reset	BOOL	FALSE	Effectue un redémarrage du régulateur. • Front FALSE -> TRUE – Passage en mode de fonctionnement "Inactif" – ErrorBits et Warnings sont remis à zéro. • Tant que Reset = TRUE, – PID_3Step reste en mode de fonctionnement "Inactif" (State = 0) – vous ne pouvez pas changer de mode de fonctionnement avec Mode et ModeActivate ou ManualEnable. – vous ne pouvez pas utiliser la boîte de dialogue de mise en service. • Front TRUE -> FALSE – Si ManualEnable = FALSE, PID_3Step passe au mode de fonctionnement qui est enregistré dans Mode. – Si Mode = 3, la commutation en mode automatique s'effectue sans à-coups en mode automatique.
ModeActivate	BOOL	FALSE	• Front FALSE -> TRUE PID_3Step passe au mode de fonctionnement qui est enregistré dans Mode".

10.2.4.5 Paramètres de sortie PID_3Step V2

Les noms des paramètres suivants sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Tableau 10-8

Paramètre	Type de données	Valeur par défaut	Description
ScaledInput	REAL	0.0	Mesure mise à l'échelle
ScaledFeedback	REAL	0.0	Signalisation de position à l'échelle Sur un actionneur sans signalisation de position, ScaledFeedback affiche la position de l'actionneur de manière très imprécise. Dans ce cas, ScaledFeedback ne peut être utilisé que pour une estimation grossière de la position réelle.
Output_UP	BOOL	FALSE	Valeur de réglage TOR pour ouvrir la vanne Si Config.OutputPerOn = FALSE, c'est le paramètre Output_UP qui est utilisé.
Output_DN	BOOL	FALSE	Valeur de réglage TOR pour fermer la vanne Si Config.OutputPerOn = FALSE, c'est le paramètre Output_DN qui est utilisé.
Output_PER	INT	0	Valeur de réglage analogique Si Config.OutputPerOn = TRUE, c'est Output_PER qui est utilisé. Utilisez Output_PER si l'actionneur mis en œuvre est une vanne adressée via une sortie analogique et commandée à l'aide d'un signal continu, par exemple 0...10 V, 4...20 mA. La valeur dans Output_PER correspond à la position finale de la vanne, p. ex. Output_PER = 13824, lorsque la vanne doit s'ouvrir à 50 %.
SetpointLimit_H	BOOL	FALSE	Quand SetpointLimit_H = TRUE, la limite supérieure absolue de la consigne est atteinte (Setpoint ≥ Config.SetpointUpperLimit). La consigne est limitée à Config.SetpointUpperLimit .
SetpointLimit_L	BOOL	FALSE	Quand SetpointLimit_L = TRUE, la limite inférieure absolue de la consigne est atteinte (Setpoint ≤ Config.SetpointLowerLimit). La consigne est limitée à Config.SetpointLowerLimit .
InputWarning_H	BOOL	FALSE	Si InputWarning_H = TRUE, la limite d'alerte supérieure de la mesure est atteinte ou dépassée.
InputWarning_L	BOOL	FALSE	Si InputWarning_L = TRUE, la limite d'alerte inférieure de la mesure est atteinte ou dépassée.
State	INT	0	Le paramètre State (Page 321) affiche le mode de fonctionnement actuel du régulateur PID. Le mode de fonctionnement peut être modifié avec le paramètre d'entrée Mode et un front montant à ModeActivate. • State = 0 : Inactif • State = 1 : Optimisation préalable • State = 2 : Optimisation fine • State = 3 : Mode automatique • State = 4 : Mode manuel • State = 5 : Accoster la valeur de réglage de remplacement • State = 6 : Mesure du temps de positionnement • State = 7 : Surveillance des erreurs

Paramètre	Type de données	Valeur par défaut	Description
			• State = 8 : Accoster la valeur de réglage de remplacement avec surveillance d'erreur • State = 10 : Mode manuel sans signaux de butée
Error	BOOL	FALSE	Si Error = TRUE, un message d'erreur au moins existe dans ce cycle.
ErrorBits	DWORD	DW#16#0	Le paramètre ErrorBits (Page 326) signale quels sont les messages d'erreur présents. ErrorBits est rémanent et réinitialisé à Reset ou ErrorAck en cas de front montant.

Voir aussi

Paramètres State et Mode V2 (Page 321)

Paramètre ErrorBits V2 (Page 326)

10.2.4.6 Paramètres d'entrée/sortie PID_3Step V2

Les noms des paramètres suivants sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Tableau 10-9

Paramètre	Type de données	Valeur par défaut	Description
Mode	INT	4	A Mode, définissez le mode dans lequel PID_3Step doit passer. Sont possibles : • Mode = 0 : Inactif • Mode = 1 : Optimisation préalable • Mode = 2 : Optimisation fine • Mode = 3 : Mode automatique • Mode = 4 : Mode manuel • Mode = 6 : Mesure du temps de positionnement • Mode = 10 : Mode manuel sans signaux de butée Le mode de fonctionnement est activé par : • Front montant à ModeActivate • Front descendant à Reset • Front descendant à ManualEnable • Démarrage à froid de la CPU, si RunModeByStartup = TRUE Mode est rémanent. Pour une description détaillée des modes de fonctionnement, voir Paramètres State et Mode V2 (Page 321).

10.2.4.7 Variables statiques PID_3Step V2

REMARQUE

Faites passer les variables identifiées par (1) seulement en mode de fonctionnement "Inactif" pour éviter un comportement erroné du régulateur PID.

Les noms des variables suivantes sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Variable	Type de données	Valeur par défaut	Description
ManualUpInternal	BOOL	FALSE	En mode manuel, chaque front montant ouvre la vanne de 5 % de la plage de réglage totale ou pour le temps de positionnement minimum du moteur. ManualUpInternal n'est exploité que si vous n'utilisez ni Output_PER ni une signalisation de position. Cette variable est utilisée dans la boîte de dialogue de mise en service.
ManualDnInternal	BOOL	FALSE	En mode manuel, chaque front montant ferme la vanne de 5 % de la plage de réglage totale ou pour le temps de positionnement minimum du moteur. ManualDnInternal n'est exploité que si vous n'utilisez ni Output_PER ni une signalisation de position. Cette variable est utilisée dans la boîte de dialogue de mise en service.
ActivateRecoverMode	BOOL	TRUE	La variable ActivateRecoverMode V2 (Page 328) détermine le comportement en cas d'erreur.
RunModeByStartup	BOOL	TRUE	Activer le mode de fonctionnement à Mode après le démarrage de la CPU Si RunModeByStartup = TRUE, au démarrage de la CPU, PID_3Step démarre dans le mode de fonctionnement enregistré sous Mode. Si RunModeByStartup = FALSE, PID_3Step reste en mode "Inactif" après démarrage de la CPU.
LoadBackUp	BOOL	FALSE	Si LoadBackUp = TRUE, le dernier jeu de paramètres PID est rechargé. Le jeu a été enregistré avant la dernière optimisation. LoadBackUp est automatiquement remis sur FALSE.
PhysicalUnit	INT	0	Unité physique de la mesure et de la consigne, par ex. °C ou °F. PhysicalUnit sert à afficher les valeurs dans les éditeurs et n'a aucune influence sur le comportement de l'algorithme de régulation dans la CPU. Lors de l'importation de PID_3Step via API Openness, la valeur par défaut de PhysicalUnit est rétablie.
PhysicalQuantity	INT	0	Grandeur physique de la mesure et de la consigne, par ex. la température PhysicalQuantity sert à afficher les valeurs dans les éditeurs et n'a aucune influence sur le comportement de l'algorithme de régulation dans la CPU. Lors de l'importation de PID_3Step via API Openness, la valeur par défaut de PhysicalQuantity est rétablie.
ErrorBehaviour	BOOL	FALSE	Quand ErrorBehaviour = FALSE, la vanne reste en cas d'erreur dans la position actuelle et le régulateur passe directement en mode de fonctionnement "Inactif" ou "Surveillance des erreurs". Quand ErrorBehaviour = TRUE, l'actionneur est amené, en cas d'erreur, à la valeur de réglage de remplacement et ce n'est qu'après que le système passe en mode de fonctionnement "Inactif" ou "Surveillance des erreurs".

Variable	Type de données	Valeur par défaut	Description
			Quand les erreurs suivantes apparaissent, il n'est plus possible d'amener la vanne à une valeur de réglage de remplacement configurée. • 2000h : Valeur invalide au paramètre Feedback_PER. • 4000h : Valeur invalide au paramètre Feedback. • 8000h : Erreur dans la signalisation de position TOR. • 20000h : Valeur invalide à la variable SavePosition.
Warning	DWORD	DW#16#0	La variable Warning (Page 321) affiche les alertes depuis Reset = TRUE ou ErrorAck =TRUE. Warning est rémanent. Les alertes cycliques (par ex. alerte de mesure) sont affichées pendant toute la durée de la cause de l'alerte. Une fois que la cause a disparu, elles sont automatiquement supprimées. Les alertes non-cycliques (par exemple, point d'inflexion pas trouvé) sont conservées et sont supprimées comme des erreurs.
SavePosition	REAL	0.0	Valeur de réglage de remplacement Si ErrorBehaviour = TRUE, l'actionneur est déplacé, en cas d'erreur, dans une position sûre pour l'installation. Dès que la valeur de réglage de remplacement est atteinte, PID_3Step change de mode en fonction de ActivateRecoverMode. La plage de valeurs admissibles dépend de la configuration. • Config.FeedbackOn = FALSE et Config.OutputPerOn = FALSE : SavePosition = 0.0 ou 100.0 • Config.FeedbackOn = TRUE ou Config.OutputPerOn = TRUE: Config.OutputUpperLimit ≥ SavePosition ≥ Config.OutputLowerLimit
CurrentSetpoint	REAL	0.0	Consigne active actuellement Cette valeur est gelée au démarrage de l'optimisation.
CancelTuningLevel	REAL	10.0	Fluctuations admissibles de la valeur de consigne pendant l'optimisation. Une optimisation est interrompue si : • Setpoint > CurrentSetpoint + CancelTuningLevel ou • Setpoint < CurrentSetpoint - CancelTuningLevel
Progress	REAL	0.0	Progrès de l'optimisation en % (0.0 à 100.0)
Config.InputPerOn(1)	BOOL	TRUE	Si InputPerOn = TRUE, c'est le paramètre Input_PER qui est utilisé. Si InputPerOn = FALSE, c'est le paramètre Input qui est utilisé.
Config.OutputPerOn(1)	BOOL	FALSE	Si OutputPerOn = TRUE, c'est le paramètre Output_PER qui est utilisé. Si OutputPerOn = FALSE, les paramètres Ouput_UP et Output_DN sont utilisés.
Config.InvertControl(1)	BOOL	FALSE	Inversion du sens de régulation Si InvertControl = TRUE, un signal d'écart croissant provoque une diminution de la valeur de réglage.
Config.FeedbackOn(1)	BOOL	FALSE	Si FeedbackOn = FALSE, une signalisation de position est simulée. Si FeedbackOn = TRUE, une signalisation de position est généralement activée.
Config.FeedbackPerOn(1)	BOOL	FALSE	FeedbackPerOn n'a d'effet que si FeedbackOn = TRUE. Si FeedbackPerOn = TRUE, l'entrée analogique est utilisée pour la signalisation de position (paramètre Feedback_PER). Si FeedbackPerOn = FALSE, le paramètre Feedback est utilisé pour la signalisation de position.

Variable	Type de données	Valeur par défaut	Description
Config.ActuatorEndStopOn(1)	BOOL	FALSE	Si ActuatorEndStopOn = TRUE, la signalisation de position TOR Actuator_L et Actuator_H est prise en compte.
Config.InputUpperLimit(1)	REAL	120.0	Limite supérieure de la mesure L'observation de cette limite est surveillée pour Input et Input_PER. A l'entrée de périphérie, la mesure peut dépasser de 18 % au plus la plage normée (dépassement haut). Un dépassement de la "limite supérieure de la mesure" ne provoque plus la signalisation d'erreur. Seuls la rupture de fil et le court-circuit sont détectés et PID_3Step se comporte comme vous en avez décidé sous Comportement en cas d'erreur. InputUpperLimit > InputLowerLimit
Config.InputLowerLimit(1)	REAL	0.0	Limite inférieure de la mesure InputLowerLimit < InputUpperLimit
Config.InputUpperWarning(1)	REAL	+3.402822e+38	Limite d'alerte supérieure de la mesure Si vous configurez InputUpperWarning en dehors des limites de la mesure, la limite supérieure absolue configurée pour la mesure est utilisée comme limite d'alerte supérieure. Si vous configurez InputUpperWarning dans les limites de la mesure, cette valeur est utilisée comme limite d'alerte supérieure. InputUpperWarning > InputLowerWarning InputUpperWarning ≤ InputUpperLimit
Config.InputLowerWarning(1)	REAL	-3.402822e+38	Limite d'alerte inférieure de la mesure Quand vous configurez InputLowerWarning en dehors des limites de la mesure, la limite inférieure absolue configurée pour la mesure est utilisée comme limite d'alerte inférieure. Si vous configurez InputLowerWarning dans les limites de la mesure, cette valeur sera utilisée comme limite d'alerte inférieure. InputLowerWarning < InputUpperWarning InputLowerWarning ≥ InputLowerLimit
Config.OutputUpperLimit(1)	REAL	100.0	Limite supérieure de la valeur de réglage La plage de valeurs suivante est autorisée : UpperPointOut ≥ OutputUpperLimit > OutputLowerLimit Pour plus de détails, voir OutputLowerLimit
Config.OutputLowerLimit(1)	REAL	0.0	Limite inférieure de la valeur de réglage La plage de valeurs suivante est autorisée : OutputUpperLimit > OutputLowerLimit ≥ LowerPointOut Lors de l'utilisation de Output_PER, une limite de la valeur de réglage de -100 % correspond à la valeur Output_PER = -27648 ; 100 % correspond à la valeur Output_PER = 27648. Si OutputPerOn = FALSE et que FeedbackOn = FALSE, OutputLowerLimit et OutputUpperLimit ne sont pas évaluées. Output_UP et Output_DN sont remises à 0 lorsque Actuator_H = TRUE ou Actuator_L = TRUE (si ActuatorEndStopOn = TRUE) ou après un temps de course de Config.VirtualActuatorLimit * Retain.TransitTime/100 (si ActuatorEndStopOn = FALSE).
Config.SetpointUpperLimit(1)	REAL	+3.402822e+38	Limite supérieure de la consigne Quand vous configurez SetpointUpperLimit en dehors des limites de la mesure, la limite supérieure absolue configurée pour la mesure est utilisée comme limite supérieure par défaut de la consigne. Quand vous configurez SetpointUpperLimit dans les limites de la mesure, cette valeur est utilisée comme limite supérieure de la consigne.

Variable	Type de données	Valeur par défaut	Description
Config.SetpointLowerLimit(1)	REAL	-3.402822e+38	Limite inférieure de la consigne Quand vous configurez SetpointLowerLimit en dehors des limites de la mesure, la limite inférieure absolue configurée pour la mesure sert de limite inférieure par défaut pour la consigne. Quand vous configurez SetpointLowerLimit dans les limites de la mesure, cette valeur est utilisée comme limite inférieure de la consigne.
Config.MinimumOnTime(1)	REAL	0.0	Plus petit temps ON Temps minimum en secondes pendant lequel le servomoteur doit être en marche. Config.MinimumOnTime n'est opérant que si Output_UP et Output_DN sont utilisés (Config.OutputPerOn = FALSE).
Config.MinimumOffTime(1)	REAL	0.0	Plus petit temps OFF Temps minimum en secondes pendant lequel le servomoteur doit être arrêté. Config.MinimumOffTime n'est opérant que si Output_UP et Output_DN sont utilisés (Config.OutputPerOn = FALSE).
Config.VirtualActuatorLimit(1)	REAL	150.0	Si toutes les conditions suivantes sont remplies, l'actionneur est déplacé dans une direction pendant la durée de VirtualActuatorLimit × Retain.TransitTime/100 maximum et l'alerte 2000h est émise : • Config.OutputPerOn = FALSE • Config.ActuatorEndStopOn = FALSE • Config.FeedbackOn = FALSE Si Config.OutputPerOn = FALSE et Config.ActuatorEndStopOn = TRUE ou Config.FeedbackOn = TRUE, l'alerte 2000h est émise. Si Config.OutputPerOn = TRUE, VirtualActuatorLimit n'est pas pris en compte. À partir de PID_3Step version 2.3, il est possible de désactiver la surveillance et la limitation du temps de course avec Config.VirtualActuatorLimit = 0.0.
Config.InputScaling.UpperPointIn(1)	REAL	27648.0	Mise à l'échelle Input_PER Haut Input_PER est converti en pourcentage à l'aide des deux couples de valeurs UpperPointOut, UpperPointIn et LowerPointOut, LowerPointIn de la structure InputScaling.
Config.InputScaling.LowerPointIn(1)	REAL	0.0	Mise à l'échelle Input_PER Bas Input_PER est converti en pourcentage à l'aide des deux couples de valeurs UpperPointOut, UpperPointIn et LowerPointOut, LowerPointIn de la structure InputScaling.
Config.InputScaling.UpperPointOut(1)	REAL	100.0	Mesure supérieure à l'échelle Input_PER est converti en pourcentage à l'aide des deux couples de valeurs UpperPointOut, UpperPointIn et LowerPointOut, LowerPointIn de la structure InputScaling.
Config.InputScaling.LowerPointOut(1)	REAL	0.0	Mesure inférieure à l'échelle Input_PER est converti en pourcentage à l'aide des deux couples de valeurs UpperPointOut, UpperPointIn et LowerPointOut, LowerPointIn de la structure InputScaling.
Config.FeedbackScaling.UpperPointIn(1)	REAL	27648.0	Mise à l'échelle Feedback_PER Haut Feedback_PER est converti en pourcentage à l'aide des deux couples de valeurs UpperPointOut, UpperPointIn et LowerPointOut, LowerPointIn de la structure FeedbackScaling.

Variable	Type de données	Valeur par défaut	Description
Config.FeedbackScaling.Lower-PointIn(1)	REAL	0.0	Mise à l'échelle Feedback_PER Bas Feedback_PER est converti en pourcentage à l'aide des deux couples de valeurs UpperPointOut, UpperPointIn et LowerPointOut, LowerPointIn de la structure FeedbackScaling.
Config.FeedbackScaling.Upper-PointOut(1)	REAL	100.0	Butée supérieure Feedback_PER est converti en pourcentage à l'aide des deux couples de valeurs UpperPointOut, UpperPointIn et LowerPointOut, LowerPointIn de la structure FeedbackScaling. La plage de valeurs admissible dépend de la configuration. • FeedbackOn = FALSE : UpperPointOut = 100.0 • FeedbackOn = TRUE : UpperPointOut = 100.0 ou 0.0 UpperPointOut ≠ LowerPointOut
Config.FeedbackScaling.Lower-PointOut(1)	REAL	0.0	Butée inférieure Feedback_PER est converti en pourcentage à l'aide des deux couples de valeurs UpperPointOut, UpperPointIn et LowerPointOut, LowerPointIn de la structure FeedbackScaling. La plage de valeurs admissible dépend de la configuration. • FeedbackOn = FALSE : LowerPointOut = 0.0 • FeedbackOn = TRUE : LowerPointOut = 0.0 ou -100.0 LowerPointOut ≠ UpperPointOut
GetTransitTime.InvertDirection	BOOL	FALSE	Quand InvertDirection = FALSE, la vanne est entièrement ouverte, fermée, puis réouverte pour déterminer le temps de positionnement. Quand InvertDirection = TRUE, la vanne est entièrement fermée, ouverte, puis refermée.
GetTransitTime.SelectFeedback	BOOL	FALSE	Quand SelectFeedback = TRUE, Feedback_PER ou Feedback est pris en compte lors de la mesure du temps de positionnement. Quand SelectFeedback = FALSE, Actuator_H et Actuator_L sont pris en compte lors de la mesure du temps de positionnement.
GetTransitTime.State	INT	0	Phase actuelle de la mesure du temps de positionnement • State = 0 : Inactif • State = 1 : Ouvrir complètement la vanne • State = 2 : Fermer complètement la vanne • State = 3 : régler la vanne sur la position cible (NewOutput) • State = 4 : Mesure du temps de positionnement terminée avec succès • State = 5 : Mesure du temps de positionnement annulée
GetTransitTime.NewOutput	REAL	0.0	Position cible pour la mesure de temps de positionnement avec signalisation de position La position cible doit se trouver dans les limites de la "Butée supérieure" et de la "Butée inférieure". La différence entre NewOutput et ScaledFeedback doit être au moins égale à 50 % de la plage de réglage admissible.
CycleTime.StartEstimation	BOOL	TRUE	Si StartEstimation = TRUE, la mesure du temps d'échantillonnage PID_3Step est lancée. Après la fin de la mesure, on a CycleTime.StartEstimation = FALSE.

Variable	Type de données	Valeur par défaut	Description
CycleTime.EnEstimation	BOOL	TRUE	Si EnEstimation = TRUE, le temps d'échantillonnage PID_3Step est calculé. Si CycleTime.EnEstimation = FALSE, la période d'échantillonnage PID_3Step n'est pas calculée et vous devez configurer CycleTime.Value correctement manuellement.
CycleTime.EnMonitoring	BOOL	TRUE	Si EnMonitoring = TRUE, le temps d'échantillonnage PID_3Step est surveillé. Si PID_3Step ne peut pas être exécuté dans la période d'échantillonnage, l'erreur 0800h est signalée et le mode de fonctionnement change. Le mode de fonctionnement vers lequel le système passe dépend de ActivateRecoverMode et ErrorBehaviour. Si EnMonitoring = FALSE, la période d'échantillonnage PID_3Step n'est pas surveillée, l'erreur 0800h n'est pas signalée et le mode de fonctionnement ne change pas.
CycleTime.Value(1)	REAL	0.1	Période d'échantillonnage PID_3Step en secondes CycleTime.Value est automatiquement déterminée et correspond normalement au temps de cycle de l'OB appelant.
CtrlParamsBackUp.SetByUser	BOOL	FALSE	Valeur enregistrée de Retain.CtrlParams.SetByUser Il est possible de recharger les valeurs de la structure CtrlParamsBackUp avec LoadBackUp = TRUE.
CtrlParamsBackUp.Gain	REAL	1.0	Gain proportionnel enregistré
CtrlParamsBackUp.Ti	REAL	20.0	Période d'intégration enregistrée en secondes
CtrlParamsBackUp.Td	REAL	0.0	Temps de dérivation enregistré en secondes
CtrlParamsBackUp.TdFiltRatio	REAL	0.2	Coefficient enregistré pour l'action par dérivation
CtrlParamsBackUp.PWeighting	REAL	1.0	Pondération de l'action P enregistrée
CtrlParamsBackUp.DWeighting	REAL	1.0	Pondération de l'action D enregistrée
CtrlParamsBackUp.Cycle	REAL	1.0	Période d'échantillonnage de l'algorithme PID enregistrée en secondes
CtrlParamsBackUp.InputDeadBand	REAL	0.0	Largeur enregistrée de la zone morte du signal d'écart
PIDSelfTune.SUT.CalculateParams	BOOL	FALSE	Les propriétés du système réglé sont enregistrées lors de l'optimisation. Si CalculateParams = TRUE, les paramètres PID sont recalculés à partir de ces propriétés. Les paramètres PID sont calculés selon la méthode paramétrée dans TuneRule. CalculateParams est mis sur FALSE après le calcul.
PIDSelfTune.SUT.TuneRule	INT	1	Calculer les paramètres pendant l'optimisation préalable selon la méthode : • SUT.TuneRule = 0 : PID rapide I (comportement de régulation plus rapide avec des amplitudes plus fortes de la valeur de réglage qu'avec SUT.TuneRule =1) • SUT.TuneRule = 1 : PID lent I (comportement de régulation plus lent avec des amplitudes plus faibles de la valeur de réglage qu'avec SUT.TuneRule = 0) • SUT.TuneRule = 2 : PID Chien, Hrones, Reswick • SUT.TuneRule = 3 : PI Chien, Hrones, Reswick • SUT.TuneRule = 4 : PID rapide II (comportement de régulation plus rapide avec des amplitudes plus fortes de la valeur de réglage qu'avec SUT.TuneRule = 5) • SUT.TuneRule = 5 : PID lent II (comportement de régulation plus lent avec des amplitudes plus faibles de la valeur de réglage qu'avec SUT.TuneRule = 4)

Variable	Type de données	Valeur par défaut	Description
			L'unique différence entre les méthodes SUT.TuneRule = 0 et 1 consiste dans le calcul du gain proportionnel par les méthodes SUT.TuneRule = 4 et 5 : Pour SUT.TuneRule = 0 et 1, le gain proportionnel est calculé à l'aide du temps de compensation du processus. Pour SUT.TuneRule = 4 et 5, ce calcul s'effectue à l'aide du temps de retard du processus. SUT.TuneRule = 4 et 5 donne une valeur plus élevée pour le gain proportionnel et ainsi un comportement de régulation plus rapide avec des amplitudes plus fortes de la valeur de réglage qu'avec SUT.TuneRule = 0 et 1.
PIDSelfTune.SUT.State	INT	0	La variable SUT.State indique la phase actuelle de l'optimisation préalable : • State = 0 : Initialiser l'optimisation préalable • State = 50 : Déterminer la position initiale sans signalisation de position • State = 100 : Calculer l'écart type • State = 200 : Déterminer le point d'inflexion • State = 300 : Déterminer le temps de montée • State = 9900 : Optimisation préalable réussie • State = 1 : Optimisation préalable échouée
PIDSelfTune.TIR.RunIn	BOOL	FALSE	La variable RunIn permet de définir l'exécution d'une optimisation fine aussi sans optimisation préalable. • RunIn = FALSE Si l'optimisation fine est démarrée depuis le mode inactif ou manuel, une optimisation préalable est lancée. Si l'optimisation fine est démarrée depuis le mode automatique, les paramètres PID existants sont utilisés pour un réglage sur la consigne. C'est seulement après cela que l'optimisation fine commence. Si l'optimisation préalable n'est pas possible, PID_3Step passe dans le mode de fonctionnement à partir duquel l'optimisation a été lancée. • RunIn = TRUE L'optimisation préalable est sautée. PID_3Step essaie d'atteindre la consigne avec la valeur de réglage mini ou maxi. Cela peut entraîner une suroscillation élevée. C'est seulement après cela que l'optimisation fine commence. RunIn est mis sur FALSE après l'optimisation fine.
PIDSelfTune.TIR.CalculateParams	BOOL	FALSE	Les propriétés du système réglé sont enregistrées lors de l'optimisation. Si CalculateParams = TRUE, les paramètres PID sont recalculés à partir de ces propriétés. Les paramètres PID sont calculés selon la méthode paramétrée dans TuneRule. CalculateParams est mis sur FALSE après le calcul.
PIDSelfTune.TIR.TuneRule	INT	0	Calculer les paramètres pendant l'optimisation fine selon la méthode :

Variable	Type de données	Valeur par défaut	Description
			• TIR.TuneRule = 0 : PID automatique • TIR.TuneRule = 1 : PID rapide (comportement de régulation plus rapide avec des amplitudes plus fortes de la valeur de réglage qu'avec TIR.TuneRule = 2) • TIR.TuneRule = 2 : PID lent (comportement de régulation plus lent avec des amplitudes plus faibles de la valeur de réglage qu'avec TIR.TuneRule = 1) • TIR.TuneRule = 3 : PID Ziegler-Nichols • TIR.TuneRule = 4 : PI Ziegler-Nichols • TIR.TuneRule = 5 : P Ziegler-Nichols Pour pouvoir répéter le calcul des paramètres PID avec TIR.CalculateParams et TIR.TuneRule = 0, 1 ou 2, il faut avoir exécuté l'optimisation fine précédente également avec TIR.TuneRule = 0, 1 ou 2. Si ce n'est pas le cas, TIR.TuneRule = 3 sera utilisé. Il est toujours possible de recalculer les paramètres PID avec TIR.CalculateParams et TIR.TuneRule = 3, 4 ou 5.
PIDSelfTune.TIR.State	INT	0	La variable TIR.State indique la phase actuelle de l' "optimisation fine" : • State = -100 L'optimisation fine n'est pas possible. Une optimisation préalable est d'abord réalisée. • State = 0 : Initialiser l'optimisation fine • State = 200 : Calculer l'écart type • State = 300 : Essayer d'atteindre la consigne avec la valeur de réglage maxi ou mini • State = 400 : Essayer d'atteindre la consigne avec les paramètres PID existants (si optimisation préalable réussie) • State = 500 : Déterminer l'oscillation et calculer les paramètres • State = 9900 : Optimisation fine réussie • State = 1 : Optimisation fine échouée
Retain.TransitTime(1)	REAL	30.0	Temps de positionnement du moteur en secondes Temps en secondes nécessaire pour le servomoteur pour faire passe la vanne de l'état fermé à l'état ouvert. TransitTime est rémanent.
Retain.CtrlParams.SetByUser(1)	BOOL	FALSE	Si SetByUser = FALSE, les paramètres PID sont déterminés automatiquement et PID_3Step travaille à la valeur de réglage avec une zone morte. La largeur de zone morte est calculée pendant l'optimisation à l'aide de l'écart type de la valeur de réglage et enregistrée dans Retain.CtrlParams.OutputDeadBand. Si SetByUser = TRUE, les paramètres PID sont entrés manuellement et PID_3 Step fonctionne avec la valeur de réglage sans zone morte. Retain.CtrlParams.OutputDeadBand = 0.0 SetByUser est rémanent.
Retain.CtrlParams.Gain(1)	REAL	1.0	Gain proportionnel actif Utilisez la variable Config.InvertControl pour inverser le sens de régulation. Des valeurs négatives au Gain inversent également le sens de régulation. Il est recommandé de régler le sens de régulation uniquement via InvertControl. Si InvertControl = TRUE et Gain < 0.0, le sens de régulation est aussi inversé. Gain est rémanent.

Variable	Type de données	Valeur par défaut	Description
Retain.CtrlParams.Ti(1)	REAL	20.0	• Ti > 0.0 : Temps d'intégration actif en secondes • Ti = 0.0 : L'action I est désactivée Ti est rémanent.
Retain.CtrlParams.Td(1)	REAL	0.0	• Td > 0.0 : Temps de dérivation actif en secondes • Td = 0.0 : L'action D est désactivée Td est rémanent.
Retain.CtrlParams.TdFiltRatio(1)	REAL	0.2	Coefficient actif pour l'action par dérivation L'effet de l'action D est retardé par le coefficient de l'action par dérivation. Action par dérivation = Temps de dérivation x Coefficient de l'action par dérivation • 0.0 : L'action D n'est active que pour un seul cycle et est donc quasiment inactive. • 0.5 : Cette valeur a fait ses preuves dans la pratique pour les systèmes réglés avec **une** constante de temps dominante. • > 1.0 : Plus le coefficient est grand, plus l'effet de l'action D est retardé. TdFiltRatio est rémanent.
Retain.CtrlParams.PWeighting(1)	REAL	1.0	Pondération active de l'action P En cas de modification de consigne, vous pouvez réduire l'action P. Les valeurs judicieuses sont comprises entre 0.0 et 1.0. • 1.0 : Action P totalement opérante si modification de la consigne • 0.0 : Action P non opérante si modification de la consigne En cas de modification de la mesure, l'action P est toujours totalement opérante. PWeighting est rémanent.
Retain.CtrlParams.DWeighting(1)	REAL	1.0	Pondération active de l'action D En cas de modification de consigne, vous pouvez réduire l'action D. Les valeurs judicieuses sont comprises entre 0.0 et 1.0. • 1.0 : En cas de modification de la consigne, l'action D est totalement opérante • 0.0 : En cas de modification de la consigne, l'action D n'est pas opérante En cas de variation de la mesure, l'action D est toujours totalement opérante. DWeighting est rémanent.
Retain.CtrlParams.Cycle(1)	REAL	1.0	Période d'échantillonnage active de l'algorithme PID en secondes, arrondi à un multiple entier supérieur du temps de cycle de l'OB appelant. Cycle est rémanent.
Retain.CtrlParams.InputDeadBand(1)	REAL	0.0	Largeur de zone morte du signal d'écart InputDeadBand est rémanent.

Voir aussi

Paramètres State et Mode V2 (Page 321)
Variable ActivateRecoverMode V2 (Page 328)
Charger des objets technologiques dans l'appareil (Page 48)

10.2.4.8 Paramètres State et Mode V2

Corrélation entre les paramètres

Le paramètre State affiche le mode de fonctionnement actuel du régulateur PID. Vous ne pouvez pas modifier le paramètre State.
Avec un front montant à ModeActivate, PID_3Step passe en mode de fonctionnement enregistré au paramètre d'entrée/sortie Mode.
Quand la CPU est mise en route ou passe de STOP à RUN, PID_3Step démarre dans le mode de fonctionnement enregistré à Mode. Pour laisser PID_3Step en mode "Inactif", mettez RunModeByStartup = FALSE.

Signification des valeurs

State	Description du mode de fonctionnement
0	Inactif Le régulateur est désactivé et ne modifie plus la position de la vanne. Le passage du mode de fonctionnement inactif au mode automatique s'effectue sans à-coups.
1	Optimisation préalable L'optimisation préalable détermine la réponse du processus à une impulsion de la valeur de sortie et recherche le point d'inflexion. Les paramètres PID sont calculés à partir de l'incrémentation maximale et du temps mort du système réglé. Les meilleurs paramètres PID sont obtenus pendant l'exécution d'une optimisation préalable et d'une optimisation fine. Conditions pour une optimisation préalable : • Le temps de positionnement du moteur est configuré ou mesuré. • Mode inactif (State = 0), manuel (State = 4) ou automatique (State = 3) • ManualEnable = FALSE • Reset = FALSE • La consigne et la mesure se trouvent dans les limites configurées. Plus la mesure est stable, plus il sera facile de déterminer des paramètres PID précis. Un bruit de la mesure est acceptable tant que la croissance de la mesure est nettement supérieure au bruit. Cela est plutôt le cas en mode de fonctionnement "Inactif" ou "Mode manuel". La consigne est gelée dans la variable CurrentSetpoint. Une optimisation est interrompue si : • Setpoint > CurrentSetpoint + CancelTuningLevel ou • Setpoint < CurrentSetpoint - CancelTuningLevel Avant que les paramètres PID soient recalculés, ils sont sauvegardés et peuvent être réactivés avec LoadBackUp. Après une optimisation préalable réussie, le régulateur passe en automatique ; si l'optimisation préalable échoue, le mode de fonctionnement change de mode en fonction de ActivateRecoverMode et de ErrorBehaviour. La phase d'optimisation préalable est indiquée par PIDSelfTune.SUT.State.

State	Description du mode de fonctionnement
2	Optimisation fine L'optimisation fine génère une oscillation constante limitée de la mesure. Les paramètres PID sont recalculés à partir de l'amplitude et de la fréquence de cette oscillation. Les paramètres PID de l'optimisation fine montrent généralement un meilleur comportement de référence et de perturbation que les paramètres PID de l'optimisation préalable. Les meilleurs paramètres PID sont obtenus pendant l'exécution d'une optimisation préalable et d'une optimisation fine. PID_3Step essaie automatiquement de créer une oscillation supérieure au bruit de la mesure. La stabilité de la mesure n'influence l'optimisation fine que de manière insignifiante. La consigne est gelée dans la variable CurrentSetpoint. Une optimisation est interrompue si : • Setpoint > CurrentSetpoint + CancelTuningLevel ou • Setpoint < CurrentSetpoint - CancelTuningLevel Les paramètres PID sont sauvegardés avant une optimisation fine. Ils peuvent être réactivés avec LoadBackUp. Conditions pour une optimisation fine : • Le temps de positionnement du moteur est configuré ou mesuré. • La consigne et la mesure se trouvent dans les limites configurées. • ManualEnable = FALSE • Reset = FALSE • Mode automatique (State = 3), inactif (State = 0) ou mode manuel (State = 4) L'optimisation fine se déroule de la manière suivante au démarrage : • Mode automatique (State = 3) Si vous souhaitez améliorer les paramètres PID existants à l'aide de l'optimisation, démarrez l'optimisation fine à partir du mode automatique. PID_3Step effectue un réglage avec les paramètres PID existants jusqu'à ce que la boucle de régulation soit en régime stationnaire et que les conditions pour une optimisation fine soient remplies. C'est seulement après cela que l'optimisation fine commence. • Inactif (State = 0) ou mode manuel (State = 4) Une optimisation préalable est lancée lorsque les conditions correspondantes sont réunies. Une régulation est effectuée avec les paramètres PID déterminés jusqu'à ce que la boucle de régulation soit en régime établi et que les conditions pour une optimisation fine soient remplies. Quand PIDSelfTune.TIR.RunIn = TRUE, l'optimisation préalable est sautée et le système essaie d'atteindre la consigne avec la valeur de sortie mini ou maxi. Cela peut entraîner une suroscillation élevée. L'optimisation fine démarre alors automatiquement. Après une optimisation fine réussie, le mode de fonctionnement passe en automatique ; si l'optimisation fine échoue, le mode de fonctionnement change de mode en fonction de ActivateRecoverMode et ErrorBehaviour. La phase d'optimisation fine est affichée avec PIDSelfTune.TIR.State.
3	Mode automatique En mode automatique, PID_3Step régule le système réglé en fonction des paramètres prédéfinis. Si l'une des conditions préalables suivantes est remplie, le système passe au mode automatique : • Optimisation préalable réussie • Optimisation fine réussie • Modification du paramètre d'entrée/sortie Mode à la valeur 3 et un front montant à ModeActivate. Le passage du mode automatique en mode manuel s'effectue sans à-coups uniquement dans l'éditeur de mise en service. Le mode automatique tient compte de la variable ActivateRecoverMode.
4	Mode manuel En mode manuel, vous spécifiez des valeurs de sortie manuelles aux paramètres Manual_UP et Manual_DN ou ManualValue. Le paramètre ErrorBits indique si l'actionneur peut être amené sur la valeur de sortie en cas d'erreur. Ce mode est également activable via ManualEnable = TRUE. Il est recommandé de changer de mode de fonctionnement uniquement via Mode et ModeActivate. Le passage du mode manuel au mode automatique s'effectue sans à-coups. En cas d'erreur, le mode manuel est également possible.

State	Description du mode de fonctionnement
5	Accoster la valeur de sortie de remplacement Ce mode de fonctionnement est activé en cas d'erreur si Errorbehaviour = TRUE et ActivateRecoverMode = FALSE.. PID_3Step met l'actionneur à la valeur de sortie de remplacement et passe ensuite au mode de fonctionnement "Inactif".
6	Mesure du temps de positionnement Le système détermine le temps nécessaire au moteur pour ouvrir entièrement la vanne depuis l'état fermé. Ce mode de fonctionnement sera activé si Mode = 6 et ModeActivate = TRUE. Quand des signaux de butée sont utilisés pour la mesure du temps de positionnement, la vanne est complètement ouverte depuis la position actuelle, complètement fermée, puis de nouveau complètement ouverte. Quand GetTransitTime.InvertDirection = TRUE, ce comportement est inversé. Quand une signalisation de position est utilisée pour la mesure du temps de positionnement, l'actionneur est mis à une position cible à partir de la position actuelle. Les limites de valeur de sortie ne sont pas prises en compte lors de la mesure du temps de positionnement. Il est possible de déplacer l'actionneur jusqu'à la butée supérieure ou inférieure.
7	Surveillance des erreurs L'algorithme de régulation est désactivé et ne modifie plus la position de la vanne. Ce mode de fonctionnement est activé à la place du mode de fonctionnement "Inactif". Toutes les conditions suivantes doivent être remplies : • Mode automatique (Mode = 3) • Errorbehaviour = FALSE • ActivateRecoverMode = TRUE • Une ou plusieurs erreurs sont apparues pour lesquelles ActivateRecoverMode (Page 328) s'applique. Dès que les erreurs ont disparu, PID_3Step repasse en mode automatique.
8	Accoster la valeur de sortie de remplacement avec surveillance d'erreur Ce mode de fonctionnement est activé à la place du mode de fonctionnement "Accoster la valeur de réglage de remplacement". PID_3Step met l'actionneur à la valeur de réglage de remplacement et passe ensuite au mode de fonctionnement "Surveillance d'erreur". Toutes les conditions suivantes doivent être remplies : • Mode automatique (Mode = 3) • Errorbehaviour = TRUE • ActivateRecoverMode = TRUE • Une ou plusieurs erreurs sont apparues pour lesquelles ActivateRecoverMode (Page 328) s'applique. Dès que les erreurs ont disparu, PID_3Step repasse en mode automatique.
10	Mode manuel sans signaux de butée Les signaux de butée ne sont **pas** pris en compte, bien que Config.ActuatorEndStopOn = TRUE. Les limites de la valeur de sortie ne sont prises en compte. Sinon, PID_3Step se comporte comme en mode manuel.

Comportement ENO

Si State = 0, alors ENO = FALSE.
Si State ≠ 0, alors ENO = TRUE.

Changement de mode de fonctionnement automatique pendant la mise en route

Après une optimisation préalable ou fine réussie, le mode automatique est activé. Le tableau suivant indique comment Mode et State évoluent pendant une optimisation préalable réussie.

Numéro de cycle	Mode	State	Action
0	4	4	Mise à 1 de Mode = 1
1	1	4	Mise à 1 de ModeActivate = TRUE
1	4	1	La valeur de State est enregistrée dans Mode L'optimisation préalable est lancée
n	4	1	Optimisation préalable terminée avec succès
n	3	3	Le mode automatique est lancé

En cas d'erreur, PID_3Step change automatiquement de mode de fonctionnement. Le tableau suivant indique comment Mode et State évoluent pendant une optimisation préalable erronée.

Numéro de cycle	Mode	State	Action
0	4	4	Mise à 1 de Mode = 1
1	1	4	Mise à 1 de ModeActivate = TRUE
1	4	1	La valeur de State est enregistrée dans Mode L'optimisation préalable est lancée
n	4	1	Optimisation préalable interrompue
n	4	4	Le mode manuel est démarré

Si ActivateRecoverMode = TRUE, le mode de fonctionnement qui est enregistré dans Mode est activé. Au démarrage de la mesure du temps de positionnement, de l'optimisation préalable ou de l'optimisation fine, PID_3Step a enregistré la valeur de State au paramètre d'entrée/sortie Mode. PID_3Step passe donc dans le mode de fonctionnement à partir duquel l'optimisation a été lancée.

Si ActivateRecoverMode = FALSE, le mode de fonctionnement "Inactif" ou "Accoster la valeur de réglage de remplacement" est activé.

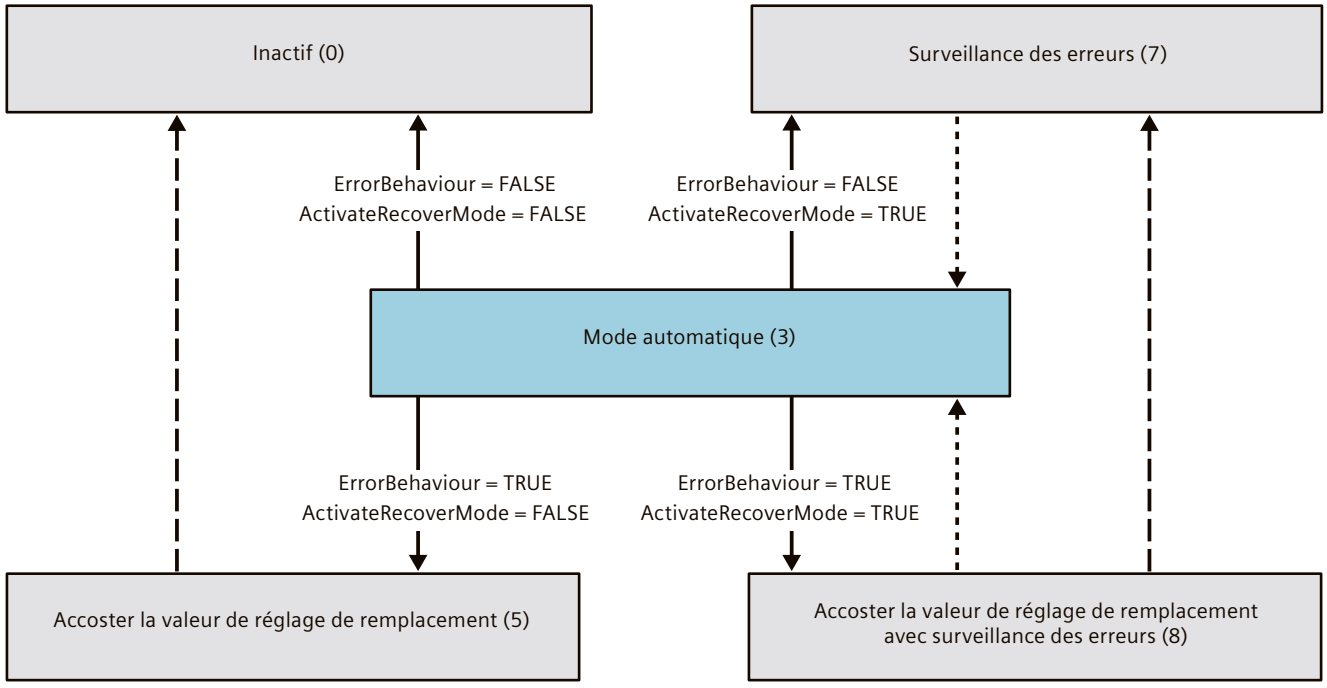
Changement automatique de mode de fonctionnement après la mesure du temps de positionnement

Si ActivateRecoverMode = TRUE, le mode de fonctionnement qui est enregistré dans Mode est activé une fois la mesure du temps de positionnement réussie.

Si ActivateRecoverMode = FALSE, le système passe en mode de fonctionnement "Inactif" une fois la mesure du temps de positionnement réussie.

Changement automatique de mode de fonctionnement en mode automatique

En cas d'erreur, PID_3Step change automatiquement de mode de fonctionnement. Le schéma suivant montre l'influence de ErrorBehaviour et ActivateRecoverMode sur ce changement de mode de fonctionnement.



Changement automatique de mode de fonctionnement en cas d'erreur

Changement automatique de mode de fonctionnement quand le fonctionnement actuel est terminé.

Changement automatique de mode de fonctionnement quand l'erreur n'est plus présente.

Voir aussi

Variable ActivateRecoverMode V2 (Page 328)

Paramètre ErrorBits V2 (Page 326)

10.2.4.9 Paramètre ErrorBits V2

En présence de plusieurs erreurs simultanées, les valeurs des ErrorBits s'affichent additionnées en binaire. L'affichage ErrorBits = 16#0000_0003, p. ex., indique la présence simultanée des erreurs 16#0000_0001 et 16#0000_0002.

En cas de signalisation en retour, PID_3Step utilise ManualValue comme valeur de réglage en mode manuel. Errorbits = 16#0001_0000 est l'exception.

ErrorBits (DW#16#...)	Description
0000_0000	Aucune erreur.
0000_0001	Le paramètre "Input" se trouve hors des limites de la mesure. • Input > Config.InputUpperLimit ou • Input < Config.InputLowerLimit Si le mode automatique était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_3Step reste en mode automatique. Si le mode Optimisation préalable, Optimisation fine ou Mesure du temps de positionnement était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_3Step passe dans le mode de fonctionnement enregistré dans Mode.
0000_0002	Valeur invalide au paramètre "Input_PER". Vérifiez si une erreur est présente à l'entrée analogique. Si le mode automatique était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_3Step passe en mode "Atteindre la valeur de réglage de remplacement avec surveillance des erreurs" ou "Surveillance des erreurs". Dès que l'erreur a disparu, PID_3Step repasse en mode automatique. Si le mode Optimisation préalable, Optimisation fine ou Mesure du temps de positionnement était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_3Step passe dans le mode de fonctionnement enregistré dans Mode.
0000_0004	Erreur pendant l'optimisation fine. L'oscillation des températures n'a pas pu être maintenue. Si avant l'apparition de l'erreur ActivateRecoverMode = TRUE, PID_3Step interrompt l'optimisation et passe dans le mode de fonctionnement enregistré dans Mode.
0000_0010	La consigne a été modifiée durant l'optimisation. La variation admissible de la consigne peut être réglée à la variable CancelTuningLevel. Si avant l'apparition de l'erreur ActivateRecoverMode = TRUE, PID_3Step interrompt l'optimisation et passe dans le mode de fonctionnement enregistré dans Mode.
0000_0020	L'optimisation préalable n'est pas autorisée pendant l'optimisation fine. Si avant l'apparition de l'erreur ActivateRecoverMode = TRUE, PID_3Step reste en mode de fonctionnement Optimisation fine.
0000_0080	Erreur lors de l'optimisation préalable. Les limites de la valeur de réglage ne sont pas configurées correctement ou la mesure ne réagit pas comme prévu. Vérifiez les points suivants : • Les limites de la valeur de réglage sont configurées correctement et conviennent au sens de la régulation. • Il est possible de modifier la valeur de réglage de sorte que la mesure se rapproche de la consigne. La valeur de réglage n'est pas limitée avant le démarrage de l'optimisation préalable par la limite de valeur de réglage correspondante et l'actionneur n'a pas encore atteint la butée correspondante. Exemple : Dans le sens normal de régulation et si la mesure est inférieure à la consigne, la valeur de réglage ne doit pas atteindre la limite supérieure et l'actionneur la butée supérieure avant le démarrage de l'optimisation préalable. • Il n'apparaît pas d'oscillations importantes de la mesure avant le démarrage de l'optimisation préalable. Si avant l'apparition de l'erreur ActivateRecoverMode = TRUE, PID_3Step interrompt l'optimisation et passe dans le mode de fonctionnement enregistré dans Mode.
0000_0100	Une erreur durant l'optimisation fine a entraîné des paramètres non valides. Si avant l'apparition de l'erreur ActivateRecoverMode = TRUE, PID_3Step interrompt l'optimisation et passe dans le mode de fonctionnement enregistré dans Mode.

ErrorBits (DW#16#...)	Description
0000_0200	Valeur invalide au paramètre "Input" : Le format numérique de la valeur est invalide. Si le mode automatique était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_3Step passe en mode "Atteindre la valeur de réglage de remplacement avec surveillance des erreurs" ou "Surveillance des erreurs". Dès que l'erreur a disparu, PID_3Step repasse en mode automatique. Si le mode Optimisation préalable, Optimisation fine ou Mesure du temps de positionnement était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_3Step passe dans le mode de fonctionnement enregistré dans Mode.
0000_0400	Le calcul de la valeur de réglage a échoué. Vérifiez les paramètres PID. Si le mode automatique était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_3Step passe en mode "Atteindre la valeur de réglage de remplacement avec surveillance des erreurs" ou "Surveillance des erreurs". Dès que l'erreur a disparu, PID_3Step repasse en mode automatique. Si le mode Optimisation préalable, Optimisation fine ou Mesure du temps de positionnement était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_3Step passe dans le mode de fonctionnement enregistré dans Mode.
0000_0800	Erreur de période d'échantillonnage : PID_3Step n'est pas appelé pendant la période d'échantillonnage de l'OB d'alarme cyclique. Il est recommandé d'appeler PID_3Step dans un OB d'alarme cyclique sans conditions et de l'activer ou le désactiver via le mode de fonctionnement au paramètre Mode. Les appels conditionnels ou l'appel dans l'OB1 peuvent avoir une influence négative sur la qualité de la régulation. La surveillance de la période d'échantillonnage peut être désactivée avec CycleTime.EnMonitoring = FALSE Si le mode automatique était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_3Step reste en mode automatique. Si le mode Optimisation préalable, Optimisation fine ou Mesure du temps de positionnement était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_3Step passe dans le mode de fonctionnement enregistré dans Mode. Si cette erreur s'est produite lors de la simulation avec PLCSIM, tenez compte des indications de la rubrique Simuler PID_3Step V2 avec PLCSIM (Page 136).
0000_1000	Valeur invalide au paramètre "Setpoint" : Le format numérique de la valeur est invalide. Si le mode automatique était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_3Step passe en mode "Atteindre la valeur de réglage de remplacement avec surveillance des erreurs" ou "Surveillance des erreurs". Dès que l'erreur a disparu, PID_3Step repasse en mode automatique. Si le mode Optimisation préalable, Optimisation fine ou Mesure du temps de positionnement était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_3Step passe dans le mode de fonctionnement enregistré dans Mode.
0000_2000	Valeur invalide au paramètre Feedback_PER. Vérifiez si une erreur est présente à l'entrée analogique. L'actionneur ne peut pas atteindre la valeur de réglage de remplacement et reste dans la position actuelle. En mode manuel, vous pouvez modifier la position de l'actionneur uniquement avec Manual_UP et Manual_DN, mais pas avec ManualValue. Si le mode automatique était activé avant l'apparition de l'erreur, que ActivateRecoverMode = TRUE et que l'erreur a disparu, PID_3Step repasse en mode automatique. Si le mode Optimisation préalable, Optimisation fine ou Mesure du temps de positionnement était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_3Step passe dans le mode de fonctionnement enregistré dans Mode.
0000_4000	Valeur invalide au paramètre Feedback. Le format numérique de la valeur est invalide. L'actionneur ne peut pas atteindre la valeur de réglage de remplacement et reste dans la position actuelle. En mode manuel, vous pouvez modifier la position de l'actionneur uniquement avec Manual_UP et Manual_DN, mais pas avec ManualValue. Si le mode automatique était activé avant l'apparition de l'erreur, que ActivateRecoverMode = TRUE et que l'erreur a disparu, PID_3Step repasse en mode automatique. Si le mode Optimisation préalable, Optimisation fine ou Mesure du temps de positionnement était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_3Step passe dans le mode de fonctionnement enregistré dans Mode.

ErrorBits (DW#16#...)	Description
0000_8000	Erreur de la signalisation de position numérique. Actuator_H = TRUE et Actuator_L = TRUE. L'actionneur ne peut pas atteindre la valeur de réglage de remplacement et reste dans la position actuelle. Le mode manuel n'est pas possible dans cet état. Pour pouvoir sortir l'actionneur de cet état, vous devez désactiver les "Signaux de butée actionneur" (Config.ActuatorEndStopOn = FALSE) ou passer en mode "Mode manuel sans signaux de butée" (Mode = 10). Si le mode automatique était activé avant l'apparition de l'erreur, que ActivateRecoverMode = TRUE et que l'erreur a disparu, PID_3Step repasse en mode automatique. Si le mode Optimisation préalable, Optimisation fine ou Mesure du temps de positionnement était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_3Step passe dans le mode de fonctionnement enregistré dans Mode.
0001_0000	Valeur invalide au paramètre ManualValue. Le format numérique de la valeur est invalide. L'actionneur ne peut pas atteindre la valeur de réglage de remplacement et reste dans la position actuelle. Saisissez une valeur valide pour ManualValue ou déplacez l'actionneur en mode manuel avec Manual_UP et Manual_DN.
0002_0000	Valeur invalide à la variable SavePosition. Le format numérique de la valeur est invalide. L'actionneur ne peut pas atteindre la valeur de réglage de remplacement et reste dans la position actuelle.
0004_0000	Valeur invalide au paramètre Disturbance. Le format numérique de la valeur est invalide. Si le mode automatique était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, Disturbance est remis à zéro. PID_3Step reste en mode automatique. Si le mode Optimisation préalable ou Optimisation fine était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_3Step passe dans le mode de fonctionnement enregistré dans Mode. Si Disturbance n'influe pas sur la valeur de réglage dans la phase actuelle, l'optimisation n'est pas interrompue. L'erreur n'a aucune influence pendant la mesure du temps de positionnement.

10.2.4.10 Variable ActivateRecoverMode V2

La variable ActivateRecoverMode détermine le comportement en cas d'erreur. Le paramètre Error indique si une erreur est actuellement présente. Quand l'erreur a disparu, Error = FALSE. Le paramètre ErrorBits indique les erreurs survenues.

IMPORTANT
Votre installation peut être endommagée. Si ActivateRecoverMode = TRUE, PID_3Step reste en mode automatique, y compris en cas de dépassement des limites de la mesure. Cela peut endommager votre installation. Configurez un comportement en cas d'erreur pour votre système réglé, qui protège votre installation de tout endommagement.

Mode automatique

ActivateRecover-Mode	Description
FALSE	En cas d'erreur, PID_3Step passe en mode "Inactif" ou "Accoster la valeur de réglage de remplacement". Le régulateur n'est activé que par un front descendant à Reset ou un front montant à ModeActivate.
TRUE	Si des erreurs apparaissent fréquemment en mode automatique, ce réglage détériore le comportement de régulation car PID_3Step alterne à chaque erreur entre la valeur de réglage calculée et la valeur de réglage de remplacement. Vérifiez alors le paramètre ErrorBits et éliminez la cause d'erreur. Quand l'une ou plusieurs des erreurs suivantes apparaissent, PID_3Step reste en mode automatique : • 0001h : Le paramètre "Input" se trouve en dehors des limites de la mesure. • 0800h : Erreur de temps d'échantillonnage • 40000h : Valeur invalide au paramètre Disturbance. Quand l'une ou plusieurs des erreurs suivantes apparaissent, PID_3Step passe en mode de fonctionnement "Accoster la valeur de réglage de remplacement avec surveillance d'erreur" ou "Surveillance d'erreur" : • 0002h : Valeur invalide au paramètre Input_PER. • 0200h : Valeur invalide au paramètre Input. • 0400h : Le calcul de la valeur de réglage a échoué. • 1000h : Valeur invalide au paramètre Setpoint. Quand l'une ou plusieurs des erreurs suivantes apparaissent, PID_3Step ne peut plus déplacer l'actionneur : • 2000h : Valeur invalide au paramètre Feedback_PER. • 4000h : Valeur invalide au paramètre Feedback. • 8000h : Erreur dans la signalisation de position TOR. • 20000h : Valeur invalide à la variable SavePosition. Le format numérique de la valeur est invalide. Ce comportement est indépendant de ErrorBehaviour. Dès que les erreurs ont disparu, PID_3Step repasse en mode automatique.

Optimisation préalable, optimisation fine et mesure du temps de positionnement

ActivateRecover-Mode	Description
FALSE	En cas d'erreur, PID_3Step passe en mode "Inactif" ou "Accoster la valeur de réglage de remplacement". Le régulateur n'est activé que par un front descendant à Reset ou un front montant à ModeActivate. Une fois la mesure du temps de positionnement réussie, le régulateur passe en mode Inactif.
TRUE	Si l'erreur suivante se produit, PID_3Step reste dans le mode actif. • 0020h : L'optimisation préalable n'est pas autorisée pendant l'optimisation fine. Les erreurs suivantes sont ignorées : • 10000h : Valeur invalide au paramètre ManualValue. • 20000h : Valeur invalide à la variable SavePosition. Pour toutes les autres erreurs, PID_3Step interrompt l'optimisation et passe dans le mode de fonctionnement à partir duquel l'optimisation a été lancée.

Mode manuel

En mode manuel, ActivateRecoverMode n'a aucun effet.

Voir aussi

Variables statiques PID_3Step V2 (Page 312)
Paramètres State et Mode V2 (Page 321)

10.2.4.11 Variable Warning V2

En présence simultanée de plusieurs alertes, les valeurs des alertes sont affichées sous forme d'addition binaire. Si l'avertissement affiché est p. ex. 16#0000_0005, cela indique la présence simultanée des avertissements 16#0000_0001 et 16#0000_0004.

Warning (DW#16#...)	Description
0000_0000	Aucune alerte n'est présente.
0000_0001	Le point d'inflexion n'a pas été trouvé pendant l'optimisation préalable.
0000_0004	La consigne a été limitée à des limites paramétrées.
0000_0008	Toutes les propriétés nécessaires du système réglé n'ont pas été déterminées pour la méthode de calcul choisie. Les paramètres PID ont été calculés avec la méthode TIR.TuneRule = 3 à titre de remplacement.
0000_0010	Impossible de modifier le mode de fonctionnement car Reset = TRUE ou ManualEnable = TRUE
0000_0020	Le temps d'échantillonnage de l'algorithme PID est limité par le temps de cycle de l'OB appelant. Afin d'obtenir de meilleurs résultats, utilisez des temps de cycle de l'OB plus courts.
0000_0040	La mesure a dépassé l'une de ses limites d'alerte.
0000_0080	Valeur invalide de Mode. Le changement de mode de fonctionnement n'est pas effectué.
0000_0100	La valeur manuelle a été limitée aux limites de la sortie du régulateur.
0000_0200	La règle mentionnée pour l'optimisation n'est pas prise en charge. Aucun paramètre PID n'est calculé.
0000_0400	Le temps de positionnement ne peut pas être mesuré, car les paramètres de l'actionneur ne correspondent pas à la méthode de mesure sélectionnée.
0000_0800	Lors de la mesure du temps de positionnement, la différence entre la position actuelle et la nouvelle valeur de réglage est trop petite. Ceci peut occasionner des résultats erronés. La différence entre la valeur de réglage actuelle et la nouvelle valeur de réglage doit au moins être égale à 50 % de la plage de réglage totale.
0000_1000	La valeur de réglage de remplacement ne peut pas être atteinte, car elle est en dehors des limites de la valeur de réglage.
0000_2000	L'actionneur est déplacé plus longtemps que Config.VirtualActuatorLimit × Retain.TransitTime dans un sens. Vérifiez si l'actionneur a atteint un signal de butée.

Les alarmes suivantes sont supprimées dès que la cause est éliminée :

- 16#0000_0001
- 16#0000_0004
- 16#0000_0008
- 16#0000_0040
- 16#0000_0100
- 16#0000_2000

Toutes les autres alertes sont supprimées avec un front montant à Reset ou ErrorAck.

10.2.5 PID_3Step V1

10.2.5.1 Description PID_3Step V1

Description

L'instruction PID_3Step permet de configurer un régulateur PID avec auto-optimisation pour les vannes ou actionneurs à comportement intégral.
Les modes suivants sont disponibles :

- Inactif
- Optimisation préalable
- Optimisation fine
- Mode automatique
- Mode manuel
- Accoster la valeur de réglage de remplacement
- Mesure du temps de positionnement
- Accoster la valeur de réglage de remplacement avec surveillance d'erreur
- Surveillance des erreurs

Les modes de fonctionnement sont décrits en détail dans le paramètre State.

Algorithme PID

PID_3Step est un régulateur PIDT1 avec anti-saturation et pondération de l'action P et D. La valeur de réglage est calculée avec la formule suivante :

$$\Delta y = K_p \cdot s \cdot \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

Δy	Valeur de réglage de l'algorithme PID
K_p	Gain proportionnel
s	Opérateur de Laplace
b	Pondération de l'action P
w	Consigne
x	Mesure
T_I	Temps d'intégration
a	Coefficient pour l'action par dérivation (T1 = a × T_D)
T_D	Temps de dérivation
c	Pondération de l'action D

Schéma fonctionnel sans signalisation de position

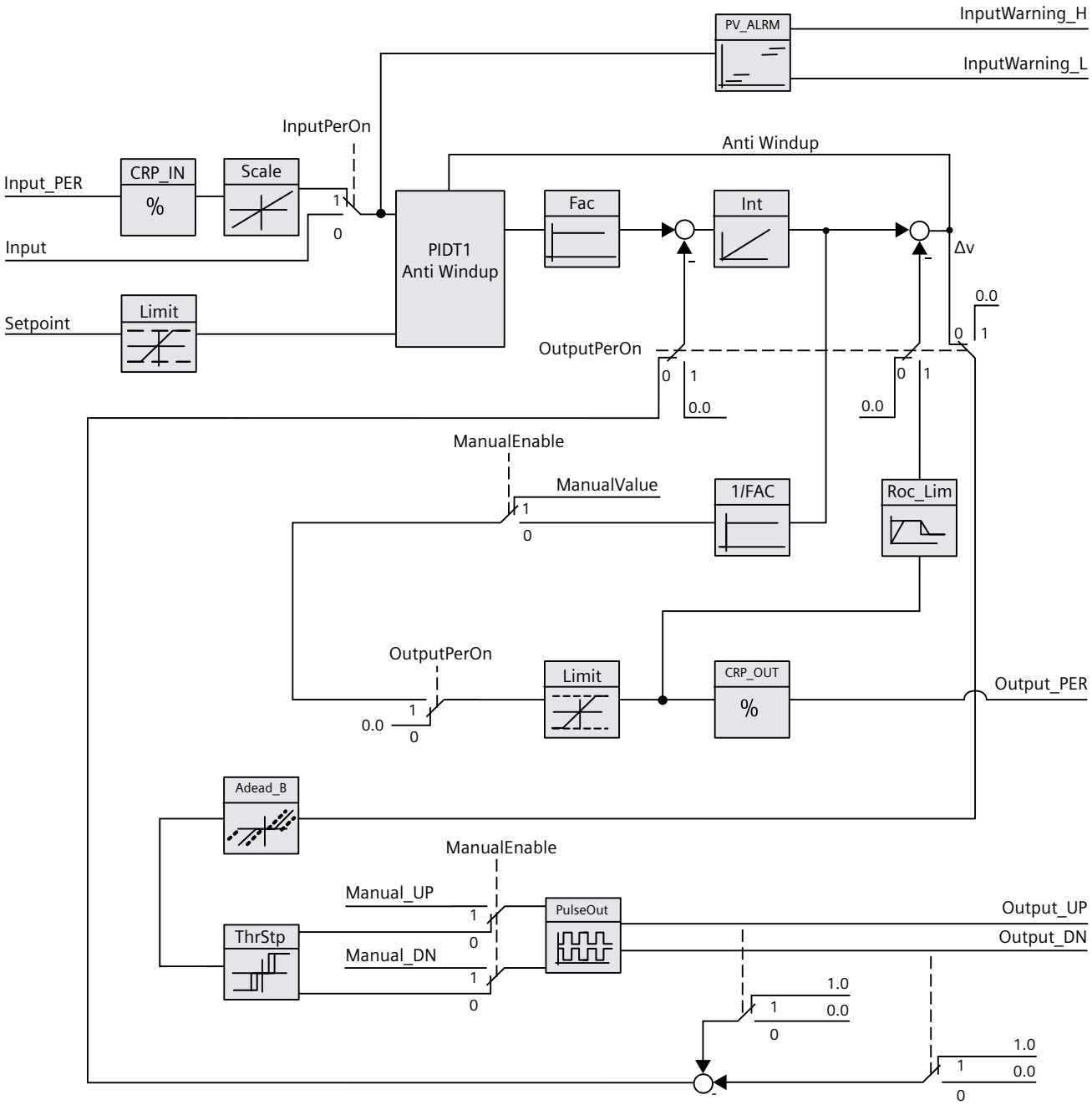


Schéma fonctionnel avec signalisation de position

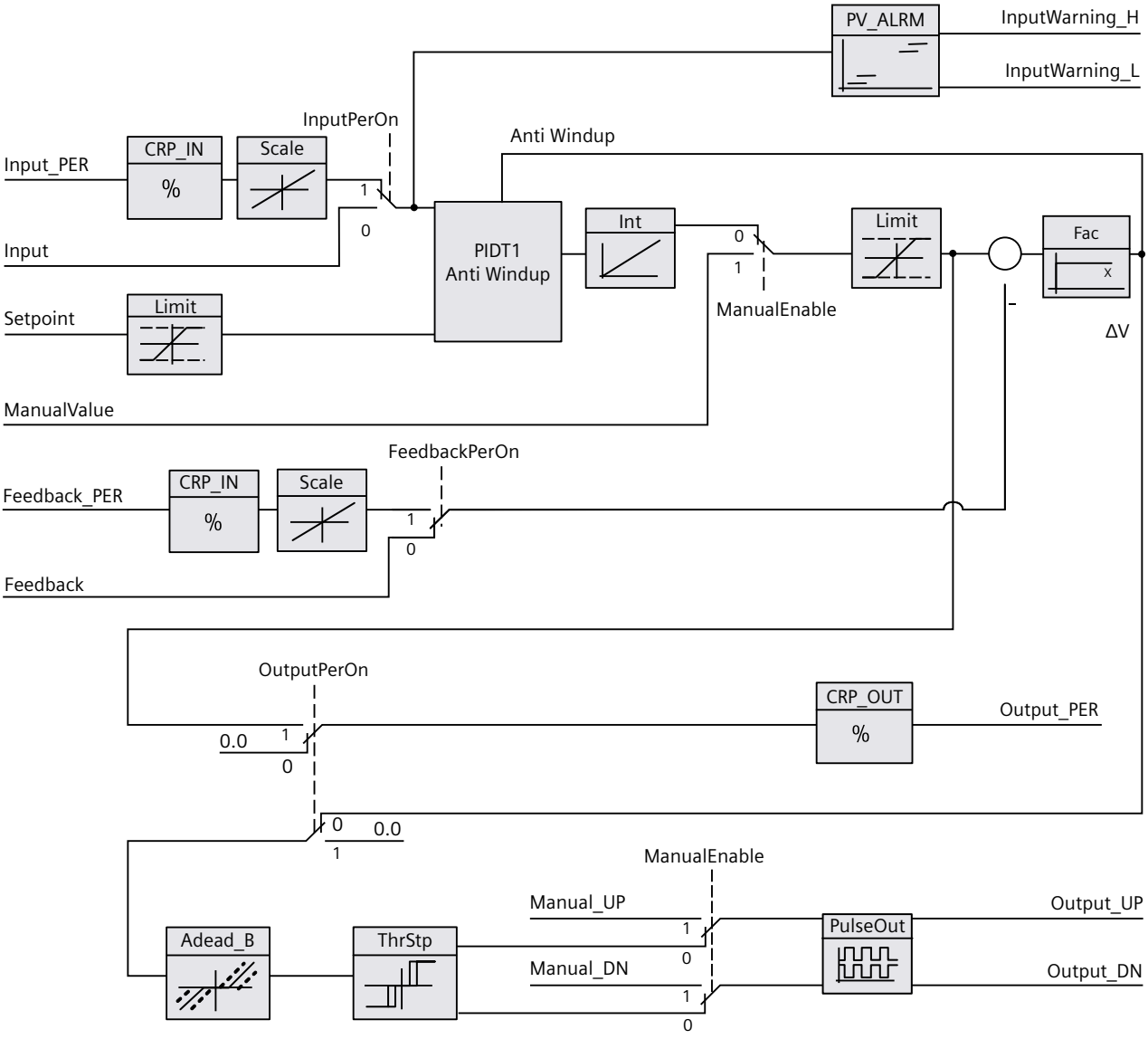
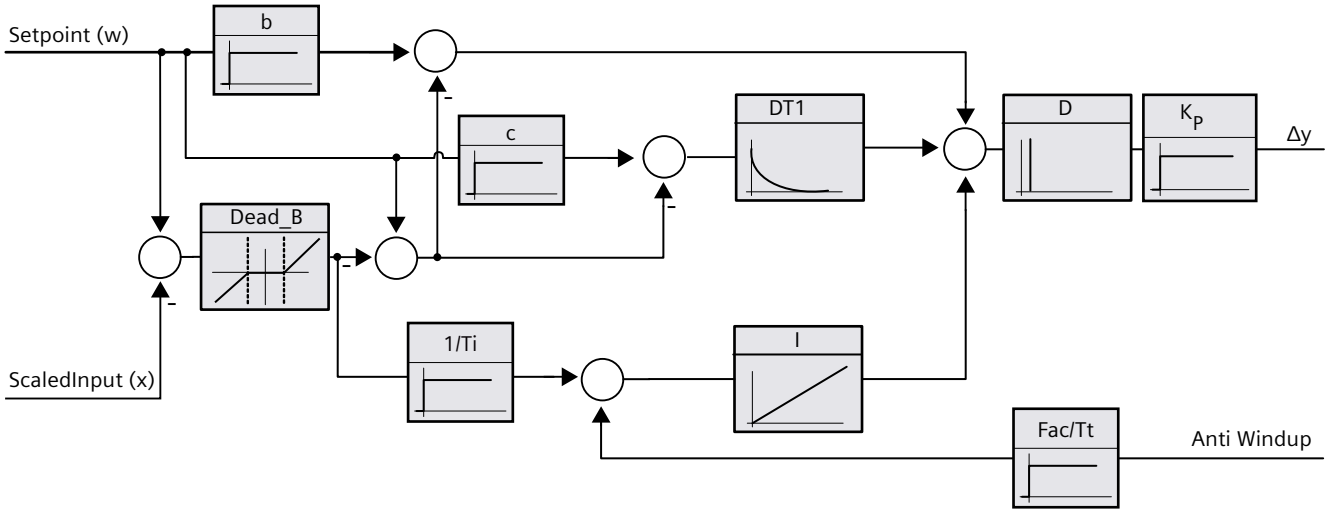


Schéma fonctionnel PIDT1 avec anti-saturation



Appel

L'appel de l'instruction PID_3Step s'effectue durant l'intervalle de temps constant du temps de cycle de l'OB appelant (de préférence dans un OB d'alarme cyclique).

Chargement dans l'appareil

Les valeurs effectives de variables rémanentes ne sont actualisées que si vous chargez entièrement PID_3Step.
Charger des objets technologiques dans l'appareil (Page 48)

Démarrage

PID_3Step est démarrée dans le dernier mode de fonctionnement actif lors du démarrage de la CPU. Pour laisser PID_3Step en mode "Inactif", mettez RunModeByStartup = FALSE.

Comportement en cas d'erreur

Si des erreurs surviennent, celles-ci sont affichées au niveau du paramètre Error. Vous configurez le comportement de PID_3Step via les variables ErrorBehaviour et ActivateRecoverMode.

ErrorBehaviour	ActivateRecoverMode	Configuration Paramètres de l'actionneur Régler Output sur	Comportement
0	FALSE	Valeur de réglage actuelle	Passage au mode de fonctionnement "Inactif" (Mode = 0)
0	TRUE	Valeur de réglage actuelle pour la durée de l'erreur	Passage au mode de fonctionnement "Surveillance d'erreur" (Mode = 7)
1	FALSE	Valeur de réglage de remplacement	Passage au mode de fonctionnement "Accoster la valeur de réglage de remplacement" (Mode = 5) Passage au mode de fonctionnement "Inactif" (Mode = 0)
1	TRUE	Valeur de réglage de remplacement pour la durée de l'erreur	Passage au mode de fonctionnement "Accoster la valeur de réglage de remplacement avec surveillance d'erreur" (Mode = 8) Passage au mode de fonctionnement "Surveillance d'erreur" (Mode = 7)

Le paramètre ErrorBits indique les erreurs survenues.

Voir aussi

Paramètres State et Retain.Mode V1 (Page 349)

Paramètre ErrorBits V1 (Page 356)

Configurer PID_3Step V1 (Page 137)

10.2.5.2 Mode de fonctionnement PID_3Step V1

Surveiller les limites de mesure

Vous définissez une limite supérieure et une limite inférieure de la mesure dans les variables Config.InputUpperLimit et Config.InputLowerLimit. Si la mesure se trouve en dehors de ces limites, un erreur se produit (ErrorBits = 0001hex).

Vous définissez une limite d'alerte supérieure et une limite d'alerte inférieure de la mesure dans les variables Config.InputUpperWarning et Config.InputLowerWarning. Si la mesure se trouve en dehors de ces limites d'alerte, une alerte survient (Warnings = 0040hex) et le paramètre de sortie InputWarning_H ou InputWarning_L passe à TRUE.

Limiter consigne

Vous définissez une limite supérieure et inférieure de la consigne dans les variables Config.SetpointUpperLimit et Config.SetpointLowerLimit. PID_3Step limite automatiquement la consigne aux limites de la mesure. Vous pouvez limiter la consigne à une plage inférieure. PID_3Step contrôle si cette plage se trouve dans les limites de la mesure. Si la consigne se trouve hors de ces limites, les limites inférieure et supérieure sont utilisées comme consigne et le paramètre de sortie SetpointLimit_H ou SetpointLimit_L passe à TRUE.

La consigne est limitée dans tous les modes de fonctionnement.

Limiter la valeur de réglage

Vous déterminez une limite supérieure et une limite inférieure de la valeur de réglage dans les variables Config.OutputUpperLimit et Config.OutputLowerLimit. Les limites de la valeur de réglage doivent se trouver entre la "butée inférieure" et la "butée supérieure".

- Butée supérieure : Config.FeedbackScaling.UpperPointOut
- Butée inférieure : Config.FeedbackScaling.LowerPointOut

Règle à appliquer :

UpperPointOut ≥ OutputUpperLimit > OutputLowerLimit ≥ LowerPointOut

Les valeurs valables de la "Butée supérieure" et de la "Butée inférieure" dépendent de :

- FeedbackOn
- FeedbackPerOn
- OutputPerOn

OutputPerOn	FeedbackOn	FeedbackPerOn	LowerPointOut	UpperPointOut
FALSE	FALSE	FALSE	non réglable (0,0 %)	non réglable (100,0 %)
FALSE	TRUE	FALSE	-100.0 % ou 0.0 %	0.0 % ou +100.0 %
FALSE	TRUE	TRUE	-100.0 % ou 0.0 %	0.0 % ou +100.0 %
TRUE	FALSE	FALSE	non réglable (100,0 %)	non réglable (100,0 %)
TRUE	TRUE	FALSE	-100.0 % ou 0.0 %	0.0 % ou +100.0 %
TRUE	TRUE	TRUE	-100.0 % ou 0.0 %	0.0 % ou +100.0 %

Quand OutputPerOn = FALSE et FeedbackOn = FALSE, vous ne pouvez pas limiter la valeur de réglage. Les sorties TOR sont remises à zéro soit avec Actuator_H = TRUE ou Actuator_L = TRUE, soit après un temps de course de 110 % du temps de positionnement du moteur.

La valeur de réglage est de 27648 pour 100 % et -27648 pour -100 %. PID_3Step doit fermer complètement la vanne. C'est pourquoi le zéro doit être contenu dans les limites de la valeur de réglage.

REMARQUE

Utilisation avec deux actionneurs ou plus

PID_3 Step ne convient pas pour l'utilisation avec deux actionneurs ou plus (p. ex. dans des applications de chauffage ou de refroidissement), car des actionneurs différents ont besoin de paramètres PID différents pour obtenir un bon comportement de régulation.

Valeur de réglage de remplacement

En cas d'erreur, PID_3Step peut fournir une valeur de réglage de remplacement et placer l'actionneur dans une position sûre, que vous spécifiez au niveau de la variable SavePosition. La valeur de réglage de remplacement doit être dans les limites de la valeur de réglage.

Surveiller la validité des signaux

La validité des valeurs des paramètres suivants est surveillée :

- Setpoint
- Input
- Input_PER
- Feedback
- Feedback_PER
- Output

Surveiller le temps d'échantillonnage PID_3Step

Le temps d'échantillonnage correspond idéalement au temps de cycle de l'OB appelant. L'instruction PID_3Step permet de mesurer l'intervalle de temps entre deux appels respectifs. Le résultat est le temps d'échantillonnage actuel. Lors de chaque changement du mode de fonctionnement et à la première mise en route, une moyenne est calculée à partir des 10 premiers temps d'échantillonnage. Si le temps d'échantillonnage actuel diverge trop de cette moyenne, une erreur survient (ErrorBits = 0800 hex).

Les conditions suivantes font passer PID_3Step en mode "Inactif" pendant l'optimisation :

- Nouvelle valeur moyenne >= 1,1 x ancienne valeur moyenne
- Nouvelle valeur moyenne <= 0,9 x ancienne valeur moyenne

En mode automatique, les conditions suivantes font passer PID_3Step en mode "Inactif" :

- Nouvelle valeur moyenne >= 1,5 x ancienne valeur moyenne
- Nouvelle valeur moyenne <= 0,5 x ancienne valeur moyenne

Temps d'échantillonnage de l'algorithme PID

Comme le système réglé nécessite un certain temps pour réagir à une modification de la valeur de réglage, il est judicieux de ne pas calculer cette valeur à chaque cycle. Le temps d'échantillonnage de l'algorithme PID est le temps entre deux calculs de valeurs de réglage. Il est déterminé pendant l'optimisation et arrondi à un multiple du temps de cycle. Toutes les autres fonctions de PID_3Step sont exécutées lors de chaque appel.

Mesurer le temps de positionnement du moteur

Le temps de positionnement du moteur est le temps en secondes requis par le moteur pour faire passer l'actionneur de l'état fermé à l'état ouvert. L'actionneur est déplacé dans un sens d'au maximum 110% du temps de positionnement du moteur. PID_3Step a besoin d'un temps de positionnement du moteur aussi exact que possible pour obtenir un bon résultat de régulation. Les indications dans la documentation de l'actionneur sont des valeurs moyennes pour ce type d'actionneur. La valeur peut être différente pour l'actionneur utilisé réellement. Vous pouvez mesurer le temps de positionnement du moteur pendant la mise en service. Les limites de valeur de réglage ne sont pas prises en compte lors de la mesure du temps de positionnement du moteur. Il est possible de déplacer l'actionneur jusqu'à la butée supérieure ou inférieure.

Le temps de positionnement du moteur est pris en compte dans le calcul de la valeur de réglage analogique ainsi que dans le calcul des valeurs de réglage TOR. Il est surtout nécessaire pour un fonctionnement correct lors de l'auto-optimisation et du comportement anti-windup. Configurez par conséquent le temps de positionnement du moteur à la valeur nécessaire au moteur pour faire passer l'actionneur de l'état fermé à l'état ouvert.

Si aucun temps de positionnement du moteur pertinent n'est actif (par ex. dans le cas d'électrovannes), si bien que la valeur de réglage est appliquée directement et entièrement au processus, utilisez à la place PID_Compact.

Sens de régulation

La plupart du temps, une augmentation de la mesure doit être atteinte avec une augmentation de la valeur de réglage. Dans ce cas, on parle d'un sens de régulation normal. Il peut être nécessaire d'inverser le sens de régulation pour les refroidissements et les régulations d'écoulement. PID_3Step ne fonctionne pas avec un gain proportionnel négatif. Si InvertControl = TRUE, un signal d'écart croissant provoque une diminution de la valeur de réglage. Le sens de régulation est pris en compte aussi pendant l'optimisation préalable et l'optimisation fine.

Voir aussi

Configurer PID_3Step V1 (Page 137)

10.2.5.3 Paramètres d'entrée PID_3Step V1

Tableau 10-10

Paramètre	Type de données	Valeur par défaut	Description
Setpoint	REAL	0.0	Consigne du régulateur PID en mode automatique
Input	REAL	0.0	Une variable du programme utilisateur est utilisée comme source de la mesure. Si vous utilisez le paramètre Input, il faut que Config.InputPerOn = FALSE.
Input_PER	WORD	W#16#0	Une entrée analogique est utilisée comme source de la mesure. Si vous utilisez le paramètre Input_PER, il faut que Config.InputPerOn = TRUE.
Actuator_H	BOOL	FALSE	Signalisation de position TOR de la vanne pour la butée supérieure Si Actuator_H = TRUE, la position de la vanne est à la butée supérieure et la vanne n'est plus déplacée dans cette direction.
Actuator_L	BOOL	FALSE	Signalisation de position TOR de la vanne pour la butée inférieure Si Actuator_L = TRUE, la position de la vanne est à la butée inférieure et la vanne n'est plus déplacée dans cette direction.
Feedback	REAL	0.0	Signalisation de position de la vanne Si vous utilisez le paramètre Feedback, il faut que Config.FeedbackPerOn = FALSE.
Feedback_PER	WORD	W#16#0	Signalisation de position analogique d'une vanne Si vous utilisez le paramètre Feedback_PER, il faut que Config.FeedbackPerOn = TRUE. Feedback_PER est mise à l'échelle avec les variables : • Config.FeedbackScaling.LowerPointIn • Config.FeedbackScaling.UpperPointIn • Config.FeedbackScaling.LowerPointOut • Config.FeedbackScaling.UpperPointOut
ManualEnable	BOOL	FALSE	• Le front FALSE -> TRUE sélectionne le mode de fonctionnement "Mode manuel", State = 4, Retain.Mode ne change pas. • Le front TRUE -> FALSE sélectionne le dernier mode de fonctionnement actif Pendant ManualEnable = TRUE, une modification de Retain.Mode n'a pas d'effet. La modification de Retain.Mode est prise en compte seulement au front TRUE -> FALSE à ManualEnable . **PID_3Step V1.1** Lorsque ManualEnable = TRUE au démarrage de la CPU, PID_3Step démarre en mode manuel. Un front montant (FALSE > TRUE) de ManualEnable n'est pas nécessaire. **PID_3Step V1.0** PID_3Step ne passe en mode manuel au démarrage de la CPU que s'il y a un front montant (FALSE->TRUE) de ManualEnable . En l'absence de ce front montant, PID_3Step démarre dans le dernier mode pour lequel ManualEnable était FALSE.
ManualValue	REAL	0.0	En mode manuel, la position absolue de la vanne est spécifiée. ManualValue n'est exploité que si vous utilisez OutputPer **ou** qu'une signalisation de position est disponible.

Paramètre	Type de données	Valeur par défaut	Description
Manual_UP	BOOL	FALSE	En mode manuel, chaque front montant ouvre la vanne de 5 % de la plage de réglage totale ou pour le temps de positionnement minimum du moteur. Manual_UP n'est exploité que si vous n'utilisez pas Output_PER et qu'une signalisation de position n'est pas disponible.
Manual_DN	BOOL	FALSE	En mode manuel, chaque front montant ferme la vanne de 5 % de la plage de réglage totale ou pour le temps de positionnement minimum du moteur. Manual_DN n'est exploité que si vous n'utilisez pas Output_PER et qu'une signalisation de position n'est pas disponible.
Reset	BOOL	FALSE	Effectue un redémarrage du régulateur. • Front FALSE -> TRUE – Passage en mode de fonctionnement "Inactif" – ErrorBits et Warning sont remis à zéro – Les valeurs intermédiaires de la régulation sont réinitialisées (les paramètres PID sont conservés) • Front TRUE -> FALSE – Passage au dernier mode de fonctionnement actif – Si le mode automatique était actif précédemment, la commutation en mode automatique s'effectue sans à-coup.

10.2.5.4 Paramètres de sortie PID_3Step V1

Tableau 10-11

Paramètre	Type de données	Valeur par défaut	Description
ScaledInput	REAL	0.0	Mesure mise à l'échelle
ScaledFeedback	REAL	0.0	Signalisation de position à l'échelle Sur un actionneur sans signalisation de position, ScaledFeedback affiche la position de l'actionneur de manière très imprécise. Dans ce cas, ScaledFeedback ne peut être utilisé que pour une estimation grossière de la position réelle.
Output_UP	BOOL	FALSE	Valeur de réglage TOR pour ouvrir la vanne Si Config.OutputPerOn = FALSE, c'est le paramètre Output_UP qui est utilisé.
Output_DN	BOOL	FALSE	Valeur de réglage TOR pour fermer la vanne Si Config.OutputPerOn = FALSE, c'est le paramètre Output_DN qui est utilisé.
Output_PER	WORD	W#16#0	Valeur de réglage analogique Si Config.OutputPerOn = TRUE, c'est Output_PER qui est utilisé. Utilisez Output_PER si l'actionneur mis en œuvre est une vanne adressée via une sortie analogique et commandée à l'aide d'un signal continu, par exemple 0...10 V, 4...20 mA. La valeur dans Output_PER correspond à la position finale de la vanne, p. ex. Output_PER = 13824, lorsque la vanne doit s'ouvrir à 50 %.

Paramètre	Type de données	Valeur par défaut	Description
SetpointLimit_H	BOOL	FALSE	Quand SetpointLimit_H = TRUE, la limite supérieure absolue de la consigne est atteinte. Dans la CPU, la consigne est limitée à la limite supérieure absolue configurée pour la consigne. La limite supérieure par défaut de la consigne est la limite supérieure absolue configurée pour la mesure. Si vous affectez à Config.SetpointUpperLimit une valeur dans les limites de la mesure, cette valeur sera utilisée comme limite supérieure de la consigne.
SetpointLimit_L	BOOL	FALSE	Quand SetpointLimit_L = TRUE, la limite inférieure absolue de la consigne est atteinte. Dans la CPU, la consigne est limitée à la limite inférieure absolue configurée pour la consigne. La limite inférieure par défaut de la consigne est la limite inférieure absolue configurée pour la mesure. Si vous affectez à Config.SetpointLowerLimit une valeur dans les limites de la mesure, cette valeur sera utilisée comme limite inférieure de la consigne.
InputWarning_H	BOOL	FALSE	Si InputWarning_H = TRUE, la limite d'alerte supérieure de la mesure est atteinte ou dépassée.
InputWarning_L	BOOL	FALSE	Si InputWarning_L = TRUE, la limite d'alerte inférieure de la mesure est atteinte ou dépassée.
State	INT	0	Le paramètre State (Page 349) affiche le mode de fonctionnement actuel du régulateur PID. La variable Retain.Mode vous permet de modifier le mode de fonctionnement. • State = 0 : Inactif • State = 1 : Optimisation préalable • State = 2 : Optimisation fine • State = 3 : Mode automatique • State = 4 : Mode manuel • State = 5 : Accoster la valeur de réglage de remplacement • State = 6 : Mesure du temps de positionnement • State = 7 : Surveillance des erreurs • State = 8 : Accoster la valeur de réglage de remplacement avec surveillance d'erreur
Error	BOOL	FALSE	Si Error = TRUE, un message d'erreur au moins existe.
ErrorBits	DWORD	DW#16#0	Le paramètre ErrorBits (Page 356) affiche les messages d'erreur.

Voir aussi

Paramètres State et Retain.Mode V1 (Page 349)
Paramètre ErrorBits V1 (Page 356)

10.2.5.5 Variables statiques PID_3Step V1

REMARQUE

Les variables qui ne sont pas mentionnées ne doivent pas être modifiées. Elles ne sont utilisées qu'en interne.

Faites passer les variables identifiées par (1) seulement en mode de fonctionnement "Inactif" pour éviter un comportement erroné du régulateur PID. Vous forcez le mode de fonctionnement "Inactif" en mettant la variable "Retain.Mode" à "0".

Tableau 10-12

Variable	Type de données	Valeur par défaut	Description
ActivateRecoverMode	BOOL	TRUE	La variable ActivateRecoverMode (Page 358) détermine le comportement en cas d'erreur.
RunModeByStartup	BOOL	TRUE	Activer le dernier mode de fonctionnement après le démarrage de la CPU Si RunModeByStartup = TRUE, le régulateur repasse au dernier mode de fonctionnement après la mise en route de la CPU. Si RunModeByStartup = FALSE, le régulateur reste inactif après la mise en route de la CPU.
PhysicalUnit	INT	0	Unité physique de la mesure et de la consigne, par ex. °C ou °F.
PhysicalQuantity	INT	0	Grandeur physique de la mesure et de la consigne, par ex. température.
ErrorBehaviour	INT	0	Quand ErrorBehaviour = 0, la vanne reste en cas d'erreur sur la position actuelle et le régulateur passe directement en mode de fonctionnement "Inactif" ou "Surveillance d'erreur". Quand ErrorBehaviour = 1, l'actionneur accoste, en cas d'erreur, la valeur de réglage de remplacement et ce n'est qu'après que le système passe en mode "Inactif" ou "Surveillance d'erreur". Quand les erreurs suivantes apparaissent, il n'est plus possible d'amener la vanne à une valeur de réglage de remplacement configurée. • 2000h : Valeur invalide au paramètre Feedback_PER. • 4000h : Valeur invalide au paramètre Feedback. • 8000h : erreur dans la signalisation de position TOR.
Warning	DWORD	DW#16#0	La variable Warning (Page 349) affiche les alertes depuis la remise à 0 ou le dernier changement du mode de fonctionnement. Les alertes cycliques (par ex. alerte de mesure) sont affichées pendant toute la durée de la cause de l'alerte. Une fois que la cause a disparu, elles sont automatiquement supprimées. Les alertes non-cycliques (par exemple, point d'inflexion pas trouvé) sont conservées et sont supprimées comme des erreurs.
SavePosition	REAL	0.0	Valeur de réglage de remplacement Si ErrorBehaviour = 1, l'actionneur est déplacé, en cas d'erreur, sur une position sûre pour l'installation et ce n'est qu'après que le système passe en mode "Inactif".

Variable	Type de données	Valeur par défaut	Description
			La plage de valeurs admissibles dépend de la configuration. • Config.FeedbackOn = FALSE et Config.OutputPerOn = FALSE : SavePosition = 0.0 ou 100.0 • Config.FeedbackOn = TRUE ou Config.OutputPerOn = TRUE: Config.OutputUpperLimit ≥ SavePosition ≥ Config.OutputLowerLimit
CurrentSetpoint	REAL	0.0	Consigne active actuellement Cette valeur est gelée au démarrage de l'optimisation.
Progress	REAL	0.0	Progrès de l'optimisation en % (0.0 à 100.0)
Config.InputPerOn(1)	BOOL	TRUE	Si InputPerOn = TRUE, c'est le paramètre Input_PER qui est utilisé. Si InputPerOn = FALSE, c'est le paramètre Input qui est utilisé.
Config.OutputPerOn(1)	BOOL	FALSE	Si OutputPerOn = TRUE, c'est le paramètre Output_PER qui est utilisé. Si OutputPerOn = FALSE, les paramètres Ouput_UP et Output_DN sont utilisés.
Config.LoadBackUp	BOOL	FALSE	Si LoadBackUp = TRUE, le dernier jeu de paramètres PID est rechargé. Le jeu a été enregistré avant la dernière optimisation. LoadBackUp est remis automatiquement à FALSE.
Config.InvertControl(1)	BOOL	FALSE	Inversion du sens de régulation Si InvertControl = TRUE, un signal d'écart croissant provoque une diminution de la valeur de réglage.
Config.FeedbackOn(1)	BOOL	FALSE	Si FeedbackOn = FALSE, une signalisation de position est simulée. Si FeedbackOn = TRUE, une signalisation de position est généralement activée.
Config.FeedbackPerOn(1)	BOOL	FALSE	FeedbackPerOn n'a d'effet que si FeedbackOn = TRUE. Si FeedbackPerOn = TRUE, l'entrée analogique est utilisée pour la signalisation de position (paramètre Feedback_PER). Si FeedbackPerOn = FALSE, le paramètre Feedback est utilisé pour la signalisation de position.
Config.ActuatorEndStopOn(1)	BOOL	FALSE	Si ActuatorEndStopOn = TRUE, la signalisation de position TOR Actuator_L et Actuator_H est prise en compte.
Config.InputUpperLimit(1)	REAL	120.0	Limite supérieure de la mesure A l'entrée de périphérie, la mesure peut dépasser de 18 % au plus la plage normée (dépassement haut). Un dépassement de la "limite supérieure de la mesure" ne provoque plus la signalisation d'erreur. Seuls la rupture de fil et le court-circuit sont détectés et PID_3Step se comporte comme vous en avez décidé sous Comportement en cas d'erreur. InputUpperLimit > InputLowerLimit
Config.InputLowerLimit(1)	REAL	0.0	Limite inférieure de la mesure InputLowerLimit < InputUpperLimit
Config.InputUpperWarning(1)	REAL	+3.40282-2e+38	Limite d'alerte supérieure de la mesure Quand vous configurez InputUpperWarning en dehors des limites de la mesure, la limite supérieure absolue configurée pour la mesure est utilisée comme limite d'alerte supérieure. Si vous configurez InputUpperWarning dans les limites de la mesure, cette valeur sera utilisée comme limite d'alerte supérieure. InputUpperWarning > InputLowerWarning InputUpperWarning ≤ InputUpperLimit

Variable	Type de données	Valeur par défaut	Description
Config.InputLowerWarning(1)	REAL	-3.402822e+38	Limite d'alerte inférieure de la mesure Quand vous configurez InputLowerWarning en dehors des limites de la mesure, la limite inférieure absolue configurée pour la mesure est utilisée comme limite d'alerte inférieure. Si vous configurez InputLowerWarning dans les limites de la mesure, cette valeur sera utilisée comme limite d'alerte inférieure. InputLowerWarning < InputUpperWarning InputLowerWarning ≥ InputLowerLimit
Config.OutputUpperLimit(1)	REAL	100.0	Limite supérieure de la valeur de réglage La plage de valeurs suivante est autorisée : UpperPointOut ≥ OutputUpperLimit > OutputLowerLimit Pour plus de détails, voir OutputLowerLimit
Config.OutputLowerLimit(1)	REAL	0.0	Limite inférieure de la valeur de réglage La plage de valeurs suivante est autorisée : OutputUpperLimit > OutputLowerLimit ≥ LowerPointOut Lors de l'utilisation de Output_PER, une limite de la valeur de réglage de -100 % correspond à la valeur Output_PER = -27648 ; 100 % correspond à la valeur Output_PER = 27648. Si OutputPerOn = FALSE et que FeedbackOn = FALSE, OutputLowerLimit et OutputUpperLimit ne sont pas évaluées. Output_UP et Output_DN sont remises à 0 lorsque Actuator_H = TRUE ou Actuator_L = TRUE (si ActuatorEndStopOn = TRUE) ou après un temps de course de 110 % * Config.TransitTime (si ActuatorEndStopOn = FALSE).
Config.SetpointUpperLimit(1)	REAL	+3.40282-2e+38	Limite supérieure de la consigne Quand vous configurez SetpointUpperLimit en dehors des limites de la mesure, la limite supérieure absolue configurée pour la mesure est utilisée comme limite supérieure par défaut de la consigne. Quand vous configurez SetpointUpperLimit dans les limites de la mesure, cette valeur est utilisée comme limite supérieure de la consigne.
Config.SetpointLowerLimit(1)	REAL	-3.402822e+38	Limite inférieure de la consigne Quand vous configurez SetpointLowerLimit en dehors des limites de la mesure, la limite inférieure absolue configurée pour la mesure sert de limite inférieure par défaut pour la consigne. Quand vous configurez SetpointLowerLimit dans les limites de la mesure, cette valeur est utilisée comme limite inférieure de la consigne.
Config.MinimumOnTime(1)	REAL	0.0	Plus petit temps ON Temps minimum en secondes pendant lequel le servomoteur doit être en marche. Config.MinimumOnTime n'est opérant que si Output_UP et Output_DN sont utilisés (Config.OutputPerOn = FALSE).
Config.MinimumOffTime(1)	REAL	0.0	Plus petit temps OFF Temps minimum en secondes pendant lequel le servomoteur doit être arrêté. Config.MinimumOffTime n'est opérant que si Output_UP et Output_DN sont utilisés (Config.OutputPerOn = FALSE).
Config.TransitTime(1)	REAL	30.0	Temps de positionnement du moteur Temps en secondes nécessaire pour le servomoteur pour faire passe la vanne de l'état fermé à l'état ouvert.

Variable	Type de données	Valeur par défaut	Description
Config.InputScaling.UpperPointIn[(1)]	REAL	27648.0	Mise à l'échelle Input_PER Haut Input_PER est converti en pourcentage à l'aide des deux couples de valeurs UpperPointOut, UpperPointIn et LowerPointOut, LowerPointIn de la structure InputScaling.
Config.InputScaling.LowerPointIn[(1)]	REAL	0.0	Mise à l'échelle Input_PER Bas Input_PER est converti en pourcentage à l'aide des deux couples de valeurs UpperPointOut, UpperPointIn et LowerPointOut, LowerPointIn de la structure InputScaling.
Config.InputScaling.UpperPointOut[(1)]	REAL	100.0	Mesure supérieure à l'échelle Input_PER est converti en pourcentage à l'aide des deux couples de valeurs UpperPointOut, UpperPointIn et LowerPointOut, LowerPointIn de la structure InputScaling.
Config.InputScaling.LowerPointOut[(1)]	REAL	0.0	Mesure inférieure à l'échelle Input_PER est converti en pourcentage à l'aide des deux couples de valeurs UpperPointOut, UpperPointIn et LowerPointOut, LowerPointIn de la structure InputScaling.
Config.FeedbackScaling.UpperPointIn[(1)]	REAL	27648.0	Mise à l'échelle Feedback_PER Haut Feedback_PER est converti en pourcentage à l'aide des deux couples de valeurs UpperPointOut, UpperPointIn et LowerPointOut, LowerPointIn de la structure FeedbackScaling.
Config.FeedbackScaling.LowerPointIn[(1)]	REAL	0.0	Mise à l'échelle Feedback_PER Bas Feedback_PER est converti en pourcentage à l'aide des deux couples de valeurs UpperPointOut, UpperPointIn et LowerPointOut, LowerPointIn de la structure FeedbackScaling.
Config.FeedbackScaling.UpperPointOut[(1)]	REAL	100.0	Butée supérieure Feedback_PER est converti en pourcentage à l'aide des deux couples de valeurs UpperPointOut, UpperPointIn et LowerPointOut, LowerPointIn de la structure FeedbackScaling. La plage de valeurs admissible dépend de la configuration. • FeedbackOn = FALSE : UpperPointOut = 100.0 • FeedbackOn = TRUE : UpperPointOut = 100.0 ou 0.0 UpperPointOut ≠ LowerPointOut
Config.FeedbackScaling.LowerPointOut[(1)]	REAL	0.0	Butée inférieure Feedback_PER est converti en pourcentage à l'aide des deux couples de valeurs UpperPointOut, UpperPointIn et LowerPointOut, LowerPointIn de la structure FeedbackScaling. La plage de valeurs admissible dépend de la configuration. • FeedbackOn = FALSE : LowerPointOut = 0.0 • FeedbackOn = TRUE : LowerPointOut = 0.0 ou -100.0 LowerPointOut ≠ UpperPointOut
GetTransitTime.InvertDirection	BOOL	FALSE	Quand InvertDirection = FALSE, la vanne est entièrement ouverte, fermée, puis réouverte pour déterminer le temps de positionnement. Quand InvertDirection = TRUE, la vanne est entièrement fermée, ouverte, puis refermée.

Variable	Type de données	Valeur par défaut	Description
GetTransitTime.SelectFeedback	BOOL	FALSE	Quand SelectFeedback = TRUE, Feedback_PER ou Feedback est pris en compte lors de la mesure du temps de positionnement. Quand SelectFeedback = FALSE, Actuator_H et Actuator_L sont pris en compte lors de la mesure du temps de positionnement.
GetTransitTime.Start	BOOL	FALSE	Quand Start = TRUE, la mesure du temps de positionnement est lancée.
GetTransitTime.State	INT	0	Phase actuelle de la mesure du temps de positionnement • State = 0 : Inactif • State = 1 : Ouvrir complètement la vanne • State = 2 : Fermer complètement la vanne • State = 3 : Régler la vanne sur la position cible (NewOutput) • State = 4 : Mesure du temps de positionnement terminée avec succès • State = 5 : Mesure du temps de positionnement annulée
GetTransitTime.NewOutput	REAL	0.0	Position cible pour la mesure de temps de positionnement avec signalisation de position La position cible doit se trouver dans les limites de la "Butée supérieure" et de la "Butée inférieure". La différence entre NewOutput et ScaledFeedback doit être au moins égale à 50 % de la plage de réglage admissible.
CycleTime.StartEstimation	BOOL	TRUE	Si StartEstimation = TRUE, la mesure du temps d'échantillonnage PID_3Step est lancée. Après la fin de la mesure, on a CycleTime.StartEstimation = FALSE.
CycleTime.EnEstimation	BOOL	TRUE	Si EnEstimation = TRUE, le temps d'échantillonnage PID_3Step est calculé.
CycleTime.EnMonitoring	BOOL	TRUE	Si EnMonitoring = TRUE, le temps d'échantillonnage PID_3Step est surveillé. Si PID_3Step ne peut pas être exécuté pendant la période d'échantillonnage, l'erreur 0800h est signalée et le mode de fonctionnement change. Le mode de fonctionnement vers lequel le système passe dépend de ActivateRecoverMode et ErrorBehaviour. Si EnMonitoring = FALSE, la période d'échantillonnage PID_3Step n'est pas surveillée, l'erreur 0800h n'est pas signalée et le mode de fonctionnement ne change pas.
CycleTime.Value(1)	REAL	0.1	Période d'échantillonnage PID_3Step en secondes CycleTime.Value est automatiquement déterminée et correspond normalement au temps de cycle de l'OB appelant.
CtrlParamsBackUp.SetByUser	BOOL	FALSE	Valeur enregistrée de Retain.CtrlParams.SetByUser Il est possible de recharger les valeurs de la structure CtrlParamsBackUp avec Config.LoadBackUp = TRUE.
CtrlParamsBackUp.Gain	REAL	1.0	Gain proportionnel enregistré
CtrlParamsBackUp.Ti	REAL	20.0	Temps d'intégration enregistré
CtrlParamsBackUp.Td	REAL	0.0	Temps de dérivation enregistré
CtrlParamsBackUp.TdFiltRatio	REAL	0.0	Coefficient de l'action par dérivation enregistré
CtrlParamsBackUp.PWeighting	REAL	0.0	Pondération de l'action P enregistrée
CtrlParamsBackUp.DWeighting	REAL	0.0	Pondération de l'action D enregistrée
CtrlParamsBackUp.Cycle	REAL	1.0	Temps d'échantillonnage enregistré de l'algorithme PID
CtrlParamsBackUp.InputDeadBand	REAL	0.0	Largeur enregistrée de la zone morte du signal d'écart

Variable	Type de données	Valeur par défaut	Description
PIDSelfTune.SUT.CalculateSUTParams	BOOL	FALSE	Les propriétés du système réglé sont enregistrées lors de l'optimisation. Si CalculateSUTParams = TRUE, les paramètres PID sont recalculés à partir de ces propriétés. Les paramètres PID sont calculés selon la méthode paramétrée dans TuneRuleSUT. CalculateSUTParams est mis sur FALSE après le calcul.
PIDSelfTune.SUT.TuneRuleSUT	INT	1	Calculer les paramètres pendant l'optimisation préalable selon la méthode : • TuneRuleSUT = 0 : PID rapide I (comportement de régulation plus rapide avec des amplitudes plus fortes de la valeur de réglage qu'avec TuneRuleSUT = 1) • TuneRuleSUT = 1 : PID lent I (comportement de régulation plus lent avec des amplitudes plus faibles de la valeur de réglage qu'avec TuneRuleSUT = 0) • TuneRuleSUT = 2 : PID Chien, Hrones, Reswick • TuneRuleSUT = 3 : PI Chien, Hrones, Reswick • TuneRuleSUT = 4 : PID rapide II (comportement de régulation plus rapide avec des amplitudes plus fortes de la valeur de réglage qu'avec TuneRuleSUT = 5) • TuneRuleSUT = 5 : PID lent II (comportement de régulation plus lent avec des amplitudes plus faibles de la valeur de réglage qu'avec TuneRuleSUT = 4) L'unique différence entre les méthodes TuneRuleSUT = 0 et 1 consiste dans le calcul du gain proportionnel par les méthodes TuneRuleSUT = 4 et 5 : Pour TuneRuleSUT = 0 et 1, le gain proportionnel est calculé à l'aide du temps de compensation du processus. Pour TuneRuleSUT = 4 et 5, ce calcul s'effectue à l'aide du temps de retard du processus. TuneRuleSUT = 4 et 5 donne une valeur plus élevée pour le gain proportionnel et ainsi un comportement de régulation plus rapide avec des amplitudes plus fortes de la valeur de réglage qu'avec TuneRuleSUT = 0 et 1.
PIDSelfTune.SUT.State	INT	0	La variable SUT.State indique la phase actuelle de l'optimisation préalable :
PIDSelfTune.TIR.RunIn	BOOL	FALSE	• RunIn = FALSE Si l'optimisation fine est démarrée depuis le mode inactif ou manuel, une optimisation préalable est lancée. Si l'optimisation fine est démarrée depuis le mode automatique, les paramètres PID existants sont utilisés pour un réglage sur la consigne. C'est seulement après cela que l'optimisation fine commence. Si l'optimisation préalable n'est pas possible, PID_3Step passe en mode de fonctionnement "Inactif". • RunIn = TRUE L'optimisation préalable est sautée. PID_3Step essaie d'atteindre la consigne avec la valeur de réglage mini ou maxi. Cela peut entraîner une suroscillation élevée. C'est seulement après cela que l'optimisation fine commence. RunIn est mis sur FALSE après l'optimisation fine.

Variable	Type de données	Valeur par défaut	Description
PIDSelfTune.TIR.CalculateTIRParams	BOOL	FALSE	Les propriétés du système réglé sont enregistrées lors de l'optimisation. Si CalculateTIRParams = TRUE, les paramètres PID sont recalculés à partir de ces propriétés. Les paramètres PID sont calculés selon la méthode paramétrée dans TuneRuleTIR. CalculateTIRParams est mis sur FALSE après le calcul.
PIDSelfTune.TIR.TuneRuleTIR	INT	0	Calculer les paramètres pendant l'optimisation fine selon la méthode : • TuneRuleTIR = 0 : PID automatique • TuneRuleTIR = 1 : PID rapide (comportement de régulation plus rapide avec des amplitudes plus fortes de la valeur de réglage qu'avec TuneRuleTIR = 2) • TuneRuleTIR = 2 : PID lent (comportement de régulation plus lent avec des amplitudes plus faibles de la valeur de réglage qu'avec TuneRuleTIR = 1) • TuneRuleTIR = 3 : PID Ziegler-Nichols • TuneRuleTIR = 4 : PI Ziegler-Nichols • TuneRuleTIR = 5 : P Ziegler-Nichols Pour pouvoir répéter le calcul des paramètres PID avec CalculateTIRParams et TuneRuleTIR = 0, 1 ou 2, il faut avoir exécuté l'optimisation fine précédente également avec TuneRuleTIR = 0, 1 ou 2. Si ce n'est pas le cas, TuneRuleTIR = 3 sera utilisé. Il est toujours possible de recalculer les paramètres PID avec CalculateTIRParams et TuneRuleTIR = 3, 4 ou 5.
PIDSelfTune.TIR.State	INT	0	La variable TIR.State affiche la phase actuelle de "l'optimisation fine" :
Retain.Mode	INT	0	Si la valeur de Retain.Mode est modifiée, le système passe à un autre mode de fonctionnement. Le mode de fonctionnement suivant est activé lorsque Mode est modifié vers : • Mode = 0 : Inactif • Mode = 1 : Optimisation préalable • Mode = 2 : Optimisation fine • Mode = 3 : Mode automatique • Mode = 4 : Mode manuel • Mode = 5 : Accoster la valeur de réglage de remplacement • Mode = 6 : Mesure du temps de positionnement • Mode = 7 : Surveillance des erreurs • Mode = 8 : Accoster la valeur de réglage de remplacement avec surveillance d'erreur Mode est rémanent.
Retain.CtrlParams.SetByUser(1)	BOOL	FALSE	Si SetByUser = FALSE, les paramètres PID sont déterminés automatiquement et PID_3Step travaille à la valeur de réglage avec une zone morte. La largeur de zone morte est calculée pendant l'optimisation à l'aide de l'écart type de la valeur de réglage et enregistrée dans Retain.CtrlParams.OutputDeadBand. Si SetByUser = TRUE, les paramètres PID sont entrés manuellement et PID_3 Step fonctionne avec la valeur de réglage sans zone morte. Retain.CtrlParams.OutputDeadBand = 0.0 SetByUser est rémanent.
Retain.CtrlParams.Gain(1)	REAL	1.0	Gain proportionnel actif Gain est rémanent.

Variable	Type de données	Valeur par défaut	Description
Retain.CtrlParams.Ti(1)	REAL	20.0	• Ti > 0.0 : Temps d'intégration actif • Ti = 0.0 : L'action I est désactivée Ti est rémanent.
Retain.CtrlParams.Td(1)	REAL	0.0	• Td > 0.0 : Temps de dérivation actif • Td = 0.0 : L'action D est désactivée Td est rémanent.
Retain.CtrlParams.TdFiltRatio(1)	REAL	0.0	Coefficient actif pour l'action par dérivation TdFiltRatio est rémanent.
Retain.CtrlParams.PWeighting(1)	REAL	0.0	Pondération active de l'action P PWeighting est rémanent.
Retain.CtrlParams.DWeighting(1)	REAL	0.0	Pondération active de l'action D DWeighting est rémanent.
Retain.CtrlParams.Cycle(1)	REAL	1.0	Période d'échantillonnage active de l'algorithme PID en secondes, arrondi à un multiple entier supérieur du temps de cycle de l'OB appelant. Cycle est rémanent.
Retain.CtrlParams.InputDeadBand(1)	REAL	0.0	Largeur de zone morte du signal d'écart InputDeadBand est rémanent.

Voir aussi

Paramètres State et Retain.Mode V1 (Page 349)
Variable ActivateRecoverMode V1 (Page 358)
Charger des objets technologiques dans l'appareil (Page 48)

10.2.5.6 Paramètres State et Retain.Mode V1

Corrélation entre les paramètres

Le paramètre State affiche le mode de fonctionnement actuel du régulateur PID. Vous ne pouvez pas modifier le paramètre State.

Pour changer de mode de fonctionnement, vous devez modifier la variable Retain.Mode. Ceci est valable également lorsque la valeur pour le nouveau mode de fonctionnement figure déjà dans Retain.Mode. Dans ce cas, réglez d'abord Retain.Mode = 0 puis ensuite Retain.Mode = 3. Si le mode de fonctionnement actuel autorise ce changement, State est mis sur la valeur de Retain.Mode.

Si PID_3Step change automatiquement le mode de fonctionnement, alors :
State != Retain.Mode.

Exemples :

- Après une optimisation préalable réussie
  State = 3 et Retain.Mode = 1
- En cas d'erreur
  State = 0 et Retain.Mode reste à la valeur en cours, par exemple Retain.Mode = 3

- ManualEnalbe = TRUE
  State = 4 et Retain.Mode reste à la valeur en cours, par exemple Retain.Mode = 3

REMARQUE

Vous souhaitez par exemple répéter une optimisation fine réussie sans terminer le mode de fonctionnement automatique avec Mode = 0.

Si vous mettez Retain.Mode à une valeur non valide pour un cycle, par exemple 9999, cela n'a aucun effet sur State. Au cycle suivant, vous réglez Mode = 2. Vous pouvez ainsi obtenir une modification de Retain.Mode sans passer d'abord au mode de fonctionnement "Inactif".

Signification des valeurs

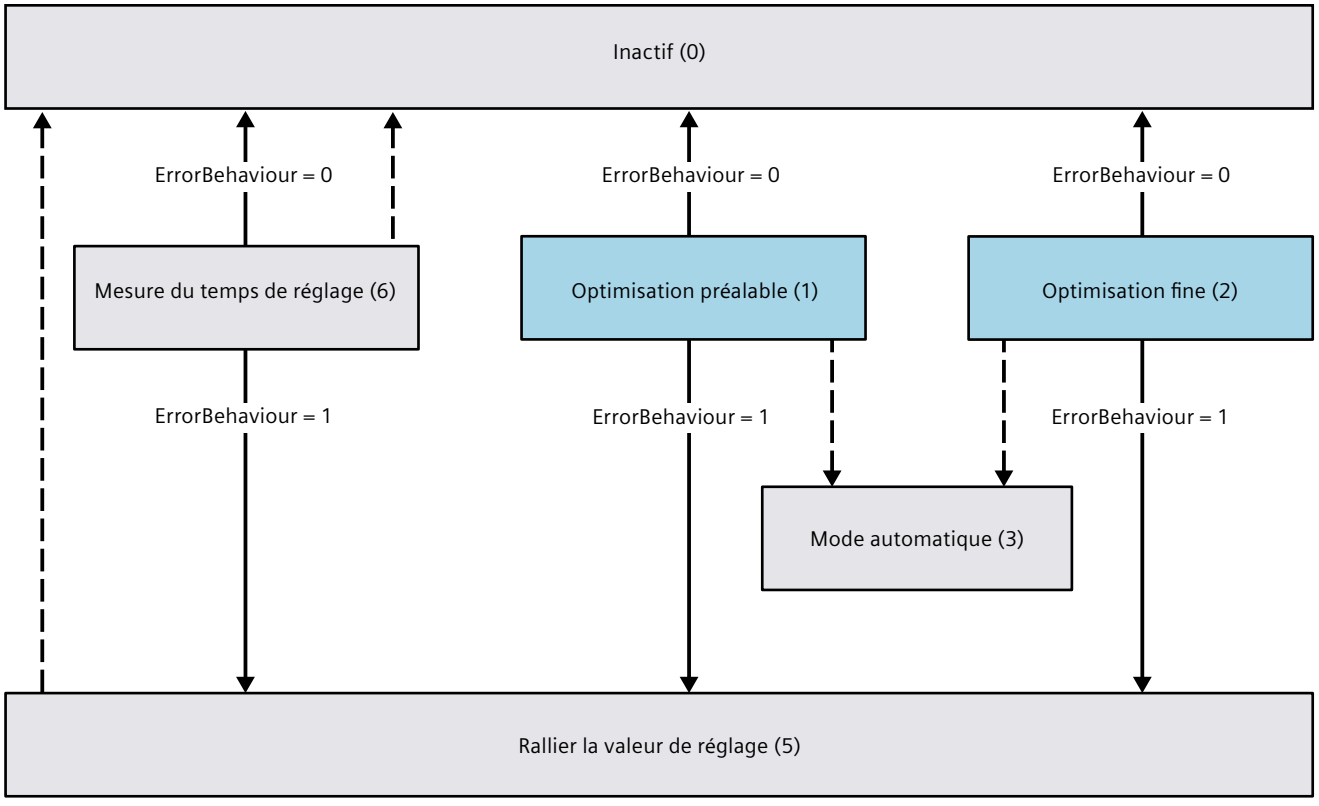
State / Retain.Mode	Description
0	Inactif Le régulateur est désactivé et ne modifie plus la position de la vanne.
1	Optimisation préalable L'optimisation préalable détermine la réponse du processus à une impulsion de la valeur de réglage et recherche le point d'inflexion. Les paramètres PID optimisés sont calculés à partir de l'incrémentation maximale et du temps mort du système réglé. Conditions pour une optimisation préalable : • State = 0 ou State = 4 • ManualEnable = FALSE • Le temps de positionnement du moteur est configuré ou mesuré. • La consigne et la mesure se trouvent dans les limites configurées. Plus la mesure est stable, plus il sera facile de déterminer des paramètres PID précis. Un bruit de la mesure est acceptable tant que la croissance de la mesure est nettement supérieure au bruit. Avant que les paramètres PID soient recalculés, ils sont sauvegardés et peuvent être réactivés avec Config.LoadBackUp. La consigne est gelée dans la variable CurrentSetpoint. Après une optimisation préalable réussie, le mode de fonctionnement passe en automatique ; si l'optimisation préalable échoue, le mode de fonctionnement passe au mode "Inactif". La phase de l'optimisation préalable est affichée avec la variable SUT.State.
2	Optimisation fine L'optimisation fine génère une oscillation constante limitée de la mesure. Les paramètres PID sont optimisés à partir de l'amplitude et de la fréquence de cette oscillation. Les différences entre la réponse du processus pendant l'optimisation préalable et l'optimisation fine sont analysées. Tous les paramètres PID sont recalculés à partir des résultats. Les paramètres PID de l'optimisation fine montrent généralement un meilleur comportement de référence et de perturbation que les paramètres PID de l'optimisation préalable. PID_3Step essaie automatiquement de créer une oscillation supérieure au bruit de la mesure. La stabilité de la mesure n'influence l'optimisation fine que de manière insignifiante. Les paramètres PID sont sauvegardés avant une optimisation fine. Ils peuvent être réactivés avec Config.LoadBackUp. La consigne est gelée dans la variable CurrentSetpoint. Conditions pour une optimisation fine : • Le temps de positionnement du moteur est configuré ou mesuré. • La consigne et la mesure se trouvent dans les limites configurées. • ManualEnable = FALSE • Mode automatique (State = 3), inactif (State = 0) ou mode manuel (State = 4)

State / Retain.Mode	Description
	L'optimisation fine se déroule de la manière suivante au démarrage : • Mode automatique (State = 3) Si vous souhaitez améliorer les paramètres PID existants à l'aide de l'optimisation, démarrez l'optimisation fine à partir du mode automatique. PID_3Step régule avec les paramètres PID existants jusqu'à ce que la boucle de régulation soit en régime établi et que les conditions pour une optimisation fine soient remplies. C'est seulement après cela que l'optimisation fine commence. • Inactif (State = 0) ou mode manuel (State = 4) Une optimisation préalable est toujours lancée en premier. Une régulation est effectuée avec les paramètres PID calculés jusqu'à ce que la boucle de régulation soit en régime établi et que les conditions pour une optimisation fine soient remplies. Quand PIDSelfTune.TIR.RunIn = TRUE, l'optimisation préalable est sautée et le système essaie d'atteindre la consigne avec la valeur de réglage mini ou maxi. Cela peut entraîner une suroscillation élevée. L'optimisation fine démarre alors automatiquement. Après une optimisation fine réussie, le régulateur passe en mode automatique ; en cas d'échec de l'optimisation fine, il passe au mode de fonctionnement "Inactif". La phase de l'optimisation fine est affichée avec la variable TIR.State.
3	Mode automatique En mode automatique, PID_3Step régule le système réglé en fonction des paramètres prédéfinis. Si l'une des conditions préalables suivantes est remplie, le système passe au mode automatique : • Optimisation préalable réussie • Optimisation fine réussie • Modification de la variable Retain.Mode sur la valeur 3. Quand la CPU est mise en route ou passe de STOP à RUN, PID_3Step démarre dans le dernier mode de fonctionnement actif. Pour laisser PID_3Step en mode "Inactif", mettez RunModeByStartup = FALSE. Le mode automatique tient compte de la variable ActivateRecoverMode.
4	Mode manuel En mode manuel, vous spécifiez des valeurs de réglage manuelles aux paramètres Manual_UP et Manual_DN ou ManualValue. Le paramètre ErrorBits indique si l'actionneur peut être amené sur la valeur de réglage en cas d'erreur. Ce mode de fonctionnement est activé si Retain.Mode = 4 ou en cas de front montant sur ManualEnable. Si ManualEnable = TRUE, seul State est modifié. Retain.Mode reste sur la même valeur. En cas de front descendant sur ManualEnable, PID_3Step repasse au mode de fonctionnement précédent. Le passage au mode automatique s'effectue sans à-coups. **PID_3Step V1.1** En cas d'erreur, le mode manuel est toujours possible. **PID_3Step V1.0** En cas d'erreur, le mode manuel dépend de la variable ActivateRecoverMode.
5	Accoster la valeur de réglage de remplacement Ce mode de fonctionnement est activé en cas d'erreur ou avec Reset = TRUE, lorsque Errorbehaviour = 1 et ActivateRecoverMode = FALSE.. PID_3Step met l'actionneur à la valeur de réglage de remplacement et passe ensuite au mode de fonctionnement "Inactif".
6	Mesure du temps de positionnement Le système détermine le temps nécessaire au moteur pour ouvrir entièrement la vanne depuis l'état fermé. Ce mode de fonctionnement sera activé lorsque GetTransitTime.Start = TRUE sera mis à 1. Quand des signaux de butée sont utilisés pour la mesure du temps de positionnement, la vanne est complètement ouverte depuis la position actuelle, complètement fermée, puis de nouveau complètement ouverte. Quand GetTransitTime.InvertDirection = TRUE, ce comportement est inversé. Quand une signalisation de position est utilisée pour la mesure du temps de positionnement, l'actionneur est mis à une position cible à partir de la position actuelle. Les limites de valeur de réglage ne sont pas prises en compte lors de la mesure du temps de positionnement. Il est possible de déplacer l'actionneur jusqu'à la butée supérieure ou inférieure.

State / Retain.Mode	Description
7	Surveillance des erreurs L'algorithme de régulation est désactivé et ne modifie plus la position de la vanne. Ce mode de fonctionnement est activé à la place du mode de fonctionnement "Inactif". Toutes les conditions suivantes doivent être remplies : • Mode = 3 (mode automatique) • Errorbehaviour = 0 • ActivateRecoverMode = TRUE • Une ou plusieurs erreurs sont apparues pour lesquelles ActivateRecoverMode (Page 358) s'applique. Dès que les erreurs ont disparu, PID_3Step repasse en mode automatique.
8	Accoster la valeur de réglage de remplacement avec surveillance d'erreur Ce mode de fonctionnement est activé à la place du mode de fonctionnement "Accoster la valeur de réglage de remplacement". PID_3Step met l'actionneur à la valeur de réglage de remplacement et passe ensuite au mode de fonctionnement "Surveillance d'erreur". Toutes les conditions suivantes doivent être remplies : • Mode = 3 (mode automatique) • Errorbehaviour = 1 • ActivateRecoverMode = TRUE • Une ou plusieurs erreurs sont apparues pour lesquelles ActivateRecoverMode (Page 358) s'applique. Dès que les erreurs ont disparu, PID_3Step repasse en mode automatique.

Changement de mode de fonctionnement automatique pendant la mise en route

En cas d'erreur, PID_3Step change automatiquement de mode de fonctionnement. Le schéma suivant montre l'influence de ErrorBehaviour sur le changement de mode de fonctionnement à partir des modes Mesure du temps de positionnement, Optimisation préalable et Optimisation fine.

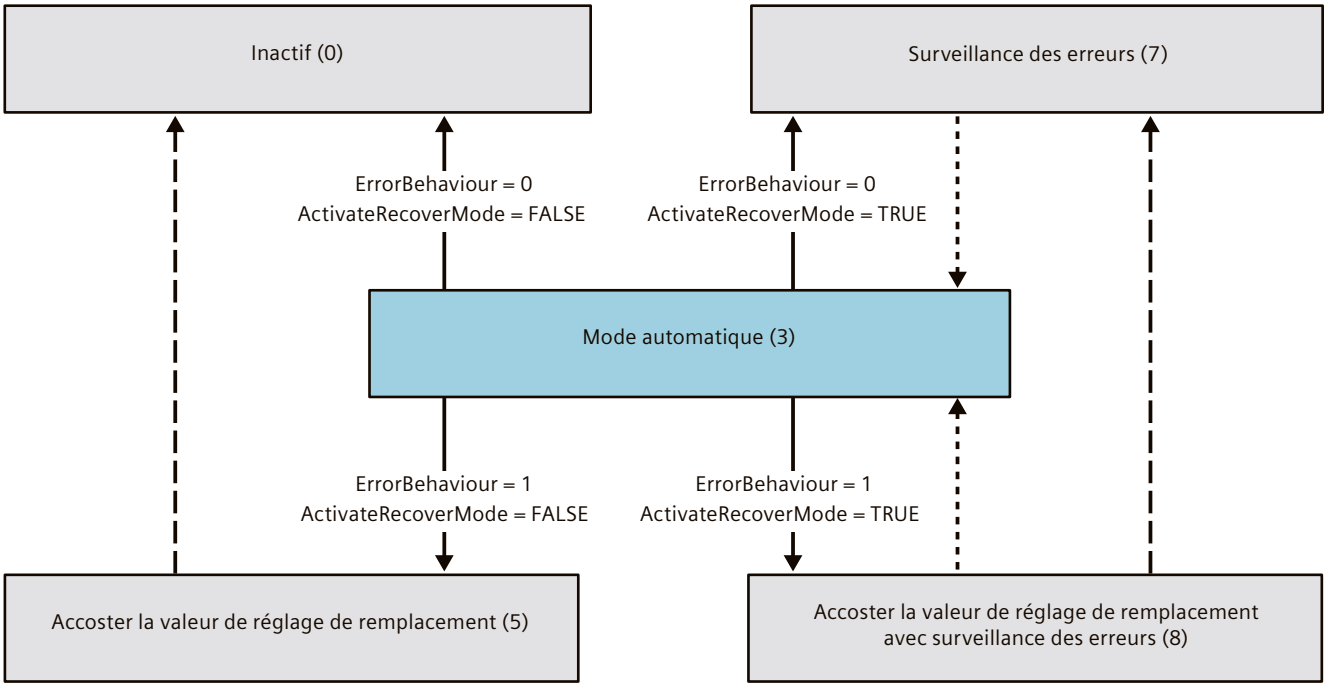


Changement automatique de mode de fonctionnement en cas d'erreur

Changement automatique de mode de fonctionnement quand le fonctionnement actuel est terminé.

Changement automatique de mode de fonctionnement en mode automatique (PID_3Step V1.1)

En cas d'erreur, PID_3Step change automatiquement de mode de fonctionnement. Le schéma suivant montre l'influence de ErrorBehaviour et ActivateRecoverMode sur ce changement de mode de fonctionnement.

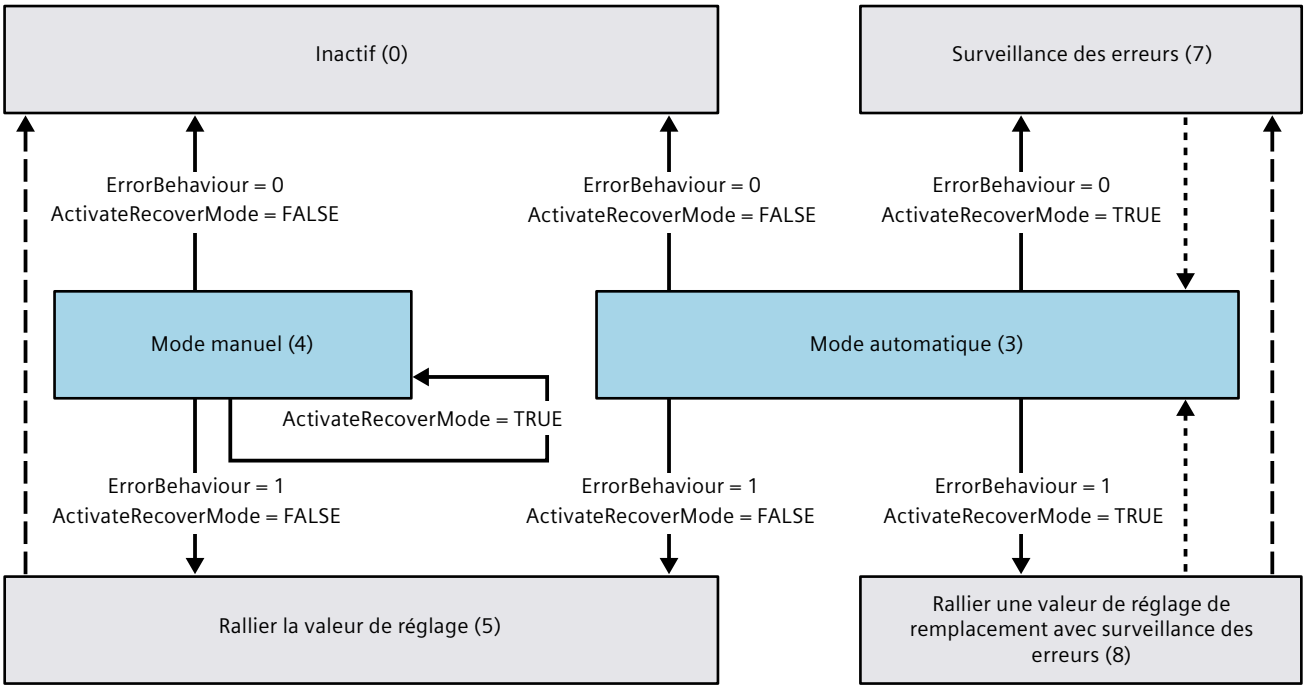


Changement automatique de mode de fonctionnement en cas d'erreur

Changement automatique de mode de fonctionnement quand le fonctionnement actuel est terminé.

Changement automatique de mode de fonctionnement en mode automatique et manuel (PID_3Step V1.0)

En cas d'erreur, PID_3Step change automatiquement de mode de fonctionnement. Le schéma suivant montre l'influence de ErrorBehaviour et ActivateRecoverMode sur ce changement de mode de fonctionnement.



Changement automatique de mode de fonctionnement en cas d'erreur

Changement automatique de mode de fonctionnement quand le fonctionnement actuel est terminé.

Changement automatique de mode de fonctionnement quand l'erreur n'est plus présente.

Voir aussi

Variable ActivateRecoverMode V1 (Page 358)
Paramètre ErrorBits V1 (Page 356)

10.2.5.7 Paramètre ErrorBits V1

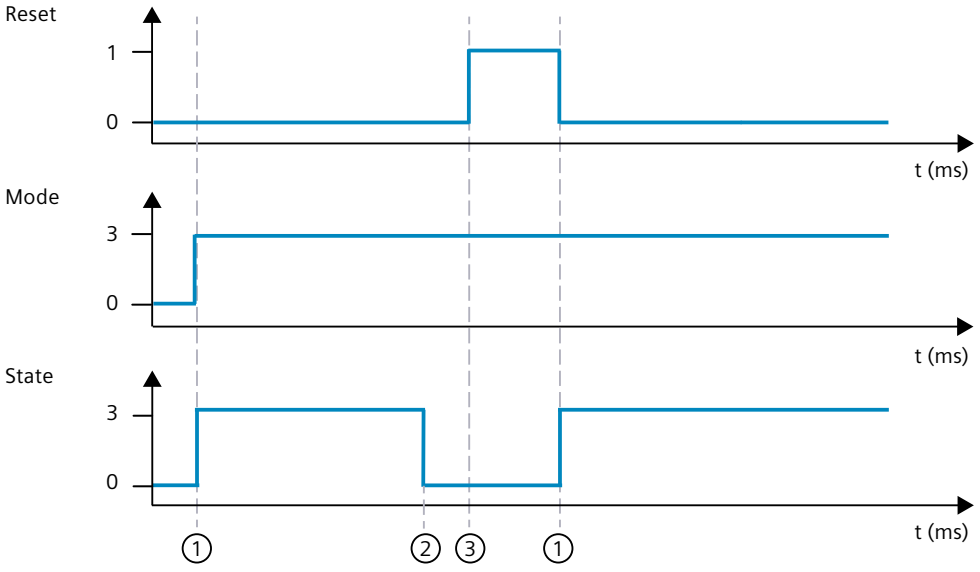
En présence de plusieurs erreurs simultanées, les valeurs des codes d'erreur s'affichent comme addition binaire. L'affichage du code d'erreur 0003, par ex., indique la présence simultanée des erreurs 0001 et 0002.

ErrorBits (DW#16#...)	Description
0000	Pas de présence d'erreur.
0001	Le paramètre "Input" se trouve en dehors des limites de la mesure. • Input > Config.InputUpperLimit ou • Input < Config.InputLowerLimit Quand ActivateRecoverMode = TRUE et ErrorBehaviour = 1, l'actionneur est amené sur la valeur de réglage de remplacement. Quand ActivateRecoverMode = TRUE et ErrorBehaviour = 0, l'actionneur reste à sa position actuelle. Quand ActivateRecoverMode = FALSE, l'actionneur reste à sa position actuelle. PID_3Step V1.1 Vous pouvez déplacer l'actionneur en mode manuel. PID_3Step V1.0 Le mode manuel n'est pas possible dans cet état. Vous ne pourrez déplacer à nouveau l'actionneur qu'après avoir éliminé l'erreur.
0002	Valeur invalide au paramètre "Input_PER". Vérifiez si une erreur est présente à l'entrée analogique. Si le mode automatique était actif avant l'apparition de l'erreur, ActivateRecoverMode = TRUE et que l'erreur a disparu, PID_3Step repasse en mode automatique.
0004	Erreur pendant l'optimisation fine. L'oscillation de la mesure n'a pas pu être maintenue.
0020	L'optimisation préalable n'est pas autorisée en mode automatique et pendant l'optimisation fine.
0080	Erreur lors de l'optimisation préalable. Les limites de la valeur de réglage ne sont pas configurées correctement ou la mesure ne réagit pas comme prévu. Vérifiez si les limites de la valeur de réglage sont configurées correctement et conviennent au sens de régulation. Assurez-vous en outre que la mesure n'oscille pas fortement avant le début de l'optimisation préalable.
0100	Une erreur durant l'optimisation fine a conduit à des paramètres invalides.
0200	Valeur invalide au paramètre "Input" : Le format numérique de la valeur est invalide. Si le mode automatique était actif avant l'apparition de l'erreur, ActivateRecoverMode = TRUE et que l'erreur a disparu, PID_3Step repasse en mode automatique.
0400	Le calcul de la valeur de réglage a échoué. Vérifiez les paramètres PID.
0800	Erreur de période d'échantillonnage : PID_3Step n'est pas appelé pendant la période d'échantillonnage de l'OB d'alarme cyclique. Si le mode automatique était actif avant l'apparition de l'erreur, ActivateRecoverMode = TRUE et que l'erreur a disparu, PID_3Step repasse en mode automatique. Si cette erreur s'est produite lors de la simulation avec PLCSIM, tenez compte des indications de la rubrique Simuler PID_3Step V1 avec PLCSIM (Page 153).
1000	Valeur invalide au paramètre "Setpoint" : Le format numérique de la valeur est invalide. Si le mode automatique était actif avant l'apparition de l'erreur, ActivateRecoverMode = TRUE et que l'erreur a disparu, PID_3Step repasse en mode automatique.
2000	Valeur invalide au paramètre Feedback_PER. Vérifiez si une erreur est présente à l'entrée analogique. L'actionneur ne peut pas être déplacé sur la valeur de réglage de remplacement et reste dans la position actuelle. Le mode manuel n'est pas possible dans cet état. Pour pouvoir sortir l'actionneur de cet état, vous devez désactiver la signalisation de position (Config. FeedbackOn = FALSE). Si le mode automatique était actif avant l'apparition de l'erreur, ActivateRecoverMode = TRUE et que l'erreur a disparu, PID_3Step repasse en mode automatique.

ErrorBits (DW#16#...)	Description
4000	Valeur invalide au paramètre Feedback. Le format numérique de la valeur est invalide. L'actionneur ne peut pas être déplacé sur la valeur de réglage de remplacement et reste dans la position actuelle. Le mode manuel n'est pas possible dans cet état. Pour pouvoir sortir l'actionneur de cet état, vous devez désactiver la signalisation de position (Config. FeedbackOn = FALSE). Si le mode automatique était actif avant l'apparition de l'erreur, ActivateRecoverMode = TRUE et que l'erreur a disparu, PID_3Step repasse en mode automatique.
8000	Erreur dans la signalisation de position TOR. Actuator_H = TRUE et Actuator_L = TRUE. L'actionneur ne peut pas être déplacé sur la valeur de réglage de remplacement et reste dans la position actuelle. Le mode manuel n'est pas possible dans cet état. Pour pouvoir sortir l'actionneur de cet état, vous devez désactiver les "Signaux de butée actionneur" (Config.ActuatorEndStopOn = FALSE). Si le mode automatique était actif avant l'apparition de l'erreur, ActivateRecoverMode = TRUE et que l'erreur a disparu, PID_3Step repasse en mode automatique.

10.2.5.8 Paramètre Reset V1

Un front montant sur Reset fait passer en mode de fonctionnement "inactif" et remet à zéro les erreurs et les avertissements. Un front descendant sur Reset fait passer au dernier mode de fonctionnement actif. Si le mode automatique était actif précédemment, la commutation en mode automatique s'effectue sans à-coup.



① Activation
① Erreur
① Reset

10.2.5.9 Variable ActivateRecoverMode V1

L'influence de la variable ActivateRecoverMode dépend de la version de l'instruction PID_3Step.

Comportement dans la version 1.1

La variable ActivateRecoverMode détermine le comportement en cas d'erreur en mode automatique. Pendant l'optimisation préalable, l'optimisation fine et la mesure du temps de positionnement, ActivateRecoverMode n'a pas d'effet.

ActivateRecover-Mode	Description
FALSE	En cas d'erreur, PID_3Step passe en mode "Inactif" ou "Accoster la valeur de réglage de remplacement". Le régulateur n'est activé que par une remise à 0 ou par une modification de Retain.Mode.
TRUE	En cas d'erreurs fréquentes en mode automatique, cette valeur détériore le comportement de régulation. Vérifiez alors le paramètre ErrorBits et éliminez la cause d'erreur. Quand l'une ou plusieurs des erreurs suivantes apparaissent, PID_3Step passe en mode de fonctionnement "Accoster la valeur de réglage de remplacement avec surveillance d'erreur" ou "Surveillance d'erreur" : • 0002h : valeur invalide du paramètre Input_PER. • 0200h : valeur invalide du paramètre Input. • 0800h : erreur de temps d'échantillonnage • 1000h : valeur invalide du paramètre Setpoint. • 2000h : valeur invalide du paramètre Feedback_PER. • 4000h : valeur invalide du paramètre Feedback. • 8000h : erreur dans la signalisation de position TOR. Avec les erreurs 2000h, 4000h et 8000h, PID_3Step ne peut **pas** accoster la valeur de réglage de remplacement configurée. Dès que les erreurs ont disparu, PID_3Step repasse en mode automatique.

Comportement dans la version 1.0

La variable ActivateRecoverMode détermine le comportement en cas d'erreur en mode automatique et en mode manuel. Pendant l'optimisation préalable, l'optimisation fine et la mesure du temps de positionnement, ActivateRecoverMode n'a pas d'effet.

ActivateRecover-Mode	Description
FALSE	En cas d'erreur, PID_3Step passe en mode "Inactif" ou "Accoster la valeur de réglage de remplacement". Le régulateur n'est activé que par une remise à 0 ou par une modification de Retain.Mode.
TRUE	**Erreurs en mode automatique** En cas d'erreurs fréquentes en mode automatique, cette valeur détériore le comportement de régulation. Vérifiez alors le paramètre ErrorBits et éliminez la cause d'erreur. Quand l'une ou plusieurs des erreurs suivantes apparaissent, PID_3Step passe en mode de fonctionnement "Accoster la valeur de réglage de remplacement avec surveillance d'erreur" ou "Surveillance d'erreur" : • 0002h : valeur invalide du paramètre Input_PER. • 0200h : valeur invalide du paramètre Input. • 0800h : erreur de temps d'échantillonnage • 1000h : valeur invalide du paramètre Setpoint. • 2000h : valeur invalide du paramètre Feedback_PER.

ActivateRecover-Mode	Description
	• 4000h : valeur invalide du paramètre Feedback. • 8000h : erreur dans la signalisation de position TOR. Avec les erreurs 2000h, 4000h et 8000h, PID_3Step ne peut **pas** accoster la valeur de réglage de remplacement configurée. Dès que les erreurs ont disparu, PID_3Step repasse en mode automatique. **Erreurs en mode manuel** Quand l'une ou plusieurs des erreurs suivantes apparaissent, PID_3Step reste en mode manuel : • 0002h : valeur invalide du paramètre Input_PER. • 0200h : valeur invalide du paramètre Input. • 0800h : erreur de temps d'échantillonnage • 1000h : valeur invalide du paramètre Setpoint. • 2000h : valeur invalide du paramètre Feedback_PER. • 4000h : valeur invalide du paramètre Feedback. • 8000h : erreur dans la signalisation de position TOR. Avec les erreurs 2000h, 4000h et 8000h, vous ne pouvez **pas** amener la vanne dans une position appropriée.

Voir aussi

Variables statiques PID_3Step V1 (Page 342)

Paramètres State et Retain.Mode V1 (Page 349)

10.2.5.10 Variable Warning V1

En présence simultanée de plusieurs alertes, les valeurs des alertes sont affichées sous forme d'addition binaire. Si l'alerte 0003 est affichée p. ex., cela indique la présence simultanée des alertes 0001 et 0002.

Warning (DW#16#...)	Description
0000	Aucune alerte n'est présente.
0001	Le point d'inflexion n'a pas été trouvé pendant l'optimisation préalable.
0002	L'oscillation était renforcée pendant l'optimisation fine.
0004	La consigne a été limitée à des limites paramétrées.
0008	Toutes les propriétés nécessaires du système réglé n'ont pas été déterminées pour la méthode de calcul choisie. Les paramètres PID ont été calculés avec la méthode TuneRuleTIR = 3 à titre de remplacement.
0010	Impossible de modifier le mode de fonctionnement car ManualEnable = TRUE
0020	Le temps d'échantillonnage de l'algorithme PID est limité par le temps de cycle de l'OB appelant. Afin d'obtenir de meilleurs résultats, utilisez des temps de cycle de l'OB plus courts.
0040	La mesure a dépassé l'une de ses limites d'alerte.
0080	valeur incorrecte sur Retain.Mode. Le changement de mode de fonctionnement n'est pas effectué.
0100	La valeur manuelle a été limitée aux limites de la sortie du régulateur.
0200	La règle utilisée pour l'optimisation ne donne pas un véritable résultat ou n'est pas prise en charge.
0400	Lors de la mesure du temps de positionnement, une méthode ne convenant pas pour l'actionneur a été choisie. Le temps de positionnement ne peut pas être mesuré, car les paramètres de l'actionneur ne correspondent pas à la méthode de mesure sélectionnée.

Warning (DW#16#...)	Description
0800	Lors de la mesure du temps de positionnement, la différence entre la position actuelle et la nouvelle valeur de réglage est trop petite. Ceci peut occasionner des résultats erronés. La différence entre la valeur de réglage actuelle et la nouvelle valeur de réglage doit au moins être égale à 50 % de la plage de réglage totale.
1000	La valeur de réglage de remplacement ne peut pas être atteinte, car elle est en dehors des limites de la valeur de réglage.

Les alarmes suivantes sont supprimées dès que la cause est éliminée :

- 0004
- 0020
- 0040
- 0100

Toutes les autres alarmes sont supprimées avec un front montant sur Reset.

10.2.5.11 Variable SUT.State V1

SUT.State	Nom	Description
0	SUT_INIT	Initialiser l'optimisation préalable
50	SUT_TPDN	Déterminer position initiale sans signalisation de position
100	SUT_STDABW	Calculer divergence standard
200	SUT_GET_POI	Déterminer point d'inflexion
300	SUT_GET_RISETM	Déterminer temps de montée
9900	SUT_IO	Optimisation préalable réussie
1	SUT_NIO	Optimisation préalable échouée

10.2.5.12 Variable TIR.State V1

TIR.State	Nom	Description
-100	TIR_FIRST_SUT	L'optimisation fine n'est pas possible. Une optimisation préalable est d'abord réalisée.
0	TIR_INIT	Initialiser l'optimisation fine
200	TIR_STDABW	Calculer divergence standard
300	TIR_RUN_IN	Essayer d'atteindre la consigne avec la valeur de réglage maxi ou mini
400	TIR_CTRLN	Essayer d'atteindre la consigne avec les paramètres PID existants (si l'optimisation préalable a réussi)
500	TIR_OSZIL	Déterminer oscillation et calculer paramètres
9900	TIR_IO	Optimisation fine réussie
1	TIR_NIO	Optimisation fine échouée

10.3 PID_Temp

10.3.1 Nouveautés PID_Temp

PID_Temp V1.1

- **Réaction de la valeur de réglage lorsque le mode de fonctionnement passe de "Inactif" à "Mode automatique"**
  La nouvelle option IntegralResetMode = 4 a été ajoutée et définie comme valeur par défaut. Avec IntegralResetMode = 4, l'action I est préréglée automatiquement de manière à ce que, lorsque le mode de fonctionnement passe de "Inactif" à "Mode automatique", le signal d'écart entraîne un saut de même signe de la valeur de réglage PID.
- **Initialisation de l'action intégrale en mode automatique**
  L'action intégrale peut être initialisée en mode automatique à l'aide des variables OverwriteInitialOutputValue et PIDCtrl.PIDInit. Cela facilite l'utilisation de PID_Temp dans des régulations en mode alternatif.

10.3.2 Compatibilité avec CPU et FW

Le tableau suivant montre les CPU et les versions de PID_Temp compatibles.

CPU	FW	PID_Temp
S7-1200	à partir de V4.2	V1.1 V1.0
	V4.1	V1.0
S7-1500	à partir de V3.0	V1.1
	V2.0 à V2.9	V1.1 V1.0
	V1.7 à V1.8	V1.0

10.3.3 Temps de traitement de la CPU et espace mémoire requis PID_Temp V1

Temps de traitement de la CPU

Temps de traitement de CPU typiques de l'objet technologique PID_Temp à partir de la version V1.0 en fonction du type de CPU et mode de fonctionnement pour CPU standard, F, T et TF.

CPU	Firm-ware	Temps de traitement de CPU typ. Mode automatique	Temps de traitement de CPU typ. Optimisation préalable et optimisation fine
CPU 1211	≥ V4.1	290 µs	450 µs
CPU 1212			
CPU 1214			
CPU 1215			
CPU 1217			
CPU 1510SP	≤ V2.9	85 µs	140 µs
CPU 1511			
CPU 1511C			
CPU 1512C			
CPU 1512SP			
CPU 1513			
CPU 1515		80 µs	110 µs
CPU 1516			
CPU 1517	≥ V1.7	12 µs	18 µs
CPU 1518		6 µs	9 µs
CPU 1510SP	≥ V3.0	75 µs	110 µs
CPU 1511			
CPU 1511C			
CPU 1512C			
CPU 1512SP			
CPU 1513			
CPU 1514SP			
CPU 1515		55 µs	90 µs
CPU 1516			

Temps de traitement de CPU typiques de l'objet technologique PID_Temp à partir de la version V1.0 en fonction du type de CPU et mode de fonctionnement pour CPU R dans l'état système RUN-Redundant.

CPU	Firm-ware	Temps de traitement de CPU typ. Mode automatique	Temps de traitement de CPU typ. Optimisation préalable et optimisation fine
CPU 1513R	≥ V3.0	110 µs	160 µs
CPU 1515R		95 µs	130 µs

Espace mémoire requis

Espace mémoire requis d'un DB d'instance de l'objet technologique PID_Temp à partir de la version V1.0.

Espace mémoire requis	Espace mémoire requis du DB d'instance de PID_Temp V1
Taille de mémoire de chargement requise	env. 4700 octets
Taille totale de la mémoire de travail requise	1280 octets
Taille de la mémoire rémanente requise	100 octets

10.3.4 PID_Temp

10.3.4.1 Description PID_Temp

Description

L'instruction PID_Temp met à disposition un régulateur PID avec optimisation intégrée pour procédés de température. PID_Temp convient aux applications de pur chauffage ou chauffage/refroidissement.

Les modes suivants sont disponibles :

- Inactif
- Optimisation préalable
- Optimisation fine
- Mode automatique
- Mode manuel
- Valeur de réglage de remplacement avec surveillance des erreurs

Les modes de fonctionnement sont décrits en détail par le paramètre State.

Algorithme PID

PID_Temp est un régulateur PIDT1 avec anti-saturation et pondération de l'action P et D. L'algorithme PID fonctionne selon la formule suivante (zone de régulation et zone morte désactivées) :

$$y = K_p \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

Le tableau ci-dessous donne la signification des symboles utilisés dans la formule et dans les figures qui suivent.

Symbole	Description	Paramètres correspondants de l'instruction PID_Temp
y	Valeur de réglage de l'algorithme PID	-
K_p	Gain proportionnel	Retain.CtrlParams.Heat.Gain Retain.CtrlParams.Cool.Gain CoolFactor
s	Opérateur de Laplace	-
b	Pondération de l'action P	Retain.CtrlParams.Heat.PWeighting Retain.CtrlParams.Cool.PWeighting
w	Consigne	CurrentSetpoint
x	Mesure	ScaledInput
T_I	Temps d'intégration	Retain.CtrlParams.Heat.Ti Retain.CtrlParams.Cool.Ti
T_D	Temps de dérivation	Retain.CtrlParams.Heat.Td Retain.CtrlParams.Cool.Td
a	Coefficient pour l'action par dérivation (action par dérivation T1 = a × T_D)	Retain.CtrlParams.Heat.TdFiltRatio Retain.CtrlParams.Cool.TdFiltRatio
c	Pondération de l'action D	Retain.CtrlParams.Heat.DWeighting Retain.CtrlParams.Cool.DWeighting
DeadZone	Largeur de zone morte	Retain.CtrlParams.Heat.DeadZone Retain.CtrlParams.Cool.DeadZone
ControlZone	Largeur de zone de régulation	Retain.CtrlParams.Heat.ControlZone Retain.CtrlParams.Cool.ControlZone

Schéma fonctionnel PID_Temp

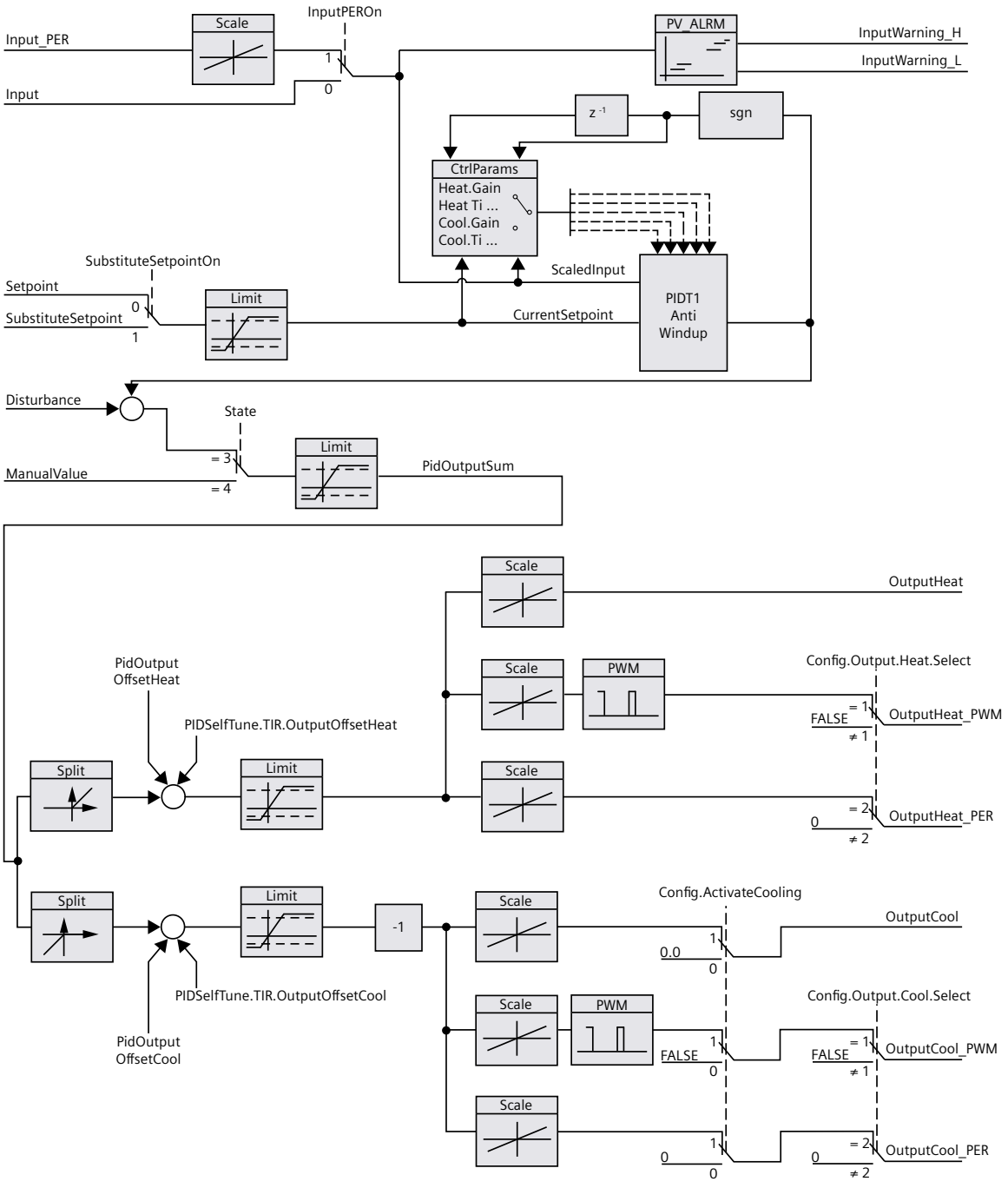
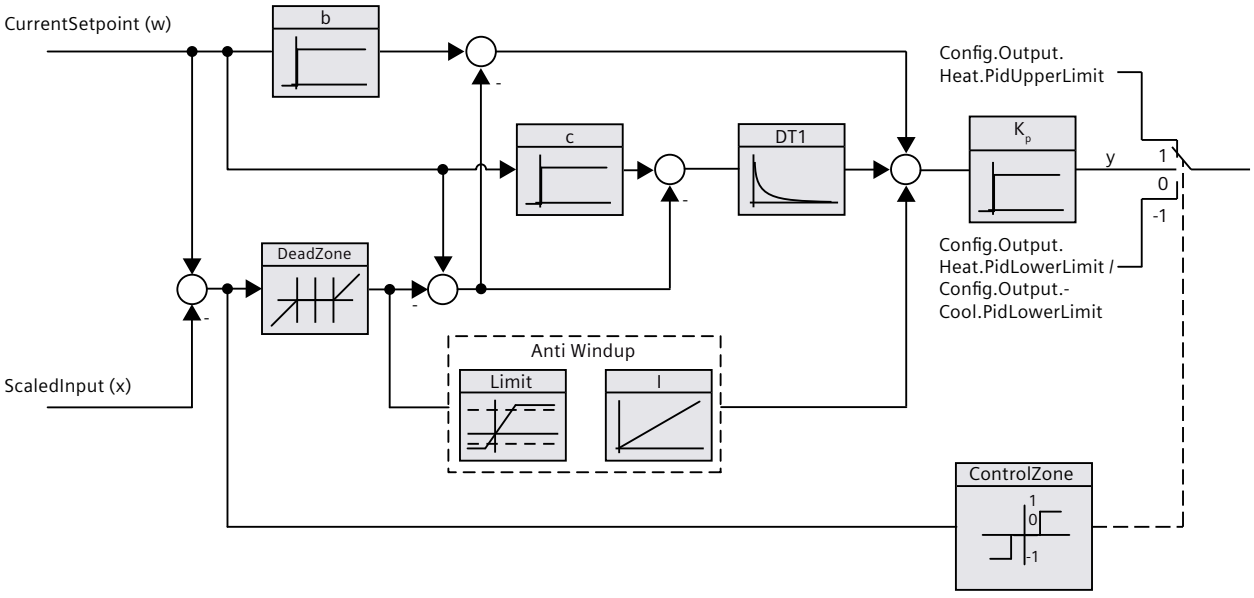


Schéma fonctionnel PIDT1 avec anti-saturation



Appel

PID_Temp est appelé à l'intervalle de temps constant d'un OB d'alarme cyclique.

Chargement dans l'appareil

Les valeurs effectives de variables rémanentes ne sont actualisées que si vous chargez entièrement PID_Temp .
Charger un objet technologique dans l'appareil (Page 48)

Démarrage

Au démarrage de la CPU, PID_Temp démarre dans le mode de fonctionnement enregistré dans le paramètre d'entrée/sortie Mode. Pour passer dans le mode de fonctionnement "Inactif" au démarrage, mettez RunModeByStartup = FALSE.

Comportement en cas d'erreur

Le comportement en cas d'erreur dépend des variables SetSubstituteOutput et ActivateRecoverMode. Quand ActivateRecoverMode = TRUE, le comportement dépend en plus de l'erreur apparue.

SetSubstituteOutput	**ActivateRecoverMode**	**Editeur de configuration > Paramètres de base sortie > Mettre PidOutputSum à**	**Comportement**
non signif.	FALSE	zéro (inactif)	Passage en mode de fonctionnement "Inactif" (State = 0) La valeur de réglage de l'algorithme PID et de toutes les sorties pour chauffage et refroidissement sont mises à 0. La mise à l'échelle des sorties pour chauffage et refroidissement n'est pas active.
FALSE	TRUE	Valeur actuelle pour la durée de l'erreur	Passage au mode de fonctionnement "Valeur de réglage de remplacement avec surveillance des erreurs" (State = 5) La valeur de réglage actuelle est transmise à l'actionneur pour la durée de l'erreur.
TRUE	TRUE	Valeur de réglage de remplacement pour la durée de l'erreur	Passage au mode de fonctionnement "Valeur de réglage de remplacement avec surveillance des erreurs" (State = 5) La valeur à SubstituteOutput est transmise à l'actionneur pour la durée de l'erreur.

PID_Temp utilise ManualValue comme valeur de réglage en mode manuel, sauf si ManualValue est invalide.

- Si ManualValue est invalide, c'est SubstituteOutput qui est utilisé.
- Si ManualValue et SubstituteOutput sont invalides, c'est Config.Output.Heat.PidLowerLimit qui est utilisé.

Le paramètre Error indique si une erreur est actuellement présente. Quand l'erreur a disparu, Error = FALSE. Le paramètre ErrorBits indique les erreurs qui se sont produites. ErrorBits est réinitialisé par un front montant à Reset ou ErrorAck.

10.3.4.2 Mode de fonctionnement de PID_Temp

Surveiller les limites de la mesure

Vous définissez une limite supérieure et une limite inférieure de la mesure dans les variables Config.InputUpperLimit et Config.InputLowerLimit. Quand la mesure se trouve en dehors de ces limites, une erreur apparaît (ErrorBits = 0000001h).

Vous définissez une limite d'alerte supérieure et une limite d'alerte inférieure de la mesure dans les variables Config.InputUpperWarning et Config.InputLowerWarning. Quand la mesure se trouve en dehors de ces limites d'alerte, une alerte apparaît (Warning = 0000040h) et le paramètre de sortie InputWarning_H ou InputWarning_L passe à TRUE.

Limiter la consigne

Vous définissez une limite supérieure et une limite inférieure de la consigne dans les variables Config.SetpointUpperLimit et Config.SetpointLowerLimit. PID_Temp limite la consigne automatiquement aux limites de la mesure. Vous pouvez limiter la consigne à une plage plus réduite. PID_Temp contrôlera si cette plage se trouve dans les limites de la mesure. Quand la consigne se trouve hors de ces limites, c'est la limite supérieure ou la limite inférieure qui est utilisée comme consigne et le paramètre de sortie SetpointLimit_H ou SetpointLimit_L passe à TRUE.

La consigne est limitée dans tous les modes de fonctionnement.

Consigne de remplacement

Vous pouvez spécifier une consigne de remplacement dans la variable SubstituteSetpoint et l'activer avec SubstituteSetpointOn = TRUE. Ceci vous permet, par exemple, de spécifier momentanément de manière directe la consigne pour un régulateur esclave dans une cascade, sans modifier le programme utilisateur. Les limites fixées pour la consigne s'appliquent aussi à la consigne de remplacement.

Chauffage et refroidissement

Avec le réglage par défaut, PID_Temp n'utilise que les sorties pour chauffage (OutputHeat, OutputHeat_PWM, OutputHeat_PER). La valeur de réglage de l'algorithme PID (PidOutputSum) est mise à l'échelle et fournie aux sorties pour le chauffage. Vous déterminez si OutputHeat_PWM ou OutputHeat_PER sera calculé, à l'aide de Config.Output.Heat.Select. OutputHeat est toujours calculé,.

Avec Config.ActivateCooling = TRUE, vous pouvez aussi activer les sorties pour refroidissement (OutputCool, OutputCool_PWM, OutputCool_PER). Les valeurs de réglage positives de l'algorithme PID (PidOutputSum) sont mises à l'échelle et fournies aux sorties pour chauffage. Les valeurs de réglage négatives de l'algorithme PID sont mises à l'échelle et fournies aux sorties pour le refroidissement. Vous déterminez si OutputCool_PWM ou OutputCool_PER sera calculé, à l'aide de Config.Output.Cool.Select. OutputCool est toujours calculé,.

Deux méthodes sont disponibles pour calculer la valeur de réglage PID quand le refroidissement est activé :

- Facteur pour le refroidissement (Config.AdvancedCooling = FALSE) :
  La valeur de réglage pour le refroidissement est calculée avec les paramètres PID pour le chauffage en tenant compte du facteur configurable pour le refroidissement Config.CoolFactor. Cette méthode convient quand l'actionneur de chauffage et l'actionneur de refroidissement ont des comportements dans le temps semblables, mais des gains différents. En choisissant cette méthode, l'optimisation préalable et l'optimisation fine pour le refroidissement, ainsi que le jeu de paramètres PID pour le refroidissement ne sont pas disponibles. Seules les optimisations pour le chauffage peuvent être exécutées.
- Commutation des paramètres PID (Config.AdvancedCooling = TRUE) :
  La valeur de réglage pour le refroidissement est calculée au moyen d'un jeu de paramètres PID propre. L'algorithme PID décide, en s'appuyant sur la valeur de réglage calculée et sur le signal d'écart, si ce sont les paramètres PID pour le chauffage ou pour le refroidissement qui sont utilisés. Cette méthode convient quand l'actionneur de chauffage et l'actionneur de refroidissement ont des comportements dans le temps et des gains différents. L'optimisation préalable et l'optimisation fine ne sont disponibles pour le refroidissement qu'avec cette méthode.

Limites et mise à l'échelle de la valeur de réglage

Selon le mode de fonctionnement, la valeur de réglage PID (PidOutputSum) est soit calculée automatiquement par l'algorithme PID, soit spécifiée par la valeur manuelle (ManualValue) ou par la valeur de réglage de remplacement configurée (SubstituteOutput).

La valeur de réglage PID est limitée en fonction de la configuration.

- Quand le refroidissement est désactivé (Config.ActivateCooling = FALSE), la limite supérieure en vigueur est Config.Output.Heat.PidUpperLimit et la limite inférieure Config.Output.Heat.PidLowerLimit.
- Quand le refroidissement est activé (Config.ActivateCooling = TRUE), la limite supérieure en vigueur est Config.Output.Heat.PidUpperLimit et la limite inférieure Config.Output.Cool.PidLowerLimit.

La valeur de réglage PID est mise à l'échelle et fournie aux sorties pour le chauffage et le refroidissement. La mise à l'échelle peut être spécifiée séparément pour chaque sortie et elle est déterminée au moyen de 2 paires de valeurs dans les structures Config.Output.Heat et Config.Output.Cool :

Sortie	Paire de valeurs	Paramètres
OutputHeat	Paire de valeurs 1	Limite supérieure de la valeur de réglage PID (chauffage) Config.Output.Heat.PidUpperLimit, Valeur de réglage supérieure mise à l'échelle (chauffage) Config.Output.Heat.UpperScaling
	Paire de valeurs 2	Limite inférieure de la valeur de réglage PID (chauffage) Config.Output.Heat.PidLowerLimit, Valeur de réglage inférieure mise à l'échelle (chauffage) Config.Output.Heat.LowerScaling
OutputHeat_PWM	Paire de valeurs 1	Limite supérieure de la valeur de réglage PID (chauffage) Config.Output.Heat.PidUpperLimit, Valeur de réglage PWM supérieure mise à l'échelle (chauffage) Config.Output.Heat.PwmUpperScaling
	Paire de valeurs 2	Limite inférieure de la valeur de réglage PID (chauffage) Config.Output.Heat.PidLowerLimit, Valeur de réglage PWM inférieure mise à l'échelle (chauffage) Config.Output.Heat.PwmLowerScaling
OutputHeat_PER	Paire de valeurs 1	Limite supérieure de la valeur de réglage PID (chauffage) Config.Output.Heat.PidUpperLimit, Valeur de réglage analogique supérieure mise à l'échelle (chauffage) Config.Output.Heat.PerUpperScaling
	Paire de valeurs 2	Limite inférieure de la valeur de réglage PID (chauffage) Config.Output.Heat.PidLowerLimit, Valeur de réglage analogique inférieure mise à l'échelle (chauffage) Config.Output.Heat.PerLowerScaling
OutputCool	Paire de valeurs 1	Limite inférieure de la valeur de réglage PID (refroidissement) Config.Output.Cool.PidLowerLimit, Valeur de réglage supérieure mise à l'échelle (refroidissement) Config.Output.Cool.UpperScaling
	Paire de valeurs 2	Limite supérieure de la valeur de réglage PID (refroidissement) Config.Output.Cool.PidUpperLimit, Valeur de réglage inférieure mise à l'échelle (refroidissement) Config.Output.Cool.LowerScaling

Quand le refroidissement est activé (Config.ActivateCooling = TRUE), Config.Output.Heat.PidLowerLimit doit avoir la valeur 0.0.

Config.Output.Cool.PidUpperLimit doit toujours avoir la valeur 0.0.

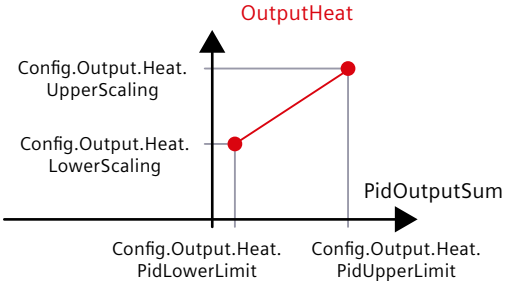
Sortie	Paire de valeurs	Paramètres
OutputCool_PWM	Paire de valeurs 1	Limite inférieure de la valeur de réglage PID (refroidissement) Config.Output.Cool.PidLowerLimit, Valeur de réglage PWM supérieure mise à l'échelle (refroidissement) Config.Output.Cool.PwmUpperScaling
	Paire de valeurs 2	Limite supérieure de la valeur de réglage PID (refroidissement) Config.Output.Cool.PidUpperLimit, Valeur de réglage PWM inférieure mise à l'échelle (refroidissement) Config.Output.Cool.PwmLowerScaling
OutputCool_PER	Paire de valeurs 1	Limite inférieure de la valeur de réglage PID (refroidissement) Config.Output.Cool.PidLowerLimit, Valeur de réglage analogique supérieure mise à l'échelle (refroidissement) Config.Output.Cool.PerUpperScaling
	Paire de valeurs 2	Limite supérieure de la valeur de réglage PID (refroidissement) Config.Output.Cool.PidUpperLimit, Valeur de réglage analogique inférieure mise à l'échelle (refroidissement) Config.Output.Cool.PerLowerScaling

Quand le refroidissement est activé (Config.ActivateCooling = TRUE), Config.Output.Heat.PidLowerLimit doit avoir la valeur 0.0.

Config.Output.Cool.PidUpperLimit doit toujours avoir la valeur 0.0.

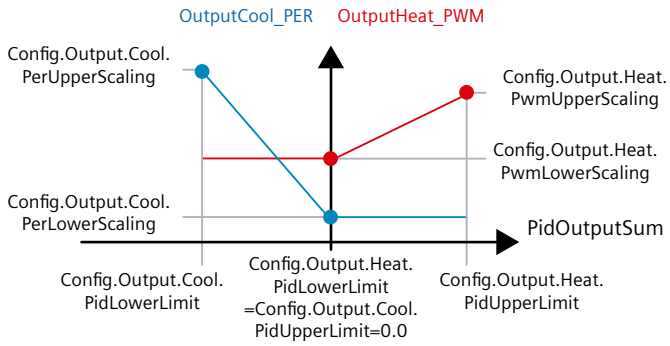
Exemple :

MIse à l'échelle de la sortie quand la sortie OutputHeat est utilisée (refroidissement désactivé ; Config.Output.Heat.PidLowerLimit peut être différent de 0.0) :



Exemple :

MIse à l'échelle de la sortie quand les sorties OutputHeat_PWM et OutputCool_PER sont utilisées (refroidissement activé ; Config.Output.Heat.PidLowerLimit doit être égal à 0.0) :



Sauf dans le mode de fonctionnement "Inactif", la valeur fournie à une sortie est toujours comprise entre sa valeur de réglage supérieure mise à l'échelle et sa valeur de réglage inférieure mise à l'échelle, par ex. pour OutputHeat toujours entre Config.Output.Heat.UpperScaling et Config.Output.Heat.LowerScaling.

Si vous voulez limiter la valeur à la sortie correspondante, vous devrez donc adapter aussi ces valeurs mises à l'échelle.

Mise en cascade

PID_Temp vous assiste lorsque vous l'utilisez dans une régulation en cascade (voir : Programmation (Page 188)).

Valeur de réglage de remplacement

En cas d'erreur, PID_Temp peut fournir une valeur de réglage de remplacement que vous spécifiez dans la variable SubstituteOutput. La valeur de réglage de remplacement doit se trouver à l'intérieur des limites de la valeur de réglage PID. Les valeurs fournies aux sorties pour le chauffage et le refroidissement et résultant de la valeur de réglage de remplacement résultent de la mise à l'échelle configurée pour la sortie.

Surveiller la validité des signaux

En cas d'utilisation, la validité des valeurs des paramètres suivants est surveillée :

- Setpoint
- SubstituteSetpoint
- Input
- Input_PER
- Disturbance
- ManualValue
- SubstituteOutput
- Paramètres PID dans les structures Retain.CtrlParams.Heat et Retain.CtrlParams.Cool.

Surveillance de la période d'échantillonnage PID_Temp

Dans le cas idéal, la période d'échantillonnage correspond au temps de cycle de l'OB d'alarme cyclique appelant. L'instruction PID_Temp mesure chaque fois l'intervalle de temps entre deux appels. Le résultat est le temps d'échantillonnage actuel. Lors de chaque changement du mode de fonctionnement et à la première mise en route, une moyenne est calculée à partir des 10 premiers temps d'échantillonnage. Quand la période d'échantillonnage actuelle diverge trop de cette valeur moyenne, une erreur apparaît (Error = 0000800h).

Une erreur survient en cours d'optimisation si :

- Nouvelle valeur moyenne >= 1,1 x ancienne valeur moyenne
- Nouvelle valeur moyenne <= 0,9 x ancienne valeur moyenne

Une erreur survient en mode automatique si :

- Nouvelle valeur moyenne >= 1,5 x ancienne valeur moyenne
- Nouvelle valeur moyenne <= 0,5 x ancienne valeur moyenne

Si vous désactivez la surveillance de la période d'échantillonnage (CycleTime.EnMonitoring = FALSE), vous pourrez appeler PID_Temp également dans l'OB1. Dans ce cas, vous devez accepter une moindre qualité de régulation du fait de la fluctuation du temps d'échantillonnage.

Pédiode d'échantillonnage de l'algorithme PID

Comme le système réglé nécessite un certain temps pour réagir à une modification de la valeur de réglage, il est judicieux de ne pas calculer cette valeur à chaque cycle. La période d'échantillonnage de l'algorithme PID est le temps écoulé entre deux calculs de valeur de réglage. Il est déterminé pendant l'optimisation et arrondi à un multiple du temps de cycle de l'OB d'alarme cyclique (période d'échantillonnage PID_Temp). Toutes les autres fonctions de PID_Temp sont exécutées à chaque appel.

Quand le refroidissement et la commutation des paramètres PID sont activés, PID_Temp utilise respectivement une période d'échantillonnage propre de l'algorithme PID pour le chauffage et le refroidissement. Dans toutes les autres configurations, il n'utilise que la période d'échantillonnage de l'algorithme PID pour le chauffage.

Quand vous utilisez OutputHeat_PWM ou OutputCool_PWM, la période d'échantillonnage de l'algorithme PID est utilisée comme période de la modulation de largeur d'impulsion. La précision du signal de sortie est déterminée par le rapport de la période d'échantillonnage de l'algorithme PID au temps de cycle de l'OB. Le temps de cycle ne devrait pas dépasser un dixième de la période d'échantillonnage de l'algorithme PID.

Quand la période d'échantillonnage de l'algorithme PID est très longue avec OutputHeat_PWM ou OutputCool_PWM, ce qui donne une longue période pour la modulation de largeur d'impulsion, vous pouvez spécifier une autre période plus courte aux paramètres Config.Output.Heat.PwmPeriode et Config.Output.Cool.PwmPeriode pour obtenir une mesure plus lissée.

Sens de régulation

PID_Temp peut être utilisé pour les applications de chauffage ou de chauffage/refroidissement et il fonctionne de manière fixe dans le sens de régulation normal.
Une augmentation de la valeur de réglage PID (PidOutputSum) doit provoquer l'augmentation de la mesure. Les valeurs fournies aux sorties pour le chauffage et le refroidissement et résultant de la valeur de réglage PID découlent de la mise à l'échelle configurée pour la sortie.
Une inversion du sens de régulation ou un gain proportionnel négatif ne sont pas pris en charge.
Si votre application nécessite une seule valeur de réglage dont l'augmentation doit provoquer la diminution de la mesure (régulation d'écoulement, par ex.), vous pourrez utiliser PID_Compact avec sens de régulation inversé.

10.3.4.3 Paramètres d'entrée de PID_Temp

Les noms des paramètres suivants sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Paramètre	Type de données	Valeur par défaut	Description
Setpoint	REAL	0.0	Consigne du régulateur PID en mode automatique Plage de valeurs autorisée : Config.SetpointUpperLimit ≥ Setpoint ≥ Config.SetpointLowerLimit Config.InputUpperLimit ≥ Setpoint ≥ Config.InputLowerLimit
Input	REAL	0.0	Une variable du programme utilisateur est utilisée comme source de la mesure. Si vous utilisez le paramètre Input, il faut que Config.InputPerOn = FALSE.
Input_PER	INT	0	Une entrée analogique est utilisée comme source de la mesure. Si vous utilisez le paramètre Input_PER, il faut que Config.InputPerOn = TRUE.
Disturbance	REAL	0.0	Grandeur perturbatrice ou valeur de commande anticipatrice
ManualEnable	BOOL	FALSE	• Un front FALSE -> TRUE active le mode de fonctionnement "Mode manuel", State = 4, Mode ne change pas. Tant que ManualEnable = TRUE, vous ne pouvez pas changer le mode de fonctionnement via un front montant à ModeActivate ni utilisez la boîte de dialogue de mise en service. • Le front FALSE -> TRUE active le mode de fonctionnement prédéfini à Mode. Il est recommandé de changer de mode de fonctionnement uniquement via Mode et ModeActivate.
ManualValue	REAL	0.0	Valeur manuelle Cette valeur est utilisée en mode manuel comme valeur de réglage PID (PidOutputSum). Les valeurs fournies aux sorties pour le chauffage et le refroidissement et résultant de cette valeur manuelle découlent de la mise à l'échelle configurée pour la sortie (structures Config.Output.Heat et Config.Output.Cool). Pour les régulateurs à sortie de refroidissement activée (Config.ActivateCooling = TRUE), vous indiquez : • une valeur manuelle positive pour fournir la valeur aux sorties pour le chauffage • une valeur manuelle négative pour fournir la valeur aux sorties pour le refroidissement

Paramètre	Type de données	Valeur par défaut	Description
			La plage de valeurs admissibles dépend de la configuration : • sortie de refroidissement désactivée (Config.ActivateCooling = FALSE) : Config.Output.Heat.PidUpperLimit ≥ ManualValue ≥ Config.Output.Heat.PidLowerLimit • sortie de refroidissement activée (Config.ActivateCooling = TRUE) : Config.Output.Heat.PidUpperLimit ≥ ManualValue ≥ Config.Output.Cool.PidLowerLimit
ErrorAck	BOOL	FALSE	• Front FALSE -> TRUE ErrorBits et Warning sont remis à zéro
Reset	BOOL	FALSE	Effectue un redémarrage du régulateur. • Front FALSE -> TRUE – passage en mode de fonctionnement "Inactif" – ErrorBits et Warning sont remis à zéro • Tant que Reset = TRUE, – PID_Temp reste en mode de fonctionnement "Inactif" (State = 0) – vous ne pouvez pas changer de mode de fonctionnement avec Mode ni ModeActivate ni ManualEnable – vous ne pouvez pas utiliser le dialogue de mise en service • Front TRUE -> FALSE – Si ManualEnable = FALSE, PID_Temp passe au mode de fonctionnement qui est enregistré dans Mode. – Si Mode = 3 (mode automatique), l'action I est traitée comme configuré avec la variable IntegralResetMode.
ModeActivate	BOOL	FALSE	• Front FALSE -> TRUE PID_Temp passe au mode de fonctionnement qui se trouve à l'entrée Mode.

10.3.4.4 Paramètres de sortie de PID_Temp

Les noms des paramètres suivants sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Paramètre	Type de données	Valeur par défaut	Description
ScaledInput	REAL	0.0	Mesure mise à l'échelle
OutputHeat	REAL	0.0	Valeur de réglage chauffage au format REAL La valeur de réglage PID (PidOutputSum) est mise à l'échelle au moyen des deux paires de valeurs Config.Output.Heat.PidUpperLimit, Config.Output.Heat.UpperScaling et Config.Output.Heat.PidLowerLimit, Config.Output.Heat.LowerScaling et fournie au format REAL à la sortie OutputHeat. OutputHeat est toujours calculée.
OutputCool	REAL	0.0	Valeur de réglage refroidissement au format REAL La valeur de réglage PID (PidOutputSum) est mise à l'échelle au moyen des deux paires de valeurs Config.Output.Cool.PidUpperLimit, Config.Output.Cool.LowerScaling et Config.Output.Cool.PidLowerLimit, Config.Output.Cool.UpperScaling et fournie au format REAL à la sortie OutputCool. OutputCool n'est calculée que si la sortie refroidissement est activée (Config.ActivateCooling = TRUE).

Paramètre	Type de données	Valeur par défaut	Description
OutputHeat_PER	INT	0	Valeur de réglage chauffage analogique La valeur de réglage PID (PidOutputSum) est mise à l'échelle au moyen des deux paires de valeurs Config.Output.Heat.PidUpperLimit, Config.Output.Heat.PerUpperScaling et Config.Output.Heat.PidLowerLimit, Config.Output.Heat.PerLowerScaling et fournie comme valeur analogique à la sortie OutputHeat_PER. OutputHeat_PER n'est calculée que si Config.Output.Heat.Select = 2.
OutputCool_PER	INT	0	Valeur de réglage refroidissement analogique La valeur de réglage PID (PidOutputSum) est mise à l'échelle au moyen des deux paires de valeurs Config.Output.Cool.PidUpperLimit, Config.Output.Cool.PerLowerScaling et Config.Output.Cool.PidLowerLimit, Config.Output.Cool.PerUpperScaling et fournie comme valeur analogique à la sortie OutputCool_PER. OutputCool_PER n'est calculée que si la sortie refroidissement est activée (Config.ActivateCooling = TRUE et Config.Output.Cool.Select = 2).
OutputHeat_PWM	BOOL	FALSE	Valeur de réglage chauffage à modulation de largeur d'impulsion La valeur de réglage PID (PidOutputSum) est mise à l'échelle au moyen des deux paires de valeurs Config.Output.Heat.PidUpperLimit, Config.Output.Heat.PwmUpperScaling et Config.Output.Heat.PidLowerLimit, Config.Output.Heat.PwmLowerScaling et fournie comme valeur à modulation de largeur d'impulsion (temps ON et OFF variables) à la sortie OutputHeat_PWM. OutputHeat_PWM n'est calculée que si Config.Output.Heat.Select = 1.
OutputCool_PWM	BOOL	FALSE	Valeur de réglage refroidissement à modulation de largeur d'impulsion La valeur de réglage PID (PidOutputSum) est mise à l'échelle au moyen des deux paires de valeurs Config.Output.Cool.PidUpperLimit, Config.Output.Cool.PwmLowerScaling et Config.Output.Cool.PidLowerLimit, Config.Output.Cool.PwmUpperScaling et fournie comme valeur à modulation de largeur d'impulsion (temps ON et OFF variables) à la sortie OutputCool_PWM. OutputCool_PWM n'est calculée que si la sortie pour le refroidissement est activée (Config.ActivateCooling = TRUE et Config.Output.Cool.Select = 1).
SetpointLimit_H	BOOL	FALSE	Quand SetpointLimit_H = TRUE, la limite supérieure absolue de la consigne est atteinte (Setpoint ≥ Config.SetpointUpperLimit ou Setpoint ≥ Config.InputUpperLimit). La consigne est limitée vers le haut au minimum de Config.SetpointUpperLimit et Config.InputUpperLimit.
SetpointLimit_L	BOOL	FALSE	Quand SetpointLimit_L = TRUE, la limite inférieure absolue de la consigne est atteinte (Setpoint ≤ Config.SetpointLowerLimit ou Setpoint ≤ Config.InputLowerLimit). La consigne est limitée vers le bas au maximum de Config.SetpointLowerLimit et Config.InputLowerLimit.
InputWarning_H	BOOL	FALSE	Quand InputWarning_H = TRUE, la limite d'alerte supérieure de la mesure est atteinte ou dépassée (ScaledInput ≥ Config.InputUpperWarning).
InputWarning_L	BOOL	FALSE	Quand InputWarning_L = TRUE, la limite d'alerte inférieure de la mesure est atteinte ou dépassée (ScaledInput ≤ Config.InputLowerWarning).

Paramètre	Type de données	Valeur par défaut	Description
State	INT	0	Les Paramètres State et Mode de PID_Temp (Page 405) indiquent le mode de fonctionnement actuel du régulateur PID. Le mode de fonctionnement peut être modifié avec le paramètre d'entrée Mode et un front montant à ModeActivate. Pour l'optimisation préalable et l'optimisation fine, vous déterminez avec Heat.EnableTuning et Cool.EnableTuning si l'opération est effectuée pour le chauffage ou le refroidissement. • State = 0 : Inactif • State = 1 : Optimisation préalable • State = 2 : Optimisation fine • State = 3 : Mode automatique • State = 4 : Mode manuel • State = 5 : Valeur de réglage de remplacement avec surveillance des erreurs
Error	BOOL	FALSE	Quand Error = TRUE, il y a au moins un message d'erreur dans ce cycle.
ErrorBits	DWORD	DW#16#0	Le Paramètre ErrorBits de PID_Temp (Page 411) montre quels messages d'erreur sont présents. ErrorBits est rémanent et il est remis à zéro par un front montant de Reset ou ErrorAck.

10.3.4.5 Paramètres d'entrée/sortie de PID_Temp

Les noms des paramètres suivants sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Paramètre	Type de données	Valeur par défaut	Description
Mode	INT	4	Vous spécifiez par Mode le mode de fonctionnement dans lequel PID_Temp doit passer. Vous avez les possibilités suivantes : • Mode = 0 : Inactif • Mode = 1 : Optimisation préalable • Mode = 2 : Optimisation fine • Mode = 3 : Mode automatique • Mode = 4 : Mode manuel Le mode de fonctionnement est activé par : • Front montant à ModeActivate • Front descendant à Reset • Front descendant à ManualEnable • Démarrage à froid de la CPU, si RunModeByStartup = TRUE Pour l'optimisation préalable et l'optimisation fine, vous déterminez avec Heat.EnableTuning et Cool.EnableTuning si l'opération est effectuée pour le chauffage ou le refroidissement. Mode est rémanent. Vous trouverez une description détaillée des modes de fonctionnement sous Paramètres State et Mode (Page 405).
Master	DWORD	DW#16#0	Interface pour régulation en cascade Quand cette instance de PID_Temp est utilisée comme régulateur esclave dans une cascade (Config.Cascade.IsSlave = TRUE), affectez au paramètre Master dans l'appel de l'instruction le paramètre Slave du régulateur maître. Exemple : Appel d'un régulateur esclave "PID_Temp_2" avec régulateur maître "PID_Temp_1" dans SCL : ----------

Paramètre	Type de données	Valeur par défaut	Description
			`"PID_Temp_2"(Master := "PID_Temp_1".Slave, Setpoint := "PID_Temp_1".OutputHeat);` --- C'est par cette interface que les régulateurs esclaves échangent avec leur régulateur maître des informations sur le mode de fonctionnement, la limitation et la consigne de remplacement. Notez bien que l'appel du régulateur maître doit précéder celui du régulateur esclave dans le même OB d'alarme cyclique. Affectation : • Bits 0 à 15 : non affectés • Bits 16 à 23 - compteur de limitation : Un régulateur esclave dont la valeur de réglage se trouve limitée incrémente ce compteur. Le régulateur maître réagit en conséquence, selon le nombre d'esclaves configuré (Config.Cascade.CountSlaves) et le mode anti-saturation (Config.Cascade.AntiWindUpMode). • Bit 24 – mode automatique des régulateurs esclaves : TRUE quand tous les régulateurs esclaves sont en mode automatique • Bit 25 – consigne de remplacement des régulateurs esclaves : TRUE quand un régulateur esclave a activé la consigne de remplacement (SubstituteSetpointOn = TRUE)
Slave	DWORD	DW#16#0	Interface pour régulation en cascade C'est par cette interface que les régulateurs esclaves échangent avec leur régulateur maître des informations sur le mode de fonctionnement, la limitation et la consigne de remplacement. Voir la description pour le paramètre Master

Voir aussi

Paramètres State et Mode de PID_Temp (Page 405)

Programmation (Page 188)

Fonction cascade avec PID_Temp (Page 186)

10.3.4.6 Variables statiques de PID_Temp

REMARQUE

Faites passer les variables identifiées par (1) seulement en mode de fonctionnement "Inactif" pour éviter un comportement erroné du régulateur PID.

Les noms des variables suivantes sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Variable	Type de données	Valeur par défaut	Description
IntegralResetMode	Int	V1.0 : 1, à partir de V1.1 : 4	La Variable IntegralResetMode (Page 419) définit comment l'action I PIDCtrl.IOutputOld est préréglée lorsque le mode de fonctionnement passe de "Inactif" à "Mode automatique".

Variable	Type de données	Valeur par défaut	Description
			Ce réglage n'agit que pendant un cycle. • IntegralResetMode = 0 : Lissage • IntegralResetMode = 1 : Supprimer • IntegralResetMode = 2 : Conserver • IntegralResetMode = 3 : Valeur par défaut • IntegralResetMode = 4 : Comme variation de la consigne (uniquement pour PID_Temp en version ≥ 1.1)
OverwriteInitialOutputValue	REAL	0.0	Si l'une des conditions suivantes est remplie, l'action par intégration PIDCtrl.IOutputOld est préréglée automatiquement comme si on avait eu PIDOutputSum = OverwriteInitialOutputValue dans le cycle précédent : • IntegralResetMode = 3 lorsque le mode de fonctionnement passe de "Inactif" à "Mode automatique" • IntegralResetMode = 3, Front TRUE -> FALSE sur le paramètre Reset et le paramètre Mode = 3 • PIDCtrl.PIDInit = TRUE en “mode automatique” (disponible à partir de PID_Temp version 1.1)
RunModeByStartup	BOOL	TRUE	Activer le mode de fonctionnement de Mode après le démarrage de la CPU • Quand RunModeByStartup = TRUE, PID_Temp démarre dans le mode de fonctionnement enregistré dans Mode, après la mise en route de la CPU. • Quand RunModeByStartup = FALSE, PID_Temp reste en mode "Inactif" après la mise en route de la CPU.
LoadBackUp	BOOL	FALSE	Quand LoadBackUp = TRUE, le dernier jeu de paramètres PID est rechargé depuis la structure CtrlParamsBackUp. Ce jeu a été enregistré avant la dernière optimisation. LoadBackUp est remis automatiquement à FALSE. L'application des paramètres s'effectue sans à-coup.
SetSubstituteOutput	BOOL	TRUE	Choix de la valeur de réglage tant qu'une erreur est présente (State = 5) : • Quand SetSubstituteOutput = TRUE et ActivateRecoverMode = TRUE, la valeur de réglage de remplacement configurée SubstituteOutput est fournie comme valeur de réglage PID tant qu'une erreur est présente. • Quand SetSubstituteOutput = FALSE et ActivateRecoverMode = TRUE, l'actionneur reste sur la valeur de réglage PID actuelle tant qu'une erreur est présente. • Quand ActivateRecoverMode = FALSE, SetSubstituteOutput n'a pas d'effet. • Quand SubstituteOutput n'est pas valide (ErrorBits = 0020000h), la valeur de réglage de remplacement ne peut pas être fournie. Dans ce cas, c'est la limite inférieure de la valeur de réglage PID pour chauffage (Config.Output.Heat.PidLowerLimit) qui est utilisée comme valeur de réglage PID.
PhysicalUnit	INT	0	Unité physique de la mesure et de la consigne, par ex. °C ou °F. PhysicalUnit sert à afficher les valeurs dans les éditeurs et n'a aucune influence sur le comportement de l'algorithme de régulation dans la CPU. Lors de l'importation de PID_Temp via API Openness, la valeur par défaut de PhysicalUnit est rétablie.

Variable	Type de données	Valeur par défaut	Description
PhysicalQuantity	INT	0	Grandeur physique de la mesure et de la consigne, par ex. température. PhysicalQuantity sert à afficher les valeurs dans les éditeurs et n'a aucune influence sur le comportement de l'algorithme de régulation dans la CPU. Lors de l'importation de PID_Temp via API Openness, la valeur par défaut de PhysicalQuantity est rétablie.
ActivateRecoverMode	BOOL	TRUE	La variable ActivateRecoverMode détermine le comportement en cas d'erreur.
Warning	DWORD	0	La variable Warning montre les avertissements depuis Reset = TRUE ou ErrorAck =TRUE. Warning est rémanente.
Progress	REAL	0.0	Progression de la phase actuelle de l'optimisation en % (0.0 à 100.0)
CurrentSetpoint	REAL	0.0	CurrentSetpoint indique toujours la consigne actuellement opérante. Cette valeur est gelée pendant l'optimisation.
CancelTuningLevel	REAL	10.0	Variation admissible de la consigne pendant l'optimisation. L'optimisation n'est abandonnée que si : • Setpoint > CurrentSetpoint + CancelTuningLevel ou • Setpoint < CurrentSetpoint - CancelTuningLevel
SubstituteOutput	REAL	0.0	La valeur de réglage de remplacement est utilisée comme valeur de réglage PID tant que les conditions suivantes sont remplies : • Il y a une ou plusieurs erreurs en mode automatique pour lesquelles ActivateRecoverMode est opérant • SetSubstituteOutput = TRUE • ActivateRecoverMode = TRUE Les valeurs fournies aux sorties pour chauffage et refroidissement et résultant de la valeur de réglage de remplacement découlent de la mise à l'échelle configurée pour la sortie (structures Config.Output.Heat et Config.Output.Cool). Pour les régulateurs à sortie de refroidissement activée (Config.ActivateCooling = TRUE), vous indiquez : • une valeur de réglage de remplacement positive pour fournir la valeur aux sorties pour le chauffage • une valeur de réglage de remplacement négative pour fournir la valeur aux sorties pour le refroidissement La plage de valeurs admissibles dépend de la configuration : • sortie de refroidissement désactivée (Config.ActivateCooling = FALSE) : Config.Output.Heat.PidUpperLimit ≥ SubstituteOutput ≥ Config.Output.Heat.PidLowerLimit • sortie de refroidissement activée (Config.ActivateCooling = TRUE) : Config.Output.Heat.PidUpperLimit ≥ SubstituteOutput ≥ Config.Output.Cool.PidLowerLimit

Variable	Type de données	Valeur par défaut	Description
PidOutputSum	REAL	0.0	Valeur de réglage PID PidOutputSum indique la valeur de réglage de l'algorithme PID. Selon le mode de fonctionnement, elle est calculée automatiquement ou spécifiée par la valeur manuelle ou par la valeur de réglage de remplacement configurée. Les valeurs fournies aux sorties pour le chauffage et le refroidissement et résultant de la valeur de réglage PID découlent de la mise à l'échelle configurée pour la sortie (structures Config.Output.Heat et Config.Output.Cool). PidOutputSum est limitée en fonction de la configuration : • sortie de refroidissement désactivée (Config.ActivateCooling = FALSE) : Config.Output.Heat.PidUpperLimit ≥ PidOutputSum ≥ Config.Output.Heat.PidLowerLimit • sortie de refroidissement activée (ConfigActivateCooling = TRUE) : Config.Output.Heat.PidUpperLimit ≥ PidOutputSum ≥ Config.Output.Cool.PidLowerLimit
PidOutputOffsetHeat	REAL	0.0	Décalage de la valeur de réglage PID (chauffage) PidOutputOffsetHeat est ajouté à la valeur qui résulte de PidOutputSum pour la branche de chauffage. Spécifiez une valeur positive pour PidOutputOffsetHeat afin d'obtenir un décalage positif aux sorties pour le chauffage. Les valeurs en résultant aux sorties pour le chauffage découlent de la mise à l'échelle configurée pour la sortie (structure Config.Output.Heat). Vous pouvez utiliser ce décalage pour les actionneurs qui nécessitent une valeur minimum fixe, comme les ventilateurs à vitesse minimum.
PidOutputOffsetCool	REAL	0.0	Décalage de la valeur de réglage PID (refroidissement) PidOutputOffsetCool est ajouté à la valeur qui résulte de PidOutputSum pour la branche de refroidissement. Spécifiez une valeur négative pour PidOutputOffsetCool afin d'obtenir un décalage positif aux sorties pour le refroidissement. Les valeurs en résultant aux sorties pour le refroidissement découlent de la mise à l'échelle configurée pour la sortie (structure Config.Output.Cool). Vous pouvez utiliser ce décalage pour les actionneurs qui nécessitent une valeur minimum fixe, comme les ventilateurs à vitesse minimum.
SubstituteSetpointOn	BOOL	FALSE	Active la consigne de remplacement comme consigne du régulateur. • FALSE = le paramètre Setpoint est utilisé. • TRUE = le paramètre SubstituteSetpoint est utilisé comme consigne SubstituteSetpointOn peut servir à spécifier directement la consigne d'un régulateur esclave dans une cascade, sans devoir modifier le programme utilisateur.

Variable	Type de données	Valeur par défaut	Description
SubstituteSetpoint	REAL	0.0	Consigne de remplacement Quand SubstituteSetpointOn = TRUE, c'est la consigne de remplacement SubstituteSetpoint qui est utilisée comme consigne. Plage de valeurs autorisée : Config.SetpointUpperLimit ≥ SubstituteSetpoint ≥ Config.SetpointLowerLimit, Config.InputUpperLimit ≥ SubstituteSetpoint ≥ Config.InputLowerLimit
DisableCooling	BOOL	FALSE	DisableCooling = TRUE désactive la branche de refroidissement pour les régulateurs chauffage/refroidissement (Config.ActivateCooling = TRUE) en mode automatique en limitant PidOutputSum à 0.0 comme limite inférieure. PidOutputOffsetCool et la mise à l'échelle pour les sorties de refroidissement restent actives. DisableCooling peut servir à optimiser des applications multizones en désactivant momentanément la branche de refroidissement tant que tous les régulateurs n'ont pas achevé leur optimisation. Ce paramètre est mis à 1 et à 0 manuellement par l'utilisateur et n'est pas remis à 0 automatiquement par l'instruction PID_Temp.
AllSlaveAutomaticState	BOOL	FALSE	Quand cette instance de PID_Temp est utilisée comme régulateur maître d'une cascade (Config.Cascade.IsMaster = TRUE), AllSlaveAutomaticState = TRUE indique que tous les régulateurs esclaves sont en mode automatique. L'optimisation. le mode manuel ou le mode automatique du régulateur maître ne peuvent être exécutés correctement que si tous les régulateurs esclaves sont en mode automatique. AllSlaveAutomaticState n'est déterminé que si vous interconnectez le régulateur maître et les régulateurs esclaves au moyen des paramètres Master et Slave. Pour des informations détaillées, voir le paramètre Master.
NoSlaveSubstituteSetpoint	BOOL	FALSE	Quand cette instance de PID_Temp est utilisée comme régulateur maître d'une cascade (Config.Cascade.IsMaster = TRUE), NoSlaveSubstituteSetpoint = TRUE indique qu'aucun régulateur esclave n'a activé sa consigne de remplacement. L'optimisation, le mode manuel ou le mode automatique du régulateur maître ne peuvent être exécutés correctement que si aucun régulateur esclave n'a activé sa consigne de remplacement. NoSlaveSubstituteSetpoint n'est déterminé que si vous interconnectez le régulateur maître et les régulateurs esclaves au moyen des paramètres Master et Slave. Pour des informations détaillées, voir le paramètre Master.
Heat.EnableTuning	BOOL	TRUE	Validation de l'optimisation pour le chauffage Heat.EnableTuning doit être mis à 1 pour les optimisations suivantes (en même temps que ou avant le démarrage avec Mode et ModeActivate) : • Optimisation préalable chauffage • Optimisation préalable chauffage et refroidissement • Optimisation fine chauffage Ce paramètre n'est pas remis à 0 automatiquement par l'instruction PID_Temp.

Variable	Type de données	Valeur par défaut	Description
Cool.EnableTuning	BOOL	FALSE	Validation de l'optimisation pour le refroidissement Cool.EnableTuning doit être mis à 1 pour les optimisations suivantes (en même temps que ou avant le démarrage avec Mode et ModeActivate) : • Optimisation préalable refroidissement • Optimisation préalable chauffage et refroidissement • Optimisation fine refroidissement N'est opérant que si la sortie de refroidissement et la commutation des paramètres PID sont activées ("Config.ActivateCooling" = TRUE et "Config.AdvancedCooling" = TRUE). Ce paramètre n'est pas remis à 0 automatiquement par l'instruction PID_Temp.
Config.InputPerOn(1)	BOOL	TRUE	Quand InputPerOn = TRUE, c'est le paramètre Input_PER qui est utilisé pour l'acquisition de la mesure. Quand InputPerOn = FALSE, c'est le paramètre Input qui est utilisé.
Config.InputUpperLimit(1)	REAL	120.0	Limite supérieure de la mesure Le respect de cette limite est surveillé pour Input et Input_PER. Quand la limite est dépassée, une erreur est émise et la réaction dépend de ActivateRecoverMode. A l'entrée de périphérie, la mesure peut dépasser la plage nominale de 18 % au plus (dépassement haut). La limite ne peut donc pas se trouver dépassée si vous utilisez l'entrée de périphérie avec la valeur par défaut de la limite supérieure et la mise à l'échelle de la mesure. Au démarrage d'une optimisation préalable, le système contrôle, au moyen de la différence entre les limites supérieure et inférieure de la mesure, si l'écart entre consigne et mesure répond aux conditions nécessaires. InputUpperLimit > InputLowerLimit
Config.InputLowerLimit(1)	REAL	0.0	Limite inférieure de la mesure Le respect de cette limite est surveillé pour Input et Input_PER. Quand la limite est dépassée par le bas, une erreur est émise et la réaction dépend de ActivateRecoverMode. InputLowerLimit < InputUpperLimit
Config.InputUpperWarning(1)	REAL	3.402822e+-38	Limite d'alerte supérieure de la mesure Le respect de cette limite est surveillé pour Input et Input_PER. Quand la limite est dépassée, un avertissement est émis avec le paramètre Warning. • Si vous configurez InputUpperWarning en dehors des limites de la mesure, la limite supérieure absolue configurée pour la mesure sera utilisée comme limite d'alerte supérieure. • Si vous configurez InputUpperWarning dans les limites de la mesure, cette valeur sera utilisée comme limite d'alerte supérieure. InputUpperWarning > InputLowerWarning
Config.InputLowerWarning(1)	REAL	-3.402822e +38	Limite d'alerte inférieure de la mesure

Variable	Type de données	Valeur par défaut	Description
			Le respect de cette limite est surveillé pour Input et Input_PER. Quand la limite est dépassée par le bas, un avertissement est émis avec le paramètre Warning. • Si vous configurez InputLowerWarning en dehors des limites de la mesure, la limite inférieure absolue configurée pour la mesure sera utilisée comme limite d'alerte inférieure. • Si vous configurez InputLowerWarning dans les limites de la mesure, cette valeur sera utilisée comme limite d'alerte inférieure. InputLowerWarning < InputUpperWarning
Config.SetpointUpperLimit(1)	REAL	3.402822e+38	Limite supérieure de la consigne Le respect de cette limite est surveillé pour Setpoint et SubstituteSetpoint. Quand la limite est dépassée, un avertissement est émis avec le paramètre Warning. • Si vous configurez SetpointUpperLimit en dehors des limites de la mesure, la limite supérieure absolue configurée pour la mesure sera utilisée comme limite supérieure de la consigne. • Si vous configurez SetpointUpperLimit dans les limites de la mesure, cette valeur sera utilisée comme limite supérieure de la consigne. SetpointUpperLimit > SetpointLowerLimit
Config.SetpointLowerLimit(1)	REAL	-3.402822e+38	Limite inférieure de la consigne Le respect de cette limite est surveillé pour Setpoint et SubstituteSetpoint. Quand la limite est dépassée par le bas, un avertissement est émis avec le paramètre Warning. • Si vous configurez SetpointLowerLimit en dehors des limites de la mesure, la limite inférieure absolue configurée pour la mesure sera utilisée comme limite inférieure de la consigne. • Si vous configurez SetpointLowerLimit dans les limites de la mesure, cette valeur sera utilisée comme limite inférieure de la consigne. SetpointLowerLimit < SetpointUpperLimit
Config.ActivateCooling(1)	BOOL	FALSE	Activer la sortie de refroidissement • Config.ActivateCooling = FALSE Seules les sorties pour le chauffage sont utilisées. • Config.ActivateCooling = TRUE Les sorties pour le chauffage et le refroidissement sont utilisées. Quand vous utilisez la sortie de refroidissement, il ne faut pas que le régulateur soit configuré comme régulateur maître (Config.Cascade.IsMaster doit être FALSE) .
Config.AdvancedCooling(1)	BOOL	TRUE	Méthode de chauffage/refroidissement • Facteur pour refroidissement (Config.AdvancedCooling = FALSE) La valeur de réglage pour le refroidissement est calculée avec les paramètres PID pour le chauffage (structure Retain.CtrlParams.Heat) en tenant compte du facteur configurable pour le refroidissement Config.CoolFactor.

Variable	Type de données	Valeur par défaut	Description
			Cette méthode convient quand l'actionneur de chauffage et l'actionneur de refroidissement ont des comportements dans le temps semblables, mais des gains différents. Avec cette méthode, l'optimisation préalable et l'optimisation fine ne sont pas disponibles pour le refroidissement. Seules les optimisations pour le chauffage peuvent être exécutées. • Commutation des paramètres PID (Config.AdvancedCooling = TRUE) La valeur de réglage pour le refroidissement est calculée au moyen de son propre jeu de paramètres PID (structure Retain.CtrlParams.Cool). Cette méthode convient quand l'actionneur de chauffage et l'actionneur de refroidissement ont des comportements dans le temps et des gains différents. L'optimisation préalable et l'optimisation fine pour refroidissement ne sont disponibles qu'avec cette méthode (Mode = 1 ou 2, Cool.EnableTuning = TRUE). Config.AdvancedCooling n'est opérant que si la sortie de refroidissement est activée (Config.ActivateCooling = TRUE).
Config.CoolFactor(1)	REAL	1.0	Facteur pour refroidissement Quand Config.AdvancedCooling = FALSE, Config.CoolFactor est pris en considération comme facteur dans le calcul de la valeur de réglage pour le refroidissement. Ceci permet de tenir compte des gains différents des actionneurs pour le chauffage et pour le refroidissement. Config.CoolFactor n'est pas réglé automatiquement ni adapté pendant l'optimisation. Vous devez donner manuellement à Config.CoolFactor la valeur correcte qui est le rapport "gain de l'actionneur chauffage / gain de l'actionneur refroidissement". Exemple : Config.CoolFactor = 2.0 signifie que le gain de l'actionneur chauffage est le double de celui de l'actionneur refroidissement. Config.CoolFactor n'est opérant que si la sortie de refroidissement est activée (Config.ActivateCooling = TRUE) et le facteur de refroidissement choisi comme méthode de chauffage/refroidissement (Config.AdvancedCooling = FALSE). Config.CoolFactor > 0.0
Config.InputScaling.UpperPointIn(1)	REAL	27648.0	Mise à l'échelle Input_PER haut Input_PER est mis à l'échelle au moyen des deux paires de valeurs UpperPointOut, UpperPointIn et LowerPointOut, LowerPointIn. Opérant seulement quand Input_PER est utilisé pour l'acquisition de la mesure (Config.InputPerOn = TRUE). UpperPointIn > LowerPointIn
Config.InputScaling.LowerPointIn(1)	REAL	0.0	Mise à l'échelle Input_PER bas Input_PER est mis à l'échelle au moyen des deux paires de valeurs UpperPointOut, UpperPointIn et LowerPointOut, LowerPointIn. Opérant seulement quand Input_PER est utilisé pour l'acquisition de la mesure (Config.InputPerOn = TRUE). LowerPointIn < UpperPointIn

Variable	Type de données	Valeur par défaut	Description
Config.InputScaling.UpperPointOut (1)	REAL	100.0	Mesure supérieure à l'échelle Input_PER est mis à l'échelle au moyen des deux paires de valeurs UpperPointOut, UpperPointIn et LowerPointOut, LowerPointIn. Opérant seulement quand Input_PER est utilisé pour l'acquisition de la mesure (Config.InputPerOn = TRUE). UpperPointOut > LowerPointOut
Config.InputScaling.LowerPointOut (1)	REAL	0.0	Mesure inférieure à l'échelle Input_PER est mis à l'échelle au moyen des deux paires de valeurs UpperPointOut, UpperPointIn et LowerPointOut, LowerPointIn. Opérant seulement quand Input_PER est utilisé pour l'acquisition de la mesure (Config.InputPerOn = TRUE). LowerPointOut < UpperPointOut
Config.Output.Heat.Select(1)	INT	1	Choix de la valeur de réglage pour le chauffage Config.Output.Heat.Select spécifie quelles sorties sont utilisées pour le chauffage : • Heat.Select = 0 - OutputHeat est utilisée • Heat.Select = 1 - OutputHeat et OutputHeat_PWM sont utilisées • Heat.Select = 2 - OutputHeat et OutputHeat_PER sont utilisées Les sorties non utilisées ne sont pas calculées et restent à leur valeur par défaut.
Config.Output.Heat.PwmPeriode(1)	REAL	0.0	Période de la modulation de largeur d'impulsion (PWM) pour chauffage (sortie OutputHeat_PWM) en secondes : • Heat.PwmPeriode = 0.0 La période d'échantillonnage de l'algorithme PID pour le chauffage (Retain.CtrlParams.Heat.Cycle) est utilisée comme période de la modulation de largeur d'impulsion. • Heat.PwmPeriode > 0.0 La valeur est arrondie à un multiple entier de la période d'échantillonnage PID_Temp (CycleTime.Value) et utilisée comme période de la modulation de largeur d'impulsion. Ce réglage permet d'obtenir une mesure plus lisse quand la période d'échantillonnage de l'algorithme PID est très longue. La valeur doit remplir les conditions suivantes : – Heat.PwmPeriode ≤ Retain.CtrlParams.Heat.Cycle, – Heat.PwmPeriode > Config.Output.Heat.MinimumOnTime – Heat.PwmPeriode > Config.Output.Heat.MinimumOffTime
Config.Output.Heat.PidUpperLimit(1)	REAL	100.0	Limite supérieure de la valeur de réglage PID pour le chauffage La valeur de réglage PID (PidOutputSum) est limitée à cette limite supérieure. Heat.PidUpperLimit constitue une paire de valeurs avec les paramètres suivants pour la mise à l'échelle de la valeur de réglage PID (PidOutputSum) sur les sorties pour le chauffage : • Heat.UpperScaling pour OutputHeat • Heat.PwmUpperScaling pour OutputHeat_PWM • Heat.PerUpperScaling pour OutputHeat_PER

Variable	Type de données	Valeur par défaut	Description
			Si vous voulez limiter la valeur à la sortie correspondante, vous devrez adapter aussi ces valeurs mises à l'échelle. Heat.PidUpperLimit > Heat.PidLowerLimit
Config.Output.Heat.PidLowerLimit(-1)	REAL	0.0	Limite inférieure de la valeur de réglage PID pour le chauffage Pour les régulateurs avec sortie de refroidissement **désactivée** (Config.ActivateCooling = FALSE), la valeur de réglage PID (PidOutputSum) est limitée à cette limite inférieure. Pour les régulateurs avec sortie de refroidissement **activée** (Config.ActivateCooling = TRUE), la valeur doit être 0.0. Heat.PidLowerLimit constitue une paire de valeurs avec les paramètres suivants pour la mise à l'échelle de la valeur de réglage PID (PidOutputSum) sur les sorties pour le chauffage : • Heat.LowerScaling pour OutputHeat • Heat.PwmLowerScaling pour OutputHeat_PWM • Heat.PerLowerScaling pour OutputHeat_PER Si vous voulez limiter la valeur à la sortie correspondante, vous devrez adapter aussi ces valeurs mises à l'échelle. La plage de valeurs admissibles dépend de la configuration : • sortie de refroidissement désactivée (Config.ActivateCooling = FALSE) : Heat.PidLowerLimit < Heat.PidUpperLimit • sortie de refroidissement activée (Config.ActivateCooling = TRUE) : Heat.PidLowerLimit = 0.0
Config.Output.Heat.UpperScaling(1)	REAL	100.0	Valeur de réglage supérieure mise à l'échelle pour le chauffage Heat.UpperScaling et Heat.PidUpperLimit constituent une paire de valeurs pour la mise à l'échelle de la valeur de réglage PID (PidOutputSum) sur la valeur de réglage pour le chauffage (OutputHeat). La valeur de OutputHeat est toujours comprise entre Heat.UpperScaling et Heat.LowerScaling. Heat.UpperScaling ≠ Heat.LowerScaling
Config.Output.Heat.LowerScaling(1)	REAL	0.0	Valeur de réglage inférieure mise à l'échelle pour le chauffage Heat.LowerScaling et Heat.PidLowerLimit constituent une paire de valeurs pour la mise à l'échelle de la valeur de réglage PID (PidOutputSum) sur la valeur de réglage pour le chauffage (OutputHeat). La valeur de OutputHeat est toujours comprise entre Heat.UpperScaling et Heat.LowerScaling. Heat.UpperScaling ≠ Heat.LowerScaling
Config.Output.Heat.PwmUpperScaling(1)	REAL	100.0	Valeur de réglage PWM supérieure mise à l'échelle pour le chauffage Heat.PwmUpperScaling et Heat.PidUpperLimit constituent une paire de valeurs pour la mise à l'échelle de la valeur de réglage PID (PidOutputSum) sur la valeur de réglage à modulation de largeur d'impulsion pour le chauffage (OutputHeat_PWM). La valeur de OutputHeat_PWM est toujours comprise entre Heat.PwmUpperScaling et Heat.PWMLowerScaling. Heat.PwmUpperScaling n'est opérant que si OutputHeat_PWM est choisi comme sortie pour le chauffage (Heat.Select = 1) 100.0 ≥ Heat.PwmUpperScaling ≥ 0.0 Heat.PwmUpperScaling ≠ Heat.PwmLowerScaling

Variable	Type de données	Valeur par défaut	Description
Config.Output.Heat.PwmLowerScaling[1]	REAL	0.0	Valeur de réglage PWM inférieure mise à l'échelle pour le chauffage Heat.PwmLowerScaling et Heat.PidLowerLimit constituent une paire de valeurs pour la mise à l'échelle de la valeur de réglage PID (PidOutputSum) sur la valeur de réglage à modulation de largeur d'impulsion pour chauffage (OutputHeat_PWM). La valeur de OutputHeat_PWM est toujours comprise entre Heat.PwmUpperScaling et Heat.PwmLowerScaling. Heat.PwmLowerScaling n'est opérant que si OutputHeat_PWM est choisi comme sortie pour le chauffage (Heat.Select = 1) 100.0 ≥ Heat.PwmLowerScaling ≥ 0.0 Heat.PwmUpperScaling ≠ Heat.PwmLowerScaling
Config.Output.Heat.PerUpperScaling[1]	REAL	27648.0	Valeur de réglage analogique supérieure mise à l'échelle pour le chauffage Heat.PerUpperScaling et Heat.PidUpperLimit constituent une paire de valeurs pour la mise à l'échelle de la valeur de réglage PID (PidOutputSum) sur la valeur de réglage analogique pour chauffage (OutputHeat_PER). La valeur de OutputHeat_PER est toujours comprise entre Heat.PerUpperScaling et Heat.PerLowerScaling. Heat.PerUpperScaling n'est opérant que si OutputHeat_PER est choisi comme sortie pour le chauffage (Heat.Select = 2) 32511.0 ≥ Heat.PerUpperScaling ≥ -32512.0 Heat.PerUpperScaling ≠ Heat.PerLowerScaling
Config.Output.Heat.PerLowerScaling[1]	REAL	0.0	Valeur de réglage analogique inférieure mise à l'échelle pour le chauffage Heat.PerLowerScaling et Heat.PidLowerLimit constituent une paire de valeurs pour la mise à l'échelle de la valeur de réglage PID (PidOutputSum) sur la valeur de réglage analogique pour le chauffage (OutputHeat_PER). La valeur de OutputHeat_PER est toujours comprise entre Heat.PerUpperScaling et Heat.PerLowerScaling. Heat.PerLowerScaling n'est opérant que si OutputHeat_PER est choisi comme sortie pour le chauffage (Heat.Select = 2) 32511.0 ≥ Heat.PerLowerScaling ≥ -32512.0 Heat.PerUpperScaling ≠ Heat.PerLowerScaling
Config.Output.Heat.MinimumOnTime[1]	REAL	0.0	Temps ON minimal de la modulation de largeur d'impulsion pour le chauffage (sortie OutputHeat_PWM) Une impulsion PWM n'est jamais plus courte que cette valeur. La valeur est arrondie à : Heat.MinimumOnTime = n × CycleTime.Value Heat.MinimumOnTime n'est opérant que si OutputHeat_PWM est choisi comme sortie pour le chauffage (Heat.Select = 1). 100000.0 ≥ Heat.MinimumOnTime ≥ 0.0
Config.Output.Heat.MinimumOffTime[1]	REAL	0.0	Temps OFF minimal de la modulation de largeur d'impulsion pour le chauffage (sortie OutputHeat_PWM) Une pause PWM n'est jamais plus courte que cette valeur. La valeur est arrondie à : Heat.MinimumOffTime = n × CycleTime.Value Heat.MinimumOffTime n'est opérant que si OutputHeat_PWM est choisi comme sortie pour le chauffage (Heat.Select = 1). 100000.0 ≥ Heat.MinimumOffTime ≥ 0.0

Variable	Type de données	Valeur par défaut	Description
Config.Output.Cool.Select(1)	INT	1	Choix de la valeur de réglage pour le refroidissement Config.Output.Cool.Select spécifie quelles sorties sont utilisées pour le refroidissement : • Cool.Select = 0 - OutputCool est utilisée • Cool.Select = 1 - OutputCool et OutputCool_PWM sont utilisées • Cool.Select = 2 - OutputCool et OutputCool_PER sont utilisées Les sorties non utilisées ne sont pas calculées et restent à leur valeur par défaut. N'est opérant que si la sortie de refroidissement est activée (Config.ActivateCooling = TRUE).
Config.Output.Cool.PwmPeriode(1)	REAL	0.0	Période de la modulation de largeur d'impulsion pour le refroidissement (sortie OutputCool_PWM) en secondes : • Cool.PwmPeriode = 0.0 et Config.AdvancedCooling = FALSE La période d'échantillonnage de l'algorithme PID pour le chauffage (Retain.CtrlParams.Heat.Cycle) est utilisée comme période de la modulation de largeur d'impulsion. • Cool.PwmPeriode = 0.0 et Config.AdvancedCooling = TRUE La période d'échantillonnage de l'algorithme PID pour le refroidissement (Retain.CtrlParams.Cool.Cycle) est utilisée comme période de la modulation de largeur d'impulsion. • Cool.PwmPeriode > 0.0 : La valeur est arrondie à un multiple entier de la période d'échantillonnage PID_Temp (CycleTime.Value) et utilisée comme période de la modulation de largeur d'impulsion. Ce réglage permet d'obtenir une mesure plus lisse quand la période d'échantillonnage de l'algorithme PID est très longue. La valeur doit remplir les conditions suivantes : – Cool.PwmPeriode ≤ Retain.CtrlParams.Cool.Cycle ou Retain.CtrlParams.Heat.Cycle – Cool.PwmPeriode > Config.Output.Cool.MinimumOnTime – Cool.PwmPeriode > Config.Output.Cool.MinimumOffTime N'est opérant que si la sortie de refroidissement est activée (Config.ActivateCooling = TRUE).
Config.Output.Cool.PidUpperLimit(1)	REAL	0.0	Limite supérieure de la valeur de réglage PID pour le refroidissement La valeur doit être 0.0. Cool.PidUpperLimit constitue une paire de valeurs avec les paramètres suivants pour la mise à l'échelle de la valeur de réglage PID (PidOutputSum) sur les sorties pour le refroidissement : • Cool.LowerScaling pour OutputCool • Cool.PwmLowerScaling pour OutputCool_PWM • Cool.PerLowerScaling pour OutputCool_PER Si vous voulez limiter la valeur à la sortie correspondante, vous devrez adapter aussi ces valeurs mises à l'échelle. N'est opérant que si la sortie de refroidissement est activée (Config.ActivateCooling = TRUE).

Variable	Type de données	Valeur par défaut	Description
			Cool.PidUpperLimit = 0.0
Config.Output.Cool.PidLowerLimit(1)	REAL	-100.0	Limite inférieure de la valeur de réglage PID pour le refroidissement Pour les régulateurs avec sortie de refroidissement activée (Config.ActivateCooling = TRUE), la valeur de réglage PID (PidOutputSum) est limitée à cette limite inférieure. Cool.PidLowerLimit constitue une paire de valeurs avec les paramètres suivants pour la mise à l'échelle de la valeur de réglage PID (PidOutputSum) sur les sorties pour le refroidissement : • Cool.UpperScaling pour OutputCool • Cool.PwmUpperScaling pour OutputCool_PWM • Cool.PerUpperScaling pour OutputCool_PER Si vous voulez limiter la valeur à la sortie correspondante, vous devrez adapter aussi ces valeurs mises à l'échelle. N'est opérant que si la sortie de refroidissement est activée (Config.ActivateCooling = TRUE). Cool.PidLowerLimit < Cool.PidUpperLimit
Config.Output.Cool.UpperScaling(1)	REAL	100.0	Valeur de réglage supérieure mise à l'échelle pour le refroidissement Cool.UpperScaling et Cool.PidLowerLimit constituent une paire de valeurs pour la mise à l'échelle de la valeur de réglage PID (PidOutputSum) sur la valeur de réglage pour le refroidissement (OutputCool). La valeur de OutputCool est toujours comprise entre Cool.UpperScaling et Cool.LowerScaling. N'est opérant que si la sortie de refroidissement est activée (Config.ActivateCooling = TRUE). Cool.UpperScaling ≠ Cool.LowerScaling
Config.Output.Cool.LowerScaling(1)	REAL	0.0	Valeur de réglage inférieure mise à l'échelle pour le refroidissement Cool.LowerScaling et Cool.PidUpperLimit constituent une paire de valeurs pour la mise à l'échelle de la valeur de réglage PID (PidOutputSum) sur la valeur de réglage pour le refroidissement (OutputCool). La valeur de OutputCool est toujours comprise entre Cool.UpperScaling et Cool.LowerScaling. N'est opérant que si la sortie de refroidissement est activée (Config.ActivateCooling = TRUE). Cool.UpperScaling ≠ Cool.LowerScaling
Config.Output.Cool.PwmUpperScaling(1)	REAL	100.0	Valeur de réglage PWM supérieure mise à l'échelle pour le refroidissement Cool.PwmUpperScaling et Cool.PidLowerLimit constituent une paire de valeurs pour la mise à l'échelle de la valeur de réglage PID (PidOutputSum) sur la valeur de réglage à modulation de largeur d'impulsion pour le refroidissement (OutputCool_PWM). La valeur de OutputCool_PWM est toujours comprise entre Cool.PwmUpperScaling et Cool.PwmLowerScaling. Cool.PwmUpperScaling n'est opérant que si la sortie de refroidissement est activée (Config.ActivateCooling = TRUE) et que vous avez choisi OutputCool_PWM comme sortie pour le refroidissement (Cool.Select = 1). 100.0 ≥ Cool.PwmUpperScaling ≥ 0.0

Variable	Type de données	Valeur par défaut	Description
			Cool.PwmUpperScaling ≠ Cool.PwmLowerScaling
Config.Output.Cool.PwmLowerScaling(1)	REAL	0.0	Valeur de réglage PWM inférieure mise à l'échelle pour le refroidissement Cool.PwmLowerScaling et Cool.PidUpperLimit constituent une paire de valeurs pour la mise à l'échelle de la valeur de réglage PID (PidOutputSum) sur la valeur de réglage à modulation de largeur d'impulsion pour le refroidissement (OutputCool_PWM). La valeur de OutputCool_PWM est toujours comprise entre Cool.PwmUpperScaling et CoolPwm.LowerScaling. Cool.PwmLowerScaling n'est opérant que si la sortie de refroidissement est activée (Config.ActivateCooling = TRUE) et que vous avez choisi OutputCool_PWM comme sortie pour le refroidissement (Cool.Select = 1). 100.0 ≥ Cool.PwmLowerScaling ≥ 0.0 Cool.PwmUpperScaling ≠ Cool.PwmLowerScaling
Config.Output.Cool.PerUpperScaling(1)	REAL	27648.0	Valeur de réglage analogique supérieure mise à l'échelle pour le refroidissement Cool.PerUpperScaling et Cool.PidLowerLimit constituent une paire de valeurs pour la mise à l'échelle de la valeur de réglage PID (PidOutputSum) sur la valeur de réglage analogique pour le refroidissement (OutputCool_PER). La valeur de OutputCool_PER est toujours comprise entre Cool.PerUpperScaling et Cool.PerLowerScaling. Cool.PerUpperScaling n'est opérant que si la sortie de refroidissement est activée (Config.ActivateCooling = TRUE) et que vous avez choisi OutputCool_PER comme sortie pour le refroidissement (Cool.Select = 2). 32511.0 ≥ Cool.PerUpperScaling ≥ -32512.0 Cool.PerUpperScaling ≠ Cool.PerLowerScaling
Config.Output.Cool.PerLowerScaling(1)	REAL	0.0	Valeur de réglage analogique inférieure mise à l'échelle pour le refroidissement Cool.PerLowerScaling et Cool.PidUpperLimit constituent une paire de valeurs pour la mise à l'échelle de la valeur de réglage PID (PidOutputSum) sur la valeur de réglage analogique pour le refroidissement (OutputCool_PER). La valeur de OutputCool_PER est toujours comprise entre Cool.PerUpperScaling et Cool.PerLowerScaling. Cool.PerLowerScaling n'est opérant que si la sortie de refroidissement est activée (Config.ActivateCooling = TRUE) et que vous avez choisi OutputCool_PER comme sortie pour le refroidissement (Cool.Select = 2). 32511.0 ≥ Cool.PerLowerScaling ≥ -32512.0 Cool.PerUpperScaling ≠ Cool.PerLowerScaling
Config.Output.Cool.MinimumOnTime(1)	REAL	0.0	Temps ON minimal de la modulation de largeur d'impulsion pour le refroidissement (sortie OutputCool_PWM) Une impulsion PWM n'est jamais plus courte que cette valeur. La valeur est arrondie à : Cool.MinimumOnTime = n × CycleTime.Value Cool.MinimumOnTime n'est opérant que si OutputCool_PWM est choisi comme sortie pour le refroidissement (Cool.Select = 1). N'est opérant que si la sortie de refroidissement est activée (Config.ActivateCooling = TRUE).

Variable	Type de données	Valeur par défaut	Description
			100000.0 ≥ Cool.MinimumOnTime ≥ 0.0
Config.Output.Cool.MinimumOff-Time(1)	REAL	0.0	Temps OFF minimal de la modulation de largeur d'impulsion pour refroidissement (sortie OutputCool_PWM) Une pause PWM n'est jamais plus courte que cette valeur. La valeur est arrondie à : Cool.MinimumOffTime = n × CycleTime.Value Cool.MinimumOffTime n'est opérant que si OutputCool_PWM est choisi comme sortie pour le refroidissement (Cool.Select = 1). N'est opérant que si la sortie de refroidissement est activée (Config.ActivateCooling = TRUE). 100000.0 ≥ Cool.MinimumOffTime ≥ 0.0
Quand vous utilisez PID_Temp dans une cascade, le régulateur maître et les régulateurs esclaves échangent des informations au moyen des paramètres Master et Slave. C'est vous qui devez faire l'interconnexion. Pour des informations détaillées, voir le paramètre Master.			
Config.Cascade.IsMaster(1)	BOOL	FALSE	Le régulateur est maître dans une cascade et fournit la consigne pour l'esclave. Mettez IsMaster = TRUE si vous utilisez cette instance de PID_Temp comme régulateur maître dans une cascade. Un régulateur maître spécifie par sa sortie la consigne d'un régulateur esclave. Une instance de PID_Temp peut être simultanément régulateur maître et régulateur esclave. Quand le régulateur est utilisé comme maître, il faut que la sortie de refroidissement soit désactivée (Config.ActivateCooling = FALSE).
Config.Cascade.IsSlave(1)	BOOL	FALSE	Le régulateur est esclave dans une cascade et reçoit sa consigne du maître. Mettez IsSlave = TRUE si vous utilisez cette instance de PID_Temp comme régulateur esclave dans une cascade. Un régulateur esclave reçoit sa consigne (paramètre Setpoint) de la sortie de son régulateur maître (paramètre OutputHeat). Une instance de PID_Temp peut être simultanément régulateur maître et régulateur esclave.
Config.Cascade.AntiWindUpMode(1)	INT	1	Comportement anti-saturation dans la cascade Vous avez les possibilités suivantes : • Anti-saturation = 0 La fonction d'anti-saturation est désactivée. Le régulateur maître ne réagit pas à la limitation de ses régulateurs esclaves. • Anti-saturation = 1 L'action I du régulateur maître est réduite selon le rapport "Esclaves en limitation" sur "Nombre d'esclaves" (paramètre "CountSlaves"). Ceci diminue les effets de la limitation sur le comportement de régulation. • Anti-saturation = 2 L'action I du régulateur maître est arrêtée dès qu'un régulateur esclave se trouve dans la limitation. N'est opérant que si le régulateur est configuré comme maître (Config.Cascade.IsMaster = TRUE).

Variable	Type de données	Valeur par défaut	Description
Config.Cascade.CountSlaves(1)	INT	1	Nombre d'esclaves asservis Indiquez ici le nombre d'esclaves directement asservis qui reçoivent leur consigne de ce régulateur maître. N'est opérant que si le régulateur est configuré comme maître (Config.Cascade.IsMaster = TRUE). 255 ≥ CountSlaves ≥ 1
CycleTime.StartEstimation	BOOL	TRUE	Quand CycleTime.EnEstimation = TRUE, CycleTime.StartEstimation = TRUE démarre le calcul automatique de la période d'échantillonnage PID_Temp (temps de cycle de l'OB appelant). Une fois la mesure terminée, CycleTime.StartEstimation est mis = FALSE.
CycleTime.EnEstimation	BOOL	TRUE	Quand CycleTime.EnEstimation = TRUE, la période d'échantillonnage PID_Temp est calculée automatiquement. Quand CycleTime.EnEstimation = FALSE, la période d'échantillonnage PID_Temp n'est pas calculée automatiquement et vous devez donner à CycleTime.Value la valeur correcte manuellement.
CycleTime.EnMonitoring	BOOL	TRUE	Quand CycleTime.EnMonitoring = FALSE, la période d'échantillonnage PID_Temp n'est pas surveillée. Si PID_Temp ne peut pas être exécuté à l'intérieur de la période d'échantillonnage, une erreur (ErrorBits=0000800h) n'est pas émise et PID_Temp ne réagit pas comme vous l'avez configuré avec ActivateRecoverMode.
CycleTime.Value(1)	REAL	0.1	Période d'échantillonnage de PID_Temp (temps de cycle de l'OB appelant) en secondes CycleTime.Value est automatiquement déterminée et correspond normalement au temps de cycle de l'OB appelant.
Il est possible de recharger les valeurs de la structure CtrlParamsBackUp avec LoadBackUp = TRUE.			
CtrlParamsBackUp.SetByUser	BOOL	FALSE	Valeur enregistrée de Retain.CtrlParams.SetByUser
CtrlParamsBackUp.Heat.Gain	REAL	1.0	Gain proportionnel enregistré pour chauffage
CtrlParamsBackUp.Heat.Ti	REAL	20.0	Temps d'intégration enregistré pour chauffage en secondes
CtrlParamsBackUp.Heat.Td	REAL	0.0	Temps de dérivation enregistré pour chauffage en secondes
CtrlParamsBackUp.Heat.TdFiltRatio	REAL	0.2	Coefficient du délai de dérivation enregistré pour chauffage
CtrlParamsBackUp.Heat.PWeighting	REAL	1.0	Pondération enregistrée de l'action P pour le chauffage
CtrlParamsBackUp.Heat.DWeighting	REAL	1.0	Pondération enregistrée de l'action D pour le chauffage
CtrlParamsBackUp.Heat.Cycle	REAL	1.0	Période d'échantillonnage enregistrée de l'algorithme PID pour le chauffage en secondes
CtrlParamsBackUp.Heat.Control-Zone	REAL	3.402822e+-38	Largeur de zone de régulation enregistrée pour le chauffage
CtrlParamsBackUp.Heat.DeadZone	REAL	0.0	Largeur de zone morte enregistrée pour le chauffage
CtrlParamsBackUp.Cool.Gain	REAL	1.0	Gain proportionnel enregistré pour le refroidissement
CtrlParamsBackUp.Cool.Ti	REAL	20.0	Temps d'intégration enregistré pour le refroidissement en secondes
CtrlParamsBackUp.Cool.Td	REAL	0.0	Temps de dérivation enregistré pour le refroidissement en secondes

Variable	Type de données	Valeur par défaut	Description
CtrlParamsBackUp.Cool.TdFiltRatio	REAL	0.2	Coefficient du délai de dérivation enregistré pour le refroidissement
CtrlParamsBackUp.Cool.PWeighting	REAL	1.0	Facteur de pondération de l'action P enregistré pour le refroidissement
CtrlParamsBackUp.Cool.DWeighting	REAL	1.0	Facteur de pondération de l'action D enregistré pour le refroidissement
CtrlParamsBackUp.Cool.Cycle	REAL	1.0	Période d'échantillonnage enregistrée de l'algorithme PID pour le refroidissement en secondes
CtrlParamsBackUp.Cool.ControlZone	REAL	3.402822e+38	Largeur de zone de régulation enregistrée pour refroidissement
CtrlParamsBackUp.Cool.DeadZone	REAL	0.0	Largeur de zone morte enregistrée pour le refroidissement
PIDSelfTune.SUT.CalculateParamsHeat	BOOL	FALSE	Les propriétés de la branche de chauffage du système réglé sont enregistrées lors de l'optimisation préalable le chauffage. Quand SUT.CalculateParamsHeat = TRUE, les paramètres PID pour le chauffage (structure Retain.CtrlParams.Heat) sont recalculés au moyen de ces propriétés. Cela permet de modifier la méthode de calcul des paramètres (paramètre PIDSelfTune.SUT.TuneRuleHeat) sans répéter l'optimisation. SUT.CalculateParamsHeat est mis sur FALSE après le calcul. N'est possible que si l'optimisation préalable a réussi (SUT.ProcParHeatOk = TRUE).
PIDSelfTune.SUT.CalculateParamsCool	BOOL	FALSE	Les propriétés de la branche refroidissement du système réglé sont enregistrées lors de l'optimisation refroidissement. Quand SUT.CalculateParamsCool = TRUE, les paramètres PID pour le refroidissement (structure Retain.CtrlParams.Cool) sont recalculés au moyen de ces propriétés. Cela permet de modifier la méthode de calcul des paramètres (paramètre PIDSelfTune.SUT.TuneRuleCool) sans répéter l'optimisation. SUT.CalculateParamsCool est mis sur FALSE après le calcul. N'est possible que si l'optimisation préalable a réussi (SUT.ProcParCoolOk = TRUE). N'est opérant que si Config.ActivateCooling = TRUE et Config.AdvancedCooling = TRUE.
PIDSelfTune.SUT.TuneRuleHeat	INT	2	Méthode pour le calcul des paramètres PID lors de l'optimisation préalable chauffage Vous avez les possibilités suivantes : • SUT.TuneRuleHeat = 0 : PID selon CHR • SUT.TuneRuleHeat = 1 : PI selon CHR • SUT.TuneRuleHeat = 2 : PID pour procédés de température selon CHR (donne un comportement de régulation plus lent et plutôt asymptotique avec suroscillation plus faible que SUT.TuneRuleHeat = 0) (CHR = Chien, Hrones et Reswick) C'est seulement avec SUT.TuneRuleHeat = 2 que la zone de régulation Retain.CtrlParams.Heat.ControlZone est réglée automatiquement lors de l'optimisation préalable chauffage.

Variable	Type de données	Valeur par défaut	Description
PIDSelfTune.SUT.TuneRuleCool	INT	2	Méthode pour le calcul des paramètres PID lors de l'optimisation préalable refroidissement Vous avez les possibilités suivantes : • SUT.TuneRuleCool = 0 : PID selon CHR • SUT.TuneRuleCool = 1 : PI selon CHR • SUT.TuneRuleCool = 2 : PID pour procédés de température selon CHR (donne un comportement de régulation plus lent et plutôt asymptotique avec suroscillation plus faible que SUT.TuneRuleCool = 0) (CHR = Chien, Hrones et Reswick) C'est seulement avec SUT.TuneRuleCool = 2 que la zone de régulation Retain.CtrlParams.Cool.ControlZone est réglée automatiquement lors de l'optimisation préalable refroidissement. SUT.TuneRuleCool n'est opérant que si la sortie de refroidissement et la commutation des paramètres PID sont activées (Config.ActivateCooling = TRUE, Config.AdvancedCooling = TRUE).
PIDSelfTune.SUT.State	INT	0	La variable SUT.State indique la phase actuelle de l'optimisation préalable : • State = 0 : Initialiser l'optimisation préalable • State = 100 : calculer l'écart type pour le chauffage • State = 200 : calculer l'écart type pour le refroidissement • State = 300 : déterminer le point d'inflexion pour le chauffage • State = 400 : déterminer le point d'inflexion pour le refroidissement • State = 500 : chauffer à la consigne une fois le point d'inflexion pour le chauffage atteint • State = 600 : chauffer à la consigne une fois le point d'inflexion pour le refroidissement atteint • State = 700 : comparer l'effet de l'actionneur chauffage et de l'actionneur refroidissement • State = 800 : chauffage et refroidissement activés • State = 900 : refroidissement activé • State = 1000 : déterminer le retard après coupure du chauffage • State = 9900 : Optimisation préalable réussie • State = 1 : Optimisation préalable échouée
PIDSelfTune.SUT.ProcParHeatOk	BOOL	FALSE	TRUE : le calcul des paramètres de processus pour l'optimisation préalable chauffage a réussi. Cette variable est mise à 1 pendant l'optimisation. Elle doit être TRUE pour le calcul des paramètres PID pour le chauffage.
PIDSelfTune.SUT.ProcParCoolOk	BOOL	FALSE	TRUE : le calcul des paramètres de processus pour l'optimisation préalable refroidissement a réussi. Cette variable est mise à 1 pendant l'optimisation. Elle doit être TRUE pour le calcul des paramètres PID pour le refroidissement.

Variable	Type de données	Valeur par défaut	Description
PIDSelfTune.SUT.AdaptDelayTime	INT	0	La variable AdaptDelayTime détermine l'adaptation du retard pour le chauffage au point de fonctionnement (pour "optimisation préalable chauffage" et "optimisation préalable chauffage et refroidissement"). Vous avez les possibilités suivantes : • SUT.AdaptDelayTime = 0 : pas d'adaptation du retard. La phase SUT.State = 1000 est sautée. Cette option donne une durée d'optimisation plus courte qu'avec SUT.AdaptDelayTime = 1. • SUT.AdaptDelayTime = 1 : adaptation du retard à la consigne dans la phase SUT.State = 1000 par coupure passagère du chauffage. Cette option donne une durée d'optimisation plus longue qu'avec SUT.AdaptDelayTime = 0. Elle peut améliorer le comportement de régulation si le comportement du processus dépend fortement du point de fonctionnement (non linéaire). Il convient de ne pas l'utiliser pour les applications multi-zones à forts couplages thermiques.
PIDSelfTune.SUT.CoolingMode	INT	0	La variable CoolingMode détermine la sortie de valeur de réglage pour le calcul des paramètres de refroidissement (lors de l'optimisation préalable chauffage et refroidissement). Vous avez les possibilités suivantes : • SUT.CoolingMode = 0 : couper le chauffage et mettre en marche le refroidissement une fois la consigne atteinte. La phase SUT.State = 700 est sautée. La phase SUT.State = 500 est suivie par la phase SUT.State = 900. Cette option peut améliorer le comportement de régulation quand le gain de l'actionneur pour le refroidissement est faible comparé à celui de l'actionneur pour le chauffage. Elle donne une durée d'optimisation plus courte qu'avec SUT.CoolingMode = 1 ou 2. • SUT.CoolingMode = 1 : mettre en marche le refroidissement en plus du chauffage une fois la consigne atteinte. La phase SUT.State = 700 est sautée. La phase SUT.State = 500 est suivie par la phase SUT.State = 800. Cette option peut améliorer le comportement de régulation quand le gain de l'actionneur pour le refroidissement est fort comparé à celui de l'actionneur pour le chauffage. • SUT.CoolingMode = 2 : une fois la consigne atteinte par chauffage, la décision de couper ou pas le chauffage est prise automatiquement dans la phase SUT.State = 700. La phase SUT.State = 500 est suivie par la phase SUT.State = 700, puis SUT.State = 800 ou SUT.State = 900. Cette option demande plus de temps que les options 0 et 1.

Variable	Type de données	Valeur par défaut	Description
PIDSelfTune.TIR.RunIn	BOOL	FALSE	Avec la variable RunIn, vous pouvez fixer le déroulement de l'optimisation fine en cas de démarrage depuis le mode automatique. • RunIn = FALSE Quand l'optimisation fine est démarrée depuis le mode automatique, les paramètres PID existants sont utilisés pour une régulation à la consigne (TIR.State = 500 ou 600). C'est seulement après cela que l'optimisation fine commence. • RunIn = TRUE PID_Temp tente d'atteindre la consigne avec la valeur de réglage minimale ou maximale (TIR.State = 300 ou 400). Cela peut entraîner une suroscillation élevée. L'optimisation fine démarre ensuite automatiquement. RunIn est mis sur FALSE après l'optimisation fine. Lorsque l'optimisation fine est démarrée depuis le mode Inactif ou depuis le mode manuel, PID_Temp se comporte comme il est décrit sous RunIn = TRUE.
PIDSelfTune.TIR.CalculateParamsHeat	BOOL	FALSE	Les propriétés de la branche de chauffage du système réglé sont enregistrées lors de l'optimisation fine chauffage. Quand TIR.CalculateParamsHeat= TRUE, les paramètres PID pour le chauffage (structure Retain.CtrlParams.Heat) sont recalculés au moyen de ces propriétés. Cela permet de modifier la méthode de calcul des paramètres (paramètre PIDSelfTune.TIR.TuneRuleHeat) sans répéter l'optimisation. TIR.CalculateParamsHeat est mis sur FALSE après le calcul. N'est possible que si l'optimisation fine chauffage a réussi auparavant (TIR.ProcParHeatOk = TRUE).
PIDSelfTune.TIR.CalculateParamsCool	BOOL	FALSE	Les propriétés de la branche refroidissement du système réglé sont enregistrées lors de l'optimisation fine refroidissement. Quand TIR.CalculateParamsCool = TRUE, les paramètres PID pour le refroidissement (structure Retain.CtrlParams.Cool) sont recalculés au moyen de ces propriétés. Cela permet de modifier la méthode de calcul des paramètres (paramètre PIDSelfTune.TIR.TuneRuleCool) sans répéter l'optimisation. TIR.CalculateParamsCool est mis sur FALSE après le calcul. N'est possible que si l'optimisation fine refroidissement a réussi auparavant (TIR.ProcParCoolOk = TRUE). N'est opérant que si Config.ActivateCooling = TRUE et Config.AdvancedCooling = TRUE.
PIDSelfTune.TIR.TuneRuleHeat	INT	0	Méthode pour le calcul des paramètres pendant l'optimisation fine chauffage Vous avez les possibilités suivantes : • TIR.TuneRuleHeat = 0 : PID automatique • TIR.TuneRuleHeat = 1 : PID rapide (comportement de régulation plus rapide avec des amplitudes plus fortes de la valeur de réglage qu'avec TIR.TuneRuleHeat = 2) • TIR.TuneRuleHeat = 2 : PID lent (comportement de régulation plus lent avec des amplitudes plus faibles de la valeur de réglage qu'avec TIR.TuneRuleHeat = 1) • TIR.TuneRuleHeat = 3 : PID ZN • TIR.TuneRuleHeat = 4 : PI ZN • TIR.TuneRuleHeat = 5 : P ZN (ZN = Ziegler-Nichols)

Variable	Type de données	Valeur par défaut	Description
			Pour pouvoir répéter le calcul des paramètres PID pour le chauffage avec TIR.CalculateParamsHeat et TIR.TuneRuleHeat = 0, 1 ou 2, il faut avoir exécuté l'optimisation fine précédente également avec TIR.TuneRuleHeat = 0, 1 ou 2. Si ce n'est pas le cas, TIR.TuneRuleHeat = 3 sera utilisé. Il est toujours possible de recalculer les paramètres PID pour chauffage avec TIR.CalculateParamsHeat et TIR.TuneRuleHeat = 3, 4 ou 5.
PIDSelfTune.TIR.TuneRuleCool	INT	0	Méthode pour le calcul des paramètres pendant l'optimisation fine refroidissement Vous avez les possibilités suivantes : • TIR.TuneRuleCool = 0 : PID automatique • TIR.TuneRuleCool = 1 : PID rapide (comportement de régulation plus rapide avec des amplitudes plus fortes de la valeur de réglage qu'avec TIR.TuneRuleCool = 2) • TIR.TuneRuleCool = 2 : PID lent (comportement de régulation plus lent avec des amplitudes plus faibles de la valeur de réglage qu'avec TIR.TuneRuleCool = 1) • TIR.TuneRuleCool = 3 : PID ZN • TIR.TuneRuleCool = 4 : PI ZN • TIR.TuneRuleCool = 5 : P ZN (ZN = Ziegler-Nichols) Pour pouvoir répéter le calcul des paramètres PID pour refroidissement avec TIR.CalculateParamsCool et TIR.TuneRuleCool = 0, 1 ou 2, il faut avoir exécuté l'optimisation fine précédente également avec TIR.TuneRuleCool = 0, 1 ou 2. Si ce n'est pas le cas, TIR.TuneRuleCool = 3 sera utilisé. Il est toujours possible de recalculer les paramètres PID pour le refroidissement avec TIR.CalculateParamsCool et TIR.TuneRuleCool = 3, 4 ou 5. N'est opérant que si la sortie de refroidissement et la commutation des paramètres PID sont activées (ConfigActivateCooling = TRUE et Config.AdvancedCooling = TRUE).
PIDSelfTune.TIR.State	INT	0	La variable TIR.State indique la phase actuelle de l'"optimisation fine" : • State = 0 : Initialiser l'optimisation fine • State = 100 : Calculer l'écart type pour le chauffage • State = 200 : Calculer l'écart type pour le refroidissement • State = 300 : Tenter d'atteindre la consigne de chauffage par régulation à 2 échelons • State = 400 : Tenter d'atteindre la consigne de refroidissement par régulation à 2 échelons • State = 500 : Tenter d'atteindre la consigne de chauffage par régulation PID • State = 600 : Tenter d'atteindre la consigne de refroidissement par régulation PID • State = 700 : Calculer l'écart type pour le chauffage • State = 800 : Calculer l'écart type pour le refroidissement • State = 900 : Déterminer l'oscillation et calculer les paramètres du chauffage • State = 1000 : Déterminer l'oscillation et calculer les paramètres du refroidissement • State = 9900 : Optimisation fine réussie • State = 1 : Optimisation fine échouée

Variable	Type de données	Valeur par défaut	Description
PIDSelfTune.TIR.ProcParHeatOk	BOOL	FALSE	TRUE : le calcul des paramètres de processus pour l'optimisation fine chauffage a réussi. Cette variable est mise à 1 pendant l'optimisation. Elle doit être vraie pour le calcul des paramètres PID pour le chauffage.
PIDSelfTune.TIR.ProcParCoolOk	BOOL	FALSE	TRUE : le calcul des paramètres de processus pour l'optimisation fine refroidissement a réussi. Cette variable est mise à 1 pendant l'optimisation. Elle doit être vraie pour le calcul des paramètres PID pour le refroidissement.
PIDSelfTune.TIR.OutputOffsetHeat	REAL	0.0	Décalage d'optimisation chauffage de la valeur de réglage PID TIR.OutputOffsetHeat est ajouté à la valeur qui résulte de PidOutputSum pour la branche chauffage. Spécifiez une valeur positive pour TIR.OutputOffsetHeat afin d'obtenir un décalage positif aux sorties pour chauffage. Les valeurs en résultant aux sorties pour le chauffage découlent de la mise à l'échelle configurée pour la sortie (Struktur Config.Output.Heat). Ce décalage d'optimisation peut servir à l'optimisation fine refroidissement pour les régulateurs dont la sortie de refroidissement et la commutation des paramètres PID sont activées (Config.ActivateCooling = TRUE, Config.AdvancedCooling = TRUE). Quand les sorties de refroidissement ne sont pas actives à proximité de la consigne pour laquelle l'optimisation doit être effectuée (PidOutputSum > 0.0), l'optimisation fine refroidissement n'est pas possible. Spécifiez dans ce cas, avant de démarrer l'optimisation, un décalage d'optimisation chauffage positif plus grand que la valeur de réglage PID (PidOutputSum) à proximité de la consigne en état stationnaire. De cette manière, les valeurs aux sorties de chauffage se trouveront augmentées et les sorties de refroidissement actives (PidOutputSum < 0.0). L'optimisation fine refroidissement sera donc possible. Quand l'optimisation fine est terminée, TIR.OutputOffsetHeat est remis à 0.0. De grandes modifications de TIR.OutputOffsetHeat en une fois peuvent provoquer des suroscillations passagères. Config.Output.Heat.PidUpperLimit ≥ PIDSelfTune.TIR.OutputOffsetHeat ≥ Config.Output.Heat.PidLowerLimit
PIDSelfTune.TIR.OutputOffsetCool	REAL	0.0	Décalage d'optimisation refroidissement de la valeur de réglage PID TIR.OutputOffsetCool est ajouté à la valeur qui résulte de PidOutputSum pour la branche refroidissement. Spécifiez une valeur négative pour TIR.OutputOffsetCool afin d'obtenir un décalage positif aux sorties pour le refroidissement. Les valeurs en résultant aux sorties pour le refroidissement découlent de la mise à l'échelle configurée pour la sortie (Struktur Config.Output.Coool). Ce décalage d'optimisation peut servir à l'optimisation fine chauffage pour les régulateurs à sortie de refroidissement activée (Config.ActivateCooling). Quand les sorties de chauffage ne sont pas actives à proximité de la consigne pour laquelle l'optimisation doit être effectuée (PidOutputSum < 0.0), l'optimisation fine chauffage n'est pas possible. Spécifiez dans ce

Variable	Type de données	Valeur par défaut	Description
			cas, avant de démarrer l'optimisation, un décalage d'optimisation refroidissement négatif plus petit que la valeur de réglage PID (PidOutputSum) à proximité de la consigne en état stationnaire. De cette manière, les valeurs aux sorties de refroidissement se trouveront augmentées et les sorties de chauffage actives (PidOutputSum > 0.0). L'optimisation fine chauffage sera donc possible. Quand l'optimisation fine est terminée, TIR.OutputOffsetCool est remis à 0.0. De grandes modifications de TIR.OutputOffsetCool en une fois peuvent provoquer des suroscillations passagères. Config.Output.Cool.PidUpperLimit ≥ PIDSelfTune.TIR.OutputOffsetCool ≥ Config.Output.Cool.PidLowerLimit
PIDSelfTune.TIR.WaitForControlIn	BOOL	FALSE	Attente lors de l'optimisation fine une fois la consigne atteinte Quand TIR.WaitForControlIn = TRUE, l'optimisation fine observe un temps d'attente entre l'arrivée à la consigne (TIR.State = 500 ou 600) et le calcul de l'écart type (TIR.State = 700 ou 800) jusqu'à ce qu'un front FALSE -> TRUE soit spécifié sur TIR.FinishControlIn. TIR.WaitForControlIn peut servir à synchroniser les optimisations des différentes zones lors de l'optimisation fine simultanée de plusieurs régulateurs dans des applications multi-zones. Ceci permet de garantir que toutes les zones ont atteint leurs consignes avant que l'optimisation proprement dite démarre. Les couplages thermiques entre les zones ont ainsi moins d'influence sur l'optimisation. TIR.WaitForControlIn n'est opérant que si l'optimisation fine est démarrée depuis le mode automatique avec PIDSelfTune.TIR.RunIn = FALSE.
PIDSelfTune.TIR.ControlInReady	BOOL	FALSE	Quand TIR.WaitForControlIn = TRUE, PID_Temp met TIR.ControlInReady = TRUE dès que la consigne est atteinte et attend qu'un front FALSE -> TRUE soit spécifié sur TIR.FinishControlIn pour exécuter les étapes suivantes de l'optimisation.
PIDSelfTune.TIR.FinishControlIn	BOOL	FALSE	Quand TIR.ControlInReady = TRUE, un front FALSE -> TRUE sur TIR.FinishControlIn met fin à l'attente et l'optimisation fine se poursuit.
PIDCtrl.IOutputOld(1)	REAL	0.0	Action I dans le dernier cycle
PIDCtrl.PIDInit	BOOL	FALSE	PIDCtrl.PIDInit est disponible à partir de PID_Temp version 1.1. Si en "mode automatique" PIDCtrl.PIDInit = TRUE, l'action intégrale PIDCtrl.IOutputOld est préréglée automatiquement comme si on avait eu PidOutputSum = OverwriteInitialOutputValue dans le cycle précédent. Cela est utilisable pour une Régulation en mode alternatif avec PID_Temp (Page 195).
Retain.CtrlParams.SetByUser(1)	BOOL	FALSE	Activer la saisie manuelle des paramètres PID Quand Retain.CtrlParams.SetByUser = TRUE, les paramètres PID sont éditables. Retain.CtrlParams.SetByUser sert à configurer le régulateur dans TIA Portal et n'a aucune influence sur le comportement de l'algorithme de régulation dans la CPU. SetByUser est rémanent.

Variable	Type de données	Valeur par défaut	Description
Retain.CtrlParams.Heat.Gain(1)	REAL	1.0	Gain proportionnel actif pour le chauffage Heat.Gain est rémanent. Heat.Gain ≥ 0.0
Retain..CtrlParams.Heat.Ti(1)	REAL	20.0	Temps d'intégration actif pour le chauffage en secondes Avec Heat.CtrlParams.Ti = 0.0, l'action I pour le chauffage est désactivée. Heat.Ti est rémanent. 100000.0 ≥ Heat.Ti ≥ 0.0
Retain.CtrlParams.Heat.Td(1)	REAL	0.0	Temps de dérivation actif pour le chauffage en secondes Avec Heat.CtrlParams.Td = 0.0, l'action D pour le chauffage est désactivée. Heat.Td est rémanent. 100000.0 ≥ Heat.Td ≥ 0.0
Retain.CtrlParams.Heat.TdFiltRatio(1)	REAL	0.2	Coefficient du délai de dérivation actif pour le chauffage L'effet de l'action D est retardé par le coefficient de l'action par dérivation. Délai de dérivation = temps de dérivation x coefficient du délai de dérivation • 0.0 : l'action D n'agit que pour un cycle et est donc quasiment sans effet. • 0.5 : Cette valeur a fait ses preuves dans la pratique pour les systèmes réglés avec **une** constante de temps dominante. • > 1.0 : plus le coefficient est grand, plus l'effet de l'action D est retardé. Heat.TdFiltRatio est rémanent. Heat.TdFiltRatio ≥ 0.0
Retain.CtrlParams.Heat.PWeighting(1)	REAL	1.0	Pondération active de l'action P pour le chauffage Vous pouvez atténuer l'action P pour les changements de consigne. Les valeurs comprises entre 0.0 et 1.0 sont judicieuses. • 1.0 : action P totalement opérante quand la consigne change • 0.0 : action P non opérante quand la consigne change L'action P est toujours totalement opérante quand la mesure change. Heat.PWeighting est rémanent. 1.0 ≥ Heat.PWeighting ≥ 0.0
Retain.CtrlParams.Heat.DWeighting(1)	REAL	1.0	Pondération active de l'action D pour chauffage Vous pouvez atténuer l'action D pour les changements de consigne. Les valeurs comprises entre 0.0 et 1.0 sont judicieuses. • 1.0 : action D totalement opérante quand la consigne change • 0.0 : action D non opérante quand la consigne change L'action D est toujours totalement opérante quand la mesure change. Heat.DWeighting est rémanent. 1.0 ≥ Heat.DWeighting ≥ 0.0

Variable	Type de données	Valeur par défaut	Description
Retain.CtrlParams.Heat.Cycle(1)	REAL	1.0	Période d'échantillonnage active de l'algorithme PID pour le chauffage en secondes CtrlParams.Heat.Cycle est déterminé pendant l'optimisation et arrondi à un multiple entier de CycleTime.Value. Quand Config.Output.Heat.PwmPeriode = 0.0, Heat.Cycle est utilisé comme période de la modulation de largeur d'impulsion pour le chauffage. Quand Config.Output.Cool.PwmPeriode = 0.0 et Config.AdvancedCooling = FALSE, Heat.Cycle est utilisé comme période de la modulation de largeur d'impulsion pour le refroidissement. Heat.Cycle est rémanent. 100000.0 ≥ Heat.Cycle > 0.0
Retain.CtrlParams.Heat.ControlZone (1)	REAL	3.402822e+-38	Largeur de zone de régulation active pour le chauffage Avec Heat.ControlZone = 3.402822e+38, la zone de régulation pour le chauffage est désactivée. Heat.ControlZone n'est réglée automatiquement que pendant l'optimisation préalable chauffage ou l'optimisation préalable chauffage et refroidissement, si PIDSelfTune.SUT.TuneRuleHeat = 2 est choisi comme méthode de calcul des paramètres. Pour les régulateurs avec sortie de refroidissement désactivée (Config.ActivateCooling = FALSE) ou les régulateurs avec sortie de refroidissement activée et facteur de refroidissement (Config.AdvancedCooling = FALSE), la zone de régulation se trouve symétriquement entre Setpoint − Heat.ControlZone et Setpoint + Heat.ControlZone. Pour les régulateurs dont la sortie de refroidissement et la commutation des paramètres PID sont activées (Config.ActivateCooling = TRUE, Config.AdvancedCooling = TRUE), la zone de régulation se trouve entre Setpoint − Heat.ControlZone et Setpoint + Cool.ControlZone. Heat.ControlZone est rémanent. Heat.ControlZone > 0.0
Retain.CtrlParams.Heat.DeadZone(1)	REAL	0.0	Largeur de zone morte active pour chauffage (voir Paramètres PID (Page 170)) Avec Heat.DeadZone = 0.0, la zone morte pour le chauffage est désactivée. Heat.DeadZone n'est pas réglé automatiquement ni adapté pendant l'optimisation. Vous devez donner à Heat.DeadZone une valeur correcte manuellement. Lorsque la zone morte est activée, un signal d'écart (écart entre consigne et mesure) permanent peut s'établir. Cela peut avoir un effet négatif sur l'exécution d'une optimisation fine. Pour les régulateurs avec sortie de refroidissement désactivée (Config.ActivateCooling = FALSE) ou les régulateurs avec sortie de refroidissement activée et facteur de refroidissement (Config.AdvancedCooling = FALSE), la zone morte se trouve symétriquement entre Setpoint − Heat.DeadZone et Setpoint + Heat.DeadZone. Pour les régulateurs dont la sortie de refroidissement et la commutation des paramètres PID sont activées (Config.ActivateCooling = TRUE, Config.AdvancedCooling = TRUE), la zone morte se trouve entre Setpoint − Heat.DeadZone et Setpoint + Cool.DeadZone. Heat.DeadZone est rémanent.

Variable	Type de données	Valeur par défaut	Description
			Heat.DeadZone ≥ 0.0
Retain.CtrlParams.Cool.Gain(1)	REAL	1.0	Gain proportionnel actif pour le refroidissement Cool.Gain est rémanent. N'est opérant que si la sortie de refroidissement et la commutation des paramètres PID sont activées (Config.ActivateCooling = TRUE, Config.AdvancedCooling = TRUE). Cool.Gain ≥ 0.0
Retain.CtrlParams.Cool.Ti(1)	REAL	20.0	Temps d'intégration actif pour le refroidissement en secondes Avec Cool.CtrlParams.Ti = 0.0, l'action I pour le refroidissement est désactivée. Cool.Ti est rémanent. N'est opérant que si la sortie de refroidissement et la commutation des paramètres PID sont activées (Config.ActivateCooling = TRUE, Config.AdvancedCooling = TRUE). 100000.0 ≥ Cool.Ti ≥ 0.0
Retain.CtrlParams.Cool.Td(1)	REAL	0.0	Temps de dérivation actif pour le refroidissement en secondes Avec Cool.CtrlParams.Td = 0.0, l'action D pour le refroidissement est désactivée. Cool.Td est rémanent. N'est opérant que si la sortie de refroidissement et la commutation des paramètres PID sont activées (Config.ActivateCooling = TRUE, Config.AdvancedCooling = TRUE). 100000.0 ≥ Cool.Td ≥ 0.0
Retain.CtrlParams.Cool.TdFiltRatio(1)	REAL	0.2	Coefficient du délai de dérivation actif pour le refroidissement L'effet de l'action D est retardé par le coefficient de l'action par dérivation. Délai de dérivation = temps de dérivation x coefficient du délai de dérivation • 0.0 : l'action D n'agit que pour un cycle et est donc quasiment sans effet. • 0.5 : cette valeur a fait ses preuves dans la pratique pour les systèmes réglés avec une constante de temps dominante. • > 1.0 : plus le coefficient est grand, plus l'effet de l'action D est retardé. Cool.TdFiltRatio est rémanent. N'est opérant que si la sortie de refroidissement et la commutation des paramètres PID sont activées (Config.ActivateCooling = TRUE, Config.AdvancedCooling = TRUE). Cool.TdFiltRatio ≥ 0.0
Retain.CtrlParams.Cool.PWeighting(1)	REAL	1.0	Pondération active de l'action P pour le refroidissement Vous pouvez atténuer l'action P pour les changements de consigne. Les valeurs comprises entre 0.0 et 1.0 sont judicieuses. • 1.0 : action P totalement opérante quand la consigne change • 0.0 : action P non opérante quand la consigne change L'action P est toujours totalement opérante quand la mesure change. Cool.PWeighting est rémanent. N'est opérant que si la sortie de refroidissement et la commutation des paramètres PID sont activées (Config.ActivateCooling = TRUE, Config.AdvancedCooling = TRUE). 1.0 ≥ Cool.PWeighting ≥ 0.0

Variable	Type de données	Valeur par défaut	Description
Retain.CtrlParams.Cool.DWeighting (1)	REAL	1.0	Pondération active de l'action D pour le refroidissement Vous pouvez atténuer l'action D pour les changements de consigne. Les valeurs comprises entre 0.0 et 1.0 sont judicieuses. • 1.0 : action D totalement opérante quand la consigne change • 0.0 : action D non opérante quand la consigne change L'action D est toujours totalement opérante quand la mesure change. Cool.DWeighting est rémanent. N'est opérant que si la sortie de refroidissement et la commutation des paramètres PID sont activées (Config.ActivateCooling = TRUE, Config.AdvancedCooling = TRUE). 1.0 ≥ Cool.DWeighting ≥ 0.0
Retain.CtrlParams.Cool.Cycle(1)	REAL	1.0	Période d'échantillonnage active de l'algorithme PID pour le refroidissement en secondes CtrlParams.Cool.Cycle est déterminé pendant l'optimisation et arrondi à un multiple entier de CycleTime.Value. Quand Config.Output.Cool.PwmPeriode = 0.0 et Config.AdvancedCooling = TRUE, Cool.Cycle est utilisé comme période de la modulation de largeur d'impulsion pour le refroidissement. Quand Config.Output.Cool.PwmPeriode = 0.0 et Config.AdvancedCooling = FALSE, Heat.Cycle est utilisé comme période de la modulation de largeur d'impulsion pour le refroidissement. Cool.Cycle est rémanent. N'est opérant que si la sortie de refroidissement et la commutation des paramètres PID sont activées (Config.ActivateCooling = TRUE, Config.AdvancedCooling = TRUE). 100000.0 ≥ Cool.Cycle > 0.0
Retain.CtrlParams.Cool.ControlZone (1)	REAL	3.402822e+-38	Largeur de zone de régulation active pour le refroidissement Avec Cool.ControlZone = 3.402822e+38, la zone de régulation pour le refroidissement est désactivée. Cool.ControlZone n'est réglée automatiquement que pendant l'optimisation préalable refroidissement ou l'optimisation préalable chauffage et refroidissement, si PIDSelfTune.SUT.TuneRuleCool = 2 est choisi comme méthode de calcul des paramètres. Cool.ControlZone est rémanent. N'est opérant que si la sortie de refroidissement et la commutation des paramètres PID sont activées (Config.ActivateCooling = TRUE, Config.AdvancedCooling = TRUE). Cool.ControlZone > 0.0
Retain.CtrlParams.Cool.DeadZone(1)	REAL	0.0	Largeur de zone morte active pour le refroidissement (voir Paramètres PID (Page 170)) Avec Cool.DeadZone = 0.0, la zone morte pour le refroidissement est désactivée. Cool.DeadZone n'est pas réglé automatiquement ni adapté pendant l'optimisation. Vous devez donner à Cool.DeadZone une valeur correcte manuellement. Lorsque la zone morte est activée, un signal d'écart (écart entre consigne et mesure) permanent peut s'établir. Cela peut avoir un effet négatif sur l'exécution d'une optimisation fine. Cool.DeadZone est rémanent.

Variable	Type de données	Valeur par défaut	Description
			N'est opérant que si la sortie de refroidissement et la commutation des paramètres PID sont activées (Config.ActivateCooling = TRUE, Config.AdvancedCooling = TRUE). Cool.DeadZone ≥ 0.0

Voir aussi

Variable ActivateRecoverMode de PID_Temp (Page 414)
Variable Warning de PID_Temp (Page 415)
Réglage multi-zones avec PID_Temp (Page 192)

10.3.4.7 Paramètres State et Mode de PID_Temp

Corrélation entre les paramètres

Le paramètre State affiche le mode de fonctionnement actuel du régulateur PID. Vous ne pouvez pas modifier le paramètre State.

Avec un front montant à ModeActivate, PID_Temp passe au mode de fonctionnement enregistré dans le paramètre d'entrée/sortie Mode.

Heat.EnableTuning et Cool.EnableTuning déterminent pour l'optimisation préalable et pour l'optimisation fine si l'opération est effectuée pour le chauffage ou pour le refroidissement.

Quand la CPU est mise en circuit ou passe d'ARRET à MARCHE, PID_Temp démarre dans le mode de fonctionnement enregistré dans Mode. Pour laisser PID_Temp en mode "Inactif", mettez RunModeByStartup = FALSE.

Signification des valeurs

State / Mode	Description du mode de fonctionnement
0	**Inactif** En mode de fonctionnement "Inactif", les valeurs de réglage suivantes sont fournies : • 0.0 comme valeur de réglage PID (PidOutputSum) • 0.0 comme valeur de réglage pour le chauffage (OutputHeat) et valeur de réglage pour refroidissement (OutputCool) • 0 comme valeur de réglage analogique pour le chauffage (OutputHeat_PER) et valeur de réglage analogique pour refroidissement (OutputCool_PER) • FALSE comme valeur de réglage PWM pour le chauffage (OutputHeat_PWM) et valeur de réglage PWM pour refroidissement (OutputCool_PWM) Ceci indépendamment des limites de valeur de réglage et de la mise à l'échelle configurées dans les structures Config.Output.Heat et Config.Output.Cool.
1	**Optimisation préalable** L'optimisation préalable détermine la réponse du processus à un échelon de la valeur de réglage et recherche le point d'inflexion. Les paramètres PID sont calculés à partir de l'incrémentation maximale et du temps mort du système réglé. Les meilleurs paramètres PID sont obtenus pendant l'exécution d'une optimisation préalable et d'une optimisation fine. Selon la configuration, PID_Temp offre différents types d'optimisation préalable : • Optimisation préalable chauffage : Un échelon de la valeur de réglage chauffage est sorti, les paramètres PID pour le chauffage sont calculés (structure Retain.CtrlParams.Heat), ensuite la régulation à la consigne est effectuée en mode automatique.

State / Mode	Description du mode de fonctionnement
	Quand le comportement du processus dépend fortement du point de fonctionnement, il est possible d'activer une adaptation du retard à proximité de la consigne avec PIDSelfTune.SUT.AdaptDelayTime. • Optimisation préalable chauffage et refroidissement : Un échelon de la valeur de réglage chauffage est sorti. Dès que la mesure s'approche de la consigne, un échelon de la valeur de réglage refroidissement est sorti. Les paramètres PID pour le chauffage (structure Retain.CtrlParams.Heat) et refroidissement (structure Retain.CtrlParams.Cool) sont calculés. Ensuite, la régulation à la consigne est effectuée en mode automatique. Quand le comportement du processus dépend fortement du point de fonctionnement, il est possible d'activer une adaptation du retard à proximité de la consigne avec PIDSelfTune.SUT.AdaptDelayTime. Suivant l'effet de l'actionneur de refroidissement par rapport à celui de l'actionneur pour le chauffage, il est possible d'influencer la qualité de l'optimisation en exploitant simultanément ou pas les sorties pour le chauffage et le refroidissement pendant l'optimisation. Vous pouvez déterminer cela avec PIDSelfTune.SUT.CoolingMode. • Optimisation préalable refroidissement : Un échelon de la valeur de réglage refroidissement est sorti et les paramètres PID pour refroidissement sont calculés (Struktur Retain.CtrlParams.Cool). Ensuite, la régulation à la consigne est effectuée en mode automatique. Si vous voulez optimiser les paramètres PID pour le chauffage et le refroidissement, vous obtiendrez un meilleur comportement de régulation en effectuant d'abord une "optimisation préalable chauffage", puis une "optimisation préalable refroidissement" qu'avec une "optimisation préalable chauffage et refroidissement". Mais l'optimisation préalable effectuée en deux étapes prend plus de temps. **Conditions générales pour l'optimisation préalable :** • L'instruction PID_Temp est appelée dans un OB d'alarme cyclique. • Mode inactif (State = 0), manuel (State = 4) ou automatique (State = 3) • ManualEnable = FALSE • Reset = FALSE • La consigne et la mesure se trouvent dans les limites configurées. **Conditions pour l'optimisation préalable chauffage :** • Heat.EnableTuning = TRUE • Cool.EnableTuning = FALSE • La mesure ne doit pas être trop proche de la consigne. |Setpoint - Input| > 0.3 * |Config.InputUpperLimit - Config.InputLowerLimit| et |Setpoint - Input| > 0.5 * |Setpoint| • La consigne est supérieure à la mesure. Setpoint > Input **Conditions pour l'optimisation préalable chauffage et refroidissement :** • Heat.EnableTuning = TRUE· • Cool.EnableTuning = TRUE • La sortie de refroidissement est activée (Config.ActivateCooling = TRUE).· • La commutation des paramètres PID est activée (Config.AdvancedCooling = TRUE). • La mesure ne doit pas être trop proche de la consigne. |Setpoint - Input| > 0.3 * |Config.InputUpperLimit - Config.InputLowerLimit| et |Setpoint - Input| > 0.5 * |Setpoint| • La consigne est supérieure à la mesure. Setpoint > Input **Conditions pour l'optimisation préalable refroidissement :** • Heat.EnableTuning = FALSE· • Cool.EnableTuning = TRUE· • La sortie de refroidissement est activée (Config.ActivateCooling = TRUE). • La commutation des paramètres PID est activée (Config.AdvancedCooling = TRUE).

State / Mode	Description du mode de fonctionnement
	• Une "optimisation préalable chauffage" ou une "optimisation préalable chauffage et refroidissement" a été effectuée avec succès (PIDSelfTune.SUT.ProcParHeatOk = TRUE), möglichst am gleichen Sollwert. • La mesure doit être proche de la consigne. |Setpoint - Input| < 0.05 * |Config.InputUpperLimit - Config.InputLowerLimit| Plus la mesure est stable, plus il sera facile de déterminer des paramètres PID précis. Un bruit de la mesure est acceptable tant que la croissance de la mesure est nettement supérieure au bruit. Les modes de fonctionnement "Inactif" ou "Mode manuel" garantissent cela avec le plus de vraisemblance. La consigne est gelée dans la variable CurrentSetpoint. L'optimisation est abandonnée quand : • Setpoint > CurrentSetpoint + CancelTuningLevel ou • Setpoint < CurrentSetpoint - CancelTuningLevel Avec PIDSelfTune.SUT.TuneRuleHeat et PIDSelfTune.SUT.TuneRuleCool, vous pouvez définir séparément la méthode de calcul des paramètres PID pour le chauffage et le refroidissement. Avant que les paramètres PID soient recalculés, ils sont sauvegardés dans la structure CtrlParamsBackUp et peuvent être réactivés avec LoadBackUp. Une optimisation préalable réussie est suivie d'un passage en mode automatique. Après l'échec d'une optimisation préalable, le changement de mode de fonctionnement dépend de ActivateRecoverMode. La phase d'optimisation préalable est indiquée par PIDSelfTune.SUT.State. Pour démarrer une optimisation préalable chauffage ou une optimisation préalable chauffage et refroidissement à partir du mode automatique, il est conseillé d'effectuer la modification nécessaire de la valeur de consigne en même temps que le front montant sur ModeActivate. Si la valeur de consigne est modifiée en premier et que l'optimisation préalable est démarrée ultérieurement, la valeur de réglage est adaptée en conséquence en mode automatique et entraîne une modification de la mesure. Cela peut avoir un effet négatif sur l'optimisation préalable ultérieure ou l'empêcher de démarrer.
2	**Optimisation fine** L'optimisation fine génère une oscillation constante limitée de la mesure. Les paramètres PID sont optimisés, pour le point de fonctionnement, à partir de l'amplitude et de la fréquence de cette oscillation. Les paramètres PID de l'optimisation fine montrent généralement un meilleur comportement de référence et de perturbation que les paramètres PID de l'optimisation préalable. Les meilleurs paramètres PID sont obtenus pendant l'exécution d'une optimisation préalable et d'une optimisation fine. PID_Temp tente automatiquement de générer une oscillation supérieure au bruit de la mesure. La stabilité de la mesure n'influence l'optimisation fine que de manière insignifiante. Selon la configuration, PID_Temp offre différents types d'optimisation fine : • Optimisation fine chauffage : PID_Temp génère une oscillation de la mesure avec des modifications périodiques de la valeur de réglage pour le chauffage et il calcule les paramètres PID pour le chauffage (Struktur Retain.CtrlParams.Heat). • Optimisation fine refroidissement : PID_Temp génère une oscillation de la mesure avec des modifications périodiques de la valeur de réglage pour refroidissement et il calcule les paramètres PID pour le refroidissement (Struktur Retain.CtrlParams.Cool). **Décalage d'optimisation momentané pour régulateurs de chauffage/refroidissement** Quand PID_Temp est utilisé comme régulateur de chauffage/refroidissement (Config.ActivateCooling = TRUE), la valeur de réglage PID (PidOutputSum) doit remplir la condition suivante à proximité de la consigne pour permettre la génération d'une oscillation de la mesure et une optimisation fine correcte : • Valeur de réglage PID positive pour l'optimisation fine chauffage • Valeur de réglage PID négative pour l'optimisation fine refroidissement

State / Mode	Description du mode de fonctionnement
	Quand cette condition n'est pas remplie, vous pouvez spécifier pour l'optimisation fine un décalage momentané qui est fourni à la sortie ayant l'effet contraire : • Décalage pour sortie de refroidissement (PIDSelfTune.TIR.OutputOffsetCool) lors de l'optimisation fine chauffage. Spécifiez, avant de démarrer l'optimisation, un décalage d'optimisation refroidissement négatif plus petit que la valeur de réglage PID (PidOutputSum) à proximité de la consigne en état stationnaire. • Décalage pour sortie de chauffage (PIDSelfTune.TIR.OutputOffsetHeat) lors de l'optimisation fine refroidissement. Spécifiez, avant de démarrer l'optimisation, un décalage d'optimisation chauffage positif plus grand que la valeur de réglage PID (PidOutputSum) à proximité de la consigne en état stationnaire. Le décalage spécifié est alors compensé par l'algorithme PID de sorte que la mesure reste proche de la consigne. Le montant du décalage permet ainsi d'adapter la valeur de réglage PID afin qu'elle remplisse la condition nommée plus haut. Pour éviter de fortes suroscillations de la mesure lorsque vous spécifiez le décalage, vous pouvez aussi augmenter ce dernier en plusieurs fois. Quand PID_Temp quitte le mode de fonctionnement optimisation fine, le décalage d'optimisation est remis à zéro. Exemple de décalage spécifié pour l'optimisation fine refroidissement : • Sans décalage : – consigne (Setpoint) = mesure (ScaledInput) = 80°C – valeur de réglage PID (PidOutputSum) = 30.0 – valeur de réglage chauffage (OutputHeat) = 30.0 – valeur de réglage refroidissement (OutputCool) = 0.0 Il n'est pas possible de générer une oscillation de la mesure autour de la consigne avec la seule sortie de refroidissement. L'optimisation fine serait ici un échec. • Avec décalage spécifié pour la sortie de chauffage (PIDSelfTune.TIR.OutputOffsetHeat) = 80.0 – consigne (Setpoint) = mesure (ScaledInput) = 80°C – valeur de réglage PID (PidOutputSum) = -50.0 – valeur de réglage chauffage (OutputHeat) = 80.0 – valeur de réglage refroidissement (OutputCool) = -50.0 Grâce au décalage spécifié pour la sortie de chauffage, la sortie de refroidissement peut générer à présent une oscillation de la mesure autour de la consigne. L'optimisation fine peut donc être effectuée correctement. **Conditions générales pour l'optimisation fine :** • L'instruction PID_Temp est appelée dans un OB d'alarme cyclique. • Aucune perturbation n'est attendue. • La consigne et la mesure se trouvent dans les limites configurées. • La boucle de régulation est en régime établi au point de fonctionnement. Le point de fonctionnement est atteint lorsque la mesure correspond à la consigne. Lorsque la zone morte est activée, un signal d'écart (écart entre consigne et mesure) permanent peut s'établir. Cela peut avoir un effet négatif sur l'exécution de l'optimisation fine. • ManualEnable = FALSE • Reset = FALSE • Mode automatique (State = 3), inactif (State = 0) ou mode manuel (State = 4) **Conditions pour l'optimisation fine chauffage :** • Heat.EnableTuning = TRUE • Cool.EnableTuning = FALSE • Quand PID_Temp est configuré comme régulateur de chauffage/refroidissement (Config.ActivateCooling = TRUE), il faut que la sortie de chauffage soit active au point de fonctionnement auquel on veut effectuer l'optimisation (PidOutputSum > 0.0 (voir décalage d'optimisation)).

State / Mode	Description du mode de fonctionnement
	Conditions pour l'optimisation fine refroidissement : • Heat.EnableTuning = FALSE • Cool.EnableTuning = TRUE • La sortie de refroidissement est activée (Config.ActivateCooling = TRUE). • La commutation des paramètres PID est activée (Config.AdvancedCooling = TRUE). • La sortie de refroidissement doit être active au point de fonctionnement auquel on veut effectuer l'optimisation (PidOutputSum < 0.0 (voir décalage d'optimisation)). **Le déroulement de l'optimisation fine dépend du mode de fonctionnement depuis lequel elle est démarrée :** • Mode automatique (State = 3) avec PIDSelfTune.TIR.RunIn = FALSE (par défaut) Si vous souhaitez améliorer les paramètres PID existants à l'aide de l'optimisation, démarrez l'optimisation fine à partir du mode automatique. PID_Temp utilise les paramètres PID existants pour la régulation jusqu'à ce que la boucle de régulation soit en régime établi et que les conditions pour une optimisation fine soient remplies. C'est seulement après cela que l'optimisation fine commence. • Mode inactif (State = 0), manuel (State = 4) ou automatique (State = 3) avec PIDSelfTune.TIR.RunIn = TRUE Tentative d'atteindre la consigne avec la valeur de réglage minimale ou maximale : – avec la valeur de réglage minimale ou maximale pour le chauffage pour l'optimisation fine chauffage – avec la valeur de réglage minimale ou maximale pour refroidissement pour l'optimisation fine refroidissement Cela peut entraîner une suroscillation élevée. L'optimisation fine démarre dès que la consigne est atteinte. Quand il n'est pas possible d'atteindre la consigne, PID_Temp n'abandonne pas l'optimisation automatiquement. La consigne est gelée dans la variable CurrentSetpoint. L'optimisation est abandonnée quand : • Setpoint > CurrentSetpoint + CancelTuningLevel ou • Setpoint < CurrentSetpoint - CancelTuningLevel Avec PIDSelfTune.TIR.TuneRuleHeat et PIDSelfTune.TIR.TuneRuleCool, vous pouvez définir séparément la méthode de calcul des paramètres PID pour le chauffage et le refroidissement. Avant que les paramètres PID soient recalculés, ils sont sauvegardés dans la structure CtrlParamsBackUp et peuvent être réactivés avec LoadBackUp. Après une optimisation fine réussie, le régulateur passe en mode automatique. Après l'échec d'une optimisation fine, le changement de mode de fonctionnement dépend de ActivateRecoverMode. La phase d'"optimisation fine" est affichée avec la PIDSelfTune.TIR.State.
3	**Mode automatique** En mode automatique, PID_Temp régule le système réglé en fonction des paramètres prédéfinis. Le système passe en mode automatique quand l'une des conditions suivantes est remplie : • Optimisation préalable terminée correctement • Optimisation fine terminée correctement • Modification du paramètre d'entrée/sortie Mode à la valeur 3 et un front montant à ModeActivate. Le passage du mode automatique en mode manuel s'effectue sans à-coups uniquement dans l'éditeur de mise en service. Le mode automatique tient compte de la variable ActivateRecoverMode.
4	**Mode manuel** En mode manuel, vous spécifiez une valeur de réglage PID manuelle au paramètre ManualValue. Les valeurs fournies aux sorties pour le chauffage et le refroidissement et résultant de cette valeur manuelle découlent de la mise à l'échelle configurée pour la sortie. Ce mode est également activable via ManualEnable = TRUE. Il est recommandé de changer de mode de fonctionnement uniquement via Mode et ModeActivate. Le passage du mode manuel au mode automatique s'effectue sans à-coups. Le mode manuel tient compte de la variable ActivateRecoverMode.

State / Mode	Description du mode de fonctionnement
5	**Valeur de réglage de remplacement avec surveillance des erreurs** L'algorithme de régulation est arrêté. La variable SetSubstituteOutput détermine quelle valeur de réglage PID (PidOutputSum) est fournie pendant ce mode de fonctionnement. • SetSubstituteOutput = FALSE : Dernière valeur de réglage PID valide • SetSubstituteOutput = TRUE : Valeur de réglage de remplacement (SubstituteOutput) Ce mode de fonctionnement ne peut pas être activé avec Mode = 5. Il est activé en cas d'erreur au lieu du mode de fonctionnement "Inactif" si toutes les conditions suivantes sont remplies : • Mode automatique (State = 3) • ActivateRecoverMode = TRUE • Une ou plusieurs erreurs sont apparues pour lesquelles ActivateRecoverMode s'applique. Dès que les erreurs ont disparu, PID_Temp repasse en mode automatique.

Comportement ENO

Quand State = 0, ENO = FALSE.
Quand State ≠ 0, ENO = TRUE.

Changement de mode de fonctionnement automatique pendant la mise en service

Après une optimisation préalable ou fine réussie, le mode automatique est activé. Le tableau suivant indique comment Mode et State évoluent pendant une optimisation préalable réussie.

N° de cycle	Mode	State	Action
0	4	4	Positionner Mode = 1
1	1	4	Positionner ModeActivate = TRUE
1	4	1	La valeur de State est enregistrée dans Mode L'optimisation préalable est lancée
n	4	1	Optimisation préalable terminée avec succès
n	3	3	Le mode automatique est lancé

En cas d'erreur, PID_Temp change automatiquement de mode de fonctionnement.
Le tableau suivant indique comment Mode et State évoluent pendant une optimisation préalable erronée.

N° de cycle	Mode	State	Action
0	4	4	Positionner Mode = 1
1	1	4	Positionner ModeActivate = TRUE
1	4	1	La valeur de State est enregistrée dans Mode L'optimisation préalable est lancée
n	4	1	Optimisation préalable interrompue
n	4	4	Le mode manuel est démarré

Quand ActivateRecoverMode = TRUE, le mode de fonctionnement qui est enregistré dans Mode est activé. Au démarrage de l'optimisation préalable ou fine, PID_Temp a enregistré la

valeur de State dans le paramètre d'entrée/sortie Mode. PID_Temp passe donc dans le mode de fonctionnement à partir duquel l'optimisation a été lancée.
Quand ActivateRecoverMode = FALSE, le système passe en mode de fonctionnement "Inactif".

Voir aussi

Paramètres de sortie de PID_Temp (Page 375)
Paramètres d'entrée/sortie de PID_Temp (Page 377)

10.3.4.8 Paramètre ErrorBits de PID_Temp

En présence de plusieurs erreurs simultanées, les valeurs des ErrorBits s'affichent additionnées en binaire. L'affichage ErrorBits = 16#0000_0003, p. ex., indique la présence simultanée des erreurs 16#0000_0001 et 16#0000_0002.

ErrorBits (DW#16#...)	Description
0000_0000	Aucune erreur.
0000_0001	Le paramètre "Input" se trouve hors des limites de la mesure. • Input > Config.InputUpperLimit ou • Input < Config.InputLowerLimit Si le mode automatique était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp reste en mode automatique. Si le mode manuel était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp reste en mode manuel. Si le mode Optimisation préalable ou Optimisation fine était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp passe au mode de fonctionnement enregistré dans Mode.
0000_0002	Valeur invalide au paramètre "Input_PER". Vérifiez si une erreur est présente à l'entrée analogique. Si le mode automatique était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp fournit la valeur de réglage de remplacement configurée. Dès que l'erreur a disparu, PID_Temp repasse en mode automatique. Si le mode manuel était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp reste en mode manuel. Si le mode Optimisation préalable ou Optimisation fine était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp passe au mode de fonctionnement enregistré dans Mode.
0000_0004	Erreur pendant l'optimisation fine. L'oscillation des températures n'a pas pu être maintenue. Si PID_Temp est utilisé comme régulateur de chauffage et de refroidissement (Config.ActivateCooling = TRUE), la valeur de réglage PID (PidOutputSum) doit remplir la condition suivante à proximité de la consigne pour permettre la génération d'une oscillation de la mesure et une optimisation fine correcte : • être positive pour l'optimisation fine chauffage • être négative pour une optimisation fine du refroidissement Si cette condition n'est pas remplie, utilisez les offsets d'optimisation (variables PIDSelfTune.TIR.OutputOffsetCool et PIDSelfTune.TIR.OutputOffsetHeat), voir Optimisation fine (Page 180). Si ActivateRecoverMode = TRUE avant l'apparition de l'erreur, PID_Temp abandonne l'optimisation et passe au mode de fonctionnement enregistré dans Mode.
0000_0008	Erreur au démarrage de l'optimisation préalable. La mesure est trop proche de la consigne ou supérieure à la consigne. Démarrez l'optimisation fine. Si ActivateRecoverMode = TRUE avant l'apparition de l'erreur, PID_Temp abandonne l'optimisation et passe au mode de fonctionnement enregistré dans Mode.
0000_0010	La consigne a été modifiée durant l'optimisation. La variation admissible de la consigne peut être réglée à la variable CancelTuningLevel. Si ActivateRecoverMode = TRUE avant l'apparition de l'erreur, PID_Temp abandonne l'optimisation et passe au mode de fonctionnement enregistré dans Mode.

ErrorBits (DW#16#...)	Description
0000_0020	L'optimisation préalable n'est pas autorisée pendant l'optimisation fine. Si ActivateRecoverMode = TRUE avant l'apparition de l'erreur, PID_Temp reste en mode de fonctionnement optimisation fine.
0000_0040	Erreur pendant l'optimisation préalable. Le refroidissement n'a pu réduire la mesure. Si ActivateRecoverMode = TRUE avant l'apparition de l'erreur, PID_Temp abandonne l'optimisation et passe au mode de fonctionnement enregistré dans Mode.
0000_0100	Une erreur durant l'optimisation fine a entraîné des paramètres non valides. Si ActivateRecoverMode = TRUE avant l'apparition de l'erreur, PID_Temp abandonne l'optimisation et passe au mode de fonctionnement enregistré dans Mode.
0000_0200	Valeur invalide au paramètre "Input" : Le format numérique de la valeur est invalide. Si le mode automatique était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp fournit la valeur de réglage de remplacement configurée. Dès que l'erreur a disparu, PID_Temp repasse en mode automatique. Si le mode manuel était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp reste en mode manuel. Si le mode Optimisation préalable ou Optimisation fine était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp passe au mode de fonctionnement enregistré dans Mode.
0000_0400	Le calcul de la valeur de réglage a échoué. Vérifiez les paramètres PID. Si le mode automatique était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp fournit la valeur de réglage de remplacement configurée. Dès que l'erreur a disparu, PID_Temp repasse en mode automatique. Si le mode Optimisation préalable ou Optimisation fine était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp passe au mode de fonctionnement enregistré dans Mode.
0000_0800	Erreur de période d'échantillonnage : PID_Temp n'est pas appelé dans la période d'échantillonnage de l'OB d'alarme cyclique. Il est recommandé d'appeler PID_Temp dans un OB d'alarme cyclique sans conditions et de l'activer ou le désactiver via le mode de fonctionnement au paramètre Mode. Les appels conditionnels ou l'appel dans l'OB1 peuvent avoir une influence négative sur la qualité de la régulation. La surveillance de la période d'échantillonnage peut être désactivée avec CycleTime.EnMonitoring = FALSE Si le mode automatique était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp reste en mode automatique. Si le mode manuel était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp reste en mode manuel. Si le mode Optimisation préalable ou Optimisation fine était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp passe au mode de fonctionnement enregistré dans Mode. Si cette erreur s'est produite lors de la simulation avec PLCSIM, tenez compte des indications de la rubrique Simuler PID_Temp avec PLCSIM (Page 199).
0000_1000	Valeur invalide au paramètre "Setpoint" ou "SubstituteSetpoint" : Le format numérique de la valeur est invalide. Si le mode automatique était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp fournit la valeur de réglage de remplacement configurée. Dès que l'erreur a disparu, PID_Temp repasse en mode automatique. Si le mode manuel était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp reste en mode manuel. Si le mode Optimisation préalable ou Optimisation fine était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp passe au mode de fonctionnement enregistré dans Mode.
0001_0000	Valeur invalide au paramètre ManualValue. Le format numérique de la valeur est invalide. Si ActivateRecoverMode = TRUE avant l'apparition de l'erreur, PID_Temp reste en mode manuel et utilise SubstituteOutput comme valeur de réglage PID. Dès que vous spécifiez une valeur valide pour ManualValue, PID_Temp l'utilise comme valeur de réglage PID.

ErrorBits (DW#16#...)	Description
0002_0000	Valeur invalide à la variable SubstituteOutput. Le format numérique de la valeur est invalide. PID_Temp reste en mode de fonctionnement "Valeur de réglage de remplacement avec surveillance d'erreur" ou en mode manuel et utilise la limite inférieure de la valeur de réglage PID pour le chauffage (Config.Output.Heat.PidLowerLimit) comme valeur de réglage PID. Dès que vous spécifiez une valeur valide pour SubstituteOutput, PID_Temp l'utilise comme valeur de réglage PID.
0004_0000	Valeur invalide au paramètre Disturbance. Le format numérique de la valeur est invalide. Si le mode automatique était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, Disturbance est mis à zéro. PID_Temp reste en mode automatique. Si le mode Optimisation préalable ou Optimisation fine était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp passe au mode de fonctionnement enregistré dans Mode. Si Disturbance n'influe pas sur la valeur de réglage dans la phase actuelle, l'optimisation n'est pas interrompue.
0020_0000	Erreur du maître en cascade : les Slaves ne sont pas en mode automatique ou ils ont activé leur consigne de remplacement et empêchent l'optimisation du maître. Si ActivateRecoverMode = TRUE avant l'apparition de l'erreur, PID_Temp abandonne l'optimisation et passe au mode de fonctionnement enregistré dans Mode.
0040_0000	L'optimisation préalable chauffage n'est pas autorisée tant que le refroidissement est activé. Si ActivateRecoverMode = TRUE avant l'apparition de l'erreur, PID_Temp abandonne l'optimisation et passe au mode de fonctionnement enregistré dans Mode.
0080_0000	La mesure doit être proche de la consigne pour démarrer l'optimisation préalable refroidissement. Si ActivateRecoverMode = TRUE avant l'apparition de l'erreur, PID_Temp abandonne l'optimisation et passe au mode de fonctionnement enregistré dans Mode.
0100_0000	Erreur au démarrage de l'optimisation : Heat.EnableTuning et Cool.EnableTuning ne sont pas à 1 ou ne correspondent pas à la configuration. Si ActivateRecoverMode = TRUE avant l'apparition de l'erreur, PID_Temp abandonne l'optimisation et passe au mode de fonctionnement enregistré dans Mode.
0200_0000	L'optimisation préalable refroidissement suppose une optimisation préalable chauffage réussie. Si ActivateRecoverMode = TRUE avant l'apparition de l'erreur, PID_Temp abandonne l'optimisation et passe au mode de fonctionnement enregistré dans Mode.
0400_0000	Erreur au démarrage de l'optimisation fine : Heat.EnableTuning et Cool.EnableTuning ne doivent pas être à 1 simultanément. Si ActivateRecoverMode = TRUE avant l'apparition de l'erreur, PID_Temp abandonne l'optimisation et passe au mode de fonctionnement enregistré dans Mode.
0800_0000	Une erreur pendant le calcul des paramètres PID a produit des paramètres non valides. Les paramètres invalides sont rejetés et les paramètres PID initiaux sont conservés sans modification. Vérifiez les points suivants si cette erreur se produit pendant une optimisation préalable : • Optimisation préalable chauffage ou optimisation préalable chauffage et refroidissement : La valeur de réglage PID n'est pas limitée par la limite supérieure de chauffage avant le démarrage de l'optimisation préalable. • Optimisation préalable refroidissement : La valeur de réglage PID n'est pas limitée par la limite inférieure de refroidissement avant le démarrage de l'optimisation préalable. Pour démarrer une optimisation préalable chauffage ou une optimisation préalable chauffage et refroidissement à partir du mode automatique, il est recommandé de modifier la valeur de consigne sur un front montant de ModeActivate. Cela permet d'éviter que la valeur de réglage PID atteigne la limite entre la modification de la consigne et le démarrage de l'optimisation préalable. Il est également possible d'y parvenir en démarrant l'optimisation préalable à partir du mode manuel ou du mode "Inactif".

ErrorBits (DW#16#...)	Description
	Il faut distinguer les cas suivants : • Si le mode automatique était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp reste en mode automatique. • Si le mode manuel était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp reste en mode manuel. • Si le mode optimisation préalable ou optimisation fine était activé avant l'apparition de l'erreur et que ActivateRecoverMode = TRUE, PID_Temp passe au mode de fonctionnement qui est enregistré dans Mode.

10.3.4.9 Variable ActivateRecoverMode de PID_Temp

La variable ActivateRecoverMode détermine le comportement en cas d'erreur. Le paramètre Error indique si une erreur est actuellement présente. Quand l'erreur a disparu, Error = FALSE. Le paramètre ErrorBits indique quelles erreurs sont apparues.

Mode automatique et mode manuel

IMPORTANT
Votre installation peut être endommagée. Si ActivateRecoverMode = TRUE, PID_Temp restera en mode automatique en cas d'erreur, même quand les limites de la mesure seront dépassées. Cela peut endommager votre installation. Configurez pour votre système réglé un comportement en cas d'erreur qui protège votre installation de tout endommagement.

ActivateRecover-Mode	Description
FALSE	En cas d'erreur, PID_Temp passe en mode "Inactif". Le régulateur n'est activé que par un front descendant à Reset ou un front montant à ModeActivate.
TRUE	**Mode automatique** Si des erreurs apparaissent fréquemment en mode automatique, ceci détériore le comportement de régulation, puisque PID_Temp alterne à chaque erreur entre la valeur de réglage PID calculée et la valeur de réglage de remplacement. Vérifiez alors le paramètre ErrorBits et éliminez la cause d'erreur. Si l'une ou plusieurs des erreurs suivantes apparaissent et que le mode automatique était actif avant l'apparition de l'erreur, PID_Temp restera en mode automatique : • 0000001h : Le paramètre "Input" se trouve en dehors des limites de la mesure. • 0000800h : Erreur de temps d'échantillonnage • 0040000h : Valeur invalide au paramètre Disturbance. • 8000000h : Erreur pendant le calcul des paramètres PID Si l'une ou plusieurs des erreurs suivantes apparaissent et que le mode automatique était actif avant l'apparition de l'erreur, PID_Temp passera en mode de fonctionnement "Valeur de réglage de remplacement avec surveillance d'erreur" : • 0000002h : Valeur invalide au paramètre Input_PER. • 0000200h : Valeur invalide au paramètre Input. • 0000400h : Le calcul de la valeur de réglage a échoué. • 0001000h : Valeur invalide au paramètre Setpoint ou SubstituteSetpoint.

ActivateRecover-Mode	Description
TRUE	Dès que les erreurs ont disparu, PID_Temp repasse en mode automatique. Si l'erreur suivante apparaît en mode de fonctionnement "Valeur de réglage de remplacement avec surveillance d'erreur", PID_Temp mettra la valeur de réglage PID sur Config.Output.Heat.PidLowerLimit tant que cette erreur sera présente : • 0020000h : Valeur invalide de la variable SubstituteOutput. Le format numérique de la valeur est invalide. Ce comportement est indépendant de SetSubstituteOutput.
	Mode manuel Si l'une ou plusieurs des erreurs suivantes apparaissent et que le mode manuel était actif avant l'apparition de l'erreur, PID_Temp restera en mode manuel : Si l'erreur suivante apparaît en mode manuel, PID_Temp mettra la valeur de réglage PID sur SubstituteOutput: tant que cette erreur sera présente : • 0010000h : Valeur invalide au paramètre ManualValue. Le format numérique de la valeur est invalide. Si l'erreur 0010000h est présente en mode manuel et que l'erreur suivante apparaît, PID_Temp mettra la valeur de réglage PID sur Config.Output.Heat.PidLowerLimit tant que cette erreur sera présente : • 0020000h : Valeur invalide de la variable SubstituteOutput. Le format numérique de la valeur est invalide. Ce comportement est indépendant de SetSubstituteOutput.

Optimisation préalable et optimisation fine

ActivateRecover-Mode	Description
FALSE	En cas d'erreur, PID_Temp passe en mode "Inactif". Le régulateur n'est activé que par un front descendant à Reset ou un front montant à ModeActivate.
TRUE	Si l'erreur suivante apparaît, PID_Temp restera en mode actif : • 0000020h : L'optimisation préalable n'est pas autorisée pendant l'optimisation fine. Les erreurs suivantes sont ignorées : • 0010000h : Valeur invalide au paramètre ManualValue. • 0020000h : Valeur invalide de la variable SubstituteOutput. Pour toutes les autres erreurs, PID_Temp abandonne l'optimisation et passe au mode de fonctionnement à partir duquel l'optimisation a été lancée.

10.3.4.10 Variable Warning de PID_Temp

En présence simultanée de plusieurs avertissements, les valeurs de la variable Warning sont affichées sous forme d'addition binaire. Si l'avertissement affiché est p. ex. 16#0000_0003, cela indique la présence simultanée des avertissements 16#0000_0001 et 16#0000_0002.

Warning (DW#16#....)	Description
0000_0000	Aucune alerte n'est présente.
0000_0001	Le point d'inflexion n'a pas été trouvé pendant l'optimisation préalable.
0000_0004	La consigne a été limitée à des limites paramétrées.
0000_0008	Toutes les propriétés nécessaires du système réglé n'ont pas été déterminées pour la méthode de calcul choisie. Les paramètres PID ont été calculés à titre de remplacement avec la méthode TIR.TuneRuleHeat = 3 et TIR.TuneRuleCool = 3.

Warning (DW#16#....)	Description
0000_0010	Impossible de modifier le mode de fonctionnement car Reset = TRUE ou ManualEnable = TRUE
0000_0020	Le temps d'échantillonnage de l'algorithme PID est limité par le temps de cycle de l'OB appelant. Afin d'obtenir de meilleurs résultats, utilisez des temps de cycle de l'OB plus courts.
0000_0040	La mesure a dépassé l'une de ses limites d'alerte.
0000_0080	Valeur invalide de Mode. Le changement de mode de fonctionnement n'est pas effectué.
0000_0100	La valeur manuelle a été limitée aux limites de la valeur de réglage PID.
0000_0200	La règle mentionnée pour l'optimisation n'est pas prise en charge. Aucun paramètre PID n'est calculé.
0000_1000	La valeur de réglage de remplacement ne peut pas être atteinte, car elle est en dehors des limites de la valeur de réglage.
0000_4000	Le choix indiqué comme valeur de réglage pour chauffage et/ou refroidissement n'est pas pris en charge. Seule la sortie OutputHeat ou OutputCool est utilisée.
0000_8000	Valeur invalide de PIDSelfTune.SUT.AdaptDelayTime. La valeur par défaut 0 est utilisée.
0001_0000	Valeur invalide de PIDSelfTune.SUT.CoolingMode. La valeur par défaut 0 est utilisée.
0002_0000	L'activation du refroidissement (variable Config.ActivateCooling) n'est pas prise en charge pour un régulateur utilisé comme maître (variable Config.Cascade.IsMaster). PID_Temp fonctionne comme régulateur de chauffage. Paramétrez la variable Config.ActivateCooling sur FALSE.
0004_0000	Valeur invalide de Retain.CtrlParams.Heat.Gain, Retain.CtrlParams.Cool.Gain ou Config.CoolFactor. PID_Temp prend uniquement en charge des valeurs positives pour le gain proportionnel (chauffage et refroidissement) et le facteur de refroidissement. Le mode automatique reste activé avec la valeur de réglage PID 0.0. L'action par intégration est arrêtée.

Les avertissements suivants sont effacés dès que leur cause est éliminée ou que vous répétez l'action avec des paramètres valides.

- 16#0000_0001
- 16#0000_0004
- 16#0000_0008
- 16#0000_0040
- 16#0000_0100

Toutes les autres alertes sont supprimées avec un front montant à Reset ou ErrorAck.

10.3.4.11 Variable PwmPeriode

Quand vous utilisez OutputHeat_PWM ou OutputCool_PWM et que la période d'échantillonnage de l'algorithme PID est très longue (Retain.CtrlParams.Heat.Cycle ou Retain.CtrlParams.Heat.Cycle), ce qui donne une longue période pour la modulation de largeur d'impulsion, vous pouvez spécifier une autre période plus courte aux paramètres Config.Output.Heat.PwmPeriode ou Config.Output.Cool.PwmPeriode pour obtenir une mesure plus lisse.

Période de la modulation de largeur d'impulsion à la sortie OutputHeat_PWM

Période de la PWM à la sortie OutputHeat_PWM en fonction de Config.Output.Heat.PwmPeriode :

- Heat.PwmPeriode = 0.0 (valeur par défaut)
  La période d'échantillonnage de l'algorithme PID pour le chauffage (Retain.CtrlParams.Heat.Cycle) est utilisée comme période de la modulation de largeur d'impulsion.
- Heat.PwmPeriode > 0.0
  La valeur est arrondie à un multiple entier de la période d'échantillonnage PID_Temp (CycleTime.Value) et utilisée comme période de la modulation de largeur d'impulsion.
  La valeur doit remplir les conditions suivantes :
  - Heat.PwmPeriode ≤ Retain.CtrlParams.Heat.Cycle
  - Heat.PwmPeriode > Config.Output.Heat.MinimumOnTime
  - Heat.PwmPeriode > Config.Output.Heat.MinimumOffTime

Période de la modulation de largeur d'impulsion à la sortie OutputCool_PWM

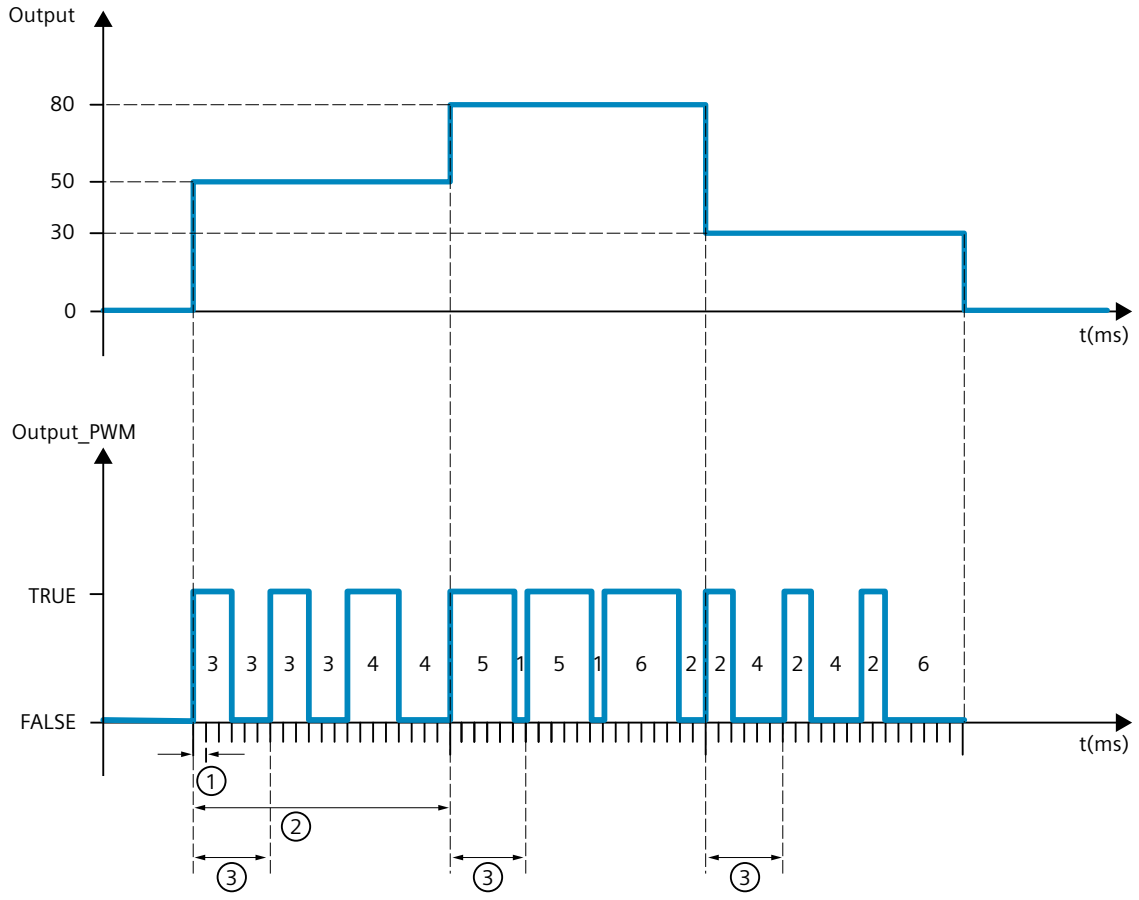
Période de la PWM à la sortie OutputCool_PWM en fonction de Config.Output.Cool.PwmPeriode et de la méthode choisie pour le chauffage/refroidissement :

- Cool.PwmPeriode = 0.0 et facteur de refroidissement (Config.AdvancedCooling = FALSE)
  La période d'échantillonnage de l'algorithme PID pour le chauffage (Retain.CtrlParams.Heat.Cycle) est utilisée comme période de la modulation de largeur d'impulsion.
- Cool.PwmPeriode = 0.0 et commutation des paramètres PID (Config.AdvancedCooling = TRUE)
  La période d'échantillonnage de l'algorithme PID pour le refroidissement (Retain.CtrlParams.Cool.Cycle) est utilisée comme période de la modulation de largeur d'impulsion.
- Cool.PwmPeriode > 0.0:
  La valeur est arrondie à un multiple entier de la période d'échantillonnage PID_Temp (CycleTime.Value) et utilisée comme période de la modulation de largeur d'impulsion.
  La valeur doit remplir les conditions suivantes :
  - Cool.PwmPeriode ≤ Retain.CtrlParams.Cool.Cycle ou Retain.CtrlParams.Heat.Cycle
  - Cool.PwmPeriode > Config.Output.Cool.MinimumOnTime
  - Cool.PwmPeriode > Config.Output.Cool.MinimumOffTime

Config.Output.Cool.PwmPeriode n'est opérant que si la sortie de refroidissement est activée (Config.ActivateCooling = TRUE).

Quand vous utilisez PwmPeriode, la précision du signal de sortie PWM est déterminée par le rapport entre PwmPeriode et la période d'échantillonnage PID_Temp (temps de cycle de l'OB). PwmPeriode doit être au moins 10 fois la période d'échantillonnage PID_Temp.
Si la période d'échantillonnage de l'algorithme PID n'est pas un multiple entier de PwmPeriode, chaque dernière période de la modulation de largeur d'impulsion (PWM) sera allongée en conséquence au sein de la période d'échantillonnage de l'algorithme PID.

Exemple de OutputHeat_PWM



① Période d'échantillonnage PID_Temp = 100,0 ms (temps de cycle de l'OB d'alarme cyclique appelant, variable CycleTime.Value)

② Période d'échantillonnage de l'algorithme PID = 2000,0 ms (variable Retain.CtrlParams.Heat.Cycle)

③ Période de la PWM pour le chauffage = 600,0 ms (variable Config.Output.Heat.PwmPeriode)

10.3.4.12 Variable IntegralResetMode

La variable IntegralResetMode détermine la valeur par défaut de l'action I PIDCtrl.IOutputOld :

- lorsque le mode de fonctionnement passe de "Inactif" à "Mode automatique"
- en cas de front TRUE -> FALSE sur le paramètre Reset et le paramètre Mode = 3

Ce réglage n'agit que pendant un cycle et n'est effectif que si l'action I est activée (variables Retain.CtrlParams.Heat.Ti et Retain.CtrlParams.Cool.Ti > 0.0)

IntegralReset-Mode	**Description**
0	Lissage La valeur de PIDCtrl.IOutputOld est préréglée de manière à ce que la commutation soit sans à-coup, c'est-à-dire que le "mode automatique" démarre à partir de la valeur de réglage = 0.0 (paramètre PidOutputSum) et qu'il n'y ait pas de saut de la valeur de réglage indépendamment du signal d'écart (consigne – mesure).
1	Supprimer Il est recommandé de mettre la pondération de l'action P (variables Retain.CtrlParams.Heat.PWeighting et Retain.CtrlParams.Cool.PWeighting) à 1.0 dans le cas où cette option est utilisée. La valeur de PIDCtrl.IOutputOld est supprimée. Si un signal d'écart est disponible, cela revient à un saut de la valeur de réglage PID. Le sens du saut de la valeur de réglage dépend de la pondération active de l'action P (variables Retain.CtrlParams.Heat.PWeighting et Retain.CtrlParams.Cool.PWeighting) et du signal d'écart : • pondération active de l'action P = 1.0 : Le saut de la valeur de réglage et le signal d'écart ont le même signe. Exemple : Si la mesure est inférieure à la consigne (signal d'écart positif), la valeur de réglage PID saute à une valeur positive. • pondération active de l'action P < 1.0 : Pour de grands signaux d'écart, le saut de la valeur de réglage PID et le signal d'écart ont le même signe. Exemple : Si la mesure est bien inférieure à la consigne (signal d'écart positif), la valeur de réglage PID saute à une valeur positive. Pour de petits signaux d'écart, le saut de la valeur de réglage PID et le signal d'écart ont des signes différents. Exemple : Si la mesure est légèrement inférieure à la consigne (signal d'écart positif), la valeur de réglage PID saute à une valeur négative. Cela n'est généralement pas souhaitable, car cela conduit à une augmentation temporaire du signal d'écart. Le signal d'écart doit être d'autant plus grand, pour obtenir un saut de valeur de réglage PID de même signe, que la pondération de l'action P configurée est faible. Il est recommandé de mettre la pondération de l'action P (variables Retain.CtrlParams.Heat.PWeighting et Retain.CtrlParams.Cool.PWeighting) à 1.0 dans le cas où cette option est utilisée. Dans le cas contraire, cela peut entraîner le comportement indésirable décrit, si le signal d'écart est faible. Une autre solution consiste à utiliser IntegralResetMode = 4. Cette option garantit des signes identiques pour le saut de la valeur de réglage PID et le signal d'écart, indépendamment de la pondération de l'action P configurée et du signal d'écart.
2	Conserver La valeur de PIDCtrl.IOutputOld n'est pas modifiée. Vous pouvez spécifier une nouvelle valeur via le programme utilisateur.
3	Valeur par défaut La valeur de PIDCtrl.IOutputOld est automatiquement préréglée comme si on avait eu PidOutputSum = OverwriteInitialOutputValue dans le dernier cycle.

IntegralReset-Mode	Description
4	Comme variation de la consigne (uniquement pour PID_Temp en version ≥ 1.1) La valeur de PIDCtrl.IOutputOld est automatiquement préréglée de manière à ce que se produise un saut de valeur de réglage PID analogue à celui d'un régulateur PI en mode automatique lorsque la consigne varie de la valeur de mesure actuelle à la valeur de consigne actuelle. Si un signal d'écart est disponible, cela revient à un saut de la valeur de réglage PID. Le saut de la valeur de réglage PID et le signal d'écart ont le même signe. Exemple : Si la mesure est inférieure à la consigne (signal d'écart positif), la valeur de réglage PID saute à une valeur positive. Ce comportement est indépendant de la pondération de l'action P configurée et du signal d'écart.

Si une valeur extérieure à la plage admissible est affectée à IntegralResetMode, PID_Temp se comporte comme avec la valeur par défaut de IntegralResetMode :

- PID_Temp jusqu'à V1.0 : IntegralResetMode = 1
- PID_Temp à partir de V1.1 : IntegralResetMode = 4

10.4 Fonctions de base PID

10.4.1 CONT_C

10.4.1.1 Description CONT_C

L'instruction CONT_C sert à la régulation de processus techniques possédant des grandeurs d'entrée et de sortie continues sur les systèmes d'automatisation SIMATIC S7. En paramétrant ce bloc, vous pouvez activer ou désactiver des fonctions partielles du régulateur PID afin de l'adapter au système réglé. En complément des fonctions de la branche de consigne et de mesure, l'instruction réalise un régulateur PID opérationnel doté d'une sortie continue pour la grandeur réglante et de la possibilité de modifier manuellement la valeur de réglage.

Application

Vous pouvez utiliser le régulateur comme régulateur PID de maintien individuel, mais aussi comme régulateur en cascade, de mélange ou de rapport dans des régulations à plusieurs boucles. Sa méthode de travail se base sur l'algorithme PID du régulateur d'échantillonnage à sortie analogique, complété le cas échéant par un étage formateur d'impulsions assurant la formation d'impulsions de sortie modulées en durée pour régulations à deux ou trois positions avec actionneurs proportionnels.

Appel

L'instruction CONT_C dispose d'une routine d'initialisation exécutée lorsque le paramètre d'entrée COM_RST = TRUE. Au moment de l'initialisation, l'intégrateur est mis sur la valeur I_ITVAL. Toutes les autres sorties sont mises à zéro. Après exécution de la routine d'initialisation, il faut mettre COM_RST = FALSE.

Les valeurs des blocs de régulation ne sont calculées correctement que si le bloc est appelé à intervalles réguliers. C'est pourquoi il convient d'appeler les blocs de régulation dans un OB d'alarme cyclique (OB 30 à OB 38). Vous spécifiez le temps d'échantillonnage au paramètre CYCLE.

Informations d'erreur

Le mot de signalisation d'erreur RET_VAL n'est pas évalué par le bloc.

10.4.1.2 Fonctionnement de CONT_C

Branche de consigne

La consigne est introduite en format à virgule flottante à l'entrée SP_INT.

Branche de valeur rélle

La mesure peut être lue en format de périphérie ou en format à virgule flottante. La fonction CRP_IN convertit la valeur de périphérie PV_PER en un nombre à virgule flottante compris entre -100 et +100 % selon la règle suivante :

Sortie de CRP_IN = PV_PER * 100 / 27648

La fonction PV_NORM normalise la sortie de CRP_IN selon la règle suivante :

Sortie de PV_NORM = (sortie de CRP_IN) *PV_FAC + PV_OFF

La valeur par défaut de PV_FAC est 1 et celle de PV_OFF est 0.

Calcul du signal d'écart

La différence entre la consigne et la mesure est appelée signal d'écart. Pour supprimer une légère oscillation continue due à la quantification de la grandeur de réglage (par ex. en cas de modulation de largeur d'impulsion avec PULSEGEN), l'écart de régulation est appliqué à une bande morte (DEADBAND). Lorsque DEADB_W = 0, la zone morte est désactivée.

Algorithme PID

L'algorithme PID fonctionne comme un algorithme de position. Les actions proportionnelle, par intégration (INT) et par dérivation (DIF) sont montées en parallèle et peuvent être activées et désactivées individuellement. Ceci permet de paramétrer des régulateurs P, PI, PD et PID. Toutefois, des régulateurs à action I seule peuvent également être paramétrés.

Mode manuel

Il est possible de commuter entre le mode manuel et le mode automatique. En mode manuel, la grandeur réglante est ajustée en fonction d'une valeur manuelle.

L'intégrateur (INT) est forcé de manière interne à LMN - LMN_P - DISV et le dérivateur (DIF) est forcé à 0 et égalisé de manière interne. Le passage au mode automatique s'effectue donc sans à-coups.

Traitement de la grandeur réglante

La valeur de réglage est limitée à des valeurs paramétrables avec la fonction LMNLIMIT. Si la grandeur d'entrée dépasse ces limites, des bits le signalent.

La fonction LMN_NORM normalise la sortie de LMNLIMIT selon la règle suivante :

LMN = (sortie de LMNLIMIT) * LMN_FAC + LMN_OFF

La valeur par défaut de LMN_FAC est 1 et celle de LMN_OFF est 0.

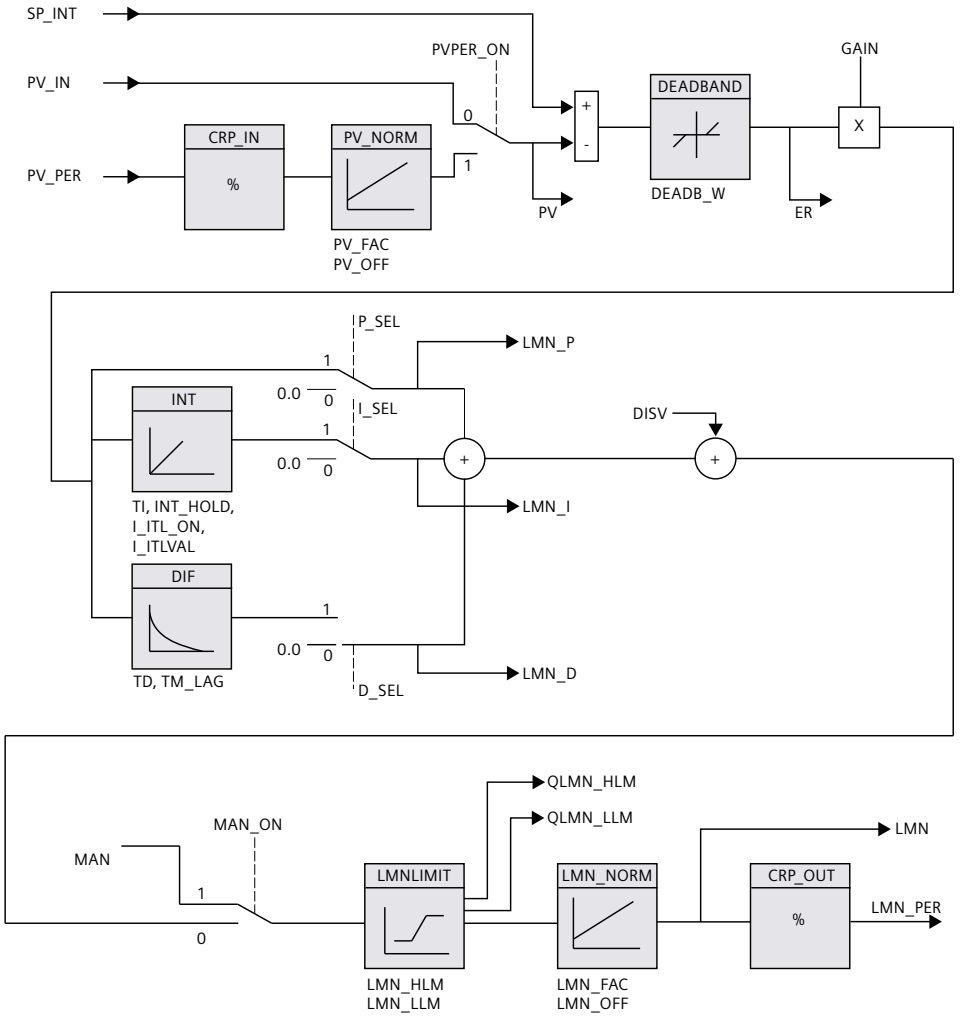
La valeur de réglage est également disponible en format périphérie. La fonction CRP_OUT convertit la valeur LMN à virgule flottante en une valeur de périphérie d'après la règle suivante :

LMN_PER = LMN * 27648 / 100

Action anticipatrice

Une perturbation additionnelle peut être appliquée à l'entrée DISV.

10.4.1.3 Schéma fonctionnel CONT_C



10.4.1.4 Paramètres d'entrée CONT_C

Les noms des paramètres suivants sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Tableau 10-13

Paramètre	Type de données	Valeur par défaut	Description
COM_RST	BOOL	FALSE	L'instruction possède une routine d'initialisation qui est exécutée quand l'entrée "Démarrage" est à 1.
MAN_ON	BOOL	TRUE	La mise à 1 de l'entrée "activation du mode manuel" interrompt la boucle de régulation. C'est une valeur manuelle qui est introduite comme valeur de réglage.
PVPER_ON	BOOL	FALSE	Pour lire la mesure de la périphérie, l'entrée PV_PER doit être interconnectée à la périphérie et l'entrée "activation de la mesure périphérie" doit être mise à 1.
P_SEL	BOOL	TRUE	Dans l'algorithme PID, les actions PID peuvent être activées et désactivées individuellement. L'action P est active quand cette entrée est à 1.
I_SEL	BOOL	TRUE	Dans l'algorithme PID, les actions PID peuvent être activées et désactivées individuellement. L'action I est active quand cette entrée est à 1.
INT_HOLD	BOOL	FALSE	Vous pouvez geler la sortie de l'intégrateur. Pour cela, cette entrée doit être à 1.
I_ITL_ON	BOOL	FALSE	La sortie de l'intégrateur peut être mise à la valeur de l'entrée I_ITLVAL. Pour ce faire, l'entrée "Activation de l'action I" doit être mise à 1.
D_SEL	BOOL	FALSE	Dans l'algorithme PID, les actions PID peuvent être activées et désactivées individuellement. L'action D est active quand cette entrée est à 1.
CYCLE	TIME	T#1s	Le temps entre deux appels du bloc doit être constant. Il est indiqué par cette entrée. CYCLE >= 1ms
SP_INT	REAL	0.0	L'entrée "consigne interne" permet de spécifier une consigne. Les valeurs autorisées sont comprises entre -100 et 100 % ou une grandeur physique 1).
PV_IN	REAL	0.0	L'entrée "Entrée mesure" permet de paramétrer une valeur de mise en service ou d'interconnecter une mesure externe au format en virgule flottante. Les valeurs autorisées sont comprises entre -100 et 100 % ou une grandeur physique 1).
PV_PER	WORD	W#16#0000	La mesure en format périphérie est interconnectée au régulateur à l'entrée "Mesure périphérie".
MAN	REAL	0.0	Cette entrée sert à introduire une valeur manuelle grâce à des fonctions de contrôle-commande. Les valeurs autorisées sont comprises entre -100 et 100 % ou une grandeur physique 2).
GAIN	REAL	2.0	L'entrée "Coefficient d'action proportionnelle" indique le gain du régulateur.
TI	TIME	T#20s	L'entrée "Temps d'intégration" détermine la réponse temporelle de l'intégrateur. TI >= CYCLE
TD	TIME	T#10s	L'entrée "Temps de dérivation" détermine la réponse temporelle du dérivateur. TD >= CYCLE
TM_LAG	TIME	T#2s	Temps de retard de l'action D L'algorithme de l'action D contient un retard pouvant être paramétré à l'entrée "Temps de retard de l'action D". TM_LAG >= CYCLE/2
DEADB_W	REAL	0.0	Le signal d'écart parcourt une zone morte. L'entrée "Largeur de zone morte" détermine la taille de la zone morte. DEADB_W >= 0.0 (%) ou une grandeur physique 1)

Paramètre	Type de données	Valeur par défaut	Description
LMN_HLM	REAL	100.0	La valeur de réglage possède toujours une limite supérieure et inférieure. L'entrée "Limitation supérieure de la valeur de réglage" indique sa limitation supérieure. Les valeurs autorisées sont des valeurs réelles à partir de LMN_LLM (%) ou une grandeur physique 2).
LMN_LLM	REAL	0.0	La valeur de réglage possède toujours une limite supérieure et inférieure. L'entrée "Limitation inférieure de la valeur de réglage" indique sa limitation inférieure. Les valeurs autorisées sont des valeurs réelles jusqu'à LMN_HLM (%) ou une grandeur physique 2).
PV_FAC	REAL	1.0	L'entrée "Facteur de mesure" est multipliée par la mesure. Cette entrée permet d'adapter l'étendue de la mesure.
PV_OFF	REAL	0.0	L'entrée "Décalage de la mesure" est additionnée à la mesure. Cette entrée permet d'adapter l'étendue de la mesure.
LMN_FAC	REAL	1.0	L'entrée "facteur de valeur de réglage" est multipliée par la valeur de réglage. Cette entrée permet d'adapter la plage de la valeur de réglage.
LMN_OFF	REAL	0.0	Cette entrée est ajoutée à la valeur de réglage. Cette entrée permet d'adapter la plage de la valeur de réglage.
I_ITLVAL	REAL	0.0	La sortie de l'intégrateur peut être positionnée à l'entrée I_ITL_ON. La valeur d'initialisation pour l'action I est à cette entrée. Les valeurs autorisées sont comprises entre -100.0 et 100.0 (%) ou une grandeur physique 2).
DISV	REAL	0.0	Pour une action anticipatrice, la perturbation est appliquée à l'entrée "perturbation". Les valeurs autorisées sont comprises entre -100.0 et 100.0 (%) ou une grandeur physique 2).

1) Paramètres dans la branche de consigne et de mesure avec la même unité

2) Paramètres dans la branche de valeur de réglage avec la même unité

10.4.1.5 Paramètres de sortie CONT_C

Les noms des paramètres suivants sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Tableau 10-14

Paramètre	Type de données	Valeur par défaut	Description
LMN	REAL	0.0	La valeur de réglage effective en format à virgule flottante est fournie à la sortie "valeur de réglage".
LMN_PER	WORD	W#16#0000	La valeur de réglage en format périphérie est connectée au régulateur à la sortie "valeur de réglage périphérie".
QLMN_HLM	BOOL	FALSE	La valeur de réglage possède toujours une limite supérieure et inférieure. La sortie "Limite supérieure de la valeur de réglage accostée" signale que la limite supérieure est atteinte.
QLMN_LLM	BOOL	FALSE	La valeur de réglage possède toujours une limite supérieure et inférieure. La sortie "Limite inférieure de la valeur de réglage accostée" signale que la limite inférieure est atteinte.
LMN_P	REAL	0.0	La sortie "action P" correspond à l'action proportionnelle de la variable réglante.
LMN_I	REAL	0.0	La sortie "action I" correspond à l'action d'intégration de la variable réglante.

Paramètre	Type de données	Valeur par défaut	Description
LMN_D	REAL	0.0	La sortie "action D" correspond à l'action de dérivation de la variable réglante.
PV	REAL	0.0	La mesure opérante est fournie à la sortie "Mesure".
ER	REAL	0.0	Le signal d'écart opérant est fourni à la sortie "Signal d'écart".

10.4.2 CONT_S

10.4.2.1 Description CONT_S

L'instruction CONT_S sert à la régulation des processus techniques à signaux de sortie binaires de la valeur de réglage pour actionneurs intégrés dans les systèmes d'automatisation SIMATIC S7. En paramétrant ce bloc, vous pouvez activer ou désactiver des fonctions partielles du régulateur pas à pas PI afin de l'adapter au système réglé. En complément des fonctions de la branche de mesure, l'instruction réalise un régulateur PID opérationnel doté d'une sortie de valeur de réglage TOR et de la possibilité de modifier manuellement la valeur de réglage. Ce régulateur pas à pas fonctionne sans signalisation de position.

Application

Ce régulateur peut être utilisé en tant que régulateur PI individuel à valeur fixe ou dans des boucles de régulation secondaires dans le cadre de concepts de régulation en cascade, de mélange ou de rapport, mais pas en tant que régulateur pilote. Sa méthode de travail se base sur l'algorithme de régulation PI du régulateur d'échantillonnage, complété par les éléments de commande générant le signal de sortie binaire à partir du signal de réglage analogique.

Appel

L'instruction CONT_S dispose d'une routine d'initialisation exécutée lorsque le paramètre d'entrée COM_RST = TRUE. Toutes les sorties sont mises à zéro. Après exécution de la routine d'initialisation, il faut mettre COM_RST = FALSE.

Les valeurs des blocs de régulation ne sont calculées correctement que si le bloc est appelé à intervalles réguliers. C'est pourquoi il convient d'appeler les blocs de régulation dans un OB d'alarme cyclique (OB 30 à OB 38). Vous spécifiez le temps d'échantillonnage au paramètre CYCLE.

Informations d'erreur

Le mot de signalisation d'erreur RET_VAL n'est pas évalué par le bloc.

10.4.2.2 Fonctionnement CONT_S

Branche de consigne

La consigne est introduite en format à virgule flottante à l'entrée SP_INT.

Branche de valeur rélle

La mesure peut être lue en format de périphérie ou en format à virgule flottante. La fonction CRP_IN convertit la valeur de périphérie PV_PER en un nombre à virgule flottante compris entre -100 et +100 % selon la règle suivante :

Sortie de CRP_IN = PV_PER * 100 / 27648

La fonction PV_NORM normalise la sortie de CRP_IN selon la règle suivante :

Sortie de PV_NORM = (sortie de CRP_IN) * PV_FAC + PV_OFF

La valeur par défaut de PV_FAC est 1 et celle de PV_OFF est 0.

Calcul du signal d'écart

La différence entre la consigne et la mesure est appelée signal d'écart. Il est appliqué à une zone morte (DEADBAND) pour atténuer une petite oscillation causée par la quantification de grandeur réglante (résolution limitée de la valeur de réglage par la vanne de régulation). Lorsque DEADB_W = 0, la zone morte est désactivée.

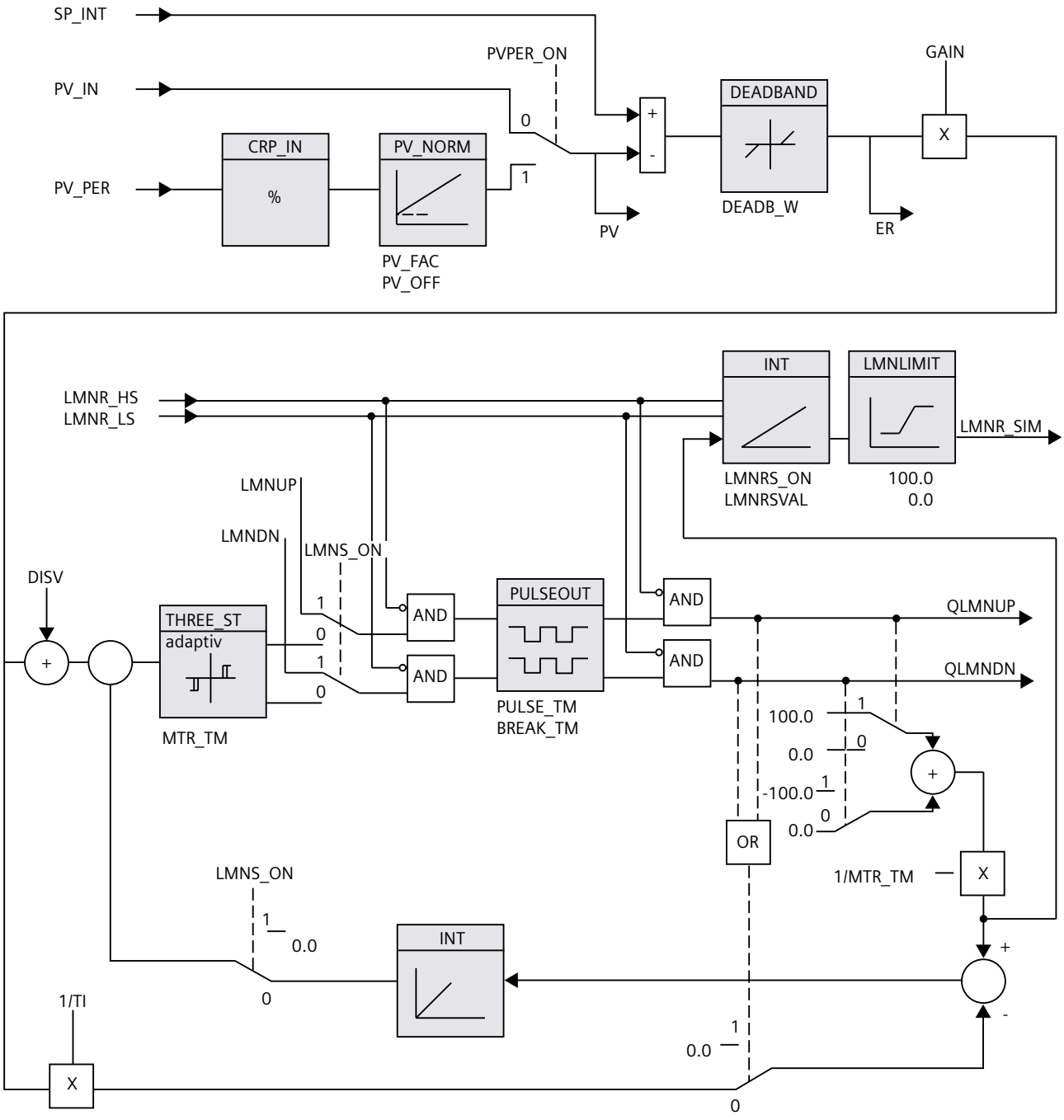
Algorithme pas-à-pas PI

L'instruction travaille sans signalisation de position. L'action I de l'algorithme PI et la signalisation théorique de position sont calculées dans **un** intégrateur (INT) et comparées en tant que valeur de rétroaction à l'action P restante. La différence est transmise à un élément fonctionnel à trois échelons (THREE_ST) ainsi qu'au formateur des impulsions (PULSEOUT) pour la valve de régulation. La fréquence de commutation du régulateur peut être réduite par l'adaptation du seuil d'action de l'élément fonctionnel à trois échelons.

Action anticipatrice

Une perturbation additionnelle peut être appliquée à l'entrée DISV.

10.4.2.3 Schéma fonctionnel CONT_S



10.4.2.4 Paramètres d'entrée CONT_S

Les noms des paramètres suivants sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Tableau 10-15

Paramètre	Type de données	Valeur par défaut	Description
COM_RST	BOOL	FALSE	L'instruction possède une routine d'initialisation qui est exécutée quand l'entrée "Démarrage" est à 1.
LMNR_HS	BOOL	FALSE	Le signal "Butée supérieure de la vanne de régulation" est appliqué à l'entrée "Signal de butée supérieure de la signalisation de position". LMNR_HS=TRUE signifie : la vanne de régulation se trouve à la butée supérieure.
LMNR_LS	BOOL	FALSE	Le signal "Butée inférieure de la vanne de régulation" est appliqué à l'entrée "Signal de butée inférieure de la signalisation de position". LMNR_LS=TRUE signifie : la vanne de régulation a atteint la butée inférieure.
LMNS_ON	BOOL	FALSE	L'entrée "Activation du mode manuel des signaux de valeur de réglage" permet de passer au traitement manuel des signaux de valeur de réglage.
LMNUP	BOOL	FALSE	En traitement manuel des signaux de valeur réglante, le signal de sortie QLMNUP est commandé par l'entrée "Signal de valeur réglante haut".
LMNDN	BOOL	FALSE	En traitement manuel des signaux de valeur de réglage, le signal de sortie QLMNDN est commandé par l'entrée "Signal bas de valeur de réglage".
PVPER_ON	BOOL	FALSE	Pour lire la mesure de la périphérie, l'entrée PV_PER doit être interconnectée à la périphérie et l'entrée "activation de la mesure périphérie" doit être mise à 1.
CYCLE	TIME	T#1s	Le temps entre deux appels du bloc doit être constant. Il est indiqué par cette entrée. CYCLE >= 1ms
SP_INT	REAL	0.0	L'entrée "consigne interne" permet de spécifier une consigne. Les valeurs autorisées sont comprises entre -100 et 100 ou une grandeur physique [1)].
PV_IN	REAL	0.0	L'entrée "Entrée mesure" permet de paramétrer une valeur de mise en service ou d'interconnecter une mesure externe au format en virgule flottante. Les valeurs autorisées sont comprises entre -100 et 100 ou une grandeur physique [1)].
PV_PER	WORD	W#16#0-000	La mesure en format périphérie est interconnectée au régulateur à l'entrée "Mesure périphérie".
GAIN	REAL	2.0	L'entrée "Coefficient d'action proportionnelle" indique le gain du régulateur.
TI	TIME	T#20s	L'entrée "Temps d'intégration" détermine la réponse temporelle de l'intégrateur. TI >= CYCLE
DEADB_W	REAL	1.0	Le signal d'écart parcourt une zone morte. L'entrée "Largeur de zone morte" détermine la taille de la zone morte. Les valeurs autorisées sont comprises entre 0 et 100 ou une grandeur physique [1)].
PV_FAC	REAL	1.0	L'entrée "Facteur de mesure" est multipliée par la mesure. Cette entrée permet d'adapter l'étendue de la mesure.
PV_OFF	REAL	0.0	L'entrée "Décalage de la mesure" est additionnée à la mesure. Cette entrée permet d'adapter l'étendue de la mesure.
PULSE_TM	TIME	T#3s	Le paramètre "Durée minimale d'impulsion" permet de régler une durée d'impulsion minimale. PULSE_TM >= CYCLE

Paramètre	Type de données	Valeur par défaut	Description
BREAK_TM	TIME	T#3s	Le paramètre "Durée minimale de pause" permet de spécifier une durée de pause minimale. BREAK_TM >= CYCLE
MTR_TM	TIME	T#30s	Le paramètre "Temps de positionnement du moteur" permet d'entrer le temps d'exécution de la vanne de régulation de butée en butée. MTR_TM >= CYCLE
DISV	REAL	0.0	Pour une action anticipatrice, la perturbation est appliquée à l'entrée "perturbation". Les valeurs autorisées sont comprises entre -100 et 100 ou une grandeur physique [2].

[1] Paramètres dans la branche de consigne et de mesure avec la même unité

[2] Paramètres dans la branche de valeur de réglage avec la même unité

10.4.2.5 Paramètres de sortie CONT_S

Les noms des paramètres suivants sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Tableau 10-16

Paramètre	Type de données	Valeur par défaut	Description
QLMNUP	BOOL	FALSE	Si la sortie" Signal haut de valeur de réglage" est à 1, la vanne de régulation doit être ouverte.
QLMNDN	BOOL	FALSE	Si la sortie" Signal bas de valeur de réglage" est à 1, la vanne de régulation doit être fermée.
PV	REAL	0.0	La mesure opérante est fournie à la sortie "Mesure".
ER	REAL	0.0	Le signal d'écart opérant est fourni à la sortie "Signal d'écart".

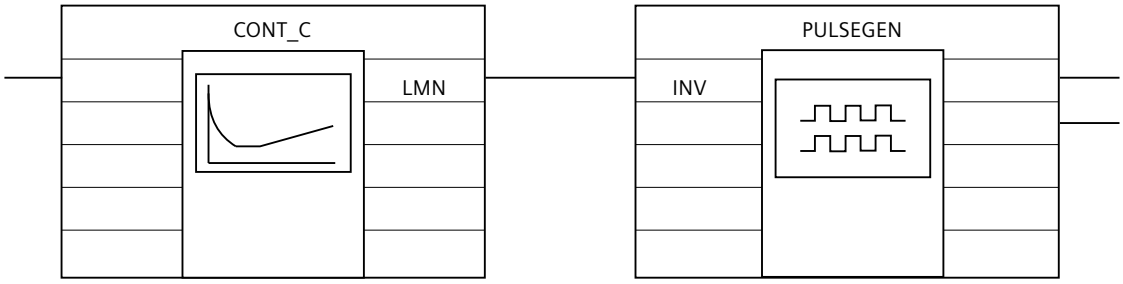
10.4.3 PULSEGEN

10.4.3.1 Description PULSGEN

L'instruction PULSEGEN sert à construire un régulateur PID à sortie d'impulsion pour actionneurs proportionnels. PULSEGEN transforme la grandeur d'entrée INV (= LMN du régulateur PID) en une série d'impulsions de période constante correspondant au cycle d'actualisation de la grandeur d'entrée.

Application

L'instruction PULSEGEN permet de réaliser des régulateurs PID à une ou deux positions avec modulation de largeur d'impulsion. La fonction est combinée le plus souvent avec le régulateur continu CONT_C.



Appel

L'instruction PULSEGEN dispose d'une routine d'initialisation exécutée lorsque le paramètre d'entrée COM_RST = TRUE. Toutes les sorties sont mises à zéro. Après exécution de la routine d'initialisation, il faut mettre COM_RST = FALSE.

Les valeurs du bloc de régulation sont calculées correctement uniquement si le bloc est appelé à intervalles réguliers. C'est pourquoi il convient d'appeler les blocs de régulation dans un OB d'alarme cyclique (OB 30 à OB 38). Vous définissez le temps d'échantillonnage avec le paramètre CYCLE.

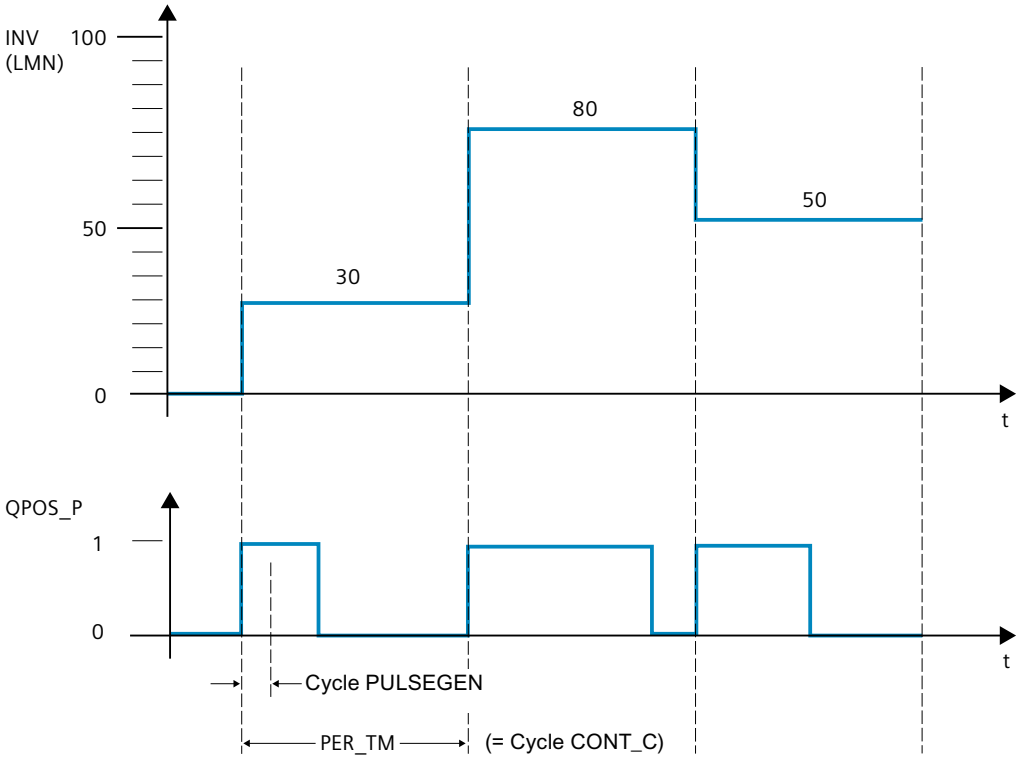
Comportement en cas d'erreur

Le mot de signalisation d'erreur RET_VAL n'est pas évalué par le bloc.

10.4.3.2 Fonctionnement PULSGEN

Modulation de la largeur d'impulsion

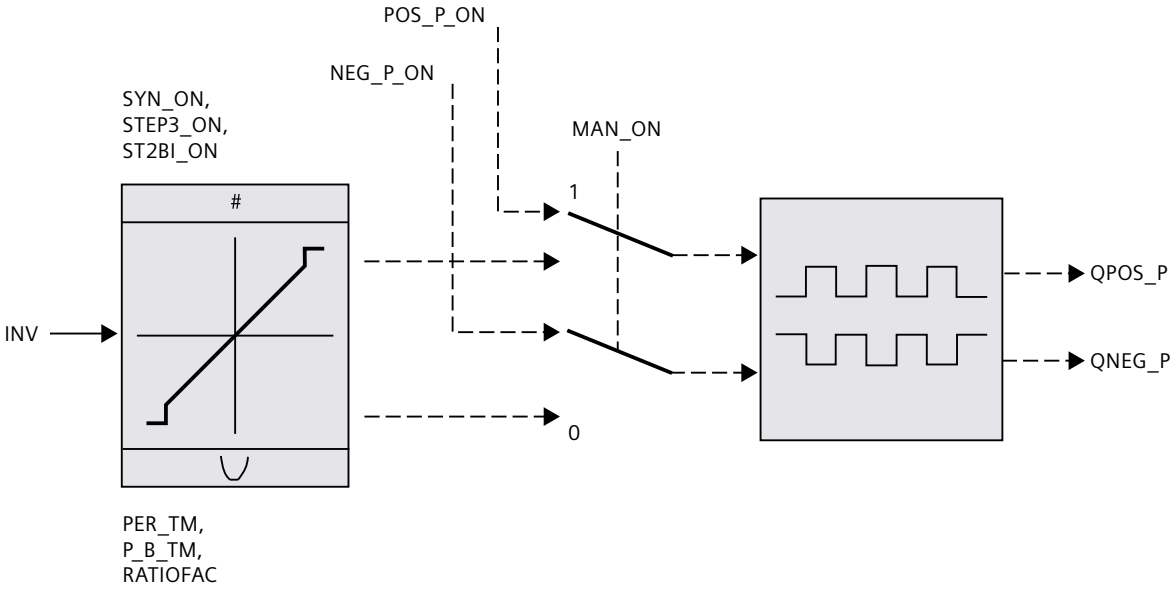
La durée d'une impulsion par durée de période est proportionnelle au paramètre d'entrée. Mais le cycle paramétré avec PER_TM n'est pas égal au cycle de traitement de l'instruction PULSEGEN. Un cycle PER_TM se compose de plusieurs cycles de traitement de l'instruction PULSEGEN, le nombre d'appels de "PULSEGEN" par cycle PER_TM étant représentatif de la précision de la largeur d'impulsion.



Une grandeur d'entrée de 30 % et 10 appels PULSEGEN par PER_TM signifient par conséquent :

- "1" à la sortie QPOS_P pour les trois premiers appels de PULSEGEN (30% de 10 appels),
- "0" à la sortie QPOS_P pour les sept appels suivants de PULSEGEN (70% de 10 appels).

Schéma fonctionnel



Précision de la grandeur réglante

Avec un "rapport d'échantillonnage" de 1:10 (appels de CONT_C par rapport aux appels de PULSEGEN), la précision de la valeur de réglage est limitée à 10% dans cet exemple, c'est-à-dire que les valeurs d'entrée transmises INV ne peuvent être converties en longueurs d'impulsion sur la sortie QPOS_P que par pas de 10%.

La précision augmente donc avec le nombre d'appels de PULSEGEN par appel de CONT_C.

Si, par exemple, PULSEGEN est appelé 100 fois plus que CONT_C, on atteint une résolution de 1% de l'étendue de la valeur de réglage.

REMARQUE

La programmation du rapport des deux types de cycles d'appel incombe à l'utilisateur.

Synchronisation automatique

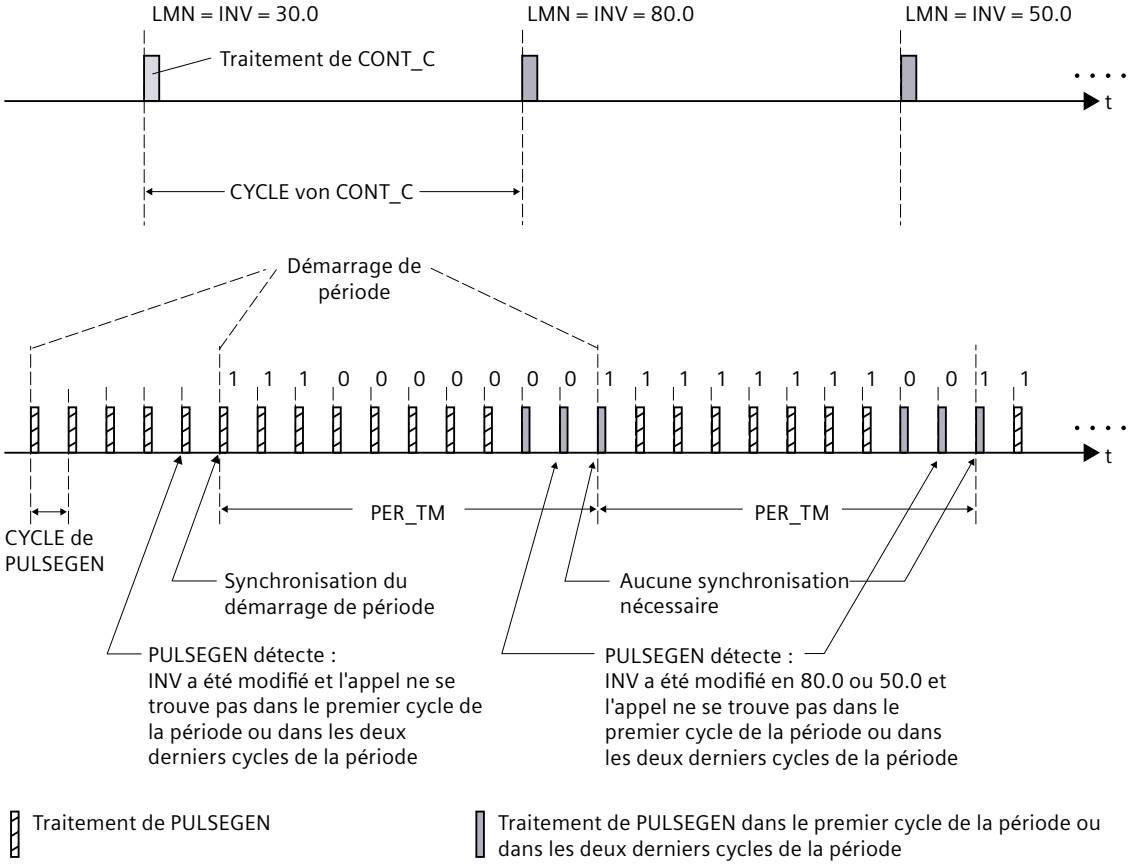
Vous pouvez synchroniser automatiquement la sortie d'impulsion avec le bloc qui met à jour la grandeur de sortie INV (par ex. CONT_C). De cette manière, vous garantissez qu'une grandeur d'entrée se modifiant sera sortie sous forme d'impulsion le plus rapidement possible.

Le formateur des impulsions exploite toujours la grandeur d'entrée INV à intervalles déterminés par la durée de période PER_TM et il transforme la valeur en une impulsion de la durée correspondante.

Mais comme INV est calculée le plus souvent dans un niveau d'alarme d'horloge plus lent, il est préférable que le formateur d'impulsions commence le plus vite possible après la mise à jour de INV à transformer la valeur discrète en une impulsion.

A cet effet, le bloc peut synchroniser lui-même le démarrage de la période selon le procédé suivant :

Quand INV a changé et que l'appel de bloc ne se trouve pas dans le premier ou dans les deux derniers cycles d'appel d'une période, une synchronisation est effectuée. La durée d'impulsion est calculée de nouveau et la sortie commence dès le prochain cycle avec une nouvelle période.



La synchronisation automatique est désactivée si SYN_ON = FALSE.

REMARQUE

Avec le début de la nouvelle période, l'ancienne valeur de INV (c'est-à-dire de LMN) est transposée de manière plus ou moins précise en le signal d'impulsion, une fois la synchronisation effectuée.

10.4.3.3 Mode de fonctionnement PULSGEN

Modes de fonctionnement

Selon le paramétrage du formateur d'impulsions, vous pouvez configurer des régulateurs PID avec soit une action à trois positions, soit une sortie à deux positions bipolaire ou unipolaire. Le tableau ci-après montre comment régler les combinaisons de commutateurs pour obtenir les différents modes.

Mode de fonctionnement	MAN_ON	STEP3_ON	ST2BI_ON
Régulation à trois échelons	FALSE	TRUE	indifférente
Régulation à deux échelons avec plage de réglage bipolaire (-100 % ... 100 %)	FALSE	FALSE	TRUE
Régulation à deux échelons avec plage de réglage unipolaire (0 % ... 100 %)	FALSE	FALSE	FALSE
Mode manuel	TRUE	indifférente	indifférente

Mode manuel en régulation à deux ou à trois positions

En mode manuel (MAN_ON = TRUE), les sorties binaires du régulateur à deux ou à trois positions peuvent être forcées au moyen des signaux POS_P_ON et NEG_P_ON indépendamment de INV.

Régulation	POS_P_ON	NEG_P_ON	QPOS_P	QNEG_P
Régulation à trois échelons	FALSE	FALSE	FALSE	FALSE
	TRUE	FALSE	TRUE	FALSE
	FALSE	TRUE	FALSE	TRUE
	TRUE	TRUE	FALSE	FALSE
Régulation à deux échelons	FALSE	indifférente	FALSE	TRUE
	TRUE	indifférente	TRUE	FALSE

10.4.3.4 Régulation à trois échelons

Régulation à trois échelons

En mode de fonctionnement "Régulation à trois échelons", il est possible de générer trois états du signal de réglage. A cet effet, les états des sorties binaires QPOS_P et QNEG_P sont affectés aux états de fonctionnement respectifs de l'actionneur. Le tableau suivant propose l'exemple d'une régulation thermique :

Signaux de sortie	chauffer	inactif	refroidir
QPOS_P	TRUE	FALSE	FALSE
QNEG_P	FALSE	FALSE	TRUE

La durée d'impulsion est calculée sur la base de la grandeur d'entrée par l'intermédiaire d'une caractéristique. Le tracé de cette caractéristique est défini par la durée minimum d'impulsion ou de pause et par le facteur de rapport. La valeur usuelle du facteur de rapport est de 1.

Les points d'inflexion des caractéristiques sont causés par la durée d'impulsion minimale et/ou la durée de pause minimale.

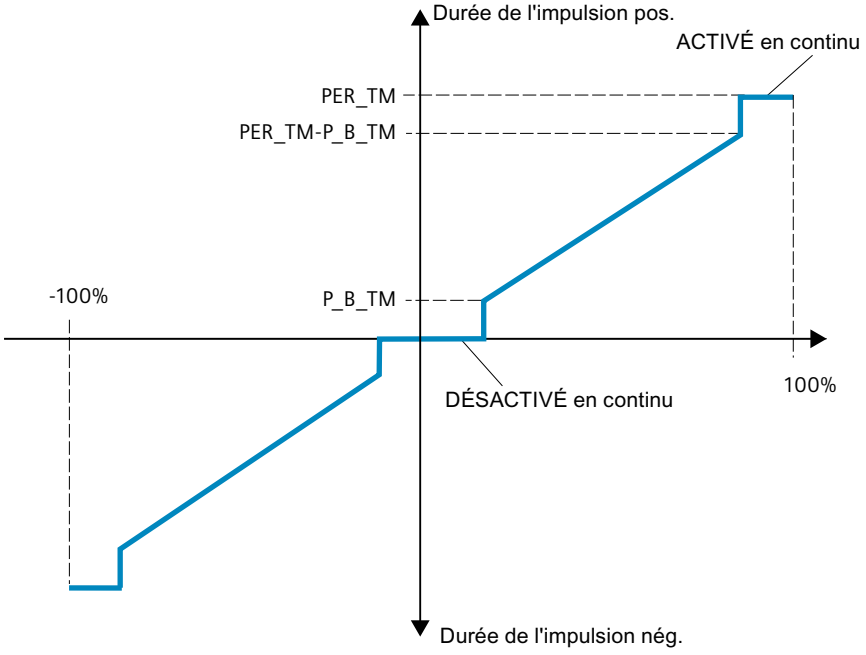
Durée d'impulsion minimale ou durée de pause minimale

Une durée d'impulsion minimale et/ou durée de pause minimale P_B_TM"minPulseIdleTime" bien paramétrée permet d'empêcher des temps d'activation ou de désactivation courts qui sont susceptibles de limiter la durée de vie des appareillages de commutation et des actionneurs. Les valeurs absolues basses de la grandeur d'entrée LMN, qui créeraient une durée d'impulsion inférieure à P_B_TM, sont rejetées. Les grandeurs d'entrée élevées, qui créeraient une durée d'impulsion supérieure à (PER_TM - P_B_TM), sont forcées à 100 % ou à -100 %.

La durée des impulsions positives ou négatives est obtenue en multipliant la grandeur d'entrée (en %) par la période :

Durée impulsion = INV / 100 * PER_TM

La figure suivante illustre une caractéristique symétrique du régulateur à trois positions (facteur de rapport = 1)



Régulation à trois échelons asymétrique

Vous pouvez modifier le rapport de la durée des impulsions positives à celle des impulsions négatives grâce au facteur de rapport RATIOFAC. Dans le cas d'un processus thermique, vous pouvez ainsi, par exemple, tenir compte de constantes de temps différentes pour le chauffage et le refroidissement.

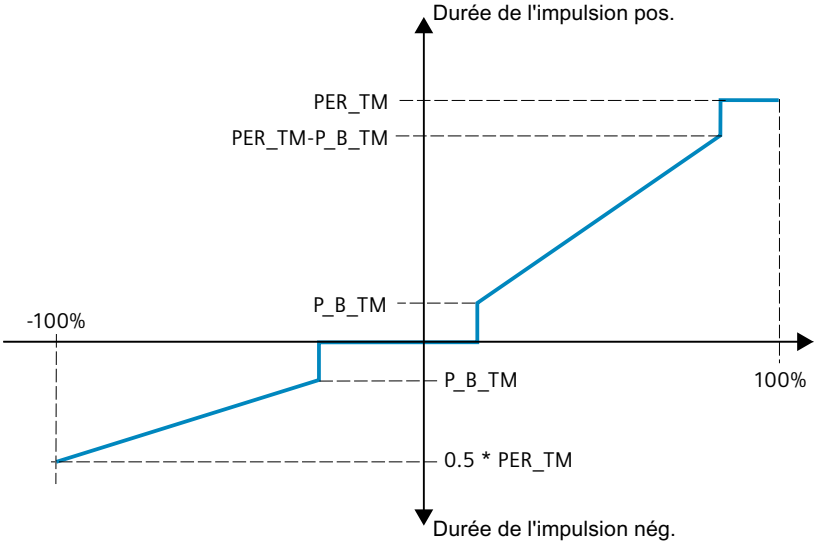
Facteur de rapport < 1

La durée d'impulsion calculée à la sortie d'impulsion négative, obtenue en multipliant la grandeur d'entrée par la période, est multipliée par le facteur de rapport.

Durée d'impulsion positive = INV /100 * PER_TM

Durée d'impulsion négative = INV / 100 * PER_TM * RATIOFAC

La figure suivante illustre une caractéristique asymétrique du régulateur à trois positions (facteur de rapport = 0,5)



Facteur de rapport > 1

La durée d'impulsion calculée à la sortie d'impulsion négative, obtenue en multipliant la grandeur d'entrée par la période, est divisée par le facteur de rapport.

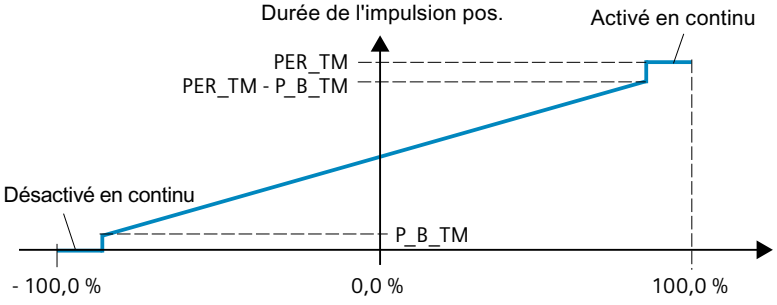
Durée d'impulsion positive = INV / 100 * PER_TM / RATIOFAC

Durée d'impulsion négative = INV / 100 * PER_TM

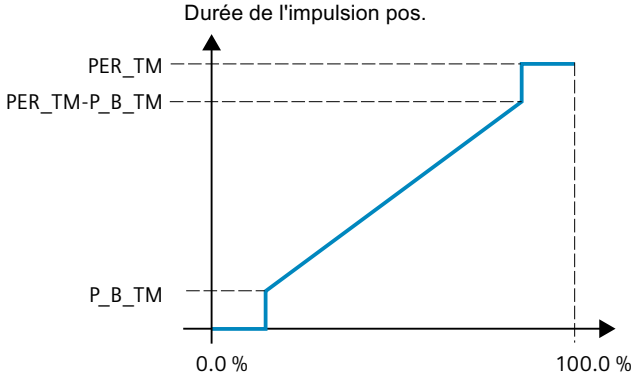
10.4.3.5 Régulation à deux échelons

Dans le cas de la régulation à deux positions, seule la sortie d'impulsions positives QPOS_P de PULSEGEN est reliée à l'actionneur de mise en marche/à l'arrêt concerné. Selon la plage de valeurs de réglage utilisée, le régulateur à deux positions a une plage de valeurs de réglage bipolaire ou unipolaire.

Régulation à deux échelons avec plage de grandeur réglante bipolaire (-100%...100%)



Régulation à deux échelons avec plage de grandeur réglante monopolaire (0 %...100 %)



QNEG_P fournit la sortie inversée au cas où la connexion du régulateur à deux positions dans la boucle de régulation exigerait un signal binaire logiquement inversé pour les impulsions de réglage.

Impulsion	Actionneur activé	Actionneur désactivé
QPOS_P	TRUE	FALSE
QNEG_P	FALSE	TRUE

10.4.3.6 Paramètre d'entrée PULSEGEN

Les valeurs des paramètres d'entrée ne sont pas limitées dans le bloc ; les paramètres ne sont pas vérifiés.

Tableau 10-17

Paramètre	Type de données	Valeur par défaut	Description
INV	REAL	0.0	Une grandeur de réglage analogique est appliquée à ce paramètre d'entrée. Les valeurs autorisées sont comprises entre -100 et 100 %.
PER_TM	TIME	T#1s	La période constante de la modulation de largeur d'impulsion est entrée à ce paramètre d'entrée. Elle correspond au temps d'échantillonnage du régulateur. Le rapport du temps d'échantillonnage du formateur d'impulsions à celui du régulateur détermine la précision de la modulation de largeur d'impulsion. PER_TM >=20*CYCLE
P_B_TM	TIME	T#50ms	Le paramètre "durée minimale d'impulsion ou de pause" permet de paramétrer le temps d'impulsion ou de pause minimal. P_B_TM >= CYCLE
RATIOFAC	REAL	1.0	Ce paramètre d'entrée permet de modifier le rapport des impulsions négatives aux impulsions positives. Ceci permet, dans le cas d'un processus thermique, de compenser des constantes de temps de chauffage et de refroidissement différentes (par ex. un processus avec chauffage électrique et refroidissement par eau). Les valeurs autorisées sont comprises entre 0.1 et 10.0.
STEP3_ON	BOOL	TRUE	Ce paramètre d'entrée permet d'activer le mode de fonctionnement concerné. Pour une régulation à trois positions, les deux signaux de sortie travaillent.
ST2BI_ON	BOOL	FALSE	Ce paramètre d'entrée permet de choisir entre les modes de fonctionnement "Régulation à deux positions pour plage de valeurs de réglage bipolaire " et "Régulation à deux positions pour plage de valeurs de réglage unipolaire". Il faut que STEP3_ON = FALSE.

Paramètre	Type de données	Valeur par défaut	Description
MAN_ON	BOOL	FALSE	Quand ce paramètre d'entrée est à 1, les sorties peuvent être forcées manuellement.
POS_P_ON	BOOL	FALSE	En mode manuel de régulation à trois positions, ce paramètre d'entrée permet de commander le signal de sortie QPOS_P. En mode manuel de régulation à deux positions, QNEG_P est toujours défini à l'inverse de QPOS_P.
NEG_P_ON	BOOL	FALSE	En mode manuel de régulation à trois positions, ce paramètre d'entrée permet de commander le signal de sortie QNEG_P. En mode manuel de régulation à deux positions, QNEG_P est toujours défini à l'inverse de QPOS_P.
SYN_ON	BOOL	TRUE	Grâce à la mise à 1 du paramètre d'entrée "Activation de la synchronisation", vous pouvez synchroniser automatiquement la sortie d'impulsion avec le bloc qui met à jour la grandeur d'entrée INV. Ceci garantit qu'une grandeur d'entrée modifiée sera sortie le plus vite possible sous forme d'impulsion.
COM_RST	BOOL	FALSE	Le bloc a un sous-programme d'initialisation qui est exécuté quand cette entrée est à 1.
CYCLE	TIME	T#10ms	Le temps entre deux appels du bloc doit être constant. Il est indiqué par cette entrée. CYCLE >= 1ms

10.4.3.7 Paramètre de sortie PULSEGEN

Tableau 10-18

Paramètre	Type de données	Valeur par défaut	Description
QPOS_P	BOOL	FALSE	Ce paramètre de sortie est mis à 1 quand il s'agit de sortir une impulsion. En cas de régulation à trois positions, c'est l'impulsion positive. En cas de régulation à deux positions, QNEG_P est toujours défini à l'inverse de QPOS_P.
QNEG_P	BOOL	FALSE	Ce paramètre de sortie est à mis 1 quand il s'agit de sortir une impulsion. En cas de régulation à trois positions, c'est l'impulsion négative. En cas de régulation à deux positions, QNEG_P est toujours défini à l'inverse de QPOS_P.

10.4.4 TCONT_CP

10.4.4.1 Description TCONT_CP

L'instruction TCONT_CP sert à la régulation de processus thermiques au moyen d'une commande continue ou pulsée. Le fonctionnement est basé sur un algorithme de régulation PID complété par des fonctions spécifiques aux processus thermiques. Une plage de régulation et une fonction de réduction de l'action P en cas d'échelon de consigne permettent d'optimiser la thermorégulation.
Grâce à son module d'optimisation, l'instruction est capable de régler elle-même les paramètres PI/PID.

Application

Il ne dessert qu'un seul actionneur, c'est-à-dire il permet soit uniquement de chauffer soit uniquement de refroidir. Dans le cas d'un processus de refroidissement, vous devez affecter une valeur négative au paramètre GAIN. La conséquence de cette inversion de régulateur est qu'en cas d'augmentation de température par ex., la grandeur réglante LMN et donc la puissance de refroidissement augmentent.

Appel

L'instruction TCONT_CP doit être appelée de manière équidistante. Veuillez utiliser une alarme cyclique (par ex. OB35 pour S7-300).
L'instruction TCONT_CP dispose d'une routine d'initialisation exécutée lorsque le paramètre d'entrée COM_RST = TRUE. Au moment de l'initialisation, l'intégrateur est mis sur la valeur I_ITVAL. Toutes les autres sorties sont mises à zéro. Après la routine d'initialisation, le bloc remet COM_RST sur FALSE. Si vous souhaitez une initialisation au démarrage de la CPU, appelez le bloc dans l'OB 100 avec COM_RST = TRUE.

Voir aussi

Mode de fonctionnement générateur d'impulsion (Page 449)
Schéma fonctionnel TCONT_CP (Page 452)

10.4.4.2 Fonctionnement TCONT_CP

Branche de consigne

La consigne est spécifiée sous forme de valeur physique à virgule flottante ou sous forme de pourcentage à l'entrée SP_INT. La consigne et la mesure intervenant dans le calcul du signal d'écart doivent avoir la même unité.

Sélection de la mesure (PVPER_ON)

Le format de la mesure est choisi en fonction de PVPER_ON : périphérie ou virgule flottante.

PVPER_ON	Entrée de la mesure
TRUE	La mesure est lue à l'entrée PV_PER via la périphérie analogique (PEW xxx).
FALSE	La mesure est lue en format à virgule flottante à l'entrée PV_IN.

Conversion du format de la mesure CRP_IN (PER_MODE)

La fonction CRP_IN effectue la conversion de la valeur de périphérie PV_PER en un format à virgule flottante en fonction du commutateur PER_MODE en appliquant la règle suivante :

PER_MODE	Sortie de CRP_IN	Type d'entrée analogique	Unité
0	PV_PER * 0.1	Thermocouples ; PT100/NI100 ; standard	°C;°F
1	PV_PER * 0.01	PT100/NI100 ; climat ;	°C;°F
2	PV_PER * 100/27648	Tension / courant	%

Normalisation de la mesure PV_NORM (PF_FAC, PV_OFFS)

La fonction PV_NORM calcule la sortie de CRP_IN selon la règle suivante :
Sortie de PV_NORM = (sortie de CRP_IN) * PV_FAC + PV_OFFS
Domaine d'application de cette règle :

- Conversion de la mesure avec PV_FAC comme facteur de mesure et PV_OFFS comme décalage de la mesure
- Normalisation d'une température en pourcentage
  Vous souhaitez entrer la consigne sous forme de pourcentage et devez à présent convertir la valeur de température mesurée en pourcentage.
- Normalisation d'un pourcentage en température
  Vous souhaitez entrer la consigne sous forme de la grandeur physique température et devez à présent convertir la valeur de tension/courant mesurée en température.

Calcul des paramètres :

- PV_FAC = plage de PV_NORM/ plage de CRP_IN;
- PV_OFFS = UG(PV_NORM) - PV_FAC * UG(CRP_IN);
  avec UG : Limite inférieure

Les valeurs par défaut (PV_FAC = 1.0 et PV_OFFS = 0.0) permettent de désactiver la normalisation. La mesure opérante est fournie à la sortie PV.

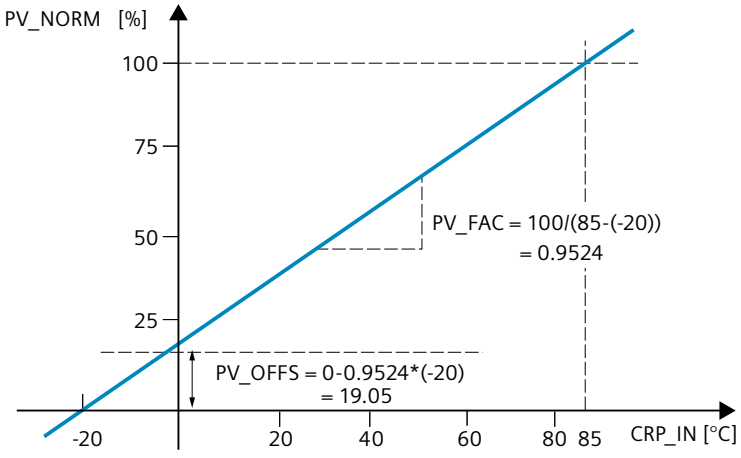
REMARQUE

Pour une régulation à impulsions, la mesure doit être transmise au bloc dans l'appel d'impulsion rapide (explication : calcul de la valeur moyenne). Dans le cas contraire, la qualité de la régulation risque de se détériorer.

Exemple de normalisation de la mesure

Si vous souhaitez spécifier la consigne sous forme d'un pourcentage et que la plage de températures est comprise entre -20 et 85 °C au niveau de CRP_IN, vous devez convertir la plage de température en pourcentage.

La figure suivante présente un exemple de conversion de la plage de température -20 à 85 ° à la plage interne 0 à 100 % :



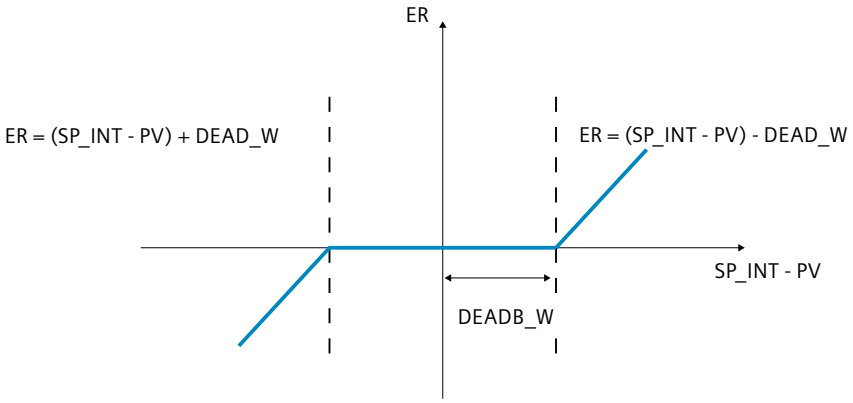
Calcul du signal d'écart

Le signal d'écart précédant la zone morte correspond à la différence entre la consigne et la mesure.

La consigne et la mesure doivent avoir la même unité.

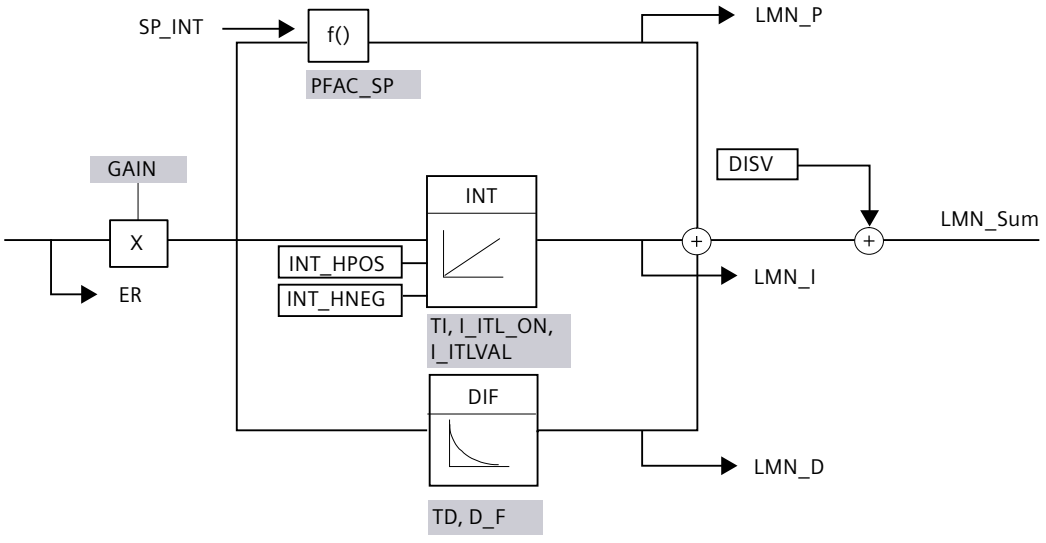
Zone morte (DEADB_W)

Le signal d'écart comprend une zone morte (DEADBAND) qui permet de neutraliser une oscillation continue de faible amplitude due à la quantification des grandeurs réglantes (p. ex. en cas de modulation de largeur d'impulsion PULSEGEN). Lorsque DEADB_W = 0.0, la zone morte est désactivée. Le signal d'écart effectif est indiqué par le paramètre ER.



Algorithme PID

La figure suivante représente le schéma fonctionnel de l'algorithme PID.

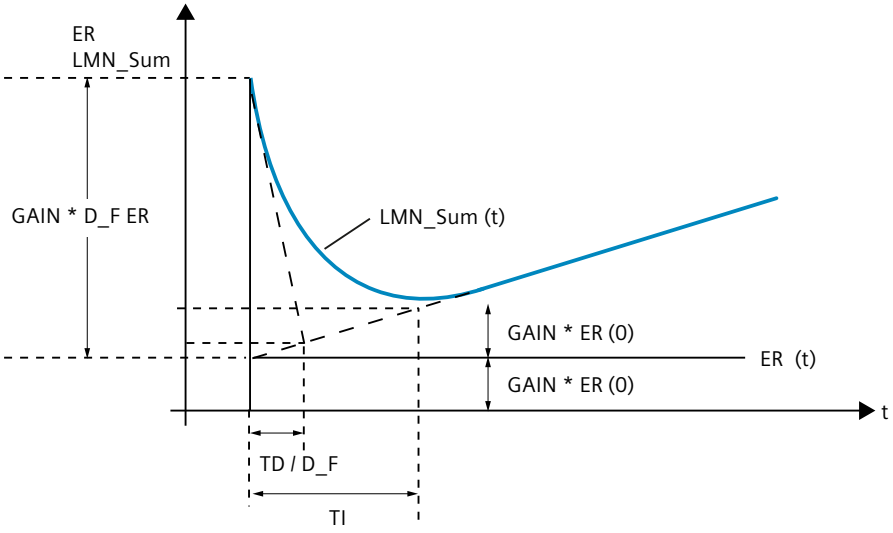


Algorithme PID (GAIN, TI, TD, D_F)

L'algorithme PID fonctionne comme un algorithme de position. Les actions proportionnelle, par intégration (INT) et par dérivation (DIF) sont montées en parallèle et peuvent être activées et désactivées individuellement. Il est donc possible de paramétrer des régulateurs P, PI, PD et PID.
La fonction d'optimisation de la régulation prend en charge les régulateurs PI et PID.
L'inversion de la régulation s'obtient par un GAIN négatif (refroidissement).
En mettant TI et TD à 0.0, vous obtenez un régulateur P pur au point de fonctionnement.
La réponse indicielle dans la plage de temps correspond à :

$$LMN\_Sum(t) = GAIN \cdot ER(0) \cdot \left(1 + \frac{1}{TI} \cdot t + D\_F \cdot e^{\frac{-t}{TD\ /\ D\_F}}\right)$$

sachant que :
LMN_Sum(t) grandeur réglante lors du fonctionnement automatique du régulateur
ER (0) pas du signal d'écart normé
GAIN gain du régulateur
TI temps d'intégration
TD temps de dérivation
D_F facteur de dérivation



Intégrateur (TI, I_ITL_ON, I_ITLVAL)

En mode manuel, il correspond à : LMN_I = LMN - LMN_P - DISV.
En cas de limitation de la valeur de réglage, l'action I est arrêtée. Elle est à nouveau activée lorsque le signal d'écart rapproche l'action par intégration I de la plage de réglage interne.
Les mesures suivantes permettent également de modifier l'action par intégration I.

- La désactivation de l'action par intégration I du régulateur s'obtient avec TI = 0.0
- Atténuation de l'action proportionnelle P en cas de modification de la consigne
- Plage de régulation
- Les limites de la valeur de réglage peuvent être modifiées en ligne

Atténuation de l'action proportionnelle P en cas de modification de la consigne (PFAC_SP)

Pour éviter un dépassement, vous pouvez atténuer l'action proportionnelle P avec le paramètre "Coefficient d'action proportionnelle en cas de modification de la consigne" (PFAC_SP). PFAC_SP vous permet de sélectionner toute valeur comprise entre 0.0 et 1.0 pour spécifier l'importance de l'action proportionnelle P en cas de modification de la consigne :

- PFAC_SP = 1.0 : en cas de modification de la consigne, l'action proportionnelle P est totalement opérante
- PFAC_SP = 0.0 : en cas de modification de la consigne, l'action proportionnelle P n'est pas opérante

L'atténuation de l'action P s'obtient par compensation de l'action I.

Dérivateur (TD, D_F)

- La désactivation de l'action D du régulateur s'obtient avec TD = 0.0
- Lorsque l'action D est activée, il convient de respecter l'équation suivante :
  TD = 0.5 * CYCLE * D_F

Paramétrage d'un régulateur P ou PD avec point de fonctionnement

Dans l'interface de paramétrage, désactivez l'action par intégration I (TI = 0.0) et le cas échéant, l'action par dérivation D (TD = 0.0). Effectuez également le paramétrage suivant :

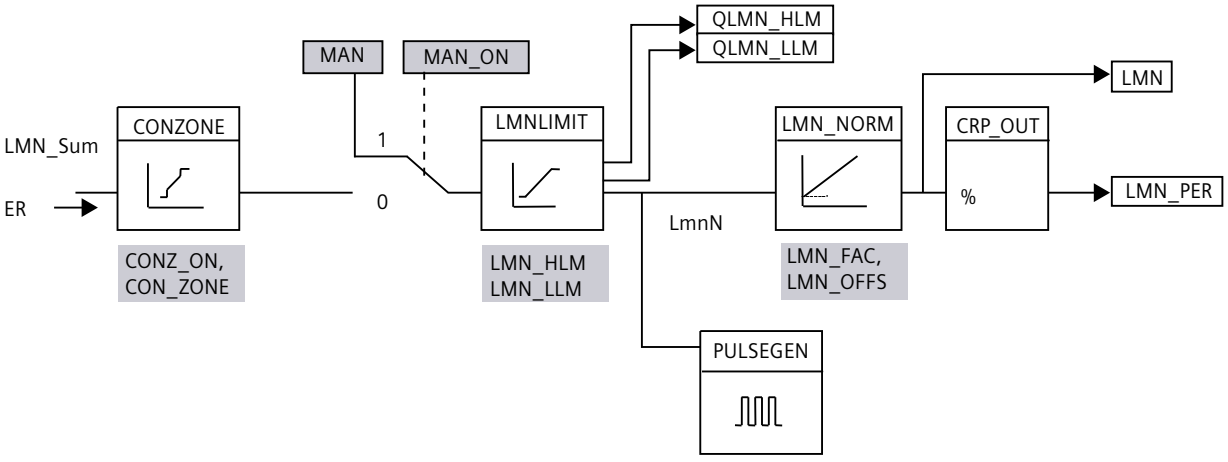
- I_ITL_ON = TRUE
- I_ITLVAL = point de fonctionnement ;

Action anticipatrice (DISV)

Vous pouvez additionner une grandeur perturbatrice à l'entrée DISV.

Calcul de la valeur de réglage

La figure suivante représente le schéma fonctionnel du calcul de la valeur de réglage :



Interface de paramétrage

Interface d'appel instruction

Interface de paramétrage, interface d'appel instruction

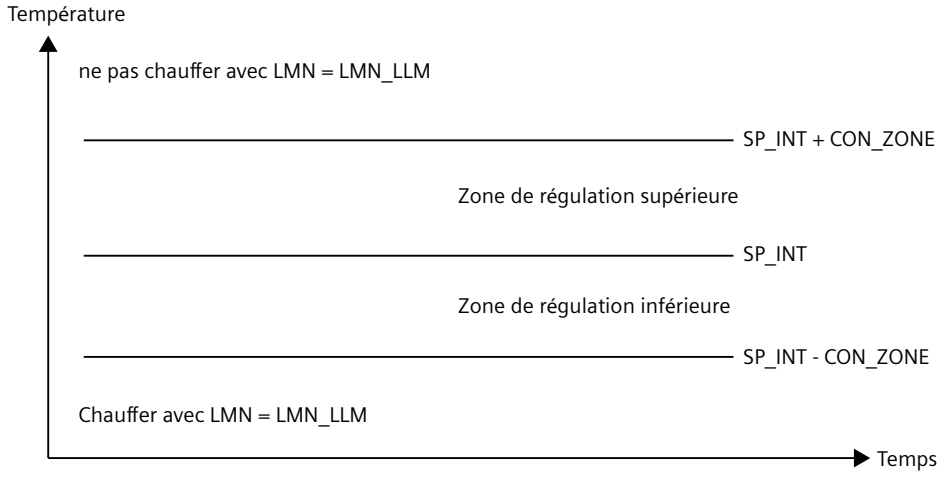
Plage de régulation (CONZ_ON, CON_ZONE)

Lorsque CONZ_ON = TRUE, le régulateur fonctionne avec une plage de régulation. Cela signifie que le régulateur fonctionne d'après l'algorithme suivant :

- Si la mesure PV dépasse la consigne SP_INT de plus de CON_ZONE, la valeur LMN_LLM est fournie comme grandeur réglante.
- Si la mesure PV est inférieure à la consigne SP_INT de plus de CON_ZONE, la valeur LMN_HLM est fournie comme grandeur réglante.
- Si la mesure PV se situe dans la plage de régulation (CON_ZONE), la grandeur réglante prend la valeur de l'algorithme PID, LMN_Sum.

REMARQUE

Le changement de la grandeur réglante de LMN_LLM ou LMN_HLM sur LMN_Sum s'effectue en respectant un hystérésis de 20 % de la plage de régulation.



REMARQUE

Avant d'activer manuellement la plage de régulation, assurez-vous que sa largeur n'est pas trop petite. Si tel est le cas, la grandeur réglante et la mesure oscilleront.

Avantage de la plage de régulation

À l'entrée dans la plage de régulation, l'action D activée entraîne une réduction très rapide de la grandeur réglante. La plage de régulation n'est donc utile que lorsque l'action D est activée. Sans plage de régulation, seule la réduction de l'action proportionnelle P permettrait de réduire la grandeur réglante. La plage de régulation conduit plus rapidement à un régime transitoire sans sur ou sous-oscillation lorsque la grandeur réglante minimale ou maximale fournie est très éloignée de la grandeur réglante stationnaire qui est requise pour le nouveau point de fonctionnement.

Mode manuel (MAN_ON, MAN)

Vous pouvez choisir entre le mode manuel et le mode automatique. En mode manuel, la grandeur réglante est ajustée en fonction d'une valeur manuelle.
L'intégrateur (INT) est forcé de manière interne à LMN - LMN_P - DISV et le dérivateur (DIF) est forcé à 0 et égalisé de manière interne. Le passage au mode automatique s'effectue donc sans à-coups.

REMARQUE

Durant l'optimisation, le paramètre MAN_ON n'est pas actif.

Limitation de la valeur de réglage LMNLIMIT (LMN_HLM, LMN_LLM)

La fonction LMNLIMIT permet de limiter la valeur de réglage entre LMN_HLM et LMN_LLM. Les bits de signalisation QLMN_HLM et QLMN_LLM indiquent que les limites sont atteintes.

En cas de limitation de la valeur de réglage, l'action I est arrêtée. Elle est à nouveau activée lorsque le signal d'écart rapproche l'action par intégration I de la plage de réglage interne.

Modification en ligne des limites de la valeur de réglage

Lorsque le domaine de la valeur de réglage est réduit et lorsque la nouvelle valeur de réglage illimitée se trouve hors des limites, l'action par intégration I et donc la valeur de réglage sont décalées.

La valeur de réglage est diminuée de la modification de sa limite. Si la valeur de réglage était illimitée avant la modification, elle prend exactement la valeur de la nouvelle limite (dans le cas présent, la description est faite pour la limite supérieure de la valeur de réglage).

Normalisation de la valeur de réglage LMN_NORM (LMN_FAC, LMN_OFFS)

La fonction LMN_NORM normalise la valeur de réglage d'après la règle suivante :

LMN = LmnN * LMN_FAC + LMN_OFFS

Domaine d'application de cette règle :

- Conversion de la valeur de réglage avec LMN_FAC comme facteur de valeur de réglage et LMN_OFFS comme décalage de valeur de réglage

La valeur de réglage est également disponible en format périphérie. La fonction CRP_OUT convertit la valeur LMN à virgule flottante en une valeur de périphérie d'après la règle suivante :

LMN_PER = LMN * 27648/100

Les valeurs par défaut (LMN_FAC = 1.0 et LMN_OFFS = 0.0) permettent de désactiver la normalisation. La valeur de réglage effective est fournie à la sortie LMN.

Sauvegarder les paramètres de régulation SAVE_PAR

Si vous estimez que les paramètres actuels pourront resservir, enregistrez-les dans des paramètres de structure spécialement prévus à cet effet dans le DB d'instance de l'instruction TCONT_CP avant de les modifier manuellement. Au cours de l'optimisation du régulateur, les paramètres enregistrés sont remplacés par les valeurs qui étaient valables avant l'optimisation.

PFAC_SP, GAIN, TI, TD, D_F, CONZ_ON et CONZONE sont écrits dans la structure PAR_SAVE.

Rechargement des paramètres enregistrés du régulateur UNDO_PAR

Les derniers paramètres de régulation sauvegardés peuvent à nouveau être activés avec cette fonction (uniquement en mode manuel).

Changement entre les paramètres PI et PID, LOAD_PID (PID_ON)

Après une optimisation, les paramètres PI et PID sont enregistrés dans les structures PI_CON et PID_CON. En mode manuel, vous pouvez remplacer les paramètres actifs par les paramètres PI ou PID avec LOAD_PID et en fonction de PID_ON.

Paramètre PID PID_ON = TRUE	Paramètres PI PID_ON = FALSE
• GAIN = PID_CON.GAIN • TI = PID_CON.TI • TD = PID_CON.TD	• GAIN = PI_CON.GAIN • TI = PI_CON.TI

REMARQUE

Les paramètres du régulateur ne peuvent être restitués avec UNDO_PAR ou LOAD_PID que si le gain du régulateur est différent de zéro :

les paramètres de LOAD_PID ne peuvent être copiés qu'à condition que GAIN <> 0 (paramètres provenant des jeux PI ou PID). Ceci permet de tenir compte du cas où aucune optimisation n'a été faite ainsi que de l'absence éventuelle de paramètres PID. Si PID_ON = TRUE et PID.GAIN = FALSE, PID_ON prend la valeur FALSE et les paramètres PI sont copiés.

- D_F, PFAC_SP sont présélectionnés automatiquement du fait de l'optimisation. Elles peuvent ensuite être modifiées par l'utilisateur. LOAD_PID ne modifie pas ces paramètres.
- Avec LOAD_PID, la plage de régulation est systématiquement recalculée (CON_ZONE = 250/GAIN), même si CONZ_ON = FALSE.

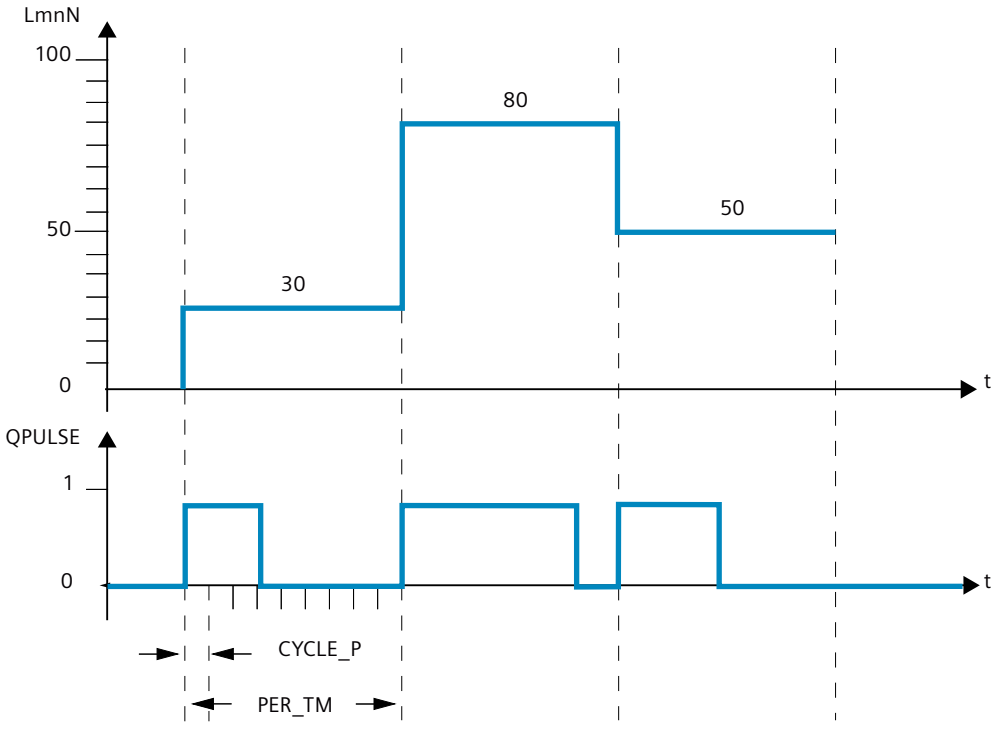
Voir aussi

Mode de fonctionnement générateur d'impulsion (Page 449)

Schéma fonctionnel TCONT_CP (Page 452)

10.4.4.3 Mode de fonctionnement générateur d'impulsion

La fonction PULSEGEN convertit la valeur de réglage analogique LmnN par la modulation de la largeur d'impulsion en une série d'impulsions de période.PER_TM. PULSEGEN est activé avec PULSE_ON = TRUE et traité dans le cycle CYCLE_P.



Une valeur de réglage LmnN = 30 % et 10 appels PULSEGEN par PER_TM signifient par conséquent :

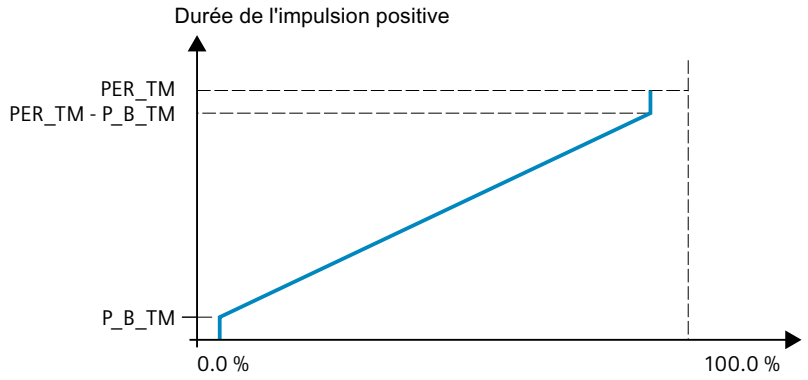
- TRUE sur la sortie QPULSE pour les trois premiers appels du PULSEGEN (30 % de 10 appels)
- FALSE sur la sortie QPULSE pour les sept appels suivants du PULSEGEN (70 % de 10 appels)

La durée d'une impulsion par période est proportionnelle à la valeur de réglage et se calcule de la manière suivante :

Durée d'impulsion = PER_TM * LmnN /100

En raison de la suppression de la durée d'impulsion minimale et/ou de la durée de pause minimale, le début et la fin de la caractéristique de déformation ont des points d'inflexion.

La figure suivante représente la régulation à deux échelons avec plage de réglage unipolaire (de 0 à 100 %) :



Durée d'impulsion minimale et/ou durée de pause minimale (P_B_TM)

De brèves durées d'activation ou de désactivation compromettent la longévité des éléments logiques et servocommandes. On peut l'éviter en paramétrant une durée d'impulsion minimale et/ou une durée de pause minimale P_B_TM.

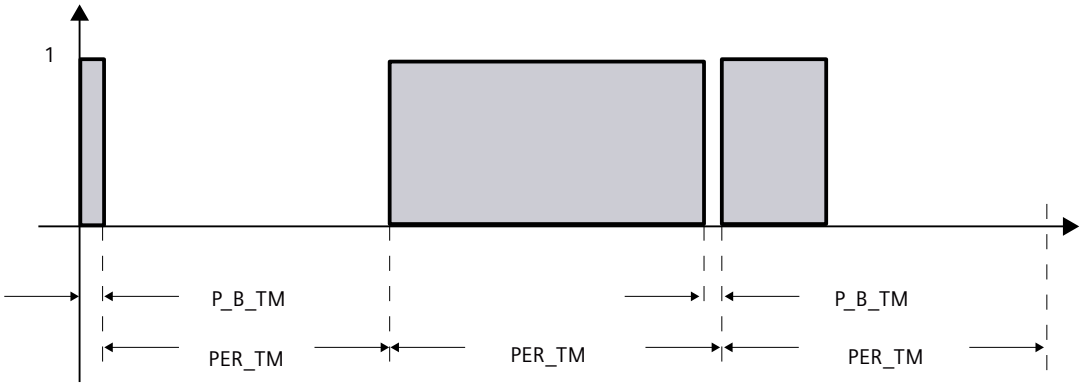
Les valeurs absolues basses de la grandeur d'entrée LmnN, qui créeraient une durée d'impulsion inférieure à P_B_TM, sont rejetées.

Les grandeurs d'entrée élevées, qui créeraient une durée d'impulsion supérieure à PER_TM - P_B_TM, sont mises sur 100 %. Cette mesure permet de réduire la dynamique de la mise en forme d'impulsions.

Consignes recommandées pour la durée d'impulsion minimale et/ou la durée de pause minimale : P_B_TM ≤ 0,1 * PER_TM.

Les points d'inflexion des caractéristiques dans la figure ci-dessus sont causés par la durée d'impulsion minimale et/ou la durée de pause minimale.

La figure suivante représente le comportement de la sortie d'impulsion :



Précision de la mise en forme d'impulsions

Plus la largeur d'impulsions CYCLE_P est petite par rapport à la période PER_TM, plus la modulation de largeur d'impulsions est précise. Pour obtenir une régulation suffisamment précise, il est conseillé de respecter le rapport suivant :
CYCLE_P ≤ PER_TM/50
Ainsi, la valeur de réglage est convertie en impulsions avec une résolution de ≤ 2 %.

REMARQUE

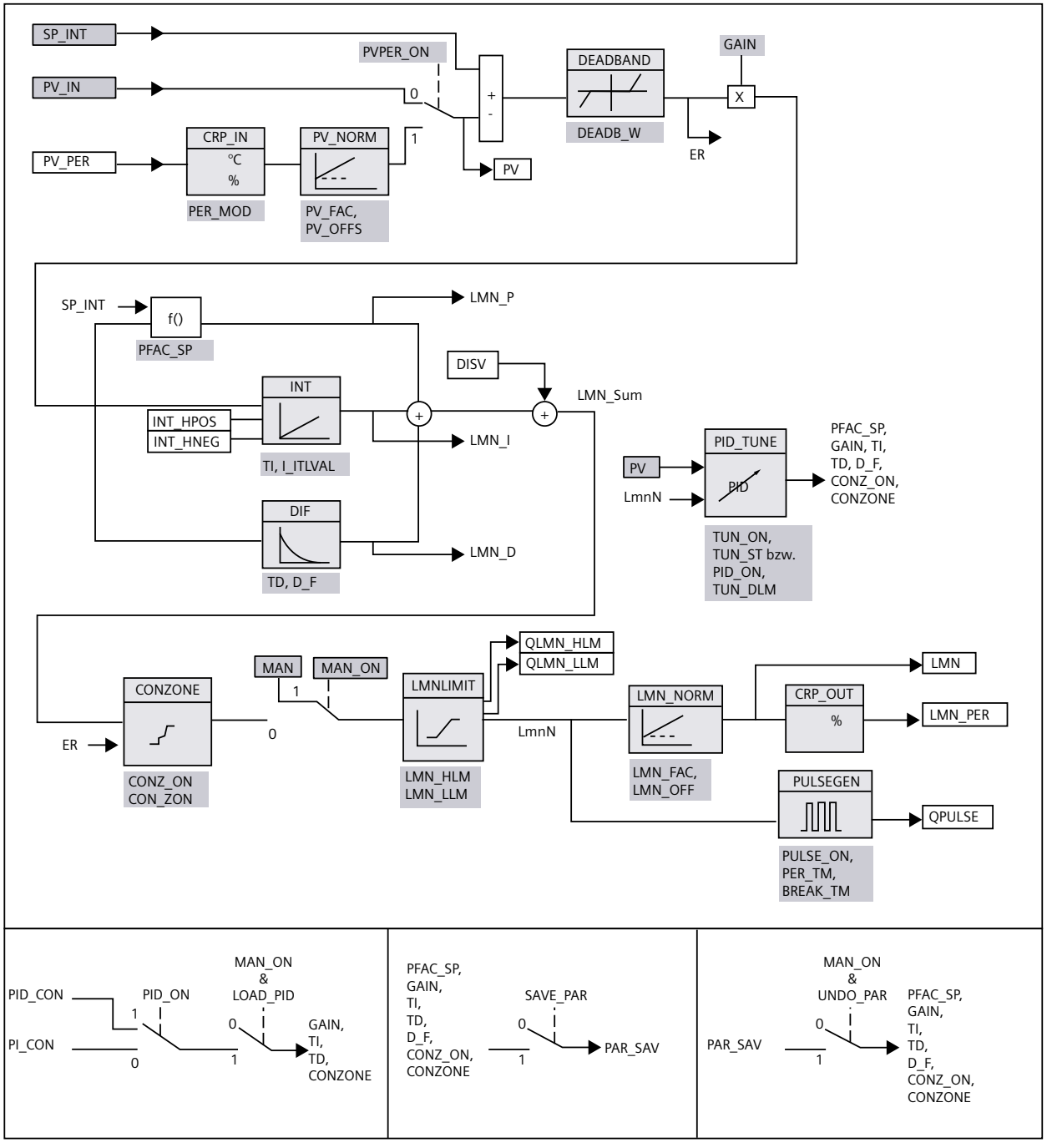
Si vous appelez le régulateur pendant le cycle du formateur d'impulsions, vous devez tenir compte du fait suivant :

En cas d'appel du régulateur pendant le cycle du formateur d'impulsions, la mesure moyenne est calculée, Ceci peut avoir pour conséquence une divergence sur la sortie PV des valeurs de celles sur l'entrée PV_IN et/ou PV_PER. Si vous souhaitez réaliser une consigne suiveuse, vous devez enregistrer la mesure au paramètre d'entrée PV_IN au moment des appels du traitement global de régulation ((QC_ACT = TRUE)). Pour les appels intermédiaires du formateur d'impulsions, vous affectez cette mesure enregistrée aux paramètres d'entrée PV_IN et SP_INT.

Voir aussi

Description TCONT_CP (Page 439)
Fonctionnement TCONT_CP (Page 440)
Schéma fonctionnel TCONT_CP (Page 452)
Paramètres d'entrée TCONT_CP (Page 453)
Paramètre de sortie TCONT_CP (Page 454)
Paramètres d'entrée/sortie TCONT_CP (Page 455)
Variables statiques TCONT_CP (Page 455)
Paramètres STATUS_H (Page 460)
Paramètre STATUS_D (Page 461)

10.4.4.4 Schéma fonctionnel TCONT_CP



Voir aussi

Description TCONT_CP (Page 439)
Fonctionnement TCONT_CP (Page 440)
Mode de fonctionnement générateur d'impulsion (Page 449)
Paramètres d'entrée TCONT_CP (Page 453)
Paramètre de sortie TCONT_CP (Page 454)
Paramètres d'entrée/sortie TCONT_CP (Page 455)
Variables statiques TCONT_CP (Page 455)
Paramètres STATUS_H (Page 460)
Paramètre STATUS_D (Page 461)

10.4.4.5 Paramètres d'entrée TCONT_CP

Les noms des paramètres suivants sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Tableau 10-19

Paramètre	Adresse	Type de données	Valeur par défaut	Description
PV_IN	0.0	REAL	0.0	L'entrée "Entrée mesure" permet de paramétrer une valeur de mise en service ou d'interconnecter une mesure externe au format en virgule flottante. Les valeurs correctes dépendent des capteurs utilisés.
PV_PER	4.0	INT	0	La mesure en format périphérie est interconnectée au régulateur à l'entrée "Mesure périphérie".
DISV	6.0	REAL	0.0	Pour une action anticipatrice, la perturbation est appliquée à l'entrée "perturbation".
INT_HPOS	10.0	BOOL	FALSE	La sortie de l'intégrateur peut être maintenue dans le sens positif. Pour ce faire, l'entrée INT_HPOS doit être mise à TRUE. Pour une régulation en cascade, la sortie INT_HPOS du régulateur pilote doit être connectée à l'entrée QLMN_HLM du régulateur asservi.
INT_HNEG	10.1	BOOL	FALSE	La sortie de l'intégrateur peut être maintenue dans le sens négatif. Pour ce faire, l'entrée INT_HNEG doit être mise à TRUE. Pour une régulation en cascade, la sortie INT_HNEG du régulateur pilote doit être connectée à l'entrée QLMN_LLM du régulateur asservi.
SELECT	12.0	INT	0	Lorsque le formateur d'impulsions est activé, il existe plusieurs possibilités d'appeler l'algorithme PID et le formateur d'impulsions : • SELECT = 0 : le régulateur est appelé dans une alarme cyclique rapide ; l'algorithme PID et la mise en forme des impulsions sont traités. • SELECT = 1 : le régulateur est appelé dans l'OB1 ; seul l'algorithme PID est traité. • SELECT = 2 : le régulateur est appelé dans une alarme cyclique rapide ; seul la mise en forme des impulsions est traitée. • SELECT = 3 : le régulateur est appelé dans une alarme cyclique plus lente ; seul l'algorithme PID est traité.

Voir aussi

Mode de fonctionnement générateur d'impulsion (Page 449)
Schéma fonctionnel TCONT_CP (Page 452)

10.4.4.6 Paramètre de sortie TCONT_CP

Les noms des paramètres suivants sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Tableau 10-20

Paramètre	Adresse	Type de données	Valeur par défaut	Description
PV	14.0	REAL	0.0	La mesure opérante est fournie à la sortie "Mesure". Les valeurs correctes dépendent des capteurs utilisés.
LMN	18.0	REAL	0.0	La valeur de réglage effective en format à virgule flottante est fournie à la sortie "valeur de réglage".
LMN_PER	22.0	INT	0	La valeur de réglage en format périphérie est connectée au régulateur à la sortie "valeur de réglage périphérie".
QPULSE	24.0	BOOL	FALSE	La valeur de réglage est fournie à la sortie QPULSE en modulation de largeur d'impulsions.
QLMN_HLM	24.1	BOOL	FALSE	La valeur de réglage possède toujours une limite supérieure et inférieure. La sortie QLMN_HLM signale que la limite supérieure est atteinte.
QLMN_LLM	24.2	BOOL	FALSE	La valeur de réglage possède toujours une limite supérieure et inférieure. La sortie QLMN_LLM signale que la limite inférieure est atteinte.
QC_ACT	24.3	BOOL	TRUE	Ce paramètre indique si l'action continue du régulateur sera traitée au prochain appel du bloc (uniquement significatif si SELECT = 0 ou 1).

Voir aussi

Mode de fonctionnement générateur d'impulsion (Page 449)
Schéma fonctionnel TCONT_CP (Page 452)
Paramètres STATUS_H (Page 460)
Paramètre STATUS_D (Page 461)

10.4.4.7 Paramètres d'entrée/sortie TCONT_CP

Les noms des paramètres suivants sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Tableau 10-21

Paramètre	Adresse	Type de données	Valeur par défaut	Description
CYCLE	26.0	REAL	0.1 s	Valeur par défaut du temps d'échantillonnage de l'algorithme PID. En phase 1, l'optimisateur calcule le temps d'échantillonnage et l'inscrit dans CYCLE. CYCLE > 0.001 s
CYCLE_P	30.0	REAL	0.02 s	Cette entrée permet de saisir le temps d'échantillonnage de la mise en forme des impulsions. En phase 1, l'instruction TCONT_CP calcule le temps d'échantillonnage et l'inscrit dans CYCLE_P. CYCLE_P > 0.001 s
SP_INT	34.0	REAL	0.0	L'entrée "consigne interne" permet de spécifier une consigne. Les valeurs correctes dépendent des capteurs utilisés.
MAN	38.0	REAL	0.0	L'entrée "valeur manuelle" permet de spécifier une valeur manuelle. En mode automatique, elle correspond à la valeur de réglage.
COM_RST	42.0	BOOL	FALSE	Le bloc dispose d'une routine d'initialisation exécutée lorsque l'entrée COM_RST est activée.
MAN_ON	42.1	BOOL	TRUE	La mise à 1 de l'entrée "activation du mode manuel" interrompt la boucle de régulation. La valeur manuelle MAN est spécifiée comme valeur de réglage.

Voir aussi

Mode de fonctionnement générateur d'impulsion (Page 449)
Schéma fonctionnel TCONT_CP (Page 452)

10.4.4.8 Variables statiques TCONT_CP

Les noms des variables suivantes sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Tableau 10-22

Paramètre	Adresse	Type de données	Valeur par défaut	Description
DEADB_W	44.0	REAL	0.0	Le signal d'écart parcourt une zone morte. L'entrée "Largeur de zone morte" détermine la taille de la zone morte. Les valeurs correctes dépendent des capteurs utilisés.
I_ITLVAL	48.0	REAL	0.0	La sortie de l'intégrateur peut être positionnée à l'entrée I_ITL_ON. La valeur d'initialisation est donnée par l'entrée "valeur d'initialisation pour l'action I". Lors du redémarrage COM_RST = TRUE, l'action I est mise à la valeur d'initialisation. Les valeurs autorisées sont comprises entre -100 et 100 %.
LMN_HLM	52.0	REAL	100.0	La valeur de réglage possède toujours une limite supérieure et inférieure. L'entrée "Limitation supérieure de la valeur de réglage" indique sa limitation supérieure. LMN_HLM > LMN_LLM

Paramètre	Adresse	Type de données	Valeur par défaut	Description
LMN_LLM	56.0	REAL	0.0	La valeur de réglage possède toujours une limite supérieure et inférieure. L'entrée "Limitation inférieure de la valeur de réglage" indique sa limitation inférieure. LMN_LLM < LMN_HLM
PV_FAC	60.0	REAL	1.0	L'entrée "facteur de mesure" est multipliée par la "mesure périphérie". Cette entrée permet d'adapter l'étendue de la mesure.
PV_OFFS	64.0	REAL	0.0	L'entrée "décalage de la mesure" est additionnée à la "mesure périphérie". Cette entrée permet d'adapter l'étendue de la mesure.
LMN_FAC	68.0	REAL	1.0	L'entrée "facteur de valeur de réglage" est multipliée par la valeur de réglage. Cette entrée permet d'adapter la plage de la valeur de réglage.
LMN_OFFS	72.0	REAL	0.0	L'entrée "décalage de valeur de réglage" est additionnée à la valeur de réglage. Cette entrée permet d'adapter la plage de la valeur de réglage.
PER_TM	76.0	REAL	1.0 s	La période de la modulation de largeur d'impulsions est spécifiée dans le paramètre PER_TM. La précision de la modulation de largeur d'impulsions est déterminée par le rapport entre la durée de période et le temps d'échantillonnage de la mise en forme des impulsions. PER_TM ≥ CYCLE
P_B_TM	80.0	REAL	0.02 s	Le paramètre « durée minimale d'impulsion ou de pause » permet de paramétrer le temps d'impulsion ou de pause minimal. P_B_TM est limité au niveau interne sur > CYCLE_P.
TUN_DLMN	84.0	REAL	20.0	L'activation du processus en vue de l'optimisation du régulateur est réalisée par un échelon de la valeur de réglage de TUN_DLMN. Les valeurs autorisées sont comprises entre -100 et 100 %.
PER_MODE	88.0	INT	0	Ce commutateur permet de spécifier le type de module AE. La mesure à l'entrée PV_PER est ainsi normalisée à la sortie PV de la manière suivante : • PER_MODE = 0 : Thermocouples ; PT100/NI100 ; standard PV_PER * 0.1 Unité : °C, °F • PER_MODE = 1 : PT100/NI100 ; climat PV_PER * 0.01 Unité : °C, °F • PER_MODE = 2 : Courant/tension PV_PER * 100/27648 Unité : %
PVPER_ON	90.0	BOOL	FALSE	Pour lire la mesure de la périphérie, l'entrée PV_PER doit être interconnectée à la périphérie et l'entrée "activation de la mesure périphérie" doit être mise à 1.
I_ITL_ON	90.1	BOOL	FALSE	La sortie de l'intégrateur peut être mise à la valeur de l'entrée I_ITLVAL. Pour ce faire, l'entrée "Activation de l'action I" doit être mise à 1.
PULSE_ON	90.2	BOOL	FALSE	PULSE_ON = TRUE permet d'activer le formateur d'impulsions.
TUN_KEEP	90.3	BOOL	FALSE	Le passage au mode automatique intervient seulement lorsque TUN_KEEP = FALSE.
ER	92.0	REAL	0.0	Le signal d'écart opérant est fourni à la sortie "Signal d'écart". Les valeurs correctes dépendent des capteurs utilisés.
LMN_P	96.0	REAL	0.0	La sortie "action P" correspond à l'action proportionnelle de la variable réglante.
LMN_I	100.0	REAL	0.0	La sortie "action I" correspond à l'action d'intégration de la variable réglante.
LMN_D	104.0	REAL	0.0	La sortie "action D" correspond à l'action de dérivation de la variable réglante.

Paramètre	Adresse	Type de données	Valeur par défaut	Description
PHASE	108.0	INT	0	La sortie PHASE indique la phase d'optimisation en cours. • PHASE = 0 : pas d'optimisation, mode automatique ou manuel • PHASE = 1 : prêt à optimiser, vérifier les paramètres, attendre activation, mesurer les temps d'échantillonnage. • PHASE = 2 : optimisation proprement dite : recherche du point d'inflexion pour une valeur de réglage constante. Ecriture du temps d'échantillonnage dans le DB d'instance. • PHASE = 3 : calcul des paramètres du processus. Sauvegarde des paramètres du régulateur valides avant l'optimisation. • PHASE = 4 : caractéristique de régulation • PHASE = 5 : asservissement du régulateur sur une nouvelle valeur de réglage. • PHASE = 7 : vérification du type de système réglé
STATUS_H	110.0	INT	0	STATUS_H indique une valeur de diagnostic pour la recherche du point d'inflexion durant le chauffage.
STATUS_D	112.0	INT	0	STATUS_D indique une valeur de diagnostic pour la recherche d'une caractéristique de régulation optimale durant le chauffage.
QTUN_RUN	114.0	BOOL	0	L'optimisation a été démarrée par application de la variable réglante d'optimisation et se trouve encore en phase 2 (recherche du point d'inflexion).
PI_CON	116.0	STRUCT		Paramètres de régulation PI
GAIN	+0.0	REAL	0.0	Gain du régulateur PI %/unité phys.
TI	+4.0	REAL	0.0 s	Temps d'intégration PI [s]
PID_CON	124.0	STRUCT		Paramètres de régulation PID
GAIN	+0.0	REAL	0.0	Gain du régulateur PID
TI	+4.0	REAL	0.0s	Temps d'intégration PID [s]
TD	+8.0	REAL	0.0s	Temps de dérivation PID [s]
PAR_SAVE	136.0	STRUCT		Les paramètres PID sont enregistrés dans cette structure.
PFAC_SP	+0.0	REAL	1.0	Coefficient d'action proportionnelle en cas de modifications de la consigne Les valeurs autorisées sont comprises entre 0.0 et 1.0.
GAIN	+4.0	REAL	0.0	Gain du régulateur %/unité phys.
TI	+8.0	REAL	40.0 s	Temps d'intégration [s]
TD	+12.0	REAL	10.0 s	Temps de dérivation [s]
D_F	+16.0	REAL	5.0	Facteur de dérivation Les valeurs autorisées sont comprises entre 5.0 et 10.0.
CON_ZONE	+20.0	REAL	100.0	Largeur de la plage de régulation Si le signal d'écart est supérieur à la largeur de la plage de régulation, la limite supérieure de la valeur de réglage est fournie comme valeur de réglage. Si le différentiel de réglage est inférieur à la largeur de la partie négative du domaine de réglage, la valeur de réglage coïncide avec la limite inférieure de ce domaine. CON_ZONE ≥ 0.0
CONZ_ON	+24.0	BOOL	FALSE	Activer plage de régulation

Paramètre	Adresse	Type de données	Valeur par défaut	Description
PFAC_SP	162.0	REAL	1.0	PFAC_SP indique l'action P effective en cas de modification de la consigne. Les valeurs possibles sont 0 ou 1. • 1 : l'action P intervient à 100 % en cas de modifications de la consigne. • 0 : l'action P n'intervient pas en cas de modifications de la consigne. Les valeurs autorisées sont comprises entre 0.0 et 1.0.
GAIN	166.0	REAL	2.0	L'entrée "Coefficient d'action proportionnelle" indique le gain du régulateur. Une inversion du sens de régulation s'obtient par une valeur négative de GAIN. %/unité phys.
TI	170.0	REAL	40.0 s	L'entrée "temps d'intégration" (temps d'action par intégration) détermine la réponse temporelle de l'intégrateur.
TD	174.0	REAL	10.0 s	L'entrée "temps de dérivation" détermine la réponse temporelle du dérivateur.
D_F	178.0	REAL	5.0	Le facteur de dérivation détermine le temps de retard de l'action D. D_F = Temps de dérivation/ "Temps de retard de l'action D" Les valeurs autorisées sont comprises entre 5.0 et 10.0.
CON_ZONE	182.0	REAL	100.0	Si le signal d'écart est supérieur à la largeur de la plage de régulation, la limite supérieure de la valeur de réglage est fournie comme valeur de réglage. Si le différentiel de réglage est inférieur à la largeur de la partie négative du domaine de réglage, la valeur de réglage coïncide avec la limite inférieure de ce domaine. Les valeurs correctes dépendent des capteurs utilisés.
CONZ_ON	186.0	BOOL	FALSE	CONZ_ON =TRUE permet d'activer la plage de régulation.
TUN_ON	186.1	BOOL	FALSE	Si TUN_ON=TRUE, la moyenne de la valeur de réglage est calculée jusqu'à ce que TUN_DLMN soit activé par un échelon de consigne ou par TUN_ST=TRUE.
TUN_ST	186.2	BOOL	FALSE	Si lors de l'optimisation, la consigne doit rester constante au point de fonctionnement, TUN_ST=1 applique un échelon de la valeur de réglage de TUN_DLMN.
UNDO_PAR	186.3	BOOL	FALSE	Charge les paramètres du régulateur PFAC_SP, GAIN, TI, TD, D_F, CONZ_ON et CON_ZONE du régulateur depuis la structure de données PAR_SAVE (uniquement en mode manuel).
SAVE_PAR	186.4	BOOL	FALSE	Sauvegarde les paramètres de régulation PFAC_SP, GAIN, TI, TD, D_F, CONZ_ON et CON_ZONE dans la structure de données PAR_SAVE.
LOAD_PID	186.5	BOOL	FALSE	Charge les paramètres du régulateur GAIN, TI,, TD en fonction de PID_ON à partir de la structure de données PI_CON ou PID_CON (mode manuel uniquement)
PID_ON	186.6	BOOL	TRUE	L'entrée PID_ON permet de spécifier si le régulateur optimisé doit fonctionner en mode PI ou PID. • Régulateur PID : PID_ON = TRUE • Régulateur PI : PID_ON = FALSE Il est toutefois possible que certains systèmes ne puissent être régulés qu'en mode PI même lorsque PID_ON = TRUE.
GAIN_P	188.0	REAL	0.0	Gain identifié pour le processus. Dans les systèmes de type I, GAIN_P est généralement sous-évalué.
TU	192.0	REAL	0.0	Délai identifié pour le processus. TU ≥ 3*CYCLE
TA	196.0	REAL	0.0	Temps de stabilisation identifié pour le processus. Dans les systèmes de type I, TA est généralement sous-évalué.

Paramètre	Adresse	Type de données	Valeur par défaut	Description
KIG	200.0	REAL	0.0	Augmentation maximale de la mesure pour un saut de grandeur réglante de 0 à 100 % [1/s] GAIN_P = 0.01 * KIG * TA
N_PTN	204.0	REAL	0.0	Ce paramètre indique l'ordre du système. Il peut avoir comme valeur un "nombre non entier". Les valeurs autorisées sont comprises entre 1.01 et 10.0.
TM_LAG_P	208.0	REAL	0.0	Constante de temps d'un modèle PTN (valeurs significatives uniquement pour N_PTN >= 2).
T_P_INF	212.0	REAL	0.0	Durée entre le déclenchement du processus et le point d'inflexion.
P_INF	216.0	REAL	0.0	Modification de la mesure du déclenchement du processus jusqu'au point d'inflexion. Les valeurs correctes dépendent des capteurs utilisés.
LMN0	220.0	REAL	0.0	Valeur de réglage au début de l'optimisation Déterminée en phase 1 (valeur moyenne). Les valeurs autorisées sont comprises entre 0 et 100 %.
PV0	224.0	REAL	0.0	Mesure au début de l'optimisation
PVDT0	228.0	REAL	0.0	Rampe de la mesure au début de l'optimisation [1/s] Signe adapté.
PVDT	232.0	REAL	0.0	Rampe momentanée de la mesure [1/s] Signe adapté.
PVDT_MAX	236.0	REAL	0.0	Variation maxi de la mesure par seconde [1/s] La dérivation maximale de la mesure au point d'inflexion (signe adapté, toujours > 0) permet de calculer TU et KIG.
NOI_PVDT	240.0	REAL	0.0	Taux de bruit dans PVDT_MAX en % Plus le taux de bruit est élevé, plus les paramètres du régulateur sont imprécis (atténués).
NOISE_PV	244.0	REAL	0.0	Bruit absolu dans la mesure Différence entre la mesure maximale et minimale en phase 1.
FIL_CYC	248.0	INT	1	Nombre de cycles de filtrage de la valeur moyenne La mesure est calculée sur FIL_CYC cycles. Au besoin, FIL_CYC est augmenté automatiquement de 1 à 1024 maxi.
POI_CMAX	250.0	INT	2	Nombre de cycles maximal après le point d'inflexion Ce temps est utilisé pour trouver un autre, c.-à-d. un meilleur point d'inflexion en cas de bruit de mesure. L'optimisation ne peut se terminer plus tôt.
POI_CYCL	252.0	INT	0	Nombre de cycles après le point d'inflexion

Voir aussi

Mode de fonctionnement générateur d'impulsion (Page 449)
Schéma fonctionnel TCONT_CP (Page 452)

10.4.4.9 Paramètres STATUS_H

STATUS_H	Description	Solution
0	Valeur par défaut ou aucun ou pas encore de nouveaux paramètres de régulation	
10000	Optimisation terminée + paramètres de régulation idoines trouvés	
2xxxx	Optimisation terminée + paramètres de régulation non fiables	
2xx2x	Point d'inflexion pas atteint (seulement en cas d'activation par échelon de consigne)	Si le régulateur oscille, atténuer les paramètres de régulation ou renouveler l'essai avec un écart de valeur de réglage plus petit TUN_DLMN
2x1xx	Erreur d'estimation (TU < 3*CYCLE)	Diminuer CYCLE et renouveler l'essai. Cas particulier d'un système pur PT1 : ne pas renouveler l'essai, atténuer éventuellement les paramètres de régulation.
2x3xx	Erreur d'estimation TU trop grande	Renouveler l'essai sous de meilleures conditions.
21xxx	Erreur d'estimation N_PTN < 1	Renouveler l'essai sous de meilleures conditions.
22xxx	Erreur d'estimation N_PTN > 10	Renouveler l'essai sous de meilleures conditions.
3xxxx	Optimisation annulée en phase 1 à cause d'un paramétrage erroné :	
30002	Ecart de valeur de réglage effectif < 5 %	Corriger l'écart de valeur de réglage TUN_DLMN.
30005	Les temps d'échantillonnage CYCLE et CYCLE_P divergent de plus de 5 % des valeurs mesurées.	Comparez CYCLE et CYCLE_P avec le temps de cycle de l'alarme cyclique et tenez compte des répartiteurs d'appel éventuellement disponibles. Vérifiez la charge de la CPU. Une CPU saturée entraîne des temps d'échantillonnage prolongés qui ne concordent pas avec CYCLE ou CYCLE_P.

REMARQUE

Si vous interrompez l'optimisation en phase 1 ou 2, STATUS_H = 0. STATUS_D affiche toutefois toujours l'état du dernier calcul du régulateur.

Plus la valeur de STATUS_D est grande, plus le rang du système réglé est élevé, plus le rapport TU/TA est grand et plus les paramètres de régulation sont légers.

Voir aussi

Mode de fonctionnement générateur d'impulsion (Page 449)

Schéma fonctionnel TCONT_CP (Page 452)

10.4.4.10 Paramètre STATUS_D

STATUS_D	Description
0	Aucun paramètre de régulation n'est calculé
110	N_PTN <= 1.5 Type de système I rapide
121	N_PTN > 1.5 Type de système I
200	N_PTN > 1.9 Type de système II (plage transitoire)
310	N_PTN >= 2.1 Type de système III rapide
320	N_PTN > 2.6 Type de système III
111, 122, 201, 311, 321	Les paramètres ont été corrigés par la phase 7.

REMARQUE

Plus la valeur de STATUS_D est grande, plus le rang du système réglé est élevé, plus le rapport TU/TA est grand et plus les paramètres de régulation sont légers.

Voir aussi

Mode de fonctionnement générateur d'impulsion (Page 449)
Schéma fonctionnel TCONT_CP (Page 452)

10.4.5 TCONT_S

10.4.5.1 Description TCONT_S

L'instruction TCONT_S sert à la régulation de processus thermiques techniques à signaux de sortie binaires de la valeur de réglage pour actionneurs à action intégrale, dans les systèmes d'automatisation SIMATIC S7. Son fonctionnement est basé sur l'algorithme de régulation PI du régulateur à échantillonnage. Ce régulateur pas à pas fonctionne sans signalisation de position.

Application

Vous pouvez également intégrer le régulateur dans un circuit en cascade, comme régulateur de position de niveau inférieur. Vous spécifiez la position de l'actionneur grâce à l'entrée de consigne SP_INT. Dans ce cas, vous devez mettre l'entrée de mesure et le paramètre TI (temps d'intégration) à zéro. Exemple d'application : réglage de température par régulation d'un système chauffage via commande impulsion-pause ou régulation d'un système de refroidissement via une vanne papillon. La fermeture complète de la vanne intervient lorsque la grandeur réglante (ER*GAIN) devient négative.

Appel

L'instruction TCONT_S doit être appelée de manière équidistante. Veuillez utiliser une alarme cyclique (par ex. OB35 pour S7-300). Le temps d'échantillonnage est spécifié dans le paramètre CYCLE.

Temps d'échantillonnage CYCLE

Le temps d'échantillonnage CYCLE doit concorder avec la différence de temps entre deux appels (temps de cycle de l'OB d'alarme cyclique compte tenu des rapports de réduction). Veillez à ce que le temps d'échantillonnage du régulateur ne soit pas supérieur à 10 % du temps d'intégration du régulateur (TI) calculé. En général, vous devez toutefois régler le temps d'échantillonnage de manière beaucoup plus faible afin d'assurer la précision requise du régulateur pas à pas.

Précision G requise	MTR_TM	CYCLE = MTR_TM*G	Commentaire
0,5 %	10 s	0,05 s	Le temps d'échantillonnage est déterminé par la précision requise du régulateur pas à pas.

Démarrage

L'instruction TCONT_S dispose d'une routine d'initialisation exécutée lorsque le paramètre d'entrée COM_RST = TRUE. Après la routine d'initialisation, le bloc remet COM_RST sur FALSE. Toutes les sorties sont mises à leurs valeurs initiales. Si vous souhaitez une initialisation au démarrage de la CPU, appelez le bloc dans l'OB 100 avec COM_RST = TRUE.

Voir aussi

Schéma fonctionnel TCONT_S (Page 466)

10.4.5.2 Fonctionnement TCONT_S

Branche de consigne

La consigne est spécifiée sous forme de valeur physique à virgule flottante ou sous forme de pourcentage à l'entrée SP_INT. La consigne et la mesure intervenant dans le calcul du signal d'écart doivent avoir la même unité.

Sélection de la mesure (PVPER_ON)

Le format de la mesure est choisi en fonction de PVPER_ON : périphérie ou virgule flottante.

PVPER_ON	Entrée de la mesure
TRUE	La mesure est lue à l'entrée PV_PER via la périphérie analogique (PEW xxx).
FALSE	La mesure est lue en format à virgule flottante à l'entrée PV_IN.

Conversion du format de la mesure CRP_IN (PER_MODE)

La fonction CRP_IN effectue la conversion de la valeur de périphérie PV_PER en un format à virgule flottante en fonction du commutateur PER_MODE en appliquant la règle suivante :

PER_MODE	Sortie de CRP_IN	Type d'entrée analogique	Unité
0	PV_PER * 0.1	Thermocouples ; PT100/NI100 ; standard	°C;°F
1	PV_PER * 0.01	PT100/NI100 ; climat ;	°C;°F
2	PV_PER * 100/27648	Tension / courant	%

Normalisation de la mesure PV_NORM (PF_FAC, PV_OFFS)

La fonction PV_NORM calcule la sortie de CRP_IN selon la règle suivante :
Sortie de PV_NORM = (sortie de CRP_IN) * PV_FAC + PV_OFFS
Domaine d'application de cette règle :

- Conversion de la mesure avec PV_FAC comme facteur de mesure et PV_OFFS comme décalage de la mesure
- Normalisation d'une température en pourcentage
  Vous souhaitez entrer la consigne sous forme de pourcentage et devez à présent convertir la valeur de température mesurée en pourcentage.
- Normalisation d'un pourcentage en température
  Vous souhaitez entrer la consigne sous forme de la grandeur physique température et devez à présent convertir la valeur de tension/courant mesurée en température.

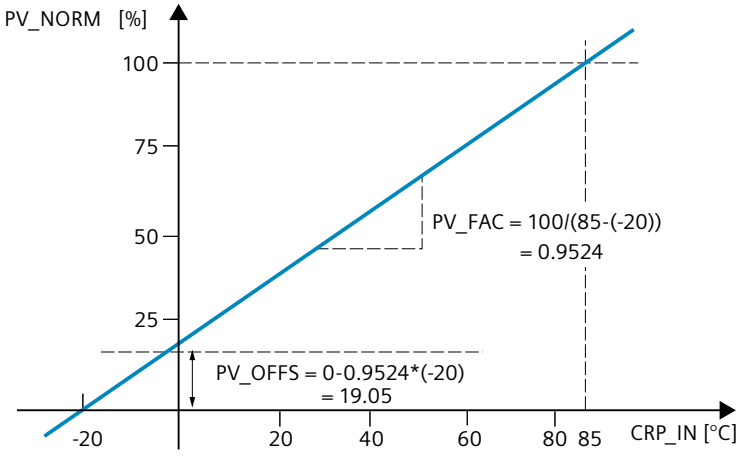
Calcul des paramètres :

- PV_FAC = plage de PV_NORM/ plage de CRP_IN;
- PV_OFFS = UG(PV_NORM) - PV_FAC * UG(CRP_IN);
  avec UG : Limite inférieure

Les valeurs par défaut (PV_FAC = 1.0 et PV_OFFS = 0.0) permettent de désactiver la normalisation. La mesure opérante est fournie à la sortie PV.

Exemple de normalisation de la mesure

Si vous souhaitez spécifier la consigne sous forme d'un pourcentage et que la plage de températures est comprise entre -20 et 85 °C au niveau de CRP_IN, vous devez convertir la plage de température en pourcentage.
La figure suivante présente un exemple de conversion de la plage de température -20 à 85 ° à la plage interne 0 à 100 % :

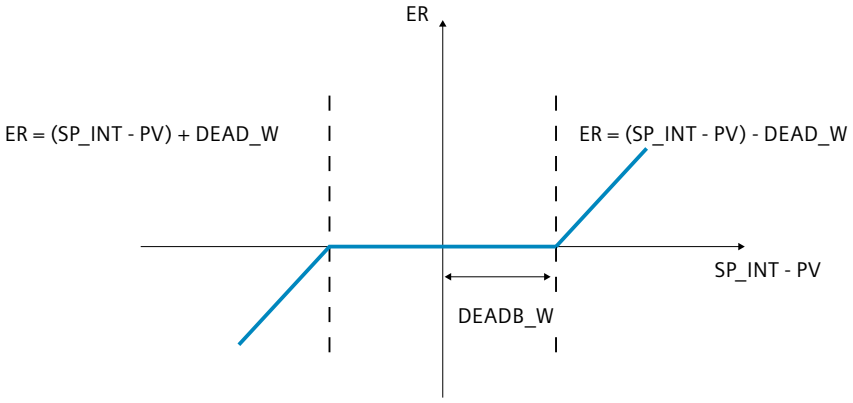


Calcul du signal d'écart

Le signal d'écart précédant la zone morte correspond à la différence entre la consigne et la mesure.
La consigne et la mesure doivent avoir la même unité.

Zone morte (DEADB_W)

Le signal d'écart comprend une zone morte (DEADBAND) qui permet de neutraliser une oscillation continue de faible amplitude due à la quantification des grandeurs réglantes (p. ex. en cas de modulation de largeur d'impulsion PULSEGEN). Lorsque DEADB_W = 0.0, la zone morte est désactivée.



Algorithme du régulateur PI pas à pas

L'instruction TCONT_S travaille sans signalisation de position. L'action par intégration I de l'algorithme PI et la signalisation théorique de position sont calculées dans un intégrateur (INT) et comparées, comme valeurs de rétroaction, à l'action proportionnelle P restante. La différence est transmise à un élément fonctionnel à trois échelons (THREE_ST) ainsi qu'au formateur des impulsions (PULSEOUT) pour la valve de régulation. La fréquence de commutation du régulateur peut être réduite par l'adaptation du seuil d'action de l'élément fonctionnel à trois échelons.

Atténuation de l'action proportionnelle P en cas de modification de la consigne

Pour éviter un dépassement, vous pouvez atténuer l'action proportionnelle P avec le paramètre "Coefficient d'action proportionnelle en cas de modification de la consigne" (PFAC_SP). PFAC_SP permet donc de sélectionner toute valeur comprise entre 0.0 et 1.0 pour spécifier l'importance de l'action proportionnelle P en cas de modification de la consigne :

- PFAC_SP = 1.0 : Action P totalement opérante en cas de modification de la consigne
- PFAC_SP = 0.0 : Aucune action P en cas de modification de la consigne

Comme c'est le cas pour le régulateur en continu, une valeur PFAC_SP < 1.0 peut réduire le dépassement si le temps de marche du moteur MTR_TM est relativement court par rapport au temps de stabilisation TA et si le rapport TU/TA < 0.2. Lorsque MTR_TM atteint 20 % de TA, les possibilités d'amélioration sont relativement faibles.

Action anticipatrice

Une perturbation additionnelle peut être appliquée à l'entrée DISV.

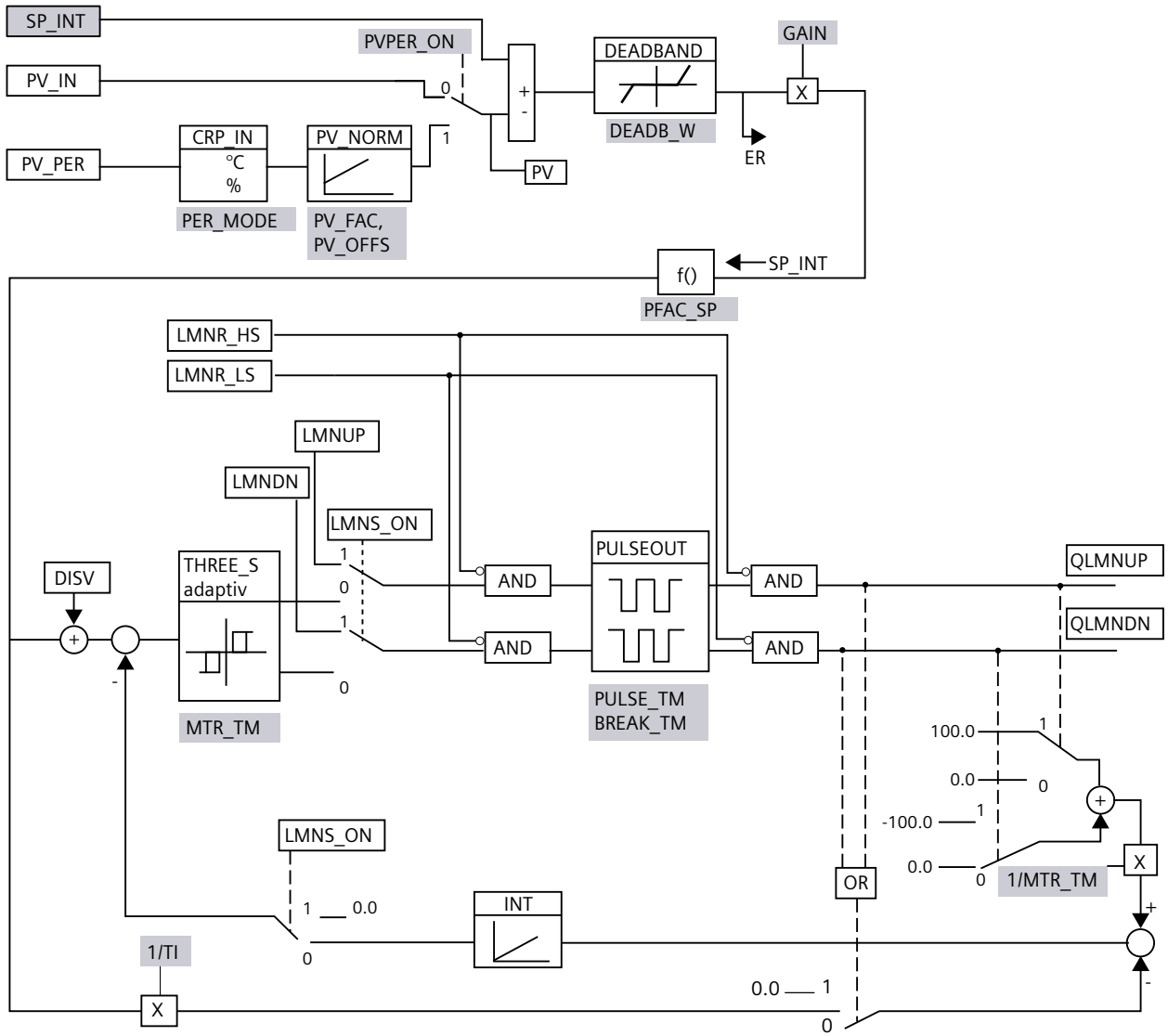
Mode manuel (LMNS_ON, LMNUP, LMNDN)

LMNS_ON permet de commuter entre les modes manuel et automatique. En mode manuel, l'actionneur est arrêté et l'intégrateur (INT) est mis à zéro par le système. LMNUP et LMNDN permettent d'ouvrir et de fermer l'actionneur. Le passage au mode automatique s'accompagne d'à-coups. Le signal d'écart existant produit un échelon de la grandeur réglante interne par l'intermédiaire de GAIN. Mais l'actionneur à action intégrale n'effectue qu'une commande du processus en forme de rampe.

Voir aussi

Schéma fonctionnel TCONT_S (Page 466)

10.4.5.3 Schéma fonctionnel TCONT_S



Interface de paramétrage

Interface d'appel instruction

Interface de paramétrage, interface d'appel instruction

Voir aussi

Description TCONT_S (Page 461)
Fonctionnement TCONT_S (Page 462)
Paramètres d'entrée TCONT_S (Page 467)
Paramètres de sortie TCONT_S (Page 468)
Paramètres d'entrée/sortie TCONT_S (Page 468)
Variables statiques TCONT_S (Page 469)

10.4.5.4 Paramètres d'entrée TCONT_S

Les noms des paramètres suivants sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Tableau 10-23

Paramètre	Adresse	Type de données	Valeur par défaut	Description
CYCLE	0.0	REAL	0.1 s	Cette entrée permet de saisir le temps d'échantillonnage pour le régulateur. CYCLE ≥ 0.001
SP_INT	4.0	REAL	0.0	L'entrée "consigne interne" permet de spécifier une consigne. Les valeurs correctes dépendent des capteurs utilisés.
PV_IN	8.0	REAL	0.0	L'entrée "Entrée mesure" permet de paramétrer une valeur de mise en service ou d'interconnecter une mesure externe au format en virgule flottante. Les valeurs correctes dépendent des capteurs utilisés.
PV_PER	12.0	INT	0	La mesure en format périphérie est interconnectée au régulateur à l'entrée "Mesure périphérie".
DISV	14.0	REAL	0.0	Pour une action anticipatrice, la perturbation est appliquée à l'entrée "perturbation".
LMNR_HS	18.0	BOOL	FALSE	Le signal "Butée supérieure de la vanne de régulation" est appliqué à l'entrée "Signal de butée supérieure de la signalisation de position". • LMNR_HS=TRUE : la vanne de régulation se trouve à la butée supérieure.
LMNR_LS	18.1	BOOL	FALSE	Le signal "Butée inférieure de la vanne de régulation" est appliqué à l'entrée "Signal de butée inférieure de la signalisation de position". • LMNR_LS=TRUE : La vanne de régulation a atteint la butée inférieure.
LMNS_ON	18.2	BOOL	TRUE	L'entrée "Activation du mode manuel des signaux de valeur de réglage" permet de passer au traitement manuel des signaux de valeur de réglage.
LMNUP	18.3	BOOL	FALSE	En mode manuel des signaux de position, cette entrée sert à commander le signal de sortie QLMNUP.
LMNDN	18.4	BOOL	FALSE	En mode manuel des signaux de valeur de réglage, le signal de sortie QLMNDN est commandé par l'entrée "Signal bas de valeur de réglage".

Voir aussi

Schéma fonctionnel TCONT_S (Page 466)

10.4.5.5 Paramètres de sortie TCONT_S

Les noms des paramètres suivants sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Tableau 10-24

Paramètre	Adresse	Type de données	Valeur par défaut	Description
QLMNUP	20.0	BOOL	FALSE	Si la sortie" Signal haut de valeur de réglage" est à 1, la vanne de régulation doit être ouverte.
QLMNDN	20.1	BOOL	FALSE	Si la sortie" Signal bas de valeur de réglage" est à 1, la vanne de régulation doit être fermée.
PV	22.0	REAL	0.0	La mesure opérante est fournie à la sortie "Mesure".
ER	26.0	REAL	0.0	Le signal d'écart opérant est fourni à la sortie "Signal d'écart".

Voir aussi

Schéma fonctionnel TCONT_S (Page 466)

10.4.5.6 Paramètres d'entrée/sortie TCONT_S

Les noms des paramètres suivants sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Tableau 10-25

Paramètre	Adresse	Type de données	Valeur par défaut	Description
COM_RST	30.0	BOOL	FALSE	Le bloc dispose d'une routine d'initialisation exécutée lorsque l'entrée COM_RST est activée.

Voir aussi

Schéma fonctionnel TCONT_S (Page 466)

10.4.5.7 Variables statiques TCONT_S

Les noms des variables suivantes sont valables aussi bien pour le bloc de données que pour l'accès via API Openness.

Tableau 10-26

Paramètre	Adresse	Type de données	Valeur par défaut	Description
PV_FAC	32.0	REAL	1.0	L'entrée "Facteur de mesure" est multipliée par la mesure. Cette entrée permet d'adapter l'étendue de la mesure.
PV_OFFS	36.0	REAL	0.0	L'entrée "Décalage de la mesure" est additionnée à la mesure. Cette entrée permet d'adapter l'étendue de la mesure. Les valeurs correctes dépendent des capteurs utilisés.
DEADB_W	40.0	REAL	0.0	Le signal d'écart parcourt une zone morte. L'entrée "Largeur de zone morte" détermine la taille de la zone morte. DEADB_W ≥ 0.0
PFAC_SP	44.4	REAL	1.0	PFAC_SP indique l'action P effective en cas de modification de la consigne. • 1 : l'action P intervient à 100% en cas de modifications de la consigne. • 0 : l'action P n'intervient pas en cas de modifications de la consigne. Les valeurs autorisées sont comprises entre 0.0 et 1.0.
GAIN	48.0	REAL	2.0	L'entrée "Coefficient d'action proportionnelle" indique le gain du régulateur. Une inversion du sens de régulation s'obtient par une valeur négative de GAIN. %/unité phys.
TI	52.0	REAL	40.0 s	L'entrée "temps d'intégration" (temps d'action par intégration) détermine la réponse temporelle de l'intégrateur.
MTR_TM	56.0	REAL	30 s	Le paramètre "Temps de positionnement du moteur" reçoit la durée d'exécution de la vanne de régulation d'une butée à l'autre. MTR_TM ≥ CYCLE
PULSE_TM	60.0	REAL	0.0 s	Le paramètre "Durée minimale d'impulsion" permet de régler une durée d'impulsion minimale.
BREAK_TM	64.0	REAL	0.0 s	Le paramètre "Durée minimale de pause" permet de spécifier une durée de pause minimale.
PER_MODE	68.0	INT	0	Ce commutateur permet de spécifier le type de module AE. La mesure à l'entrée PV_PER est ainsi normalisée à la sortie PV de la manière suivante : • PER_MODE = 0 : Thermocouples ; PT100/NI100 ; standard PV_PER * 0.1 Unité : °C, °F • PER_MODE = 1 : PT100/NI100 ; climat PV_PER * 0.01 Unité : °C, °F • PER_MODE = 2 : Courant/tension PV_PER * 100/27648 Unité : %
PVPER_ON	70.0	BOOL	FALSE	Pour lire la mesure de la périphérie, l'entrée PV_PER doit être interconnectée à la périphérie et l'entrée "activation de la mesure périphérie" doit être mise à 1.

Voir aussi

Schéma fonctionnel TCONT_S (Page 466)

10.4.6 Fonctions système intégrées

10.4.6.1 CONT_C_SF

CONT_C_SF

L'instruction CONT_C_SF est intégrée dans les CPU Compact S7-300. L'instruction ne doit pas être transmise lors du chargement dans la CPU S7-300. La fonction correspond à l'instruction CONT_C.

Voir aussi

Description CONT_C (Page 420)
Fonctionnement de CONT_C (Page 421)
Schéma fonctionnel CONT_C (Page 422)
Paramètres d'entrée CONT_C (Page 423)
Paramètres de sortie CONT_C (Page 424)

10.4.6.2 CONT_S_SF

CONT_S_SF

L'instruction CONT_S_SF est intégrée dans les CPU Compact S7-300. L'instruction ne doit pas être transmise lors du chargement dans la CPU S7-300. La fonction correspond à l'instruction CONT_S.

Voir aussi

Description CONT_S (Page 425)
Fonctionnement CONT_S (Page 426)
Schéma fonctionnel CONT_S (Page 427)
Paramètres d'entrée CONT_S (Page 428)
Paramètres de sortie CONT_S (Page 429)

10.4.6.3 PULSEGEN_SF

PULSEGEN_SF

L'instruction PULSEGEN_SF est intégrée dans les CPU Compact S7-300. L'instruction ne doit pas être transmise lors du chargement dans la CPU S7-300. La fonction correspond à l'instruction PULSEGEN.

Voir aussi

Description PULSGEN (Page 430)
Fonctionnement PULSGEN (Page 431)
Mode de fonctionnement PULSGEN (Page 434)
Régulation à trois échelons (Page 434)
Régulation à deux échelons (Page 436)
Paramètre d'entrée PULSEGEN (Page 437)
Paramètre de sortie PULSEGEN (Page 438)

10.5 Polyligne

10.5.1 Compatibilité avec CPU et FW

Le tableau suivant montre la compatibilité entre les CPU et les versions de Polyline:

CPU	FW	Polyline
S7-1200	à partir de V4.2	V1.0
CPU basées sur S7-1500	à partir de V2.0	V1.0

10.5.2 Description de polyligne

Description

L'instruction Polyline reproduit la valeur d'entrée Input sur la valeur de réglage Output à l'aide d'une courbe caractéristique. La courbe caractéristique est définie comme un tracé polygonal comportant 50 nœuds d'interpolation maximum. Une interpolation linéaire est effectuée entre ces nœuds d'interpolation. Vous pouvez adapter le tracé polygonal à la courbe caractéristique souhaitée au moyen du nombre et de la configuration des nœuds d'interpolation.

L'instruction Polyline peut par exemple être utilisée pour linéariser un comportement non-linéaire de capteurs et d'actionneurs.

Calcul d'interpolation

Polyline calcule, par une interpolation linéaire, la valeur de réglage du paramètre Output pour la valeur d'entrée au paramètre Input, qui se trouve entre les nœuds d'interpolation x_i et x_{i+1}. L'interpolation linéaire est calculée selon la formule suivante :

$$Output = \frac{(Input - x_i)}{(x_{i+1} - x_i)} (y_{i+1} - y_i) + y_i$$

Le paramètre Reset = TRUE permet également de spécifier une valeur de réglage alternative via le paramètre SubstituteOutput.

Données relatives au tracé polygonal

Les paires de valeurs pour le tracé polygonal se trouvent dans la zone Static de l'instruction.

REMARQUE

- Le nombre minimum de paires de valeurs à configurer est de 2.
- Le nombre maximum de paires de valeurs à configurer est de 50.
- Pour une configuration valide, les valeurs de x doivent être spécifiées par ordre croissant.

Pour pouvoir modifier les données relatives au tracé polygonal sans que les modifications ne soient immédiatement effectives, les paire de valeurs du tracé polygonal doivent être dupliquées et figurer dans les structures suivantes :

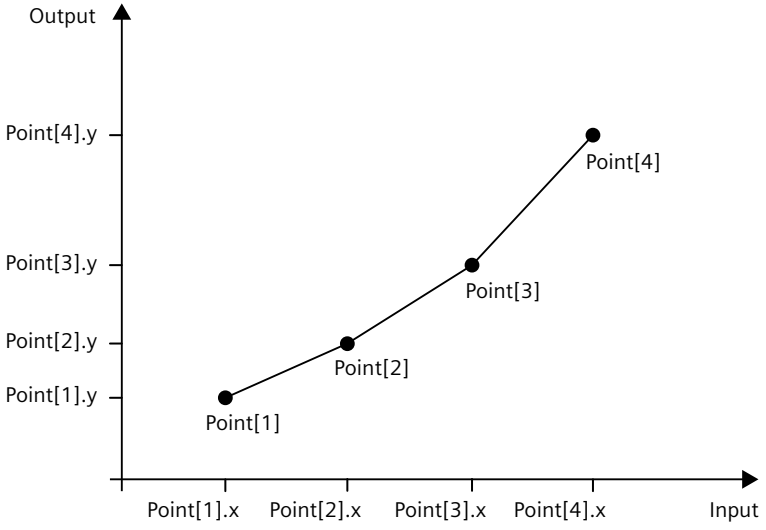
- UserData
  Les données relatives au tracé polygonal de cette structure peuvent être éditées.
  Utilisez cette structure pour définir ou modifier les données relatives au tracé polygonal. Les modifications de cette structure ne se répercutent sur le calcul d'interpolation que lorsque le contrôle et la copie des données sont déclenchés dans la structure WorkingData. Cela se produit par la configuration de Validate = TRUE ou automatiquement pendant le premier traitement de Polyline après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN.
  La préconfiguration des valeurs dans cette structure ne constitue pas une configuration valide. Pour utiliser ces valeurs pour le calcul d'interpolation, modifiez les variables et attribuez-leur des valeurs valides.

- WorkingData
  Les données relatives au tracé polygonal de cette structure ne peuvent pas être éditées. Ces données sont utilisées pour le calcul d'interpolation. Ne modifiez pas manuellement les données de cette structure.

Les deux structures ont le même type de données et donc le même contenu :

- NumberOfUsedPoints
  Nombre de nœuds d'interpolation utilisés pour le calcul d'interpolation.
- Point
  Le tableau (Array) avec 50 éléments contient des paires de valeurs des nœuds d'interpolation Point[i].x et Point[i].y avec l'indice "i" allant de 1 à 50.

La figure ci-dessous montre un tracé polygonal comportant quatre nœuds d'interpolation :



Appel

Dans un OB, Polyline est appelé en tant que bloc de données d'instance unique, dans une FC, Polyline est appelé en tant que bloc de données d'instance unique ou bloc de données d'instance de paramètres et, dans un FB, Polyline peut être appelée aussi bien comme bloc de données d'instance unique que comme bloc de données de multi-instance ou bloc de données d'instance de paramètres.

Aucun objet technologique n'est créé lors de l'appel de l'instruction. La configuration des polylignes est disponible dans la fenêtre d'inspection de l'éditeur de programmation.

Démarrage

Les variables des structures UserData et WorkingData ne sont pas rémanentes. Ces variables sont initialisées à l'aide des valeurs initiales après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN.
Si vous modifiez les valeurs actuelles dans la structure UserData en mode en ligne et que ces valeurs doivent être conservées après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN, sauvegardez ces valeurs dans les valeurs initiales du bloc de données.
Lors du premier appel de l'instruction Polyline après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN, la validité des données est automatiquement contrôlée dans la structure UserData. Si le contrôle est positif, les données sont transmises à la structure WorkingData.

Comportement en cas d'erreur

L'instruction Polyline détecte différentes erreurs pouvant survenir lors du calcul d'interpolation. Le résultat du calcul d'interpolation peut être émis en dépit de la présence d'une erreur à la sortie. Si aucun calcul correct du résultat d'interpolation n'est possible à cause d'une erreur, une valeur de réglage de remplacement est émise à la sortie.
Vous définissez, pour la variable ErrorMode, la variable de réglage de remplacement de la manière suivante en cas d'erreur rendant impossible tout calcul correct du résultat d'interpolation :

ErrorMode	Output
0	Valeur du paramètre Input
1	Valeur du paramètre SubstituteOutput
2	Dernier résultat valide du calcul d'interpolation 0.0, en l'absence de résultat valide

Les points suivants s'appliquent en plus pour toutes les valeurs de la variable ErrorMode :

- Si la valeur de réglage de remplacement n'a pas de de valeur REAL valide, 0.0 est émis comme valeur de réglage.
- La valeur de réglage de remplacement est limitée à la plage de valeurs du type de données REAL. Ce n'est qu'ensuite que la valeur de réglage de remplacement est émise au paramètre Output.
- La variable ErrorMode n'est active que lorsque le paramètre Reset = FALSE . Si le paramètre Reset = TRUE, la valeur du paramètre SubstituteOutput est émise au paramètre Output.

Le paramètre Error indique si une erreur est actuellement présente. Quand l'erreur a disparu, Error est réglé sur FALSE. Le paramètre ErrorBits indique les erreurs qui se sont produites. ErrorBits est rémanent et n'est réinitialisé que par un front montant au paramètre Reset ou ErrorAck.

10.5.3 Fonctionnement de la polyligne

Données relatives au tracé polygonal

Pour modifier les données relatives au tracé polygonal, éditez les valeurs dans la structure UserData. Ensuite, la validité des valeurs est contrôlée et transmise à la structure WorkingData. Les valeurs du calcul d'interpolation ne sont utilisées que dans la structure WorkingData.

Les valeurs sont contrôlées et transmises lorsque

- vous réglez le paramètre Validate sur TRUE tandis que le paramètre Reset est réglé sur FALSE.
- Polyline est appelée pour la première fois après le passage de CPU de l'état de fonctionnement STOP à l'état RUN, tandis que le paramètre Reset est réglé sur FALSE.
  Si la polyligne a déjà été appelée par ex. dans le OB100, aucun nouveau contrôle automatique des valeurs n'est effectué en cas d'appels ultérieurs.

Si les données relatives au tracé polygonal dans la structure UserData ne sont pas valides, les données relatives au tracé polygonal précédentes dans la structure WorkingData sont conservées et un message d'erreur correspondant s'affiche. S'il s'agit du premier contrôle, aucune valeur valide n'est disponible dans la structure WorkingData et un message d'erreur correspondant s'affiche. Dans ce cas, la valeur de réglage de remplacement que vous avez configurée avec la variable ErrorMode est définie pour le paramètre Output.

Le contrôle et la transmission des valeurs de la structure UserData nécessite plus de temps de traitement de la CPU que le calcul d'interpolation. Dans les applications minutées, la première exécution de la polyligne peut avoir lieu dans l'OB de démarrage 100. Ainsi, le contrôle et la transmission, uniques mais chronophages, des données relatives au tracé polygonal sont déjà terminés avant les parties de programme d'application cycliques.

Validité des données relatives au tracé polygonal

Lors du contrôle des valeurs dans la structure UserData, les valeurs doivent remplir les conditions suivantes pour qu'un tracé polygonal valide soit disponible pour le calcul d'interpolation :

- $2 \leq$ UserData.NumberOfUsedPoints ≤ 50
- UserData.Point[j].x < UserData.Point[j+1].x avec l'indice j = 1..(UserData.NumberOfUsedPoints – 1)
- -3.402823e+38 $\leq$ UserData.Point[i].x $\leq$ 3.402823e+38 avec l'indice i = 1..UserData.NumberOfUsedPoints
- -3.402823e+38 $\leq$ UserData.Point[i].y $\leq$ 3.402823e+38 avec l'indice i = 1..UserData.NumberOfUsedPoints
- UserData.Point[i].x and UserData.Point[i].y sont des valeurs valides pour REAL ($\neq$ NaN) avec l'indice i = 1..UserData.NumberOfUsedPoints

Si une ou plusieurs conditions ne sont pas remplies pendant le contrôle, les valeurs de la structure UserData ne sont pas reprises dans la structure WorkingData. Un message d'erreur correspondant est émis au paramètre ErrorBits (Page 481).

La préconfiguration des valeurs dans la structure UserData ne constitue pas une configuration valide. Pour pouvoir utiliser ces variables pour le calcul d'interpolation, modifiez les variables et attribuez-leur des valeurs valides.

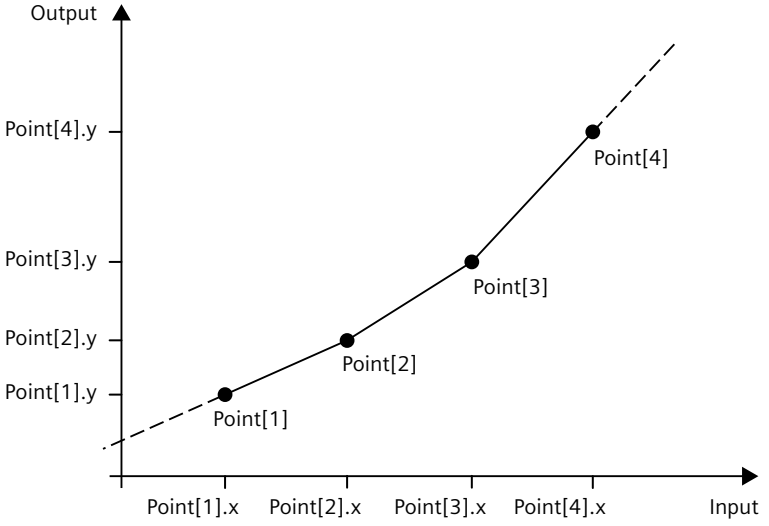
REMARQUE

Si votre application nécessite plus que le nombre maximal de 50 nœuds d'interpolation, veuillez utiliser deux instances de polyligne ou plus.

Calculer une valeur de réglage

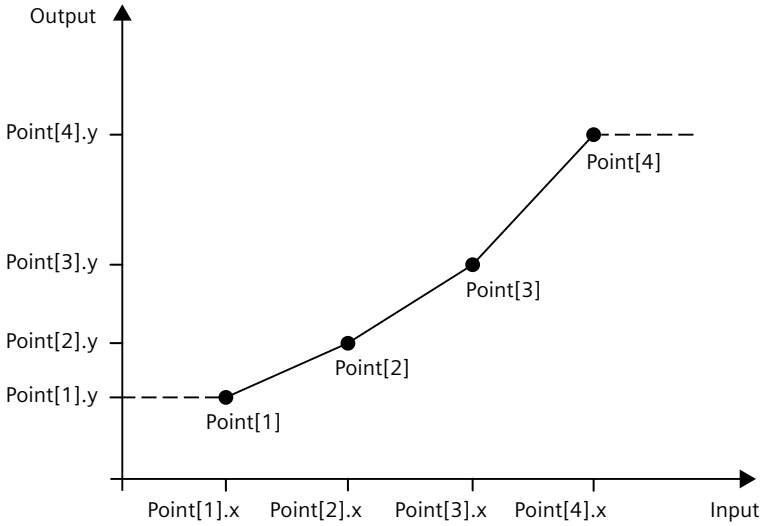
Si la valeur d'entrée au paramètre Input est inférieure à la première valeur de x ou supérieure à la dernière valeur de x des nœuds d'interpolation utilisés, effectuez les réglages suivants pour le paramètre Output à la variable OutOfRangeMode :

- OutOfRangeMode = 0
  La valeur de réglage est extrapolée avec la pente des deux premiers ou derniers nœuds d'interpolation.



Lorsque la variable OutOfRangeMode se trouve en-dehors de la plage de valeurs autorisée de 0 à 1, la préconfiguration par défaut 0 s'applique.

- OutOfRangeMode = 1
  La valeur de réglage est limitée à la valeur y du premier ou dernier nœud d'interpolation.



Le paramètre Output a une plage de valeurs autorisée d'un type de données REAL, allant de -3,402823e+38 à 3,402823e+38. La validité de la valeur de réglage au paramètre Output est contrôlée à chaque exécution de l'instruction Polyline. Si le calcul d'interpolation donne une valeur REAL non valide, la valeur de réglage est remplacée par le paramètre de la variable ErrorMode.

Comportement de validation EN/ENO

Si l'une des conditions suivantes est remplie, la sortie de validation ENO est réglée sur FALSE :

- L'entrée de validation EN est réglée sur TRUE et le paramètre Output est défini par une valeur de réglage de remplacement pour les messages d'erreur ErrorBits ≥ 16#0001_0000.
- L'entrée de validation EN est réglée sur FALSE.

Sinon, la sortie de validation ENO est réglée sur TRUE.

Nœuds d'interpolation actuellement utilisés

La variable NextXIndex émet l'indice de la valeur de x supérieure suivante pour la valeur d'entrée actuelle. Ainsi, vous pouvez déterminer les nœuds d'interpolation utilisés pour le calcul d'interpolation actuel.

WorkingData.Point[NextXIndex-1].x < Input ≤ WorkingData.Point[NextXIndex].x

Exemple :

- Si la valeur du paramètre Input est comprise entre WorkingData.Point[3].x et WorkingData.Point[4].x, alors la variable NextXIndex a la valeur 4.
- Si la valeur du paramètre Input est inférieure à WorkingData.Point[1].x, alors la variable NextXIndex a la valeur 1.
- Si la valeur du paramètre Input est supérieure à WorkingData.Point[WorkingData.NumberOfUsedPoints].x et donc supérieure à la dernière valeur de x du tracé polygonal, alors la variable NextXIndex a la valeur de la variable WorkingData.NumberOfUsedPoints + 1. Ainsi, la valeur maximale autorisée de la variable NextXIndex est égale à 51.

10.5.4 Paramètres d'entrée de la polyligne

Paramètre	Type de données	Valeur par défaut	Description
Input	REAL	0.0	Valeur d'entrée
SubstituteOutput	REAL	0.0	SubstituteOutput est utilisé comme valeur de réglage de remplacement, si • Reset = TRUE ou • aucun calcul correct du résultat d'interpolation n'est possible à cause d'une erreur avec le message d'erreur ErrorBits ≥ 16#0001_0000 et que ErrorMode est configuré à la valeur 1 .
Validate	BOOL	FALSE	Si Validate est réglé sur TRUE, la validité des données relatives au tracé polygonal dans UserData est contrôlée et ces données sont reprises selon WorkingData.
ErrorAck	BOOL	FALSE	Supprime les messages d'erreur • Front FALSE -> TRUE ErrorBits est réinitialisé
Reset	BOOL	FALSE	Exécute un redémarrage de l'instruction • Front FALSE -> TRUE ErrorBits est réinitialisé. • Tant que Reset est réglé sur TRUE, la valeur de réglage de remplacement SubstituteOutput est émise à la sortie. • Tant que Reset est réglé sur FALSE, le calcul d'interpolation est exécuté.

10.5.5 Paramètres de réglage de la polyligne

Paramètre	Type de données	Valeur par défaut	Description
Output	REAL	0.0	Valeur de réglage
Error	BOOL	FALSE	Le réglage de Error sur TRUE indique la présence d'au moins une erreur actuellement.
ErrorBits	DWORD	DW#16#0	Le paramètre ErrorBits (Page 481) signale quels sont les messages d'erreur présents. ErrorBits est rémanent et réinitialisé sur Reset ou ErrorAck en cas de front montant.

10.5.6 Variables statiques de la polyligne

Variable	Type de données	Valeur par défaut	Description
UserData	AuxFct_Point-Table	-	Zone de saisie pour les données relatives au tracé polygonal Les données relatives au tracé polygonal dans la structure UserData peuvent être éditées. Les modifications de cette structure ne se répercutent sur le calcul d'interpolation que lorsque le contrôle et la copie des données sont déclenchés dans la structure WorkingData.
UserData.NumberOfUsedPoints	INT	0	Nombre de nœuds d'interpolation utilisés pour le calcul d'interpolation Plage de valeurs autorisée : 2 à 50
UserData.Point	Array[1..50] of AuxFct_Point	-	Nœuds d'interpolation pour le calcul d'interpolation Le tableau (Array) comportant 50 éléments de type AuxFct_Point contient les paires de valeurs des nœuds d'interpolation.
UserData.Point[i]	AuxFct_Point	-	Nœud d'interpolation pour le calcul d'interpolation Un élément comportant l'indice "i" du tableau (Array) "Point".
UserData.Point[i].x	REAL	0.0	Valeur de x du nœud d'interpolation Plage de valeurs autorisée : Point[i].x < Point[i+1].x
UserData.Point[i].y	REAL	0.0	Valeur y du nœud d'interpolation
WorkingData	AuxFct_Point-Table	-	Zone d'affichage des données relatives au tracé polygonal actuellement effectives Les données relatives au tracé polygonal de la structure WorkingData ne peuvent pas être éditées. Elles sont utilisées pour le calcul d'interpolation.
WorkingData.NumberOfUsedPoints	INT	0	Nombre de nœuds d'interpolation utilisés pour le calcul d'interpolation Plage de valeurs autorisée : 2 à 50

Variable	Type de données	Valeur par défaut	Description
WorkingData.Point	Array[1..50] of AuxFct_Point	-	Nœuds d'interpolation pour le calcul d'interpolation Le tableau (Array) comportant 50 éléments de type AuxFct_Point contient les paires de valeurs des nœuds d'interpolation.
WorkingData.Point[i]	AuxFct_Point	-	Nœud d'interpolation pour le calcul d'interpolation Un élément comportant l'indice "i" du tableau (Array) "Point".
WorkingData.Point[i].x	REAL	0.0	Valeur de x du nœud d'interpolation Plage de valeurs autorisée : Point[i].x < Point[i+1].x
WorkingData.Point[i].y	REAL	0.0	Valeur y du nœud d'interpolation
ErrorMode	INT	0	Choix de la valeur de réglage de remplacement en cas d'erreur • 0 = Input • 1 = SubstituteOutput • 2 = dernière valeur de réglage valide Plage de valeurs autorisée : 0 à 2
OutOfRangeMode	INT	0	Choix de la valeur de réglage lorsque la valeur d'entrée est en-dehors des valeurs de x définies • 0 = conserver la pente • 1 = valeur de y du premier / dernier nœud d'interpolation Plage de valeurs autorisée : 0 à 1
NextXIndex	INT	2	Indice de la valeur de x suivante Sert à visualiser l'indice des points d'appui utilisés pour le calcul d'interpolation actuel. La condition suivante s'applique : WorkingData.Point[NextXIndex-1].x < Input ≤ WorkingData.Point[NextXIndex].x Ne modifiez pas cette valeur manuellement.

10.5.7 Paramètre ErrorBits

En présence de plusieurs erreurs simultanées, les valeurs des ErrorBits s'affichent comme addition binaire. L'affichage de ErrorBits = 16#0000_0003, par ex., indique la présence simultanée des erreurs 16#0000_0001 et 16#0000_0002.

Avec Polyline, les erreurs émises au paramètre ErrorBits sont réparties en deux catégories :

- les erreurs déclenchant des messages d'erreur ErrorBits < 16#0001_0000
- les erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000

Erreurs déclenchant des messages d'erreur ErrorBits < 16#0001_0000

En présence d'une ou de plusieurs erreurs déclenchant des messages d'erreur ErrorBits < 16#0001_0000, Polyline réagit de la manière suivante :

- La valeur de réglage est déterminée comme suit en dépit de cette erreur :
  - Calcul d'interpolation si Reset = FALSE
  - Sortie de SubstituteOutput si Reset = TRUE
- Le paramètre de sortie Error est mis à 1.
- La sortie de validation ENO n'est pas modifiée.

Le paramètre de sortie Error est supprimé dès que l'erreur a disparu.

ErrorBits (DW#16#...)	**Description**
0000_0000	Pas d'erreur.
0000_0001	**Cause d'erreur et réaction en cas d'erreur :** Le paramètre Output a été limité à -3.402823e+38 ou +3.402823e+38. **Solution :** Si la valeur d'interpolation est émise à la sortie (Reset = FALSE et ErrorBits < 16#0001_0000), vérifiez les variables suivantes utilisées dans le calcul d'interpolation : • Input • WorkingData.Point[i].x • WorkingData.Point[i].y Si ErrorBits ≥ 16#0001_0000 et Reset = FALSE, la valeur de réglage de remplacement est limitée à sa sortie. Vérifiez ensuite les paramètres suivants en fonction de la valeur paramétrée à la variable ErrorMode : • Input • SubstituteOutput Si Reset = TRUE, vérifiez le paramètre SubstituteOutput.
0000_0002	**Cause d'erreur :** Une ou plusieurs variables de la structure UserData ont des valeurs non valides, tandis que les données relatives au tracé polygonal sont vérifiées (Validate = TRUE et Reset = FALSE). **Réaction en cas d'erreur :** Les données relatives au tracé polygonal de la structure UserData ne sont pas reprises dans la structure WorkingData de telle sorte que les modifications effectuées dans la structure UserData ne sont pas appliquées. Le FB Polyline poursuit le calcul d'interpolation avec les valeurs de tracé polygonal non modifiées et valides dans la structure WorkingData. **Solution :** Vérifiez que les conditions suivantes sont remplies lorsque le paramètre Validate est réglé sur TRUE : • 2 ≤ UserData.NumberOfUsedPoints ≤ 50 • UserData.Point[j].x < UserData.Point[j+1].x avec l'indice j = 1..(UserData.NumberOfUsedPoints - 1) • -3.402823e+38 ≤ UserData.Point[i].x ≤ 3.402823e+38 avec l'indice i =1..UserData.NumberOfUsedPoints

ErrorBits (DW#16#...)	Description
	• -3.402823e+38 ≤ UserData.Point[i].y ≤ 3.402823e+38 avec l'indice i = 1..UserData.NumberOfUsedPoints • UserData.Point[i].x et UserData.Point[i].y sont des valeurs valides pour REAL (≠ NaN) avec l'indicei = 1..UserData.NumberOfUsedPoints

Erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000

En présence d'une ou de plusieurs erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000, Polyline réagit de la manière suivante :

- La valeur de réglage ne peut pas être déterminée comme prévu. À la place, c'est la valeur de réglage de remplacement qui est émise.
- Le paramètre de sortie Error est mis à 1.
- La sortie de validation ENO est réglée sur FALSE.

Dès que les erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000 ont disparu, Polyline réagit de la manière suivante :

- La valeur de réglage est déterminée comme suit :
  - Calcul d'interpolation si Reset = FALSE
  - Sortie de SubstituteOutput si Reset = TRUE
- La sortie de validation ENO est réglée sur TRUE.

Le paramètre de sortie Error est supprimé dès que l'erreur a disparu.

<table>
<tr><th>ErrorBits (DW#16#...)</th><th>Description</th></tr>
<tr><td>0001_0000</td><td>Cause d'erreur :
Le paramètre SubstituteOutput ou Input qui est utilisé comme valeur de réglage n'a pas de valeur REAL valide.
Réaction en cas d'erreur :
La sortie est réglée sur 0.0.
Solution :
Vérifiez que le paramètre utilisé comme valeur de réglage est une valeur REAL valide (≠ NaN p. ex. 16#7FFF_FFFF). Le paramètre utilisé comme valeur de réglage dépend de Reset et de ErrorMode :
<table>
<tr><th>Reset</th><th>ErrorMode</th><th>Valeur de réglage</th></tr>
<tr><td>FALSE</td><td>0</td><td>Input</td></tr>
<tr><td>FALSE</td><td>1</td><td>SubstituteOutput</td></tr>
<tr><td>TRUE</td><td>-</td><td>SubstituteOutput</td></tr>
</table></td></tr>
<tr><td>0002_0000</td><td>Cause d'erreur :
Le paramètre Input n'a pas de valeur REAL valide, tandis que le calcul d'interpolation est exécuté (Reset = FALSE).
Réaction en cas d'erreur :
La valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output.
0.0 est émis comme valeur de réglage si ErrorMode = 0.
La variable NextXIndex n'est pas actualisée tant que le paramètre Input n'a pas de valeur REAL valide.
Solution :
Vérifiez que le paramètre Input est une valeur REAL valide (≠ NaN p. ex. 16#7FFF_FFFF).</td></tr>
</table>

ErrorBits (DW#16#...)	Description
0004_0000	**Cause d'erreur :** Le calcul d'interpolation donne une valeur REAL non valide pour le paramètre Output. **Réaction en cas d'erreur :** La valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output. La variable NextXIndex peut avoir une valeur invalide tant que cette erreur est présente. **Solution :** Vérifiez la validité des valeurs REAL dans la structure WorkingData. **Informations complémentaires :** Si vous souhaitez modifier les données relatives au tracé polygonal, éditez d'abord la structure UserData et définissez ensuite le paramètre Validate = TRUE. Ne modifiez pas manuellement les données de la structure WorkingData.
0008_0000	**Cause d'erreur :** Une ou plusieurs variables de la structure UserData ont des valeurs non valides, tandis que les données relatives au tracé polygonal sont vérifiées. **Réaction en cas d'erreur :** Les données relatives au tracé polygonal de la structure UserData ne sont pas reprises dans la structure WorkingData afin que les valeurs de la structure UserData ne soient pas appliquées. Le FB Polyline n'émet pas la valeur d'interpolation au paramètre Output car la structure WorkingData ne contient aucune donnée relative au tracé polygonal valide. La valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output. **Solution :** Vérifiez que les conditions suivantes sont remplies lorsque les données du tracé polygonal sont vérifiées : • 2 ≤ UserData.NumberOfUsedPoints ≤ 50 • UserData.Point[j].x < UserData.Point[j+1].x avec l'indice j = 1..(UserData.NumberOfUsedPoints – 1) • -3.402823e+38 ≤ UserData.Point[i].x ≤ 3.402823e+38 avec l'indice i = 1..UserData.NumberOfUsedPoints • -3.402823e+38 ≤ UserData.Point[i].y ≤ 3.402823e+38 avec l'indice i = 1..UserData.NumberOfUsedPoints • UserData.Point[i].x et UserData.Point[i].y sont des valeurs valides pour REAL (≠ NaN) avec l'indice i = 1..UserData.NumberOfUsedPoints **Informations complémentaires :** Les données relatives au tracé polygonal dans la structure UserData sont vérifiées si • le paramètre Validate est réglé sur TRUE tandis que le paramètre Reset est réglé sur FALSE ou • Polyline après que chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN est appelé pour la première fois avec le paramètre Reset = FALSE. Notez que les variables des structures UserData et WorkingData ne sont pas toutes rémanentes. Ces variables sont initialisées à l'aide des valeurs initiales après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN.

10.6 SplitRange

10.6.1 Compatibilité avec CPU et FW

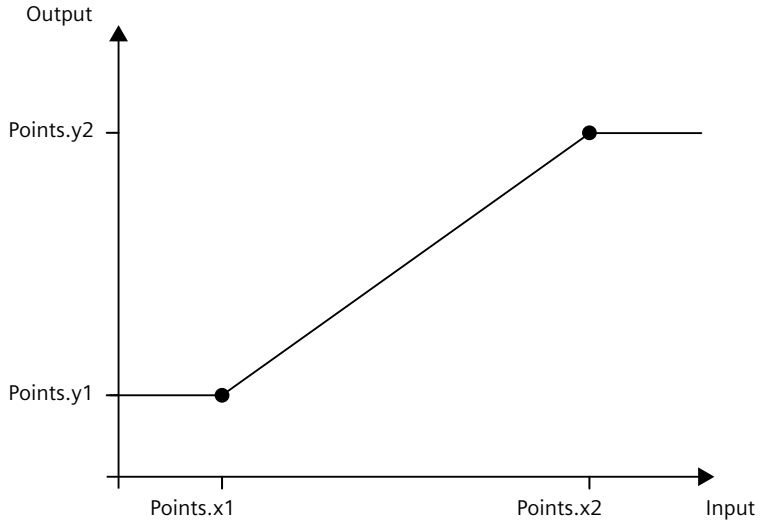
Le tableau suivant montre la compatibilité entre les CPU et les versions de SplitRange :

CPU	FW	SplitRange
S7-1200	à partir de V4.2	V1.0
CPU basées sur S7-1500	à partir de V2.0	V1.0

10.6.2 Description de SplitRange

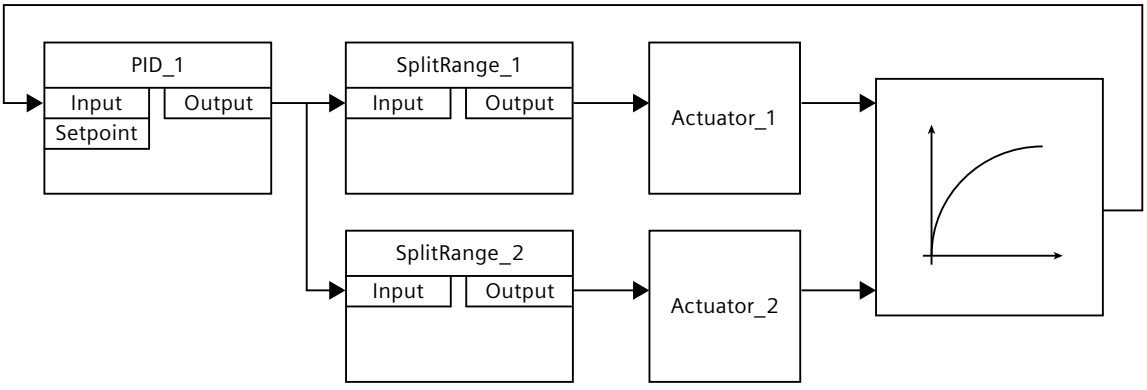
Description

L'instruction SplitRange convertit la valeur d'entrée en valeur de réglage. La valeur d'entrée se situe dans la plage de valeurs qui est délimitée par Points.x1 et Points.x2 La valeur d'entrée se situe dans la plage de valeurs qui est délimitée par Points.y1 et Points.y2.
La figure suivante représente la caractéristique d'un exemple de configuration de l'instruction SplitRange :



Utilisez SplitRange si vous avez besoin de réguler un processus influencé par plusieurs actionneurs. SplitRange divise la plage des valeurs de réglage du régulateur PID en plusieurs sous-intervalles. Attribuez un sous-intervalle à chaque actionneur. Le programme utilisateur appelle le bloc une fois par sous-intervalle. La valeur d'entrée de chaque instance SplitRange est liée à la valeur de réglage du régulateur PID.

La figure suivante montre, à titre illustratif, un circuit de régulation avec deux instances de SplitRange et deux actionneurs :



Validité des données SplitRange

Les paires de valeurs dans la structure Points définissent la plage des valeurs d'entrées et la plage des valeurs de réglage de SplitRange. Les deux paires de valeur se trouvent dans la plage statique du bloc SplitRange.
SplitRange vérifie à chaque appel si les conditions suivantes sont remplies, pour que des valeurs valides pour le calcul de la valeur de réglage soient disponibles :

- Points.x1 < Points.x2
- Points.x1, Points.y1, Points.x2 et Points.y2 se situent dans la plage de valeurs admissibles -3.402823e+38 à 3.402823e+38
- Points.x1, Points.y1, Points.x2 et Points.y2 sont des valeurs REAL valides (≠ NaN par ex. 16#7FFF_FFFF)

Si une ou plusieurs conditions ne sont pas remplies, aucun calcul correct de la valeur de réglage n'est possible. Un message d'erreur correspondant est émis au paramètre ErrorBits.
La préconfiguration des valeurs de x et y avec 0.0 ne constitue pas une configuration valide. Pour utiliser ces variables pour le calcul de la valeur de réglage, modifiez les variables et attribuez-leur des valeurs valides.

Comportement de validation EN/ENO

Si l'une des conditions suivantes est remplie, la sortie de validation ENO est réglée sur FALSE :

- L'entrée de validation EN est réglée sur TRUE et le paramètre Output est défini par une valeur de réglage de remplacement pour les messages d'erreur ErrorBits ≥ 16#0001_0000.
- L'entrée de validation EN est réglée sur FALSE.

Sinon, la sortie de validation ENO est réglée sur TRUE.

Appel

Dans un OB ou une FC, l'instruction SplitRange est appelée comme DB d'instance unique. Dans un FB, l'instruction SplitRange peut être appelée aussi bien comme DB d'instance unique que comme DB de multi-instance ou DB d'instance de paramètre.
Aucun objet technologique n'est créé lors de l'appel de l'instruction. Vous ne disposez pas d'interface de paramétrage et mise en service. Vous paramétrez SplitRange directement via le DB d'instance et vous mettez en service SplitRange via une table de visualisation du programme utilisateur dans la CPU ou HMI.

Démarrage

Les variables de la plage statique de SplitRange ne sont pas rémanentes. Ces variables sont initialisées à l'aide des valeurs initiales après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN.
Si vous modifiez les valeurs actuelles dans la structure Points en mode en ligne et que ces valeurs doivent être conservées après le passage de la CPU de l'état de fonctionnement STOP à l'état RUN, sauvegardez ces valeurs dans les valeurs initiales du bloc de données.

Comportement en cas d'erreur

L'instruction SplitRange détecte différentes erreurs pouvant survenir lors du calcul de la valeur de réglage. Le résultat du calcul peut être émis en dépit de la présence d'une erreur à la sortie. Si aucun calcul correct de la valeur de réglage n'est possible à cause d'une erreur, une valeur de réglage de remplacement est émise à la sortie.
Vous définissez, pour la variable ErrorMode, la variable de réglage de remplacement de la manière suivante en cas d'erreur rendant impossible tout calcul correct de la valeur de réglage :

ErrorMode	Output
0	Valeur du paramètre Input
1	Valeur du paramètre SubstituteOutput
2	Dernier résultat valide du calcul de la valeur de réglage 0.0, en l'absence de résultat valide

Les points suivants s'appliquent en plus pour toutes les valeurs de la variable ErrorMode :

- Si la valeur de réglage de remplacement n'a pas de de valeur REAL valide, 0.0 est émis comme valeur de réglage.
- La valeur de réglage de remplacement est limitée à la plage de valeurs du type de données REAL. Ce n'est qu'ensuite que la valeur de réglage de remplacement est émise au paramètre Output.
- La variable ErrorMode n'est active que lorsque le paramètre Reset = FALSE . Si le paramètre Reset = TRUE, la valeur du paramètre SubstituteOutput est émise au paramètre Output.

Le paramètre Error indique si une erreur est actuellement présente. Quand l'erreur a disparu, Error est réglé sur FALSE. Le paramètre ErrorBits indique les erreurs qui se sont produites. ErrorBits est rémanent et n'est réinitialisé que par un front montant au paramètre Reset ou ErrorAck.

10.6.3 Paramètre d'entrée SplitRange

Paramètre	Type de données	Valeur par défaut	Description
Input	REAL	0.0	Valeur d'entrée
SubstituteOutput	REAL	0.0	SubstituteOutput est utilisé comme valeur de réglage de remplacement, si • Reset = TRUE ou • aucun calcul correct de la valeur de réglage n'est possible à cause d'une erreur avec le message d'erreur ErrorBits ≥ 16#0001_0000 et que ErrorMode est configuré à la valeur 1.
ErrorAck	BOOL	FALSE	Supprime les messages d'erreur • Front FALSE -> TRUE ErrorBits est réinitialisé
Reset	BOOL	FALSE	Exécute un redémarrage de l'instruction • Front FALSE -> TRUE ErrorBits est réinitialisé. • Tant que Reset est réglé sur TRUE, la valeur de réglage de remplacement SubstituteOutput est émise à la sortie. • Tant que Reset est réglé sur FALSE, le calcul de la valeur de réglage est exécuté.

10.6.4 Paramètre de sortie SplitRange

Paramètre	Type de données	Valeur par défaut	Description
Output	REAL	0.0	Valeur de réglage
ErrorBits	DWORD	DW#16#0	Le paramètre ErrorBits (Page 488) signale quels sont les messages d'erreur présents. ErrorBits est rémanent et réinitialisé sur Reset ou ErrorAck en cas de front montant.
Error	BOOL	FALSE	Le réglage de Error sur TRUE indique la présence d'au moins une erreur actuellement.

10.6.5 Variables statiques SplitRange

Variable	Type de données	Valeur par défaut	Description
Points	AuxFct_SplitRange_Points	-	Données du nœud d'interpolation
Points.x1	REAL	0.0	Valeur x du nœud d'interpolation 1 Plage de valeurs autorisée : Points.x1 < Points.x2
Points.y1	REAL	0.0	Valeur y du nœud d'interpolation 1
Points.x2	REAL	0.0	Valeur x du nœud d'interpolation 2 Plage de valeurs autorisée : Points.x1 < Points.x2
Points.y2	REAL	0.0	Valeur y du nœud d'interpolation 2
ErrorMode	INT	0	Choix de la valeur de réglage de remplacement en cas d'erreur • 0 = Input • 1 = SubstituteOutput • 2 = dernière valeur de réglage valide Plage de valeurs autorisée : 0 à 2

10.6.6 Paramètre ErrorBits

En présence de plusieurs erreurs simultanées, les valeurs des ErrorBits s'affichent comme addition binaire. L'affichage de ErrorBits = 16#0000_0003, par ex., indique la présence simultanée des erreurs 16#0000_0001 et 16#0000_0002.

Avec SplitRange, les erreurs émises au paramètre ErrorBits sont réparties en deux catégories :

- les erreurs déclenchant des messages d'erreur ErrorBits < 16#0001_0000
- les erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000

Erreurs déclenchant des messages d'erreur ErrorBits < 16#0001_0000

En présence d'une ou de plusieurs erreurs déclenchant des messages d'erreur ErrorBits < 16#0001_0000, SplitRange réagit de la manière suivante :

- La valeur de réglage est déterminée comme suit en dépit de cette erreur :
  - Calcul de la valeur de réglage si Reset = FALSE
  - Sortie de SubstituteOutput si Reset = TRUE
- Le paramètre de sortie Error est mis à 1.
- La sortie de validation ENO n'est pas modifiée.

Le paramètre de sortie Error est supprimé dès que l'erreur a disparu.

ErrorBits (DW#16#...)	Description
0000_0000	Pas d'erreur.
0000_0001	**Cause d'erreur et réaction en cas d'erreur :** Le paramètre Output a été limité à -3.402823e+38 ou +3.402823e+38. **Solution :** Si ErrorBits ≥ 16#0001_0000 et Reset = FALSE, la valeur de réglage de remplacement est limitée à sa sortie. Vérifiez ensuite les paramètres suivants en fonction de la valeur paramétrée à la variable ErrorMode : • Input • SubstituteOutput Si Reset = TRUE, vérifiez le paramètre SubstituteOutput.

Erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000

En présence d'une ou de plusieurs erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000, SplitRange réagit de la manière suivante :

- La valeur de réglage ne peut pas être déterminée comme prévu. À la place, c'est la valeur de réglage de remplacement qui est émise.
- Le paramètre de sortie Error est mis à 1.
- La sortie de validation ENO est réglée sur FALSE.

Dès que les erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000 ont disparu, SplitRange réagit de la manière suivante :

- La valeur de réglage est déterminée comme suit :
  - Calcul de la valeur de réglage si Reset = FALSE
  - Sortie de SubstituteOutput si Reset = TRUE
- La sortie de validation ENO est réglée sur TRUE.

Le paramètre de sortie Error est supprimé dès que l'erreur a disparu.

<table>
<tr><th>ErrorBits (DW#16#...)</th><th>Description</th></tr>
<tr><td>0001_0000</td><td>Cause d'erreur :
Le paramètre SubstituteOutput ou Input qui est utilisé comme valeur de réglage n'a pas de valeur REAL valide.
Réaction en cas d'erreur :
La sortie est réglée sur 0.0.
Solution :
Vérifiez que le paramètre utilisé comme valeur de réglage est une valeur REAL valide (≠ NaN p. ex. 16#7FFF_FFFF). Le paramètre utilisé comme valeur de réglage dépend de Reset et de ErrorMode :
<table>
<tr><td>Reset</td><td>ErrorMode</td><td>Valeur de réglage</td></tr>
<tr><td>FALSE</td><td>0</td><td>Input</td></tr>
<tr><td>FALSE</td><td>1</td><td>SubstituteOutput</td></tr>
<tr><td>TRUE</td><td>-</td><td>SubstituteOutput</td></tr>
</table></td></tr>
<tr><td>0002_0000</td><td>Cause d'erreur :
Le paramètre Input n'a pas de valeur REAL valide, tandis que le calcul de la valeur de réglage est exécuté (Reset = FALSE).
Réaction en cas d'erreur :
La valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output.
0.0 est émis comme valeur de réglage si ErrorMode = 0.
Solution :
Vérifiez que le paramètre Input est une valeur REAL valide (≠NaN p. ex. 16#7FFF_FFFF).</td></tr>
<tr><td>0004_0000</td><td>Causes d'erreurs possibles :
• Une ou plusieurs variables de la structure Points ont des valeurs non valides.
• Le calcul de la valeur de réglage donne une valeur REAL non valide pour le paramètre Output.
Réaction en cas d'erreur :
La valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output.
Solution :
Vérifiez que les conditions suivantes sont remplies :
1. Points.x1 < Points.x2
2. Points.x1, Points.y1, Points.x2 et Points.y2 se situent dans la plage de valeurs admissible de -3.402823e+38 à 3.402823e+38
3. Points.x1, Points.y1, Points.x2 et Points.y2 sont des valeurs REAL valides (≠ NaN par ex. 16#7FFF_FFFF)</td></tr>
</table>

ErrorBits (DW#16#...)	Description
	Informations complémentaires : Notez que les variables de la structure Points ne sont pas toutes rémanentes. Ces variables sont initialisées à l'aide des valeurs initiales après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN.

10.7 RampFunction

10.7.1 Compatibilité avec CPU et FW

Le tableau suivant montre la compatibilité entre les CPU et les versions deRampFunction :

CPU	FW	RampFunction
S7-1200	à partir de V4.2	V1.0
CPU basées sur S7-1500	à partir de V2.0	V1.0

10.7.2 Description de RampFunction

Description

L'instruction RampFunction limite la vitesse de modification d'un signal. RampFunction émet un saut de signal à l'entrée comme fonction de rampe de la valeur de réglage.

Utilisez RampFunction pour éviter des sauts de signal, par ex. dans les cas suivants :

- Entre la source de consigne et l'entrée de consigne du régulateur pour obtenir un comportement de guidage plus doux, sans influer sur le comportement de perturbation.
- Entre la sortie du régulateur et l'entrée de l'actionneur, pour ménager l'actionneur, par ex. un moteur avec entraînement, ou le processus.

Les valeurs limites suivantes sont réglables pour la vitesse de modification :

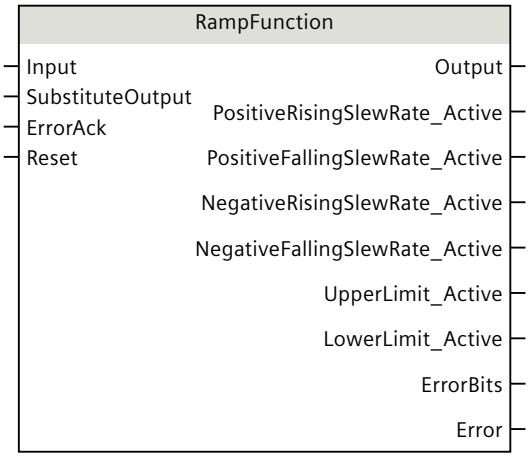
- Augmentation de la vitesse de modification dans la plage de valeurs positive
- Diminution de la vitesse de modification dans la plage de valeurs positive
- Augmentation de la vitesse de modification dans la plage de valeurs négative
- Diminution de la vitesse de modification dans la plage de valeurs négative

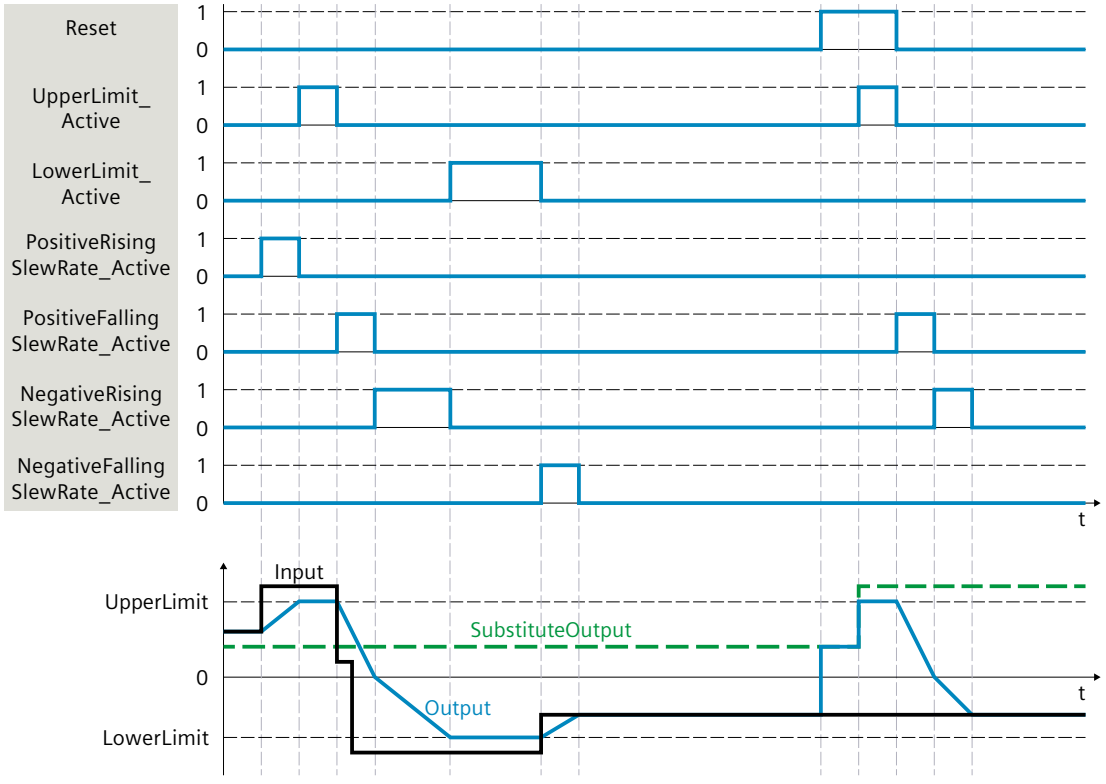
De plus, l'instruction RampFunction limite la valeur de réglage à la valeur limite inférieure et la valeur limite supérieure.

Si la limite de la vitesse de modification ou la limite inférieure ou supérieure est atteinte, RampFunction met le bit de sortie correspondant sur TRUE.

Diagramme fonctionnel

La figure suivante représente l'instruction RampFunction et un exemple de diagramme fonctionnel :





Appel

Dans un OB ou une FC, l'instruction RampFunction est appelée comme DB d'instance unique. Dans un FB, l'instruction RampFunction peut être appelée aussi bien comme DB d'instance unique que comme DB de multi-instance ou DB d'instance de paramètre.
Aucun objet technologique n'est créé lors de l'appel de l'instruction. Vous ne disposez pas d'interface de paramétrage et mise en service. Vous paramétrez RampFunction directement via le DB d'instance et vous mettez en service RampFunction via une table de visualisation du programme utilisateur dans la CPU ou HMI.

Démarrage

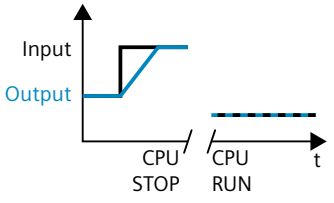
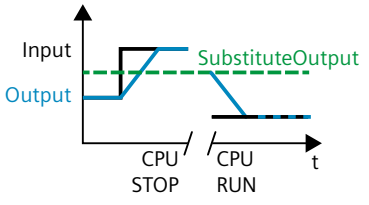
Les variables de la plage statique de RampFunction ne sont pas rémanentes. Ces variables sont initialisées à l'aide des valeurs initiales après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN.
Si vous modifiez les valeurs actuelles des valeurs limites en mode en ligne et que ces valeurs doivent être conservées après le changement d'état de fonctionnement de la CPU, sauvegardez ces valeurs dans les valeurs initiales du bloc de données.
Vous définissez la valeur d'initialisation pour le paramètre Output sur la variable StartMode.
La valeur d'initialisation est émise au premier appel de RampFunction après le

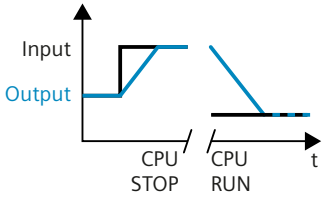
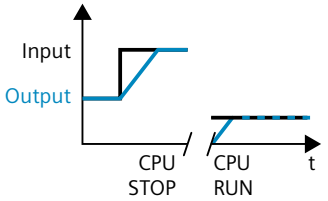
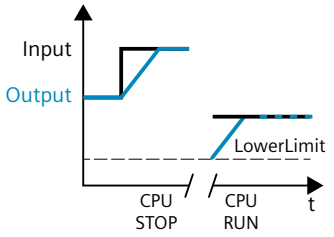
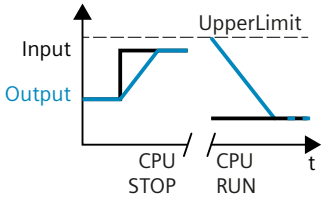
- changement d'état de fonctionnement de la CPU

ou

- l'exécution de "Charger les valeurs de départ comme valeurs en cours" (uniquement pour l'option "Toutes les valeurs", et pas pour l'option "Consignes uniquement")

au paramètre Output.
Lors des appels suivants, RampFunction calcule la valeur de réglage, à partir de cette valeur d'initialisation, sur la base de la valeur d'entrée et des valeurs limites pour la vitesse de modification.
Le tableau suivant présente la dépendance entre la variable StartMode et le paramètre Output. Les valeurs de la colonne Output sont émises au paramètre Output après le changement d'état de fonctionnement de la CPU :

StartMode	Output	Exemple
0	Valeur du paramètre Input	
1	Valeur du paramètre SubstituteOutput	

StartMode	Output	Exemple
2	Reste inchangé. Le paramètre Output est rémanent.	
3	0.0	
4	Valeur de de la variable LowerLimit	
5	Valeur de de la variable UpperLimit	

Les points suivants s'appliquent en plus pour toutes les valeurs de la variable StartMode :

- Si les valeurs des variables UpperLimit et LowerLimit sont valides, la valeur d'initialisation est limitée à la plage de valeurs de ces variables. Ce n'est qu'ensuite que la valeur d'initialisation est émise au paramètre Output.
- Si la valeur d'initialisation n'est pas une valeur REAL valide, la valeur de réglage de remplacement est émise au paramètre Output. La valeur de réglage de remplacement est configurée par la variable ErrorMode. La valeur de réglage de remplacement est limitée par la plage de valeurs des variables UpperLimit et LowerLimit. Si la valeur de réglage de remplacement n'est pas non plus une valeur REAL valide, 0.0 est émise au paramètre Output. Lors des appels suivants, l'instruction calcule la valeur de réglage à partir de cette valeur de réglage de remplacement.
- La variable StartMode n'est active que lorsque le paramètre Reset = FALSE lors du premier appel de l'instruction et si aucune erreur n'apparaît avec le message d'erreur ErrorBits ≥ 16#0002_0000. Lorsque le paramètre Reset = TRUE, la valeur du paramètre SubstituteOutput est émise au paramètre Output. En présence d'une erreur déclenchant un message d'erreur ErrorBits ≥ 16#0002_0000, la valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output.

Comportement en cas d'erreur

L'instruction RampFunction détecte différentes erreurs pouvant survenir lors du calcul de la valeur de réglage. Le résultat de ce calcul peut être émis en dépit de la présence d'une erreur à la sortie. Si aucun calcul correct de la valeur de réglage n'est possible à cause d'une erreur, une valeur de réglage de remplacement est émise à la sortie.
Vous définissez, pour la variable ErrorMode, la valeur de réglage de remplacement en cas d'erreur rendant impossible tout calcul correct de la valeur de réglage.
Le tableau suivant présente la dépendance entre la valeur de la variable ErrorMode et la valeur de réglage de remplacement que RampFunction émet au paramètre Output :

ErrorMode	Output
0	Valeur du paramètre Input
1	Valeur du paramètre SubstituteOutput
2	La dernière valeur de réglage valide au paramètre Output
3	0.0
4	Valeur de de la variable LowerLimit
5	Valeur de de la variable UpperLimit

Les points suivants s'appliquent en plus pour toutes les valeurs de la variable ErrorMode :

- Si la valeur de réglage de remplacement n'a pas de de valeur réelle valide, 0.0 est émis comme valeur de réglage.
- Si les valeurs des variables UpperLimit et LowerLimit sont valides, la valeur de réglage de remplacement est limitée à la plage de valeurs de ces variables. Ce n'est qu'ensuite que la valeur de réglage de remplacement est émise au paramètre Output.
- La variable ErrorMode n'est active que lorsque le paramètre Reset = FALSE . Si le paramètre Reset = TRUE, la valeur du paramètre SubstituteOutput est émise au paramètre Output.
- En présence d'une erreur qui empêche un calcul correct de la valeur de réglage, RampFunction passe, au paramètre Output, de la valeur de réglage calculé à la valeur de réglage de remplacement. Ce faisant, un saut de la valeur de réglage peut se produire selon la valeur de la variable ErrorMode.

Le paramètre Error indique si une erreur est actuellement présente. Quand l'erreur a disparu, Error est réglé sur FALSE. Le paramètre ErrorBits indique les erreurs qui se sont produites. ErrorBits est rémanent et n'est réinitialisé que par un front montant au paramètre Reset ou ErrorAck.

10.7.3 Fonctionnement de RampFunction

Limiter la vitesse de modification

Quatre valeurs limites peuvent être configurées pour la vitesse de modification du signal d'entrée. La valeur limite qui s'applique actuellement dépend des facteurs suivants :

- Signe de la valeur de réglage au paramètre Output
- Sens de modification de la valeur absolue de la valeur de réglage au paramètre Output

Le tableau suivant montre les variables opérantes pour la valeur limite de la vitesse de modification en fonction du paramètre Output :

Output	Variable opérante
Output ≥ 0 et |Output| montant	PositiveRisingSlewRate
Output ≥ 0 et |Output| descendant	PositiveFallingSlewRate
Output < 0 et |Output| montant	NegativeRisingSlewRate
Output < 0 et |Output| descendant	NegativeFallingSlewRate

La valeur absolue des valeurs limites de la vitesse de modification définit la modification maximale de la valeur de réglage par seconde.

Exemple :

Le scénario suivant est valable pour l'exemple :

- PositiveRisingSlewRate = 10.0
- Temps d'appel de RampFunction = 0.1 s
- Input > Output ≥ 0.0

Résultat :

La valeur de réglage Output augmente de 1.0 par appel (10.0 par seconde) jusqu'à ce que la valeur du paramètre Input soit atteinte.

Pour désactiver la limitation de la vitesse de modification pour une ou plusieurs plages, réglez la variable correspondante sur la valeur 3.402823e+38.

Si la valeur de réglage Output est actuellement limitée par une valeur limite de la vitesse de modification, RampFunction règle alors le bit de sortie correspondant sur TRUE :

- PositiveRisingSlewRate_Active
- PositiveFallingSlewRate_Active
- NegativeRisingSlewRate_Active
- NegativeFallingSlewRate_Active

Si le paramètre Reset est réglé sur TRUE, les valeurs limites de la vitesse de modification ne sont pas opérantes. Ainsi, des sauts au paramètre SubstituteOutput aboutissent à des sauts au paramètre Output.

RampFunction vérifie à chaque appel si les conditions suivantes sont remplies pour les variables PositiveRisingSlewRate, PositiveFallingSlewRate, NegativeRisingSlewRate et NegativeFallingSlewRate :

- Les valeurs se situent dans la plage de valeurs admissible allant de 0.0 non inclus à 3.402823e+38
- Les valeurs sont des valeurs REAL valides (≠ NaN, par ex. 16#7FFF_FFFF)

Si une ou plusieurs conditions ne sont pas remplies, la valeur de réglage de remplacement est émise au paramètre Output. Le message d'erreur correspondant est émis au paramètre ErrorBits.

Limiter la valeur de réglage

La valeur de réglage Output est toujours limitée à la plage de valeurs des variables UpperLimit et LowerLimit tant que ces variables ont des valeurs valides.

Si la valeur de réglage Output est actuellement limitée par cette plage de valeurs, RampFunction règle alors le bit de sortie correspondant sur TRUE :

- UpperLimit_Active
- LowerLimit_Active

La limitation de la valeur de réglage a une priorité plus élevée que la limitation de la vitesse de modification. Ainsi, les modifications des variables UpperLimit et LowerLimit aboutissent à des sauts de la valeur de réglage Output, si cela est nécessaire pour respecter les valeurs limites des variables UpperLimit et LowerLimit. La limitation de la vitesse de modification n'est pas prise en compte dans ce cas.

Exemple :

Si la valeur limite UpperLimit est réduite de 100,0 à 80,0, tandis que les valeurs des paramètres Input et Output s'élèvent à 90,0, la valeur de réglage Output saute à 80.0. La valeur de réglage Output saute à 80,0, que cela provoque ou non un dépassement de la valeur limite configurée pour la vitesse de modification.

RampFunction vérifie à chaque appel si les conditions suivantes sont remplies :

- LowerLimit < UpperLimit
- LowerLimit et UpperLimit se situent dans la plage de valeurs admissible -3.402823e+38 à 3.402823e+38
- LowerLimit et UpperLimit sont des valeurs REAL valides (≠ NaN par ex. 16#7FFF_FFFF)

Si une ou plusieurs conditions ne sont pas remplies, la valeur de réglage de remplacement est émise au paramètre Output. Le message d'erreur correspondant est émis au paramètre ErrorBits.

En plus, RampFunction vérifie à chaque appel si la valeur de réglage Output a la plage de valeurs admissible d'un type de données REAL, soit de -3.402823e+38 à 3.402823e+38. Si le calcul de la valeur de réglage donne une valeur REAL non valide, la valeur de réglage de remplacement est émise au paramètre Output. Vous configurez la valeur de réglage de remplacement sur la variable ErrorMode.

Comportement de validation EN/ENO

Si l'une des conditions suivantes est remplie, la sortie de validation ENO est réglée sur FALSE :

- L'entrée de validation EN est réglée sur TRUE et le paramètre Output est défini par une valeur de réglage de remplacement pour les messages d'erreur ErrorBits ≥ 16#0001_0000.
- L'entrée de validation EN est réglée sur FALSE.

Sinon, la sortie de validation ENO est réglée sur TRUE.

Mesure automatique du temps de cycle

Pour le calcul de la vitesse de modification de la valeur de réglage, RampFunction a besoin du temps qui s'est écoulé depuis le dernier appel de RampFunction.

Le temps de cycle est automatiquement mesuré par défaut et est émis au niveau de la variable CycleTime.Value à partir du deuxième appel. RampFunction mesure le temps de cycle à chaque appel de l'instruction et est donc utilisable dans des cycles d'appel non équidistants, par ex. dans l'OB1.

Notez que des appels conditionnels de l'instruction, les points d'arrêt actifs ou le chargement d'instantanés comme valeurs en cours rallongent le temps de cycle en cas de mesure automatique du temps de cycle.

Si la mesure du temps de cycle ne fournit aucun résultat valide, RampFunction calcule la valeur de réglage actuelle à l'aide du dernier temps de cycle valide. De plus, RampFunction émet un message d'erreur au paramètre ErrorBits.

Si vous désactivez la mesure automatique du temps de cycle en mettant à 1 la variable CycleTime.EnableMeasurement = FALSE, vous devez spécifier le temps de cycle manuellement au niveau de la variable CycleTime.Value. RampFunction vérifie la validité de la variable CycleTime.Value à chaque appel.

Mesure automatique du temps de cycle avec points d'arrêt

Si des points d'arrêt sont actifs entre deux appels de RampFunction, la mesure automatique du temps de cycle donne le temps réel qui s'est écoulé entre deux appels. Si un point d'arrêt est actif, la CPU se trouve à l'état de fonctionnement ATTENTE.

REMARQUE

Les points d'arrêt actifs allongent l'intervalle de temps entre deux appels de RampFunction.

Avec un intervalle de temps plus long entre deux appels, la modification maximale autorisée de la valeur de réglage augmente également au paramètre Output.

Exemple :

Le scénario suivant est valable pour l'exemple :

- PositiveRisingSlewRate = 10.0
- Temps d'appel de RampFunction = 0.1 s
- Input > Output ≥ 0.0

Résultat sans points d'arrêt :

La valeur de réglage Output augmente de 1.0 par appel jusqu'à ce que la valeur du paramètre Input soit atteinte.

Résultat pour un point d'arrêt actif pendant dix secondes :

Lors de l'appel suivant, la valeur de réglage Output augmente de 100.0.

Si vous n'avez pas besoin du calcul de la valeur de réglage sur la base du temps réel avec des points d'arrêt actifs, exécutez alors les étapes suivantes :

- Désactivez la mesure automatique du temps de cycle en mettant à 1 la variable CycleTime.EnableMeasurement = FALSE.
- Saisissez le temps de cycle manuellement au niveau de la variable CycleTime.Value.

10.7.4 Paramètre d'entrée RampFunction

Paramètre	Type de données	Valeur par défaut	Description
Input	REAL	0.0	Valeur d'entrée
SubstituteOutput	REAL	0.0	SubstituteOutput est utilisé comme valeur de réglage de remplacement, si • Reset = TRUE ou • aucun calcul correct de la valeur de réglage n'est possible à cause d'une erreur avec le message d'erreur ErrorBits ≥ 16#0001_0000 et que ErrorMode est configuré à la valeur 1.
ErrorAck	BOOL	FALSE	Supprime les messages d'erreur • Front FALSE -> TRUE ErrorBits est réinitialisé
Reset	BOOL	FALSE	Exécute un redémarrage de l'instruction • Front FALSE -> TRUE ErrorBits est réinitialisé. • Tant que Reset est réglé sur TRUE, la valeur de réglage de remplacement SubstituteOutput est émise à la sortie. • Tant que Reset est réglé sur FALSE, le calcul de la valeur de réglage est exécuté.

10.7.5 Paramètre de sortie RampFunction

Paramètre	Type de données	Valeur par défaut	Description
Output	REAL	0.0	Valeur de réglage
PositiveRisingSlewRate_Active	BOOL	FALSE	Si PositiveRisingSlewRate_Active = TRUE, la valeur de réglage est actuellement limitée par PositiveRisingSlewRate .
PositiveFallingSlewRate_Active	BOOL	FALSE	Si PositiveFallingSlewRate_Active = TRUE, la valeur de réglage est actuellement limitée par PositiveFallingSlewRate.
NegativeRisingSlewRate_Active	BOOL	FALSE	Si NegativeRisingSlewRate_Active = TRUE, la valeur de réglage est actuellement limitée par NegativeRisingSlewRate.
NegativeFallingSlewRate_Active	BOOL	FALSE	Si NegativeFallingSlewRate_Active = TRUE, la valeur de réglage est actuellement limitée par NegativeFallingSlewRate.
UpperLimit_Active	BOOL	FALSE	Si UpperLimit_Active = TRUE, la valeur de réglage est actuellement limitée par UpperLimit.
LowerLimit_Active	BOOL	FALSE	Si LowerLimit_Active = TRUE, la valeur de réglage est actuellement limitée par LowerLimit.
ErrorBits	DWORD	DW#16#0	Le paramètre ErrorBits (Page 500) signale quels sont les messages d'erreur présents. ErrorBits est rémanent et réinitialisé sur Reset ou ErrorAck en cas de front montant.
Error	BOOL	FALSE	Error = TRUE indique la présence d'au moins une erreur actuellement.

10.7.6 Variables statiques RampFunction

Variable	Type de données	Valeur par défaut	Description
PositiveRisingSlewRate	REAL	10.0	Valeur limite positive pour la vitesse de modification de la valeur de réglage par seconde en cas d'augmentation de la valeur absolue Lorsque PositiveRisingSlewRate = 3.402823e+38, cette limitation de la vitesse de modification est désactivée. Plage de valeurs autorisée : > 0.0
PositiveFallingSlewRate	REAL	10.0	Valeur limite positive pour la vitesse de modification de la valeur de réglage par seconde en cas de diminution de la valeur absolue Lorsque PositiveFallingSlewRate = 3.402823e+38, cette limitation de la vitesse de modification est désactivée. Plage de valeurs autorisée : > 0.0
NegativeRisingSlewRate	REAL	10.0	Valeur limite négative pour la vitesse de modification de la valeur de réglage par seconde en cas d'augmentation de la valeur absolue Lorsque NegativeRisingSlewRate = 3.402823e+38, cette limitation de la vitesse de modification est désactivée Plage de valeurs autorisée : > 0.0
NegativeFallingSlewRate	REAL	10.0	Valeur limite négative pour la vitesse de modification de la valeur de réglage par seconde en cas de diminution de la valeur absolue Lorsque NegativeFallingSlewRate = 3.402823e+38, cette limitation de la vitesse de modification est désactivée. Plage de valeurs autorisée : > 0.0
UpperLimit	REAL	100.0	Limite supérieure de la valeur de réglage Plage de valeurs autorisée : > LowerLimit
LowerLimit	REAL	0.0	Limite inférieure de la valeur de réglage Plage de valeurs autorisée : < UpperLimit
ErrorMode	INT	2	Choix de la valeur de réglage de remplacement en cas d'erreur • 0 = Input • 1 = SubstituteOutput • 2 = dernière valeur de réglage valide • 3 = 0.0 • 4 = LowerLimit • 5 = UpperLimit Plage de valeurs autorisée : 0 à 5
StartMode	INT	2	Choix de la valeur de réglage pour le premier appel de l'instruction • 0 = Input • 1 = SubstituteOutput • 2 = dernière valeur de réglage • 3 = 0.0 • 4 = LowerLimit • 5 = UpperLimit Plage de valeurs autorisée : 0 à 5

Variable	Type de données	Valeur par défaut	Description
CycleTime	AuxFct_CycleTime	-	Données du temps de cycle
CycleTime.Value	REAL	0.1	Intervalle de temps entre deux appels de l'instruction en secondes Plage de valeurs autorisée : > 0.0
CycleTime.EnableMeasurement	BOOL	TRUE	Mesure automatique du temps de cycle • FALSE = désactivée • TRUE = activée

10.7.7 Paramètre ErrorBits

En présence de plusieurs erreurs simultanées, les valeurs des ErrorBits s'affichent comme addition binaire. L'affichage de ErrorBits = 16#0000_0003, par ex., indique la présence simultanée des erreurs 16#0000_0001 et 16#0000_0002.

Avec RampFunction, les erreurs émises au paramètre ErrorBits sont réparties en deux catégories :

- les erreurs déclenchant des messages d'erreur ErrorBits < 16#0001_0000
- les erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000

Erreurs déclenchant des messages d'erreur ErrorBits < 16#0001_0000

En présence d'une ou de plusieurs erreurs déclenchant des messages d'erreur ErrorBits < 16#0001_0000, RampFunction réagit de la manière suivante :

- La valeur de réglage est déterminée comme suit en dépit de cette erreur :
  - Calcul de la valeur de réglage si Reset = FALSE
  - Sortie de SubstituteOutput si Reset = TRUE
- Le paramètre de sortie Error est mis à 1.
- La sortie de validation ENO n'est pas modifiée.

Le paramètre de sortie Error est supprimé dès que l'erreur a disparu.

ErrorBits (DW#16#...)	Description
0000_0000	Pas d'erreur.
0000_0002	**Cause d'erreur :** La mesure du temps de cycle ne donne pas de valeur valide, tandis que le calcul de la valeur de réglage est exécuté (Reset = FALSE). **Réaction en cas d'erreur :** Si une valeur valide du temps de cycle a déjà été mesurée, RampFunction calcule la valeur de réglage à partir de cette dernière valeur de la variable CycleTime.Value. Si aucune valeur valide du temps de cycle n'a été mesurée préalablement, RampFunction continue d'émettre la valeur de réglage configurée avec la variable StartMode au paramètre Output.

Erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000

En présence d'une ou de plusieurs erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000, RampFunction réagit de la manière suivante :

- La valeur de réglage ne peut pas être déterminée comme prévu. À la place, c'est la valeur de réglage de remplacement qui est émise.
- Le paramètre de sortie Error est mis à 1.
- La sortie de validation ENO est réglée sur FALSE.
- La limitation de la valeur de réglage reste opérante tant que les variables LowerLimit et UpperLimit ont des valeurs valides.
- La limitation de la vitesse de modification n'est plus opérante. Dans l'un des scénarios suivants, des sauts peuvent survenir à la valeur de réglage :
  - Lorsqu'une erreur est détectée, RampFunction passe de la valeur de réglage calculée à la valeur de réglage de remplacement. Un saut se produit ou non en fonction de la valeur de la variable ErrorMode.
  - La valeur de réglage de remplacement est modifiée tandis qu'elle est opérante.

Dès que les erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000 ont disparu, RampFunction réagit de la manière suivante :

- La valeur de réglage est déterminée comme suit :
  - Calcul de la valeur de réglage si Reset = FALSE
  - Sortie de SubstituteOutput si Reset = TRUE
- La sortie de validation ENO est réglée sur TRUE.

Le paramètre de sortie Error est supprimé dès que l'erreur a disparu.

<table>
<tr><th>ErrorBits (DW#16#...)</th><th colspan="3">Description</th></tr>
<tr><td>0001_0000</td><td colspan="3">Cause d'erreur :
Le paramètre SubstituteOutput ou une autre variable qui est utilisée comme valeur de réglage n'a pas de valeur REAL valide.
Réaction en cas d'erreur :
La sortie est mise à 0.0 et est limitée par les variables LowerLimit et UpperLimit.
Solution :
Vérifiez que la variable utilisée comme valeur de réglage est une valeur REAL valide (≠ NaN p. ex. 16#7FFF_FFFF). La variable utilisée comme valeur de réglage dépend de Reset et de ErrorMode :
<table>
<tr><th>Reset</th><th>ErrorMode</th><th>Valeur de réglage</th></tr>
<tr><td>FALSE</td><td>0</td><td>Input</td></tr>
<tr><td>FALSE</td><td>1</td><td>SubstituteOutput</td></tr>
<tr><td>FALSE</td><td>4</td><td>LowerLimit</td></tr>
<tr><td>FALSE</td><td>5</td><td>UpperLimit</td></tr>
<tr><td>TRUE</td><td>-</td><td>SubstituteOutput</td></tr>
</table></td></tr>
<tr><td>0002_0000</td><td colspan="3">Cause d'erreur :
Le paramètre Input n'a pas de valeur REAL valide, tandis que le calcul de la valeur de réglage est exécuté (Reset = FALSE).
Réaction en cas d'erreur :
La valeur de réglage de remplacement, qui est configurée au niveau de la variable ErrorMode et qui est limitée par les variables UpperLimit et LowerLimit, est émise au paramètre Output.
0.0 est émis comme valeur de réglage si ErrorMode = 0.
Solution :
Vérifiez que le paramètre Input est une valeur REAL valide (≠ NaN p. ex. 16#7FFF_FFFF).</td></tr>
</table>

ErrorBits (DW#16#...)	Description
0004_0000	**Cause d'erreur :** Le calcul de la valeur de réglage donne une valeur REAL non valide pour le paramètre Output. **Réaction en cas d'erreur :** La valeur de réglage de remplacement, qui est configurée au niveau de la variable ErrorMode et qui est limitée par les variables UpperLimit et LowerLimit, est émise au paramètre Output. **Solution :** Vérifiez toutes les variables utilisées pour le calcul de la valeur de réglage : • Input • PositiveRisingSlewRate • PositiveFallingSlewRate • NegativeRisingSlewRate • NegativeFallingSlewRate • CycleTime.Value Les valeurs de ces variables sont valides. Le calcul de la valeur de réglage dans cette combinaison de variables échoue.
0008_0000	**Cause d'erreur :** La variable LowerLimit ou UpperLimit a une valeur valide. **Réaction en cas d'erreur :** La valeur suivante est émise au paramètre Output en fonction du paramètre Reset : • Reset = FALSE La valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output. • Reset = TRUE La valeur du paramètre SubstituteOutput est émise au paramètre Output. Dans les deux cas, le paramètre Ouput est limité à la plage de valeurs du type de données REAL allant de -3.402823e+38 à 3.402823e+38. **Solution :** Vérifiez que les conditions suivantes sont remplies : 1. LowerLimit < UpperLimit 2. LowerLimit et UpperLimit se situent dans la plage de valeurs admissible -3.402823e+38 à 3.402823e+38 3. LowerLimit et UpperLimit sont des valeurs REAL valides (≠ NaN par ex. 16#7FFF_FFFF)
0010_0000	**Cause d'erreur :** Au moins l'une des variables suivantes n'a pas de valeur valide, tandis que le calcul de la valeur de réglage est exécuté (Reset = FALSE) : 1. PositiveRisingSlewRate 2. PositiveFallingSlewRate 3. NegativeRisingSlewRate 4. NegativeFallingSlewRate **Réaction en cas d'erreur :** La valeur de réglage de remplacement, qui est configurée au niveau de la variable ErrorMode et qui est limitée par les variables UpperLimit et LowerLimit, est émise au paramètre Output. **Solution :** Vérifiez que les conditions suivantes sont remplies pour les quatre variables susmentionnées : • Les valeurs se situent dans la plage de valeurs admissible allant de 0.0 non inclus à 3.402823e+38 • Les valeurs sont des valeurs REAL valides (≠ NaN par ex. 16#7FFF_FFFF)

ErrorBits (DW#16#...)	Description
0020_0000	**Cause d'erreur :** La variable (configurée avec StartMode) pour l'initialisation du paramètre Output lors du premier appel de l'instruction n'a pas de valeur REAL valide. **Réaction en cas d'erreur :** Lors du premier appel de l'instruction, la valeur de réglage de remplacement qui est configurée au niveau de la variable ErrorMode et qui est limitée par les variables LowerLimit et UpperLimit, est émise au paramètre Output. Lors des appels suivants, RampFunction calcule la valeur de réglage à partir de cette valeur de réglage de remplacement. **Solution :** Vérifiez que la variable pour l'initialisation du paramètre Output est une valeur REAL valide (≠ NaN p. ex. 16#7FFF_FFFF). Si Reset = FALSE, l'initialisation n'est active lors du premier appel de l'instruction qu'après le passage de la CPU de l'état de fonctionnement STOP à l'état RUN. La variable utilisée pour l'initialisation du paramètre Output dépend de StartMode : • StartMode = 1: Substitute Output • StartMode = 2: Output
0040_0000	**Cause d'erreur :** La variable CycleTime.Value n'a pas de valeur valide, tandis que le calcul de la valeur de réglage est exécuté (Reset = FALSE). **Réaction en cas d'erreur :** La valeur de réglage de remplacement, qui est configurée au niveau de la variable ErrorMode et qui est limitée par les variables UpperLimit et LowerLimit, est émise au paramètre Output. **Solution :** Vérifiez que les conditions suivantes sont remplies : • 0.0 < CycleTime.Value ≤ 3.402823e+38 • CycleTime.Value est une valeur REAL valide (≠ NaN, par ex. 16#7FFF_FFFF) **Informations complémentaires :** Pour un calcul automatique de la valeur de la variable CycleTime.Value, réglez la variable CycleTime.EnableMeasurement sur TRUE.

10.8 RampSoak

10.8.1 Compatibilité avec CPU et FW

Le tableau suivant montre la compatibilité entre les CPU et les versions de RampSoak:

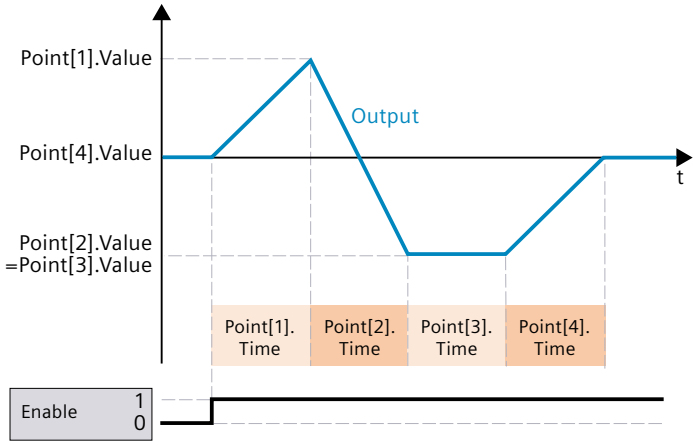
CPU	FW	RampSoak
S7-1200	à partir de V4.2	V1.0
CPU basées sur S7-1500	à partir de V2.0	V1.0

10.8.2 Description RampSoak

Description

Avec l'instruction RampSoak, vous générez une valeur de réglage qui suit un profil programmable dans le temps. Chaque point de ce profil a une valeur cible et une valeur de temps. Lorsque le profil s'exécute, la valeur cible du point actuel est atteinte dans la limite de la valeur de temps.

La vue suivante montre un profil avec 4 points :



L'instruction RampSoak peut être utilisée par exemple pour fournir un profil de consignes en vue de réaliser un régulateur industriel de température Rampe/palier.

Calcul de la valeur de réglage

La valeur de réglage est calculée par interpolation à partir du point actuel et du point précédent selon la formule suivante :

$$\text{Output}(t) = \text{Point}[i].\text{Value} - \frac{\text{RemainingTime\_Point}}{\text{Point}[i].\text{Time}} (\text{Point}[i].\text{Value} - \text{Point}[i-1].\text{Value})$$

i représente la valeur du paramètre CurrentPoint.

Le temps écoulé permet de déterminer les points du profil actuellement utilisés pour le calcul de la valeur de réglage.

Appel

L'instruction RampSoak est appelée comme DB d'instance unique dans un OB ou une FC. Lorsqu'elle est appelée dans un FB, l'instruction RampSoak peut être appelée comme DB d'instance unique ou comme DB multi-instance et DB d'instance paramètres.

Aucun objet technologique n'est créé lors de l'appel de l'instruction. Vous ne disposez pas d'une interface de paramétrage et de mise en service. Vous paramétrez directement l'instruction RampSoak via le DB d'instance et la mettez en service RampSoak via une table de visualisation du programme utilisateur dans la CPU ou l'IHM.

Démarrage

Les variables dans la plage statique de RampSoak et donc aussi les données de profil dans les structures UserData et WorkingData ne sont pas rémanentes. Ces variables sont initialisées à l'aide des valeurs initiales après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN.

Si vous modifiez les valeurs actuelles dans la structure UserData en mode en ligne et que ces valeurs doivent être conservées après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN, sauvegardez ces valeurs dans les valeurs initiales du bloc de données.

Lors du premier appel de l'instruction RampSoak après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN, la validité des données est automatiquement contrôlée dans la structure UserData. Si le contrôle est positif, les données sont transmises à la structure WorkingData.

Vous pouvez utiliser la variable StartMode (Page 513) pour spécifier le comportement de démarrage de l'instruction RampSoak lors du premier appel après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN.

Comportement en cas d'erreur

Si la valeur de réglage ne peut pas être calculée correctement, l'instruction RampSoak renvoie une valeur de réglage de remplacement et une erreur avec un message d'erreur ErrorBits >= 16#0002_0000 à la place. Vous pouvez utiliser la variable ErrorMode (Page 523) pour définir la valeur de réglage de remplacement comme suit :

Error-Mode	Output
0	Valeur de la variable WorkingData.StartValue Assurez-vous que les données de profil dans UserData ont été vérifiées avec le paramètre Validate et appliquées dans les WorkingData. La valeur par défaut 0.0 de WorkingData.StartValue est utilisée si les données de profil n'ont pas encore été vérifiées et appliquées.
1	Valeur du paramètre SubstituteOutput
2	La dernière valeur de réglage valide de l'exécution du profil 0.0, s'il n'y a pas de valeur de réglage valide de l'exécution du profil.
3	0.0

Les points suivants s'appliquent en plus pour toutes les valeurs de la variable ErrorMode :

- Si la valeur de réglage de remplacement n'est pas une valeur REAL valide, 0.0 est sorti comme valeur de réglage.
- La valeur de réglage de remplacement est limitée à la plage de valeurs -3,402823e+38 .. +3,402823e+38 du type de données REAL. Ce n'est qu'alors que la valeur de réglage de remplacement est sortie au paramètre Output.
- La variable ErrorMode n'est active que lorsque le paramètre Reset = FALSE. Si le paramètre Reset = TRUE est défini, la valeur du paramètre SubstituteOutput ou 0.0 est sortie au paramètre Output.

Le paramètre Error indique si une erreur est actuellement présente. Quand l'erreur a disparu, Error est réglé sur FALSE. Le paramètre ErrorBits indique les erreurs qui se sont produites. ErrorBits est rémanent et n'est réinitialisé que par un front montant au paramètre Reset ou ErrorAck.

10.8.3 Fonctionnement RampSoak

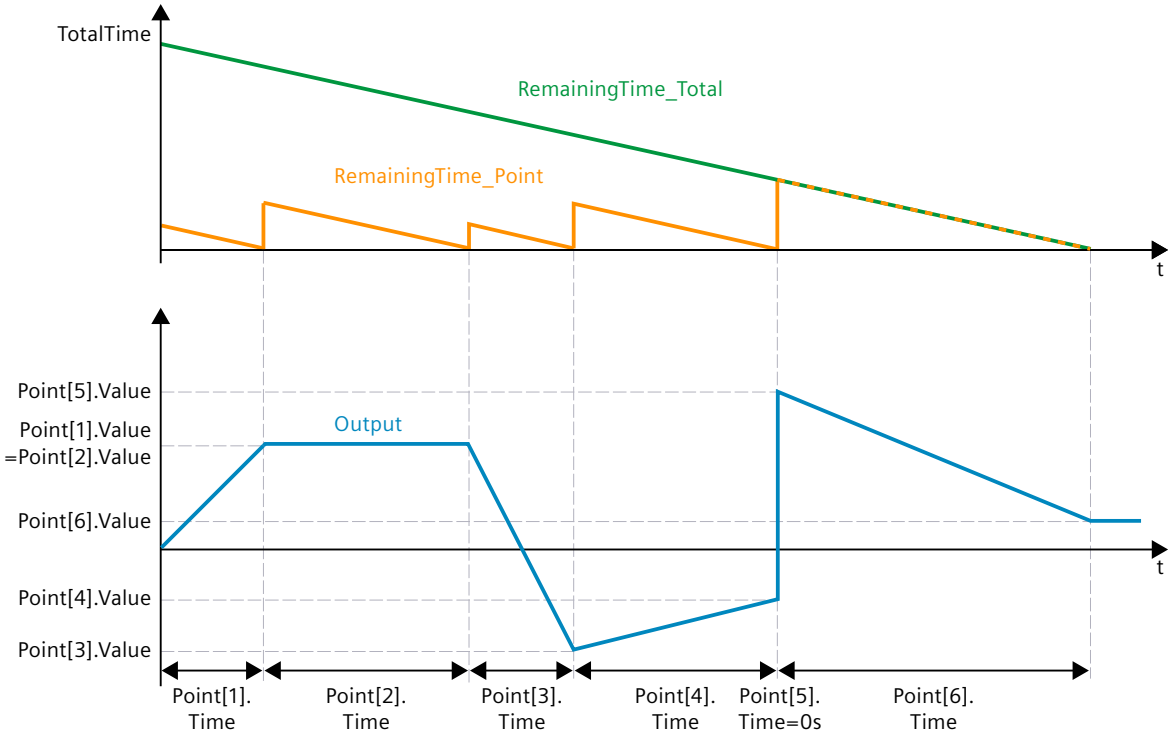
10.8.3.1 Configurer et vérifier les données de profil

Configurer les données de profil

Vous configurez le profil dans la structure statique UserData.
Celle-ci contient les éléments suivants :

- NumberOfUsedPoints
  Nombre de points d'interpolation utilisés pour le profil.
- StartValue
  Valeur de réglage optionnelle lorsque l'exécution du profil est démarrée (Page 513), arrêtée (Page 516) ou terminée.
- Point
  La matrice avec 50 éléments contient des paires de valeurs des points d'interpolation Point[i].Value et Point[i].Time :
  - Point[i].Value
    La valeur de réglage est modifiée progressivement jusqu'à cette valeur tant que ce point est actif et atteint cette valeur dans le temps Point[i].Time.
  - Point[i].Time
    Cette valeur définit la durée du point en secondes.

La figure suivante montre un profil avec 6 points. Le point numéro 5 a une durée de 0 seconde, ce qui entraîne un échelon de la valeur de réglage.



Vérifier les données de profil

Avant que les données de profil ne soient utilisées pour l'exécution du profil, une vérification des données de profil dans la structure UserData est requise.

Le contrôle est déclenché comme suit :

- Lorsque vous réglez le paramètre Validate sur TRUE.
- Si vous mettez pour la première fois le paramètre Enable sur TRUE après le passage de la CPU de l'état de fonctionnement STOP à l'état RUN et que les données de profil n'ont pas encore été vérifiées.

Les conditions suivantes sont vérifiées pour les données de profil dans la structure UserData :

- 1 ≤ UserData.NumberOfUsedPoints ≤ 50
- NextPoint ≤ UserData.NumberOfUsedPoints
- Lorsque l'exécution du profil est activée (Enable = TRUE) :
  - CurrentPoint ≤ UserData.NumberOfUsedPoints
- -3.402823e+38 ≤ UserData.StartValue ≤ 3.402823e+38
- -3.402823e+38 ≤ UserData.Point[i].Value ≤ 3.402823e+38 avec l'indice *i* = 1..UserData.NumberOfUsedPoints
- 0.0 ≤ UserData.Point[i].Time ≤ 3.402823e+38 avec l'indice *i* = 1..UserData.NumberOfUsedPoints
- 0.0 < UserData.Point[1].Time + UserData.Point[2].Time +...+ UserData.Point[UserData.NumberOfUsedPoints].Time ≤ 3.402823e+38

Effets de la vérification des données de profil et messages d'erreur possibles

Si les conditions sont remplies, les nouvelles données de profil dans la structure UserData sont transférées dans la structure WorkingData et utilisées pour l'exécution du profil. Le paramètre TotalTime est actualisé. Le paramètre RemainingTime_Total est actualisé, lors de l'exécution du profil.

REMARQUE

La valeur et la durée du point actuel sont mises en cache et restent inchangées même après une vérification réussie des nouvelles données de profil jusqu'à ce que le point actuel soit terminé. Les nouvelles données de profil seront utilisées à partir du point suivant.

Si l'une des conditions n'est pas remplie, les données de profil précédentes de la structure WorkingData restent inchangées. Une erreur avec le message d'erreur ErrorBits = 16#0000_0004 (Page 523) est présente.

Si le paramètre Enable ou le paramètre Next est défini sur TRUE mais qu'il n'y a pas de données de profil valides dans la structure WorkingData, une erreur sera présente avec le message d'erreur ErrorBits = 16#0008_0000 (Page 523).

REMARQUE

Si vous ne modifiez pas les valeurs par défaut dans la structure UserData avant le contrôle, le contrôle des données de profil échoue.

REMARQUE

Vous ne pouvez pas modifier les valeurs hors ligne de la structure WorkingData. Si vous souhaitez modifier les données de profil, éditez d'abord la structure UserData et définissez ensuite le paramètre Validate = TRUE.

10.8.3.2 Exécuter le profil

Pour exécuter un profil et calculer la valeur de réglage, vous avez besoin de données de profil vérifiées et d'un front montant sur le paramètre Enable.

Démarrer et arrêter l'exécution du profil - paramètre Enable

Avec un front montant sur le paramètre Enable, vous démarrez l'exécution du profil et le calcul de la valeur de réglage. L'exécution du profil commence à partir du point spécifié dans le paramètre d'entrée/sortie NextPoint.

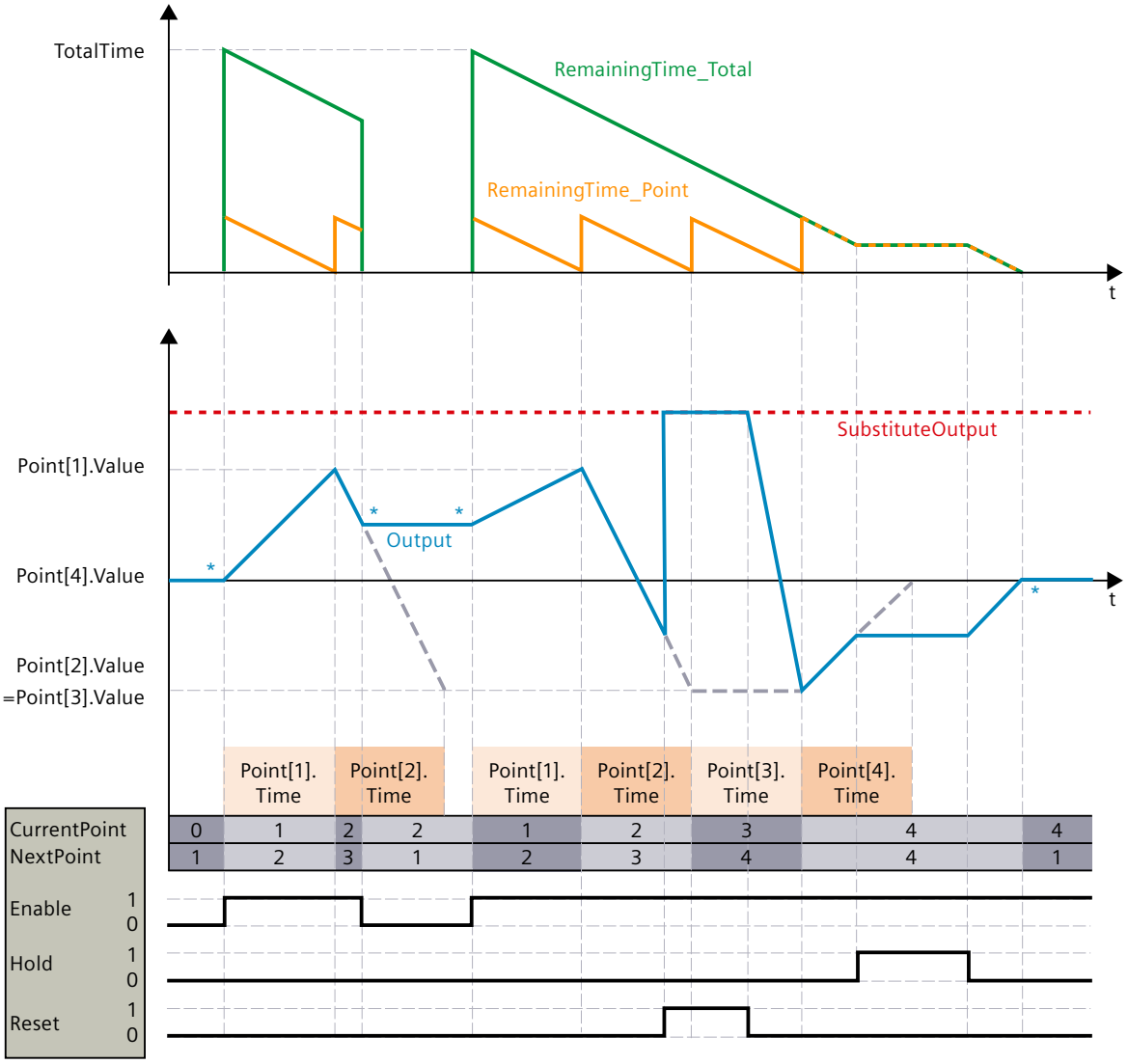
Vous arrêtez l'exécution du profil avec un front négatif sur le paramètre Enable.

Si le profil est arrêté ou entièrement exécuté, le paramètre d'entrée/sortie NextPoint est automatiquement remis à 1. L'exécution du profil suivant démarre donc à partir du premier point du profil si le paramètre d'entrée/sortie NextPoint n'est pas modifié.

Vous pouvez configurer le comportement lors du démarrage, de l'arrêt et de la fin de l'exécution du profil à l'aide des variables statiques StartMode (Page 513) et StopMode (Page 516).

L'exécution du profil ne détermine que la valeur de réglage sur le paramètre Output, si Reset = FALSE. Si RESET = TRUE est défini, une exécution du profil activée se poursuit en arrière-plan.

La figure suivante montre comment les paramètres d'entrée Enable, Reset et Hold déterminent l'exécution du profil :



Dans cet exemple, le comportement de démarrage et d'arrêt de l'exécution du profil (marqué d'un *) avec la valeur par défaut StartMode = StopMode = 2 (démarrer depuis la dernière valeur de réglage / conserver la dernière valeur de réglage) est affiché.

Arrêter l'exécution du profil - paramètre Hold

Avec le paramètre Hold = TRUE, vous arrêtez l'exécution du profil activée.
Le paramètre Hold = TRUE a l'influence suivante :

- Les paramètres Output, CurrentPoint, NextPoint, RemainingTime_Total et RemainingTime_Point restent inchangés.
- Un front aux paramètres Enable ou Next est retardé et n'est effectif que si Hold = FALSE.
- Si le paramètre Reset passe de TRUE à FALSE, le paramètre Hold influence les modifications apportées au paramètre Output. Pour plus d'informations, référez-vous à la description du paramètre Reset.

Le paramètre Hold n'a aucune influence sur le traitement des paramètres Validate et ErrorAck.

Écraser la valeur de réglage - paramètre Reset

Avec le paramètre Reset = TRUE, vous réinitialisez l'instruction et écrasez la valeur de réglage du paramètre Output avec la valeur du paramètre SubstituteOutput.
Une exécution du profil activée continue en arrière-plan. Les paramètres Enable, Hold et Next peuvent toujours être utilisés.
Si l'exécution du profil est encore activée (Enable=TRUE et le profil pas encore complètement exécuté) et que le paramètre Reset passe de TRUE à FALSE, le comportement dépend du paramètre Hold comme suit :

- Hold = FALSE
  Le paramètre Output passe progressivement de SubstituteOutput à la valeur du point actuel et l'atteint pendant le temps restant du point actuel (RemainingTime_Point).
- Hold = TRUE
  Le paramètre Output passe de SubstituteOutput à la valeur de l'exécution du profil arrêtée.

Si l'exécution du profil est désactivée (Enable=FALSE ou profil complètement exécuté) et que le paramètre Reset passe de TRUE à FALSE, le paramètre Output passe à la dernière valeur de l'exécution du profil.

Poursuivre l'exécution du profil avec un point spécifique - paramètre Next

Vous continuez l'exécution du profil avec un front montant sur le paramètre Next avec Point[NextPoint].
Si l'exécution du profil est désactivée, le paramètre Next = TRUE a l'effet suivant :

- Le paramètre Output prend la valeur de Point[NextPoint].Value.
- Le paramètre CurrentPoint prend la valeur de NextPoint.
- Le paramètre NextPoint est incrémenté. Si NextPoint est le dernier point du profil, alors NextPoint prend la valeur 1.

Si l'exécution du profil est activée, le paramètre Next = TRUE a l'effet suivant :

- Le paramètre Output passe progressivement de sa valeur actuelle à la valeur de Point[NextPoint].Value et l'atteint pendant Point[NextPoint].Time.
- Le paramètre RemainingTime_Point prend la valeur de Point[NextPoint].Time.
- Le paramètre RemainingTime_Total est actualisé en conséquence.

- Le paramètre CurrentPoint prend la valeur de NextPoint.
- Le paramètre NextPoint est incrémenté. Si NextPoint est le dernier point du profil, alors la nouvelle valeur dépend de StopMode. Pour plus d'informations, référez-vous à la description du paramètre NextPoint.

REMARQUE

Lors du premier appel de l'instruction RampSoak après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN, le comportement est déterminé par StartMode. Le paramètre Next n'est effectif que pour les appels suivants.

Vous trouverez un exemple de profil avec le paramètre Next dans le paragraphe suivant "Poursuivre l'exécution du profil avec un point spécifique - paramètre NextPoint".

Poursuivre l'exécution du profil avec un point spécifique - paramètre NextPoint

Le paramètre NextPoint détermine avec quel point l'exécution du profil se poursuit. Lorsque le point actuel est terminé, l'instruction RampSoak actualise automatiquement la valeur du paramètre NextPoint au point suivant du profil.
En modifiant la valeur du paramètre NextPoint, vous pouvez déterminer que l'exécution du profil doit continuer avec un point différent.
Le paramètre NextPoint peut affecter le comportement de l'exécution du profil lorsqu'elle est désactivée comme suit :

- Avec un front montant au paramètre Next, le paramètre Output prend la valeur de Point[NextPoint].Value.
- Lorsque l'exécution du profil démarre, NextPoint représente le premier CurrentPoint. Le paramètre Output passe alors progressivement de la valeur définie par StartMode à la valeur de Point[NextPoint].Value et l'atteint pendant Point[NextPoint].Time.

Le paramètre NextPoint peut affecter le comportement de l'exécution du profil lorsqu'elle est activée comme suit :

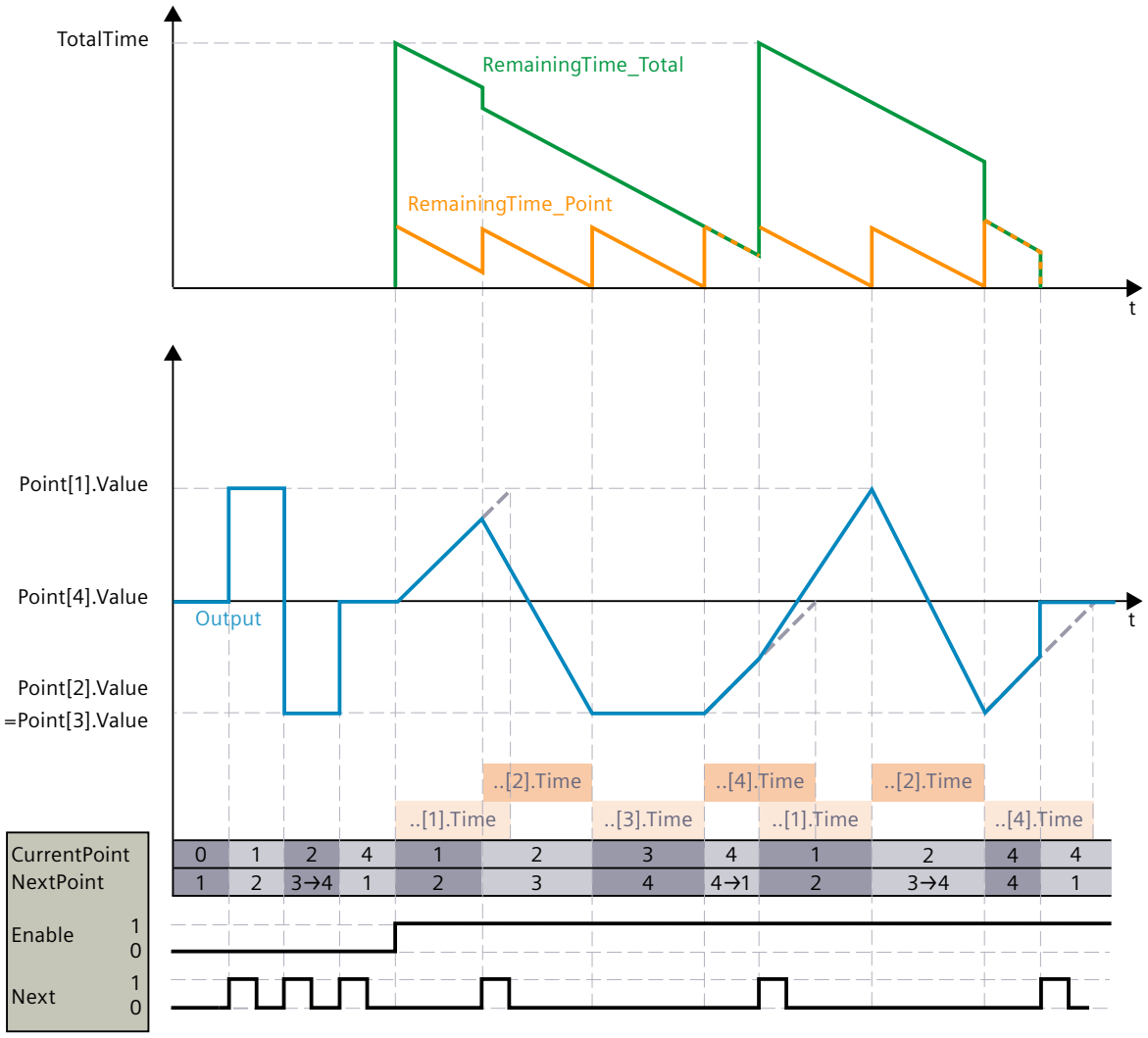
- Sans front montant au paramètre Next :
  Lorsque le point actuel est atteint, le régulateur passe au NextPoint.
- Avec un front montant au paramètre Next :
  Le régulateur passe immédiatement au NextPoint.

Si le dernier point d'exécution du profil est activé (CurrentPoint = WorkingData.NumberOfUsedPoints), StopMode détermine comment la valeur de NextPoint est automatiquement actualisée comme suit :

- StopMode <> 4
  Pendant que le dernier point est activé : NextPoint = CurrentPoint = WorkingData.NumberOfUsedPoints.
  L'exécution du profil se termine après le dernier point.
- StopMode = 4
  Pendant que le dernier point est activé : NextPoint = 1.
  L'exécution du profil redémarre après le dernier point.

Si le profil est arrêté ou complètement exécuté, le paramètre d'entrée/sortie NextPoint est automatiquement remis à 1. L'exécution du profil suivant démarre donc à partir du premier point du profil si le paramètre d'entrée/sortie NextPoint n'est pas modifié.

La figure suivante montre comment les paramètres Next et NextPoint déterminent l'exécution du profil. Si la modification automatique de NextPoint est ensuite écrasée par la saisie de l'utilisateur, cela est indiqué par → :



Dans cet exemple, le comportement d'arrêt de l'exécution du profil avec StopMode <> 4.

10.8.3.3 Configurer le comportement de démarrage - variable statique StartMode

Vous pouvez utiliser la variable StartMode pour déterminer le comportement de l'instruction RampSoak dans les cas suivants :

- Lorsque l'instruction RampSoak est exécutée pour la première fois après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN (redémarrage).
- L'exécution du profil démarre.
- L'exécution de "Charger les valeurs initiales comme valeurs actuelles" (uniquement pour l'option "Toutes les valeurs", et pas pour l'option "Valeurs de réglage uniquement")

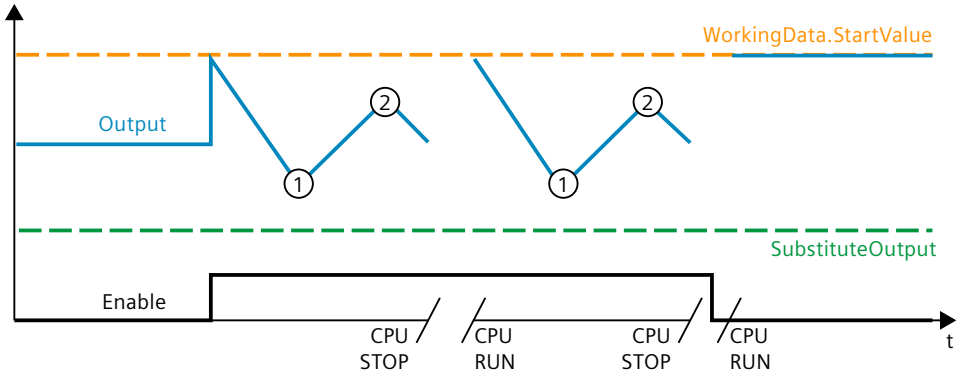
Vous avez le choix entre les options suivantes de la variable StartMode :

- StartMode = 0

  Le paramètre Output prend la valeur de WorkingData.StartValue.

  L'exécution du profil démarre à partir de cette valeur si Enable = TRUE est défini après un redémarrage de la CPU ou si Enable passe de FALSE à TRUE.

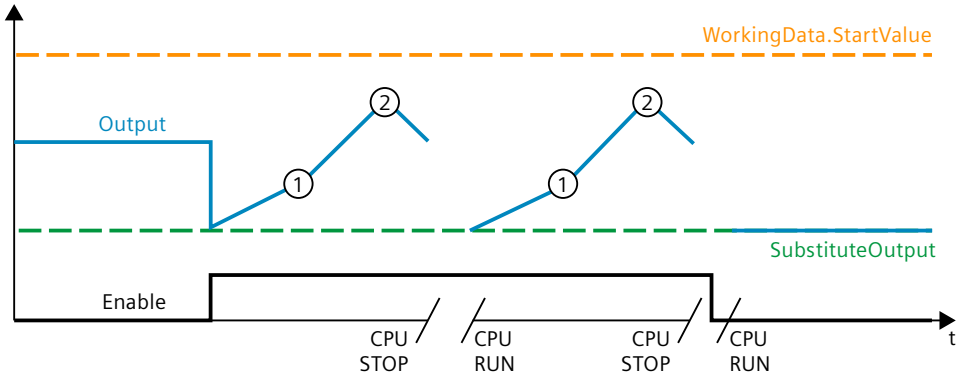
  La figure suivante montre le démarrage de l'exécution du profil et le redémarrage de la CPU avec StartMode = 0 :



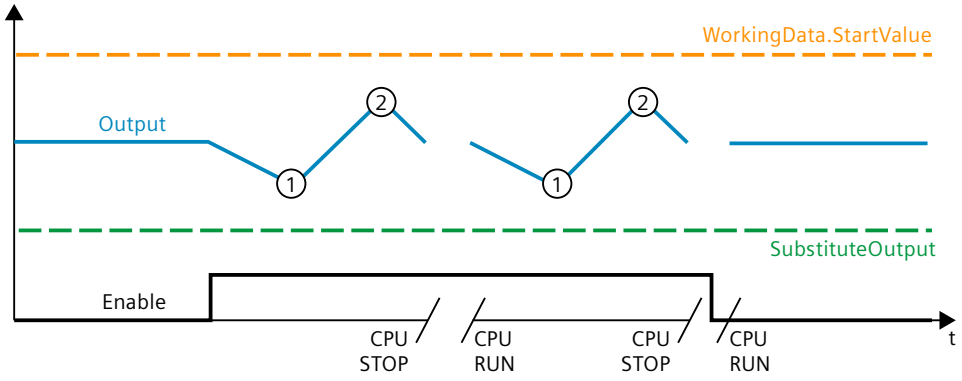
REMARQUE

Avec ce réglage, les données de profil doivent être vérifiées lors du premier appel de l'instruction après le redémarrage de la CPU et appliquées dans les WorkingData. Sinon, la valeur par défaut 0.0 de WorkingData.StartValue est utilisée.

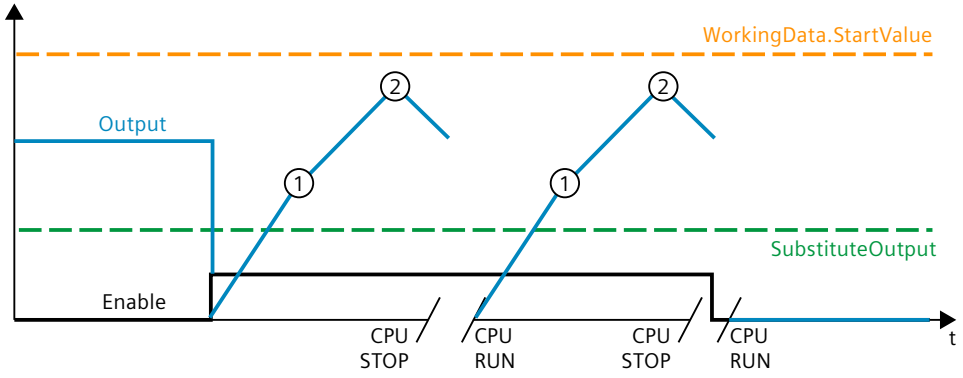
- StartMode = 1
  Le paramètre Output prend la valeur de SubstituteOutput.
  L'exécution du profil démarre à partir de cette valeur si Enable = TRUE est défini après un redémarrage de la CPU ou si Enable passe de FALSE à TRUE.
  La figure suivante montre le démarrage de l'exécution du profil et le redémarrage de la CPU avec StartMode = 1 :



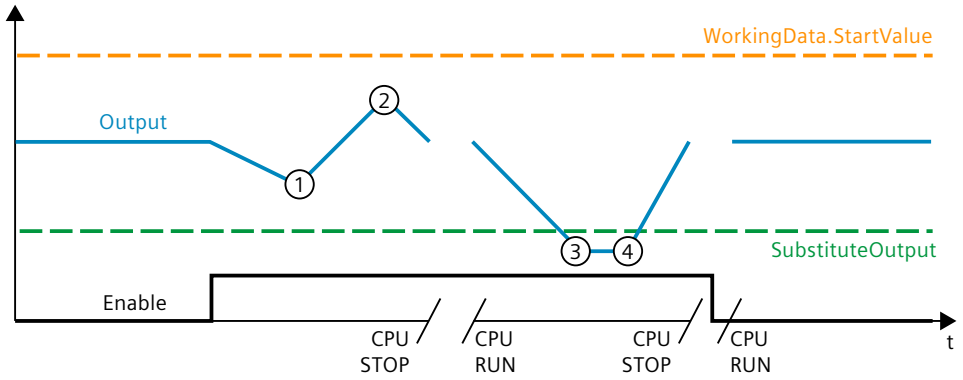
- StartMode = 2
  Le paramètre Output reste inchangé.
  L'exécution du profil démarre à partir de la valeur inchangée du paramètre Output, si Enable = TRUE est défini après un redémarrage de la CPU ou si Enable passe de FALSE à TRUE.
  La figure suivante montre le démarrage de l'exécution du profil et le redémarrage de la CPU avec StartMode = 2 :



- StartMode = 3
  Le paramètre Output prend la valeur 0.0.
  L'exécution du profil démarre à partir de cette valeur si Enable = TRUE est défini après un redémarrage de la CPU ou si Enable passe de FALSE à TRUE.
  La figure suivante montre le démarrage de l'exécution du profil et le redémarrage de la CPU avec StartMode = 3 :



- StartMode = 4
  Le paramètre Output reste inchangé.
  L'exécution du profil se poursuit si Enable = TRUE reste défini après le redémarrage de la CPU et si l'exécution du profil a été activée avant le redémarrage.
  L'exécution du profil démarre à partir de la valeur inchangée du paramètre Output, si Enable = TRUE est défini après un redémarrage de la CPU et l'exécution du profil était désactivé avant le redémarrage ou si Enable passe de FALSE à TRUE.
  La figure suivante montre le démarrage de l'exécution du profil et le redémarrage de la CPU avec StartMode = 4 :



Les points suivants s'appliquent en plus pour toutes les valeurs de la variable StartMode :

- Le paramètre Enable, la variable StartMode et les données de profil des structures UserData et WorkingData ne sont pas rémanentes. Ces variables sont initialisées avec les valeurs initiales après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN. Vérifiez que ces variables ont des valeurs appropriées lors du premier appel de l'instruction RampSoak après le passage de la CPU de l'état de fonctionnement STOP à l'état RUN, afin d'atteindre le comportement souhaité.
- La valeur sélectionnée par StartMode est limitée à la plage de valeurs du type de données REAL. Ce n'est qu'alors que la valeur est sortie au paramètre Output.

- Si la valeur sélectionnée par StartMode n'est pas une valeur REAL valide, la valeur de réglage de remplacement est sortie au paramètre Output. La valeur de réglage de remplacement est configurée par la variable ErrorMode et est limitée à la plage de valeurs du type de données REAL. Si l'exécution du profil est activée, elle démarre à partir de cette valeur de réglage de remplacement.
- La variable StartMode n'affecte le paramètre Output que si le paramètre Reset = FALSE est défini et qu'aucune erreur n'apparaît avec le message d'erreur ErrorBits ≥ 16#0002_0000 en même temps. Si le paramètre Reset = TRUE, la valeur du paramètre SubstituteOutput est sortie au paramètre Output. En présence d'une erreur avec message d'erreur ErrorBits ≥ 16#0002_0000, la valeur de réglage de remplacement qui est configurée dans la variable ErrorMode est sortie au paramètre Output.

10.8.3.4 Configurer le comportement d'arrêt - variable statique StopMode

Vous pouvez utiliser la variable StopMode pour déterminer le comportement de l'instruction RampSoak dans les cas suivants :

- L'exécution du profil se termine lorsque le dernier nœud est atteint.
- L'exécution du profil est arrêtée en réinitialisant Enable.

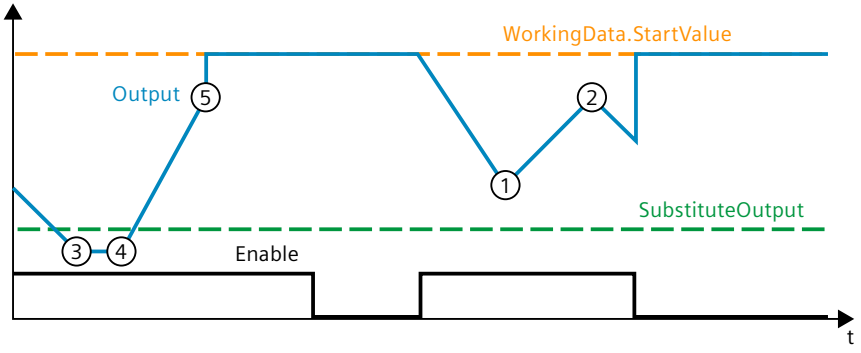
La valeur définie par StopMode est utilisée comme valeur de réglage jusqu'à une nouvelle action, par exemple le début de l'exécution du profil est déclenché.

Vous avez le choix entre les options suivantes de la variable StopMode :

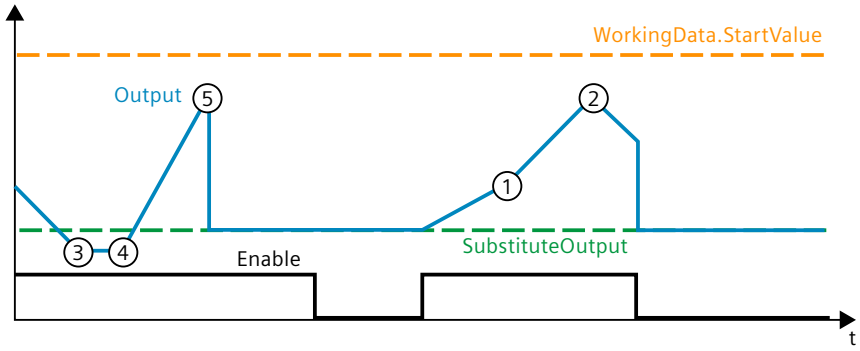
- StopMode = 0

  Le paramètre Output prend la valeur de WorkingData.StartValue.

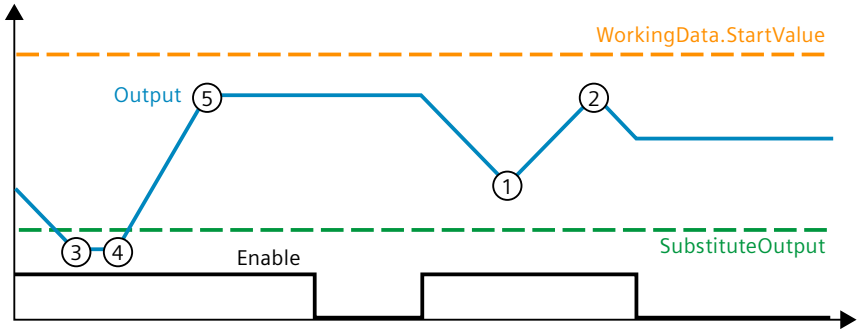
  La vue suivante montre comment l'exécution d'un profil à 5 nœuds avec StopMode = 0 est terminée et arrêtée :



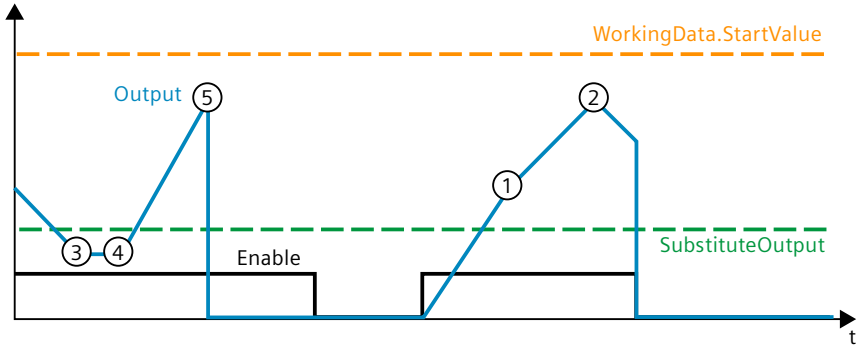
- StopMode = 1
  Le paramètre Output prend la valeur de SubstituteOutput.
  La vue suivante montre comment l'exécution d'un profil à 5 nœuds avec StopMode = 1 est terminée et arrêtée :



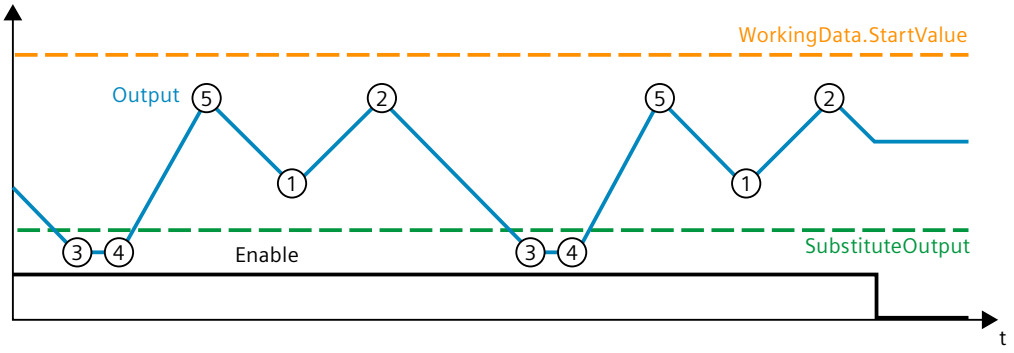
- StopMode = 2
  Le paramètre Output prend la dernière valeur valide de l'exécution du profil.
  La vue suivante montre comment l'exécution d'un profil à 5 nœuds avec StopMode = 2 est terminée et arrêtée :



- StopMode = 3
  Le paramètre Output prend la valeur 0.0.
  La vue suivante montre comment l'exécution d'un profil à 5 nœuds avec StopMode = 3 est terminée et arrêtée :



- StopMode = 4
  Lorsque l'exécution du profil est terminée avec le dernier nœud, elle redémarre automatiquement et poursuit avec WorkingData.Point[1]. Tant que Enable n'est pas réinitialisé, l'exécution du profil est répétée.
  Si l'exécution du profil est arrêtée avec Enable = FALSE, alors le paramètre Output prend la dernière valeur valide de l'exécution du profil.
  La vue suivante montre comment l'exécution d'un profil à 5 nœuds avec StopMode = 4 est terminée et arrêtée :



Les points suivants s'appliquent en plus pour toutes les valeurs de la variable StopMode :

- La valeur sélectionnée par StopMode est limitée à la plage de valeurs du type de données REAL. Ce n'est qu'ensuite que celle-ci est émise au paramètre Output.
- Si la valeur sélectionnée par StopMode n'est pas une valeur REAL valide, la valeur de réglage de remplacement est émise au paramètre Output, puis maintenue. La valeur de réglage de remplacement est configurée par la variable ErrorMode et est limitée à la plage de valeurs du type de données REAL.
- La variable StopMode n'affecte le paramètre Output que si le paramètre Reset = FALSE est défini et qu'aucune erreur n'apparaît avec le message d'erreur ErrorBits ≥ 16#0002_0000 en même temps. Lorsque le paramètre Reset = TRUE, la valeur du paramètre SubstituteOutput est émise au paramètre Output. En présence d'une erreur déclenchant un message d'erreur ErrorBits ≥ 16#0002_0000, la valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output.

10.8.3.5 Mesure du temps de cycle

Mesure automatique du temps de cycle

Pour le calcul de la valeur de réglage, RampSoak a besoin du temps qui s'est écoulé depuis le dernier appel de RampSoak.

Le temps de cycle est automatiquement mesuré par défaut et est émis au niveau de la variable CycleTime.Value à partir du deuxième appel. RampSoak mesure le temps de cycle à chaque appel de l'instruction et est donc utilisable dans des cycles d'appel non équidistants, par exemple dans l'OB1.

Notez que des appels conditionnels de l'instruction, les points d'arrêt actifs ou le chargement d'instantanés comme valeurs en cours rallongent le temps de cycle en cas de mesure automatique du temps de cycle.

Si la mesure du temps de cycle ne fournit aucun résultat valide, RampSoak calcule la valeur de réglage actuelle à l'aide du dernier temps de cycle valide. De plus, RampSoak émet un message d'erreur au paramètre ErrorBits.

Si vous désactivez la mesure automatique du temps de cycle en définissant la variable CycleTime.EnableMeasurement = FALSE, vous devez spécifier le temps de cycle manuellement au niveau de la variable CycleTime.Value. RampSoak vérifie la validité de la variable CycleTime.Value à chaque appel.

Mesure automatique du temps de cycle avec points d'arrêt

Si des points d'arrêt sont actifs entre deux appels de RampSoak, la mesure automatique du temps de cycle donne le temps réel qui s'est écoulé entre deux appels. Si un point d'arrêt est actif, la CPU se trouve à l'état de fonctionnement ATTENTE.

REMARQUE

Les points d'arrêt actifs allongent l'intervalle de temps entre deux appels de RampSoak.

Avec un intervalle de temps plus long entre deux appels, la modification de la valeur de réglage augmente également au paramètre Output. En fonction du temps écoulé et des données de profil configurées, des nœuds peuvent être ignorés.

Si vous n'avez pas besoin du calcul de la valeur de réglage sur la base du temps réel avec des points d'arrêt actifs, exécutez alors les étapes suivantes :

- Désactivez la mesure automatique du temps de cycle en définissant la variable CycleTime.EnableMeasurement = FALSE.
- Saisissez le temps de cycle manuellement au niveau de la variable CycleTime.Value.

10.8.3.6 Comportement de validation EN/ENO

Si l'une des conditions suivantes est remplie, la sortie de validation ENO prend la valeur FALSE :

- L'entrée de validation EN prend la valeur TRUE et une erreur est signalée par le message d'erreur ErrorBits ≥ 16#0001_0000 .
- L'entrée de validation EN prend la valeur FALSE.

Dans tous les autres cas, la sortie de validation ENO prend la valeur TRUE.

10.8.4 Paramètre d'entrée RampSoak

Paramètre	Type de données	Valeur par défaut	Description
Enable	BOOL	FALSE	L'exécution du profil et le calcul de la valeur de réglage sont exécutés sur front montant du paramètre Enable. L'exécution du profil est arrêtée sur front descendant du paramètre Enable.
Hold	BOOL	FALSE	Si Hold a la valeur TRUE, l'exécution du profil est interrompue. La valeur de réglage reste constante.
Next	BOOL	FALSE	L'exécution du profil se poursuit au Point[NextPoint] sur front montant du paramètre Next,
SubstituteOutput	REAL	0.0	SubstituteOutput est utilisé comme valeur de réglage de remplacement si Reset = TRUE ou l'un des modes suivants est actuellement en vigueur : • ErrorMode = 1 • StartMode = 1 • StopMode = 1
Validate	BOOL	FALSE	Si Validate prend la valeur TRUE, la validité des données de profil dans les UserData est contrôlée et ces données sont appliquées dans WorkingData.
ErrorAck	BOOL	FALSE	Supprime les messages d'erreur. • Le front FALSE -> TRUE ErrorBits est remis à 0.
Reset	BOOL	FALSE	Réinitialise l'instruction. • Le front FALSE -> TRUE ErrorBits est remis à 0. • Tant que Reset a la valeur TRUE, la valeur de réglage de remplacement SubstituteOutput est fournie à la sortie. Une exécution simultanée éventuelle du profil se poursuivra en arrière-plan. • Tant que Reset a la valeur FALSE, la valeur de réglage est définie par l'exécution du profil.

10.8.5 Paramètre de sortie RampSoak

Paramètre	Type de données	Valeur par défaut	Description
Output	REAL	0.0	Valeur de réglage La valeur de réglage est rémanente.
CurrentPoint	INT	0	Numéro du point actuellement utilisé pour exécuter le profil et calculer la valeur de réglage.
TotalTime	REAL	0.0	Temps total du profil (somme des temps de tous les points utilisés) en secondes
RemainingTime_Total	REAL	0.0	Temps restant du profil en secondes
RemainingTime_Point	REAL	0.0	Temps restant du point actuel en secondes
ErrorBits	DWORD	16#0	Le paramètre ErrorBits (Page 523) signale quels sont les messages d'erreur présents. ErrorBits est rémanent et réinitialisé à Reset ou ErrorAck en cas de front montant.
Error	BOOL	FALSE	Le réglage de Error sur TRUE indique la présence d'au moins une erreur actuellement.

10.8.6 Paramètre d'entrée/sortie RampSoak

Paramètre	Type de données	Valeur par défaut	Description
NextPoint	Int	1	Numéro du point qui sera utilisé ensuite. Plage de valeurs autorisée : 1 à WorkingData.NumberOfUsedPoints

10.8.7 Variables statiques RampSoak

Variable	Type de données	Valeur par défaut	Description
UserData	AuxFct_RampSoak_Profile	-	Les données de profil sont saisies dans la structure UserData. Les données de profil dans la structure UserData peuvent être éditées. Les modifications de cette structure ne se répercutent sur l'exécution du profil qu'une fois que le contrôle et la copie des données dans la structure WorkingData ont été lancés.
UserData.NumberOfUsedPoints	INT	0	Nombre de points d'interpolation utilisés du profil Plage de valeurs autorisée : 1 à 50
UserData.StartValue	REAL	0.0	StartValue est utilisé comme valeur de réglage optionnelle si l'un des modes suivants est actuellement en vigueur : • ErrorMode = 0 • StartMode = 0 • StopMode = 0
UserData.Point	Array[1..50] of AuxFct_RampSoak_Point	-	Points d'interpolation du profil
UserData.Point[i].Value	REAL	0.0	Valeur de réglage de ce point
UserData.Point[i].Time	REAL	0.0	Durée de ce point en secondes Plage de valeurs admissible : Point[i].Time ≥ 0.0
WorkingData	AuxFct_RampSoak_Profile	-	Les données de profil actuellement en vigueur sont affichées dans la structure WorkingData. Les données de profil dans la structure WorkingData ne peuvent pas être éditées.
WorkingData.NumberOfUsedPoints	INT	0	Nombre de points d'interpolation utilisés du profil Plage de valeurs autorisée : 1 à 50
WorkingData.StartValue	REAL	0.0	WorkingData.StartValue est utilisé comme valeur de réglage optionnelle si l'un des modes suivants est actuellement en vigueur : • ErrorMode = 0 • StartMode = 0 • StopMode = 0
WorkingData.Point	Array[1..50] of AuxFct_RampSoak_Point	-	Points d'interpolation du profil
WorkingData.Point[i].Value	REAL	0.0	Valeur de réglage de ce point
WorkingData.Point[i].Time	REAL	0.0	Durée de ce point en secondes Plage de valeurs admissible : Point[i].Time ≥ 0.0

Variable	Type de données	Valeur par défaut	Description
ErrorMode	INT	2	Sélection de la valeur de réglage de remplacement en cas d'erreur • 0 = WorkingData.StartValue • 1 = SubstituteOutput • 2 = dernière valeur de réglage valide de l'exécution du profil • 3 = 0.0 Plage de valeurs admissible : 0 à 3 Si la valeur de ErrorMode ne correspond pas à la plage de valeurs admissible, alors ErrorMode = 2.
StartMode	INT	2	Choix du comportement de démarrage • 0 = WorkingData.StartValue • 1 = SubstituteOutput • 2 = démarrer à partir de la dernière valeur de réglage • 3 = 0.0 • 4 = continuer à partir de la dernière valeur de réglage Plage de valeurs admissible : 0 à 4 Si la valeur de StartMode ne correspond pas à la plage de valeurs admissible, alors StartMode = 2.
StopMode		2	Choix du comportement d'arrêt • 0 = WorkingData.StartValue • 1 = SubstituteOutput • 2 = dernière valeur de réglage valide de l'exécution du profil • 3 = 0.0 • 4 = mode cyclique Plage de valeurs admissible : 0 à 4 Si la valeur de StopMode ne correspond pas à la plage de valeurs admissible, alors StopMode = 2.
CycleTime	AuxFct_CycleTime	-	Données du temps de cycle
CycleTime.Value	REAL	0.1	Temps de cycle en secondes (intervalle entre 2 appels) Plage de valeurs admissible : CycleTime.Value > 0.0
CycleTime.EnableMeasurement	BOOL	TRUE	Mesure automatique du temps de cycle • FALSE = désactivée • TRUE = activée

10.8.8 Paramètre ErrorBits

En présence de plusieurs erreurs simultanées, les valeurs des ErrorBits s'affichent additionnées en binaire. L'affichage de ErrorBits = 16#0000_0003, par ex., indique la présence simultanée des erreurs 16#0000_0001 et 16#0000_0002.

Avec RampSoak, les erreurs signalées au paramètre ErrorBits sont réparties en deux catégories :

- les erreurs avec les messages d'erreur ErrorBits < 16#0001_0000
  La valeur de réglage peut être calculée malgré l'erreur.
- les erreurs avec les messages d'erreur ErrorBits ≥ 16#0001_0000
  L'erreur empêche le calcul de la valeur de réglage. Une valeur de réglage de remplacement est sortie.

Erreurs avec les messages d'erreur ErrorBits < 16#0001_0000

En présence d'une ou de plusieurs erreurs avec les messages d'erreur ErrorBits < 16#0001_0000, l'instruction RampSoak réagit de la manière suivante :

- La valeur de réglage est calculée comme suit malgré cette erreur :
  - Si Reset = FALSE, calcul de la valeur de réglage par l'exécution du profil
  - Sortie de SubstituteOutput si Reset = TRUE
- Le paramètre de sortie Error est mis à 1.
- La sortie de validation ENO n'est pas modifiée.

Le paramètre de sortie Error est supprimé dès que l'erreur a disparu.

ErrorBits (DW#16#...)	Description
0000_0000	Pas d'erreur.
0000_0001	**Cause de l'erreur et réaction à l'erreur :** Le paramètre Output a été limité à -3.402823e+38 ou +3.402823e+38. **Solution :** Si ErrorBits < 16#0001_0000 et Reset = FALSE, la valeur de réglage est limitée tandis qu'elle est déterminée par StartMode ou StopMode. Vérifiez ensuite les paramètres suivants en fonction de la valeur paramétrée aux variables StartMode ou StopMode : • WorkingData.StartValue • SubstituteOutput Si ErrorBits≥ 16#0001_0000 et Reset = FALSE, la valeur de réglage de remplacement est limitée à sa sortie. Vérifiez ensuite les paramètres suivants en fonction de la valeur paramétrée à la variable ErrorMode: • WorkingData.StartValue • SubstituteOutput Si Reset = TRUE, vérifiez le paramètre SubstituteOutput. **Informations complémentaires :** Si vous souhaitez modifier WorkingData.StartValue, éditez d'abord UserData.StartValue et définissez ensuite le paramètre Validate = TRUE. Ne modifiez pas manuellement les données de la structure WorkingData.
0000_0002	**Cause de l'erreur :** La mesure du temps de cycle ne donne pas de valeur valide, tandis que l'exécution du profil est activée (Enable = TRUE). **Réaction à l'erreur :** Si une valeur valide du temps de cycle a déjà été mesurée, l'instruction RampSoak poursuit l'exécution du profil à partir de cette dernière valeur de la variable CycleTime.Value. Si aucune valeur valide du temps de cycle n'a été mesurée préalablement, RampSoak continue de sortir la valeur de réglage configurée avec la variable StartMode au paramètre Output.

ErrorBits (DW#16#...)	Description
0000_0004	**Cause de l'erreur :** Une ou plusieurs variables de la structure UserData ont des valeurs non valides, tandis que les données de profil sont vérifiées. **Réaction à l'erreur :** Les données de profil de la structure UserData ne sont pas reprises dans la structure WorkingData de sorte que les modifications effectuées dans la structure UserData ne sont pas appliquées. **Solution :** Vérifiez que les conditions suivantes sont remplies lorsque les données de profil sont vérifiées : • 1 ≤ UserData.NumberOfUsedPoints ≤ 50 • -3.402823e+38 ≤ UserData.Point[i].Value ≤ 3.402823e+38 avec l'indice *i* = 1..UserData.NumberOfUsedPoints • 0.0 ≤ UserData.Point[i].Time ≤ 3.402823e+38 avec l'indice *i* = 1..UserData.NumberOfUsedPoints • -3.402823e+38 ≤ UserData.StartValue ≤ 3.402823e+38 • NextPoint ≤ UserData.NumberOfUsedPoints • 0.0 < UserData.Point[1].Time + UserData.Point[2].Time +...+ UserData.Point[UserData.NumberOfUsedPoints].Time ≤ 3.402823e+38 • Si l'exécution du profil est activée : CurrentPoint ≤ UserData.NumberOfUsedPoints **Informations complémentaires :** Les données de profil dans la structure UserData sont vérifiées dans les cas suivants : • Lorsque le paramètre Validate prend la valeur TRUE. • Ou lorsqu'après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN, l'instruction RampSoak est appelé pour la première fois avec le paramètre Enable = TRUE et que les données de profil n'ont pas encore été vérifiées. Notez que les variables des structures UserData et WorkingData ne sont pas rémanentes. Ces variables sont initialisées avec les valeurs initiales après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN.
0000_0008	**Cause de l'erreur :** Le paramètre NextPoint a une valeur non valide. **Réaction à l'erreur :** NextPoint est réinitialisé à la dernière valeur valide. **Solution :** Vérifiez que la condition suivante est remplie : • 1 ≤ NextPoint ≤ WorkingData.NumberOfUsedPoints

Erreurs avec les messages d'erreur ErrorBits ≥ 16#0001_0000

En présence d'une ou de plusieurs erreurs avec les messages d'erreur ErrorBits ≥ 16#0001_0000, l'instruction RampSoak réagit de la manière suivante :

- La valeur de réglage ne peut pas être calculée comme prévu. Le tableau suivant montre la réaction du paramètre Output et l'exécution du profil.
- Le paramètre de sortie Error est mis à 1.
- La sortie de validation ENO prend la valeur FALSE.

Dès que les erreurs avec messages d'erreur ErrorBits ≥ 16#0001_0000 ont disparu, RampSoak réagit de la manière suivante :

- La valeur de réglage est calculée comme suit :
  - Si Reset = FALSE, calcul de la valeur de réglage par l'exécution du profil
  - Sortie de SubstituteOutput si Reset = TRUE
- La sortie de validation ENO prend la valeur TRUE.

Le paramètre de sortie Error est supprimé dès que l'erreur a disparu.

<table>
<tr><th>ErrorBits (DW#16#...)</th><th>Description</th></tr>
<tr><td>0001_0000</td><td>Cause de l'erreur :
Le paramètre SubstituteOutput ou la variable SubstituteOutput est actuellement utilisé pour déterminer la valeur de réglage, mais n'a pas de valeur REAL valide.
Réaction à l'erreur :
Si Reset = TRUE et SubstituteOutput est une valeur REAL valide, SubstituteOutput continue d'être fournie au paramètre Output.
Dans tous les autres cas, le paramètre Output est défini sur 0.0.
Solution :
Vérifiez que le paramètre SubstituteOutput et la variable WorkingData.StartValue sont des valeurs REAL valides (≠NaN par exemple 16#7FFF_FFFF). La variable utilisée dépend de Reset, d'erreurs éventuelles et de ErrorMode :
<table>
<tr><th>Reset</th><th>ErrorBits</th><th>ErrorMode</th><th>Variable utilisée</th></tr>
<tr><td>-</td><td>>= 16#0002_0000</td><td>0</td><td>WorkingData.StartValue</td></tr>
<tr><td>-</td><td>>= 16#0002_0000</td><td>1</td><td>SubstituteOutput</td></tr>
<tr><td>TRUE</td><td>-</td><td>-</td><td>SubstituteOutput</td></tr>
</table>
Informations complémentaires :
Si vous souhaitez modifier WorkingData.StartValue, éditez d'abord UserData.StartValue et définissez ensuite le paramètre Validate = TRUE. Ne modifiez pas manuellement les données de la structure WorkingData.</td></tr>
<tr><td>0004_0000</td><td>Cause de l'erreur :
Le calcul pendant l'exécution du profil donne une valeur REAL non valide.
Réaction à l'erreur :
L'exécution du profil est interrompue.
Si Reset = FALSE, la valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est sortie au paramètre Output, puis maintenue.
Si Reset = TRUE, SubstituteOutput continue d'être fournie au paramètre Output.
Solution :
Vérifiez la validité des valeurs REAL dans la structure WorkingData et redémarrez l'exécution du profil si nécessaire.
Informations complémentaires :
Si vous souhaitez modifier les données de profil, éditez d'abord la structure UserData et définissez ensuite le paramètre Validate = TRUE. Ne modifiez pas manuellement les données de Struktur WorkingData.</td></tr>
<tr><td>0008_0000</td><td>Cause de l'erreur :
Le paramètre Enable ou le paramètre Next est défini sur TRUE mais il n'y a pas de données de profil valides dans la structure WorkingData.
Réaction à l'erreur :
Le paramètre Enable et le paramètre Next ne sont pas effectifs.
Si Reset = FALSE, la valeur de réglage de remplacement qui est configurée sur la variable ErrorMode est sortie au paramètre Output.
Si Reset = TRUE, SubstituteOutput continue d'être fournie au paramètre Output.
Solution :
Entrez des données de profil valides dans la structure UserData, puis définissez le paramètre Validate = TRUE. Ainsi les données de profil sont transférées à la structure WorkingData après un contrôle.
Informations complémentaires :
Si le paramètre Enable ou le paramètre Next sont toujours définis sur TRUE, ils prennent effet dès qu'il existe des données de profil valides dans la structure WorkingData. Un autre front montant n'est pas nécessaire.
Notez que les variables des structures UserData et WorkingData ne sont pas rémanentes. Ces variables sont initialisées avec les valeurs initiales après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN.</td></tr>
</table>

ErrorBits (DW#16#...)	Description
0010_0000	**Cause de l'erreur :** La variable (configurée avec StopMode) qui détermine la valeur initiale lorsque l'exécution du profil se termine ou est arrêtée n'a pas de valeur REAL valide. **Réaction à l'erreur :** Si Reset = FALSE, la valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est sortie au paramètre Output, puis maintenue. Si Reset = TRUE, SubstituteOutput continue d'être fournie au paramètre Output. **Solution :** Vérifiez que la variable est une valeur REAL valide (≠ NaN par exemple 16#7FFF_FFFF). La variable utilisée dépend de StopMode : • StopMode = 0: WorkingData.StartValue • StopMode = 1: SubstituteOutput **Informations complémentaires :** Si vous souhaitez modifier WorkingData.StartValue, éditez d'abord UserData.StartValue et définissez ensuite le paramètre Validate = TRUE. Ne modifiez pas manuellement les données de la structure WorkingData.
0020_0000	**Cause de l'erreur :** La variable (configurée avec StartMode) qui détermine la valeur initiale lorsque l'instruction est appelée pour la première fois ou lorsque l'exécution du profil est démarrée n'a pas de valeur REALvalide. **Réaction à l'erreur :** Si Reset = FALSE, la valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est sortie au paramètre Output, puis maintenue. Lorsque l'exécution du profil est activée (Enable = TRUE), elle démarre à partir de cette valeur de réglage de remplacement. Si Reset = TRUE, SubstituteOutput continue d'être fournie au paramètre Output. **Solution :** Vérifiez que la variable est une valeur REAL valide (≠ NaN par exemple 16#7FFF_FFFF). La variable utilisée dépend de StartMode : • StartMode = 0: WorkingData.StartValue • StartMode = 1: SubstituteOutput • StartMode = 2: Output **Informations complémentaires :** Si vous souhaitez modifier WorkingData.StartValue, éditez d'abord UserData.StartValue et définissez ensuite le paramètre Validate = TRUE. Ne modifiez pas manuellement les données de la structure WorkingData.
0040_0000	**Cause de l'erreur :** La variable CycleTime.Value a une valeur non valide, tandis que l'exécution du profil est activée (Enable = TRUE). **Réaction à l'erreur :** L'exécution du profil est interrompue. Si Reset = FALSE, la valeur de réglage de remplacement qui est configurée sur la variable ErrorMode est sortie au paramètre Output. Si Reset = TRUE, SubstituteOutput continue d'être fournie au paramètre Output. L'exécution du profil se poursuivra dès que cette erreur ne sera plus présente. Si l'exécution du profil est arrêtée au préalable, la valeur de réglage de remplacement est alors maintenue. **Solution :** Vérifiez que les conditions suivantes sont remplies : • 0.0 < CycleTime.Value ≤ 3.402823e+38 • CycleTime.Value est une valeur REAL valide (≠ NaN, p. ex. 16#7FFF_FFFF) **Informations complémentaires :** Pour un calcul automatique de la valeur de la variable CycleTime.Value, réglez la variable CycleTime.EnableMeasurement sur TRUE.

10.9 Filter_PT1

10.9.1 Compatibilité avec CPU et FW

Le tableau suivant montre sur quelle CPU vous pouvez utiliser quelle version de Filter_PT1 :

CPU	FW	Filter_PT1
S7-1200	à partir de V4.2	V1.0
CPU basées sur S7-1500	à partir de V2.0	V1.0

10.9.2 Description Filter_PT1

Description

L'instruction Filter_PT1 est un élément de transmission proportionnel avec un retard de premier ordre, également appelé élément PT1.

Vous pouvez utiliser Filter_PT1 aux fins suivantes :

- Filtre passe-bas pour atténuer les contenus à haute fréquence dans un signal, comme par exemple un bruit.
- Élément de temporisation pour lisser des sauts de signal, par exemple la valeur de consigne ou la valeur de réglage d'un régulateur.
- Bloc de simulation de processus pour réaliser une boucle de régulation fermée au sein de la CPU. Vous pouvez ainsi par exemple tester des régulateurs avant la mise en service.

Vous pouvez régler les paramètres de filtre suivants :

- Gain proportionnel (Gain)
- Constante de temps de retard (Lag)

REMARQUE

Différences entre un élément PT1 à action continue et Filter_PT1

Filter_PT1 étant exécuté dans un programme API, Filter_PT1 est une mise en œuvre à temps discret d'un élément PT1. Les systèmes à temps discret n'ont pas les mêmes propriétés que le modèle à action continue correspondant. Les systèmes à temps discret peuvent cependant reproduire correctement un système à action continue en fonction du temps de cycle : Plus le temps de cycle est petit et constant, plus l'écart entre les propriétés de Filter_PT1 et les propriétés de l'élément PT1 à action continue est petit. Les propriétés d'un élément PT1 à action continue sont la fonction de transfert, le comportement temporel et la réponse en fréquence décrits ci-dessous.

Pour une reproduction correcte de la réponse en fréquence, il est recommandé de régler un temps de cycle maximal d'un dixième de la durée de période la plus courte des composants de signal d'entrée. Par exemple, un signal avec éléments de fréquence de jusqu'à 50 Hz a une durée de période la plus courte de 20 ms. Pour obtenir reproduction correcte de la réponse en fréquence, il est par exemple recommandé de régler un temps de cycle maximal de 2 ms.

Fonction de transmission d'un élément PT1

La formule suivante indique la fonction de transfert d'un élément PT1, avec s équivalent à l'opérateur Laplace :

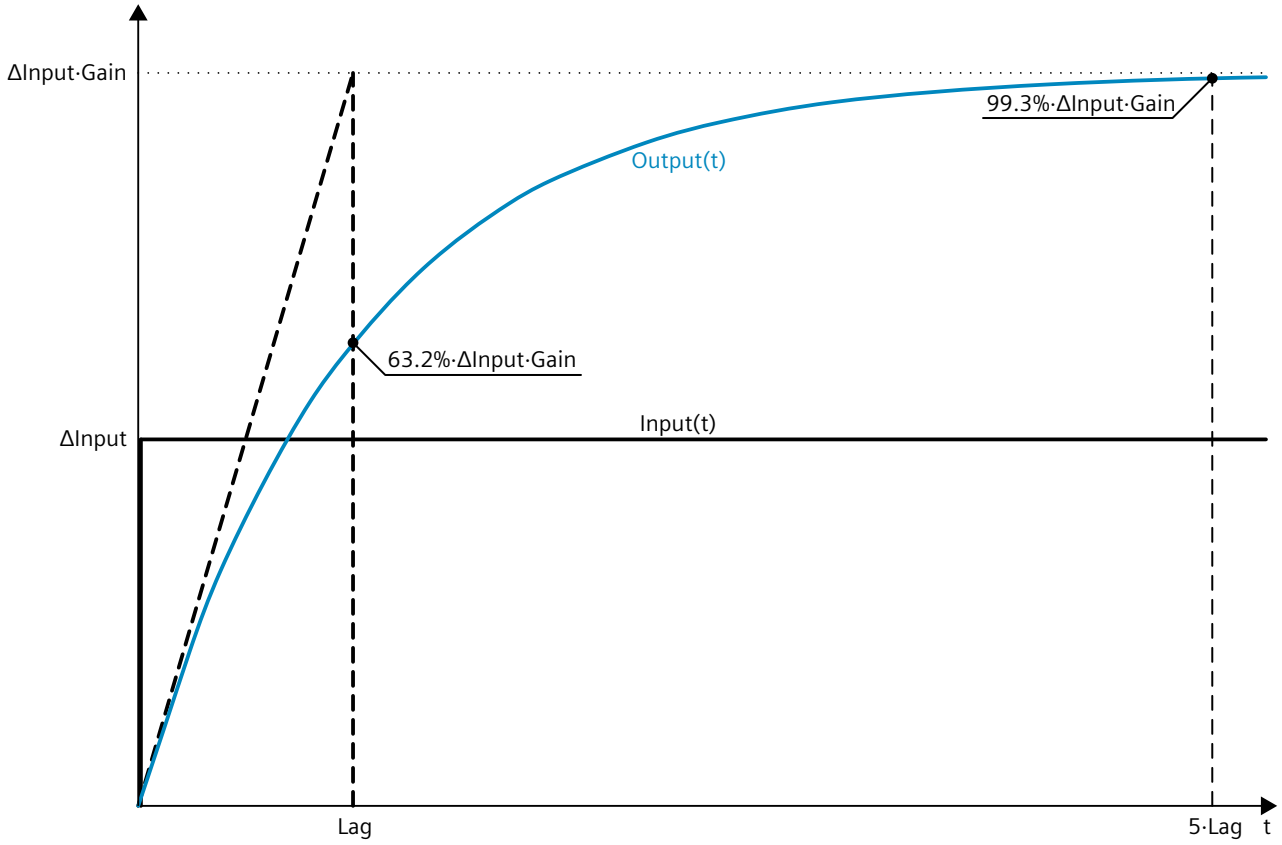
$$G(s) = \frac{\text{Output(s)}}{\text{Input(s)}} = \frac{\text{Gain}}{1 + \text{Lag} \cdot s}$$

Comportement temporel d'un élément PT1

La réponse indicielle est la réaction de la valeur de réglage à un saut de la valeur d'entrée. La réponse indicielle pour un saut de la valeur d'entrée de 0 à ΔInput peut être calculée avec la formule suivante :

$$Output(t) = \Delta Input \cdot Gain \cdot (1 - e^{-\frac{t}{Lag}})$$

La figure ci-dessous montre la réponse indicielle d'un élément PT1 :



Réponse en fréquence d'un élément PT1

La réponse en fréquence d'un élément de transmission est décrite par la réponse d'amplitude et la réponse de phase.

La réponse d'amplitude décrit l'amplification d'un signal par l'élément de transmission en fonction de la pulsation du signal.

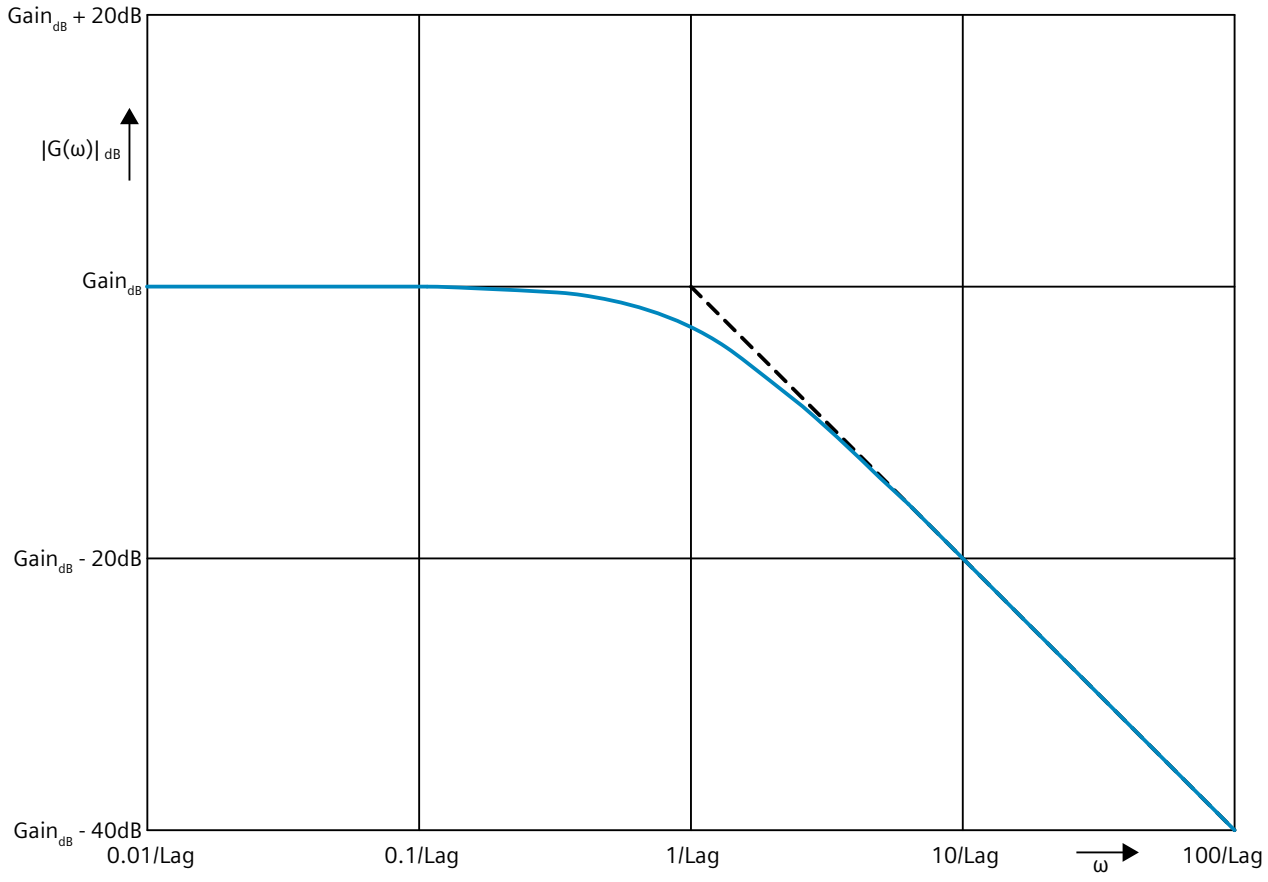
L'équation suivante décrit la réponse d'amplitude d'un élément PT1 :

$$|G(\omega)| = \frac{\text{Gain}}{\sqrt{1 + (\omega \cdot \text{Lag})^2}}$$

|G(ω)| Amplification d'un signal en fonction de la pulsation

ω Pulsation

La figure suivante montre la réponse d'amplitude d'un élément PT1 :



La réponse de phase décrit le décalage de phases d'un signal par l'élément de transmission en fonction de la pulsation du signal.

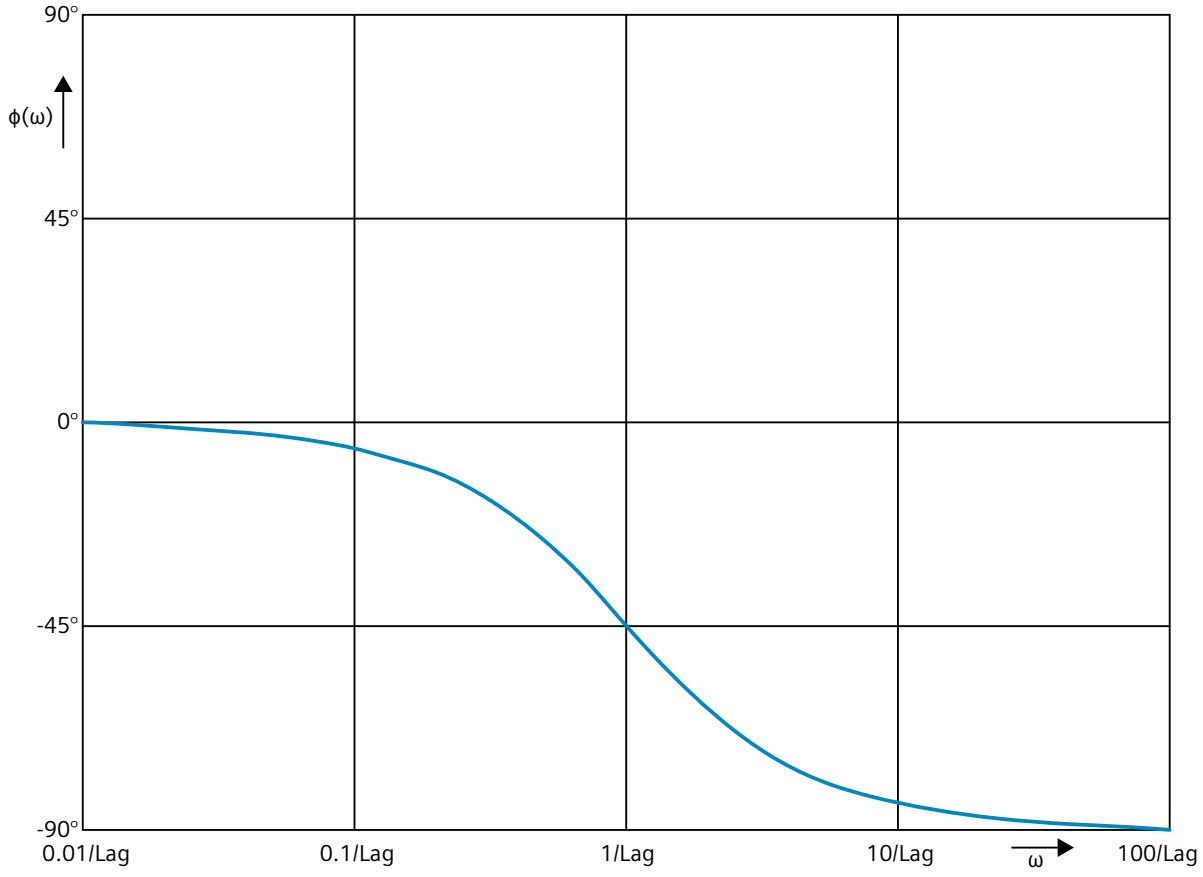
L'équation suivante décrit la réponse de phase d'un élément PT1 :

$$\varphi(\omega) = -\tan^{-1}(\omega \cdot \text{Lag})$$

φ(ω) Décalage de phases en fonction de la pulsation

ω Pulsation

La figure suivante montre la réponse de phase d'un élément PT1 :



Appel

Dans un OB ou une FC, l'instruction Filter_PT1 est appelée comme DB d'instance unique. Dans un FB, l'instruction Filter_PT1 peut être appelée aussi bien comme DB d'instance unique que comme DB de multi-instance ou DB d'instance de paramètre.
Aucun objet technologique n'est créé lors de l'appel de l'instruction. Vous ne disposez pas d'une interface de paramétrage et de mise en service. Vous paramétrez Filter_PT1 directement via le DB d'instance et vous mettez en service Filter_PT1 via une table de visualisation du programme utilisateur dans la CPU ou HMI.

Démarrage

Les variables de la plage statique de Filter_PT1 ne sont pas rémanentes. Ces variables sont initialisées à l'aide des valeurs initiales après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN. Si vous modifiez les valeurs actuelles en mode en ligne et que ces valeurs doivent être conservées après le changement d'état de fonctionnement de la CPU, sauvegardez ces valeurs dans les valeurs initiales du bloc de données.

Vous définissez la valeur d'initialisation pour le paramètre Output sur la variable StartMode.

La valeur d'initialisation est émise au premier appel de Filter_PT1 après le

- changement d'état de fonctionnement de la CPU

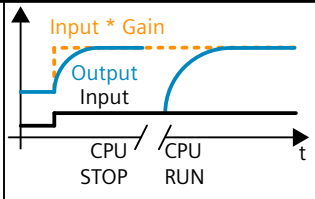
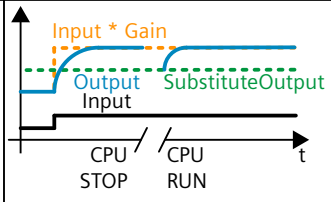
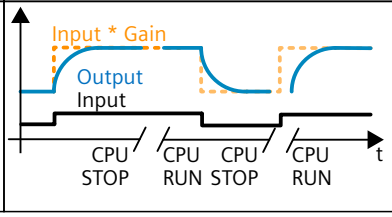
ou

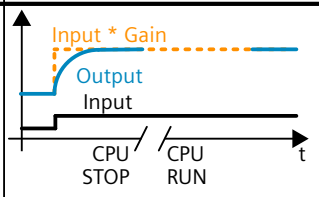
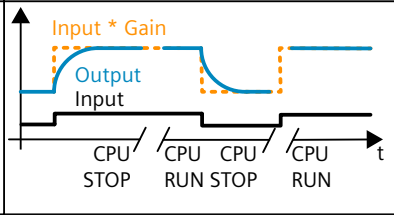
- l'exécution de "Charger les valeurs de départ comme valeur en cours" (uniquement pour l'option "Toutes les valeurs", et pas pour l'option "Consignes uniquement")

au paramètre Output.

Lors des appels suivants, Filter_PT1 calcule la valeur de réglage, à partir de cette valeur d'initialisation, sur la base de la valeur d'entrée et de la configuration du filtre.

Le tableau suivant présente la dépendance entre la variable StartMode et le paramètre Output. Les valeurs dans la colonne Output sont émises au paramètre Output après le changement d'état de fonctionnement de la CPU :

StartMode	Output	Exemple
0	Valeur du paramètre Input	
1	Valeur du paramètre SubstituteOutput	
2	Reste inchangé Le paramètre Output est rémanent. Paramètre par défaut Utilisé si StartMode n'est pas dans la plage 0...4	

StartMode	Output	Exemple
3	0.0	
4	Valeur du produit Input * Gain	

Les points suivants s'appliquent en plus pour toutes les valeurs de la variable StartMode :

- La valeur d'initialisation est limitée à la plage de valeurs du type de données REAL. Ce n'est qu'ensuite que la valeur d'initialisation est émise au paramètre Output.
- Si la valeur d'initialisation n'est pas une valeur REALvalide, la valeur de réglage de remplacement est émise au paramètre Output. La valeur de réglage de remplacement est configurée par la variable ErrorMode. La valeur de réglage de remplacement est limitée à la plage de valeurs du type de données REAL avant d'être émise au paramètre Output. Si la valeur de réglage de remplacement n'est pas non plus une valeur REAL valide, 0.0 est émise au paramètre Output. Lors des appels suivants, l'instruction calcule la valeur de réglage à partir de cette valeur de réglage de remplacement.
- La variable StartMode n'est active que lorsque le paramètre Reset = FALSE lors du premier appel de l'instruction et si aucune erreur n'apparaît avec le message d'erreur ErrorBits ≥ 16#0002_0000. Lorsque le paramètre Reset = TRUE, la valeur du paramètre SubstituteOutput est émise au paramètre Output. En présence d'une erreur déclenchant un message d'erreur ErrorBits ≥ 16#0002_0000, la valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output.

Comportement en cas d'erreur

L'instruction Filter_PT1 détecte différentes erreurs pouvant survenir lors du calcul de la valeur de réglage. Le résultat de ce calcul peut être émis en dépit de la présence d'une erreur à la sortie. Si aucun calcul correct de la valeur de réglage n'est possible à cause d'une erreur, une valeur de réglage de remplacement est émise à la sortie.

Vous définissez, pour la variable ErrorMode, la valeur de réglage de remplacement en cas d'erreur rendant impossible tout calcul correct de la valeur de réglage.
Le tableau suivant présente la dépendance entre la valeur de la variable ErrorMode et la valeur de réglage de remplacement que Filter_PT1 émet au paramètre Output :

ErrorMode	Output
0	Valeur du paramètre Input
1	Valeur du paramètre SubstituteOutput

ErrorMode	Output
2	La dernière valeur de réglage de filtre valide 0.0, si aucune valeur de réglage de filtre valide n'est disponible Paramètre par défaut Utilisé si ErrorMode n'est pas dans la plage 0...4
3	0.0
4	Valeur du produit Input * Gain

Les points suivants s'appliquent en plus pour toutes les valeurs de la variable ErrorMode :

- Si la valeur de réglage de remplacement n'a pas de de valeur REAL valide, 0.0 est émis comme valeur de réglage.
- La valeur de réglage de remplacement est limitée à la plage de valeurs du type de données REAL. Ce n'est qu'ensuite que la valeur de réglage de remplacement est émise au paramètre Output.
- La variable ErrorMode n'est active que lorsque le paramètre Reset = FALSE. Lorsque le paramètre Reset = TRUE, la valeur du paramètre SubstituteOutput est émise au paramètre Output.

Le paramètre Error indique si une erreur est actuellement présente. Quand l'erreur a disparu, Error est réglé sur FALSE. Le paramètre ErrorBits indique les erreurs qui se sont produites. ErrorBits est rémanent et n'est réinitialisé que par un front montant au paramètre Reset ou ErrorAck.

10.9.3 Fonctionnement de Filter_PT1

Comportement Reset

Le comportement de Filter_PT1 dépend du paramètre Reset de la manière suivante :

- Lorsque le paramètre Reset =TRUE, la valeur du paramètre SubstituteOutput est émise au paramètre Output.
- Lorsque le paramètre Reset = FALSE, la valeur émise au paramètre Output est calculée par l'algorithme de filtre.
- Si le paramètre Reset passe de FALSE à TRUE, la valeur au paramètre Output devient directement la valeur du paramètre SubstituteOutput. Ce changement peut se faire par à-coups. Le paramètre ErrorBits est également réinitialisé.
- Si le paramètre Reset passe de TRUE à FALSE, l'algorithme de filtre est réglé de manière à ce que le changement se fasse sans à-coups.

Comportement de validation EN/ENO

Si l'une des conditions suivantes est remplie, la sortie de validation ENO est réglée sur FALSE :

- L'entrée de validation EN est réglée sur TRUE et le paramètre Output est défini par une valeur de réglage de remplacement pour les messages d'erreur ErrorBits ≥ 16#0001_0000.
- L'entrée de validation EN est réglée sur FALSE.

Sinon, la sortie de validation ENO est réglée sur TRUE.

Mesure automatique du temps de cycle

Pour le calcul de la valeur de réglage, Filter_PT1 a besoin du temps qui s'est écoulé depuis le dernier appel de Filter_PT1.

Le temps de cycle est automatiquement mesuré par défaut et est émis au niveau de la variable CycleTime.Value à partir du deuxième appel. Filter_PT1 mesure le temps de cycle à chaque appel de l'instruction et est donc utilisable dans des cycles d'appel non équidistants, par ex. dans l'OB1.

Notez que des appels conditionnels, les points d'arrêt actifs ou le chargement d'instantanés comme valeurs en cours rallongent le temps de cycle en cas de mesure automatique du temps de cycle. Un temps de cycle trop élevé est détecté comme erreur avec le message d'erreur ErrorBits = 16#0008_0000.

Si la mesure du temps de cycle ne fournit aucun résultat valide, Filter_PT1 calcule la valeur de réglage actuelle à l'aide du dernier temps de cycle valide. De plus, Filter_PT1 émet un message d'erreur au paramètre ErrorBits.

Si vous désactivez la mesure automatique du temps de cycle en définissant la variable CycleTime.EnableMeasurement = FALSE, vous devez spécifier le temps de cycle manuellement au niveau de la variable CycleTime.Value. Filter_PT1 vérifie la validité de la variable CycleTime.Value à chaque appel.

Mesure automatique du temps de cycle avec points d'arrêt

Si des points d'arrêt sont actifs entre deux appels de Filter_PT1, la mesure automatique du temps de cycle donne le temps réel qui s'est écoulé entre deux appels. Si un point d'arrêt est actif, la CPU se trouve à l'état de fonctionnement ATTENTE.

REMARQUE

Les points d'arrêt actifs allongent l'intervalle de temps entre deux appels de Filter_PT1.

Avec un intervalle de temps plus long entre deux appels, la modification de la valeur de réglage augmente également au paramètre Output.

Il est en outre possible que des intervalles plus longs violent la condition Lag ≥ CycleTime.Value/2 et qu'une erreur avec message d'erreur ErrorBits = 16#0008_0000 soit alors détectée.

Si vous n'avez pas besoin du calcul de la valeur de réglage sur la base du temps réel avec des points d'arrêt actifs, exécutez alors les étapes suivantes :

- Désactivez la mesure automatique du temps de cycle en définissant la variable CycleTime.EnableMeasurement = FALSE.
- Saisissez le temps de cycle manuellement au niveau de la variable CycleTime.Value.

10.9.4 Paramètre d'entrée Filter_PT1

Paramètre	Type de données	Valeur par défaut	Description
Input	REAL	0.0	Valeur d'entrée
SubstituteOutput	REAL	0.0	SubstituteOutput est utilisé comme valeur de réglage de remplacement si • Reset = TRUE ou • si aucun calcul correct de la valeur de réglage n'est possible à cause d'une erreur avec le message d'erreur ErrorBits ≥ 16#0001_0000 et que ErrorMode est configuré à la valeur 1.
ErrorAck	BOOL	FALSE	Supprime les messages d'erreur • Front FALSE -> TRUE ErrorBits est réinitialisé.
Reset	BOOL	FALSE	Exécute un redémarrage de l'instruction • Front FALSE -> TRUE ErrorBits est réinitialisé. • Tant que Reset est réglé sur TRUE, la valeur de réglage de remplacement SubstituteOutput est émise à la sortie. • Tant que Reset est réglé sur FALSE, le calcul de la valeur de réglage est exécuté.

10.9.5 Paramètre de sortie Filter_PT1

Paramètre	Type de données	Valeur par défaut	Description
Output	REAL	0.0	Valeur de réglage La valeur de réglage est rémanente.
ErrorBits	DWORD	DW#16#0	Le paramètre ErrorBits (Page 536) signale quels sont les messages d'erreur présents. ErrorBits est rémanent et réinitialisé à Reset ou ErrorAck en cas de front montant.
Error	BOOL	FALSE	Le réglage de Error sur TRUE indique la présence d'au moins une erreur actuellement.

10.9.6 Variables statiques Filter_PT1

Variable	Type de données	Valeur par défaut	Description
Gain	REAL	1.0	Gain proportionnel
Lag	REAL	25.0	Constante de temps de retard en secondes Plage de valeurs autorisée : Lag ≥ CycleTime.Value/2
ErrorMode	INT	2	Sélection de la valeur de réglage de remplacement en cas d'erreur • 0 = Input • 1 = SubstituteOutput • 2 = dernière valeur de réglage de filtre valide • 3 = 0.0 • 4 = Input * Gain Plage de valeurs autorisée : 0 à 4

Variable	Type de données	Valeur par défaut	Description
StartMode	INT	2	Sélection de la valeur de réglage pour le premier appel de l'instruction • 0 = Input • 1 = SubstituteOutput • 2 = dernière valeur de réglage • 3 = 0.0 • 4 = Input * Gain Plage de valeurs autorisée : 0 à 4
CycleTime	AuxFct_CycleTime	-	Données du temps de cycle
CycleTime.Value	REAL	0.1	Temps de cycle en secondes (intervalle entre 2 appels) Plage de valeurs autorisée : CycleTime.Value > 0.0
CycleTime.EnableMeasurement	BOOL	TRUE	Mesure automatique du temps de cycle • FALSE = désactivée • TRUE = activée

10.9.7 Paramètre ErrorBits

En présence de plusieurs erreurs simultanées, les valeurs des ErrorBits s'affichent comme addition binaire. L'affichage de ErrorBits = 16#0000_0003, par ex., indique la présence simultanée des erreurs 16#0000_0001 et 16#0000_0002.

Avec Filter_PT1, les erreurs émises au paramètre ErrorBits sont réparties en deux catégories :

- les erreurs déclenchant des messages d'erreur ErrorBits < 16#0001_0000
- les erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000

Erreurs déclenchant des messages d'erreur ErrorBits < 16#0001_0000

En présence d'une ou de plusieurs erreurs déclenchant des messages d'erreur ErrorBits < 16#0001_0000, Filter_PT1 réagit de la manière suivante :

- La valeur de réglage est déterminée comme suit en dépit de cette erreur :
  - Calcul de la valeur de réglage par l'algorithme de filtre si Reset = FALSE
  - Sortie de SubstituteOutput si Reset = TRUE
- Le paramètre de sortie Error est mis à 1.
- La sortie de validation ENO n'est pas modifiée.

Le paramètre de sortie Error est supprimé dès que l'erreur a disparu.

ErrorBits (DW#16#...)	Description
0000_0000	Pas d'erreur.
0000_0001	**Cause d'erreur et réaction en cas d'erreur :** Le paramètre Output a été limité à -3.402823e+38 ou +3.402823e+38. **Solution :** Si la valeur déterminée par la fonction de filtre est émise à la sortie (Reset = FALSE et ErrorBits < 16#0001_0000), vérifiez les variables suivantes utilisées dans le calcul de filtre : • Input • Gain • Lag • CycleTime.Value

ErrorBits (DW#16#...)	Description
	Si ErrorBits≥ 16#0001_0000 et Reset = FALSE, la valeur de réglage de remplacement est limitée à sa sortie. Vérifiez ensuite les paramètres suivants en fonction de la valeur paramétrée à la variable ErrorMode: • Input • SubstituteOutput • Le produit de Input et Gain Si Reset = TRUE, vérifiez le paramètre SubstituteOutput.
0000_0002	**Cause d'erreur :** La mesure du temps de cycle ne donne pas de valeur valide, tandis que le calcul de la valeur de réglage est exécuté (Reset = FALSE). **Réaction en cas d'erreur :** Si une valeur valide du temps de cycle a déjà été mesurée, Filter_PT1 calcule la valeur de réglage à partir de cette dernière valeur de la variable CycleTime.Value. Si aucune valeur valide du temps de cycle n'a été mesurée préalablement, Filter_PT1 continue d'émettre la valeur de réglage configurée avec la variable StartMode au paramètre Output.

Erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000

En présence d'une ou de plusieurs erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000, Filter_PT1 réagit de la manière suivante :

- La valeur de réglage ne peut pas être déterminée comme prévu. À la place, c'est la valeur de réglage de remplacement qui est émise.
- Le paramètre de sortie Error est mis à 1.
- La sortie de validation ENO est réglée sur FALSE.

Dès que les erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000 ont disparu, Filter_PT1 réagit de la manière suivante :

- La valeur de réglage est déterminée comme suit :
  - Calcul de la valeur de réglage par l'algorithme de filtre si Reset = FALSE
  - Sortie de SubstituteOutput si Reset = TRUE
- La sortie de validation ENO est réglée sur TRUE.

Le paramètre de sortie Error est supprimé dès que l'erreur a disparu.

<table>
<tr><th>ErrorBits (DW#16#...)</th><th colspan="3">Description</th></tr>
<tr><td>0001_0000</td><td colspan="3">Cause d'erreur :
Le paramètre SubstituteOutput ou une autre variable qui est utilisée comme valeur de réglage n'a pas de valeur REAL valide.
Réaction en cas d'erreur :
La sortie est réglée sur 0.0.
Solution :
Vérifiez que la variable utilisée comme valeur de réglage est une valeur REAL valide (≠NaN p. ex. 16#7FFF_FFFF). La variable utilisée comme valeur de réglage dépend de Reset et de ErrorMode :</td></tr>
<tr><td></td><td>Reset</td><td>ErrorMode</td><td>Valeur de réglage</td></tr>
<tr><td></td><td>FALSE</td><td>0</td><td>Input</td></tr>
<tr><td></td><td>FALSE</td><td>1</td><td>SubstituteOutput</td></tr>
<tr><td></td><td>FALSE</td><td>4</td><td>Produit de Input et Gain</td></tr>
<tr><td></td><td>TRUE</td><td>-</td><td>SubstituteOutput</td></tr>
</table>

ErrorBits (DW#16#...)	Description
0002_0000	**Cause d'erreur :** Le paramètre Input n'a pas de valeur REAL valide, tandis que le calcul de la valeur de réglage est exécuté (Reset = FALSE). **Réaction en cas d'erreur :** La valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output. 0.0 est émis comme valeur de réglage si ErrorMode = 0. **Solution :** Vérifiez que le paramètre Input est une valeur REAL valide (≠NaN p. ex. 16#7FFF_FFFF).
0004_0000	**Cause d'erreur :** Le calcul de la valeur de réglage donne une valeur REAL non valide pour le paramètre Output. **Réaction en cas d'erreur :** La valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output. **Solution :** Vérifiez toutes les variables utilisées pour le calcul de la valeur de réglage : • Input • Gain • Lag • CycleTime.Value Les valeurs de ces variables sont valides. Le calcul de la valeur de réglage dans cette combinaison de variables échoue.
0008_0000	**Cause d'erreur :** La variable Gain ou Lag a une valeur valide, tandis que le calcul de la valeur de réglage est exécuté (Reset = FALSE). **Réaction en cas d'erreur :** La valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output. **Solution :** Vérifiez que les conditions suivantes pour les valeurs des variables Gain et Lag sont remplies : • -3.402823e+38 ≤ Gain ≤ 3.402823e+38 • CycleTime.Value/2 ≤ Lag ≤ 3.402823e+38 • Les valeurs sont des valeurs REAL valides (≠ NaN par ex. 16#7FFF_FFFF) **Informations complémentaires :** Tenez compte du fait que la condition CycleTime.Value/2 ≤ Lag n'est pas respectée dans les scénarios suivants : • L'intervalle entre deux appels de Filter_PT1 est plus long que 2 * Lag, p. ex. en raison d'appels conditionnels dans l'exécution du programme ou de points d'arrêt actifs. • Un instantané du DB d'instance Filter_PT1 est chargé comme valeurs actuelles dans la CPU et la création de l'instantané remonte à plus de 2 * Lag. Dans ces scénarios, une erreur avec message d'erreur ErrorBits = 16#0008_0000 est détectée en cas de mesure automatique du temps de cycle.
0020_0000	**Cause d'erreur :** La variable (configurée avec StartMode) pour l'initialisation du paramètre Output lors du premier appel de l'instruction n'a pas de valeur REAL valide. **Réaction en cas d'erreur :** Lors du premier appel de l'instruction, la valeur de réglage de remplacement qui est configurée au niveau de la variable ErrorMode est émise au paramètre Output. Lors des appels suivants, Filter_PT1 calcule la valeur de réglage à partir de cette valeur de réglage de remplacement. **Solution :**

ErrorBits (DW#16#...)	Description
	Vérifiez que la variable pour l'initialisation du paramètre Output est une valeur REAL valide (≠ NaN p. ex. 16#7FFF_FFFF). Si Reset = FALSE, l'initialisation n'est active lors du premier appel de l'instruction qu'après le passage de la CPU de l'état de fonctionnement STOP à l'état RUN. La variable utilisée pour l'initialisation du paramètre Output dépend de StartMode : • StartMode = 1: SubstituteOutput • StartMode = 2: Output • StartMode = 4: Produit de Input et Gain
0040_0000	**Cause d'erreur :** La variable CycleTime.Value n'a pas de valeur valide, tandis que le calcul de la valeur de réglage est exécuté (Reset = FALSE). **Réaction en cas d'erreur :** La valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output. **Solution :** Vérifiez que les conditions suivantes sont remplies : • 0.0 < CycleTime.Value ≤ 3.402823e+38 • CycleTime.Value est une valeur REAL valide (≠ NaN, p. ex. 16#7FFF_FFFF) **Informations complémentaires :** Pour un calcul automatique de la valeur de la variable CycleTime.Value, réglez la variable CycleTime.EnableMeasurement sur TRUE.

10.10 Filter_PT2

10.10.1 Compatibilité avec CPU et FW

Le tableau suivant montre sur quelle CPU vous pouvez utiliser quelle version de Filter_PT2 :

CPU	FW	Filter_PT2
S7-1200	à partir de V4.2	V1.0
CPU basées sur S7-1500	à partir de V2.0	V1.0

10.10.2 Description Filter_PT2

Description

L'instruction Filter_PT2 est un élément de transmission proportionnel avec un retard de second ordre, également appelé élément PT2.

Vous pouvez utiliser Filter_PT2 aux fins suivantes :

- Filtre passe-bas pour atténuer les contenus à haute fréquence dans un signal, comme par exemple un bruit.
- Élément de temporisation pour lisser des sauts de signal, par exemple la valeur de consigne ou la valeur de réglage d'un régulateur.
- Bloc de simulation de processus pour réaliser une boucle de régulation fermée au sein de la CPU. Vous pouvez ainsi par exemple tester des régulateurs avant la mise en service.

Vous pouvez régler les paramètres de filtre suivants :

- Gain proportionnel (Gain)
- Constante de temps (TimeConstant)
- Atténuation (Damping)

REMARQUE

Différences entre un élément PT2 à action continue et Filter_PT2

Filter_PT2 Étant exécuté dans un programme API, Filter_PT2 est une mise en œuvre à temps discret d'un élément PT2. Les systèmes à temps discret n'ont pas les mêmes propriétés que le modèle à action continue correspondant. Les systèmes à temps discret peuvent cependant reproduire correctement un système à action continue en fonction du temps de cycle : Plus le temps de cycle est petit et constant, plus l'écart entre les propriétés de Filter_PT2 et les propriétés de l'élément PT2 à action continue est petit. Les propriétés d'un élément PT2 à action continue sont la fonction de transfert, le comportement temporel et la réponse en fréquence décrits ci-dessous.

Pour une reproduction correcte de la réponse en fréquence, il est recommandé de régler un temps de cycle maximal d'un dixième de la durée de période la plus courte des composants de signal d'entrée. Par exemple, un signal avec éléments de fréquence de jusqu'à 50 Hz a une durée de période la plus courte de 20 ms. Pour obtenir reproduction correcte de la réponse en fréquence, il est par exemple recommandé de régler un temps de cycle maximal de 2 ms.

Fonction de transmission d'un élément PT2

La formule suivante indique la fonction de transfert d'un élément PT2, avec s équivalent à l'opérateur Laplace :

$$G(s) = \frac{\text{Output(s)}}{\text{Input(s)}} = \frac{\text{Gain}}{1 + 2 \cdot \text{Damping} \cdot \text{TimeConstant} \cdot s + \text{TimeConstant} \cdot s}$$

Si Damping ≥ 1, l'élément PT2 peut être décrit comme deux éléments PT1 connectés en série :

$$G(s) = \frac{\text{Output(s)}}{\text{Input(s)}} = \frac{\text{Gain}}{(1 + \text{Lag}_1 \cdot s) \cdot (1 + \text{Lag}_2 \cdot s)}$$

Les constantes de temps de retard des deux éléments PT1 connectés en série sont calculées comme suit :

$$Lag_{1/2} = TimeConstant \cdot \left(Damping \pm \sqrt{Damping^2 - 1} \right)$$

Comportement temporel d'un élément PT2

La réponse indicielle est la réaction de la valeur de réglage à un saut de la valeur d'entrée.
La réponse indicielle pour un saut de la valeur d'entrée de 0 à ΔInput peut être calculée avec les formules suivantes :
La formule suivante est valable pour Damping < 1 :

$$Output(t) = \Delta Input \cdot Gain \cdot \left(1 - e^{-\frac{Damping}{TimeConstant} \cdot t} \cdot \left[\cos\left(\frac{\sqrt{1 - Damping^2}}{TimeConstant} \cdot t\right) + \frac{Damping}{\sqrt{1 - Damping^2}} \cdot \sin\left(\frac{\sqrt{1 - Damping^2}}{TimeConstant} \cdot t\right)\right]\right)$$

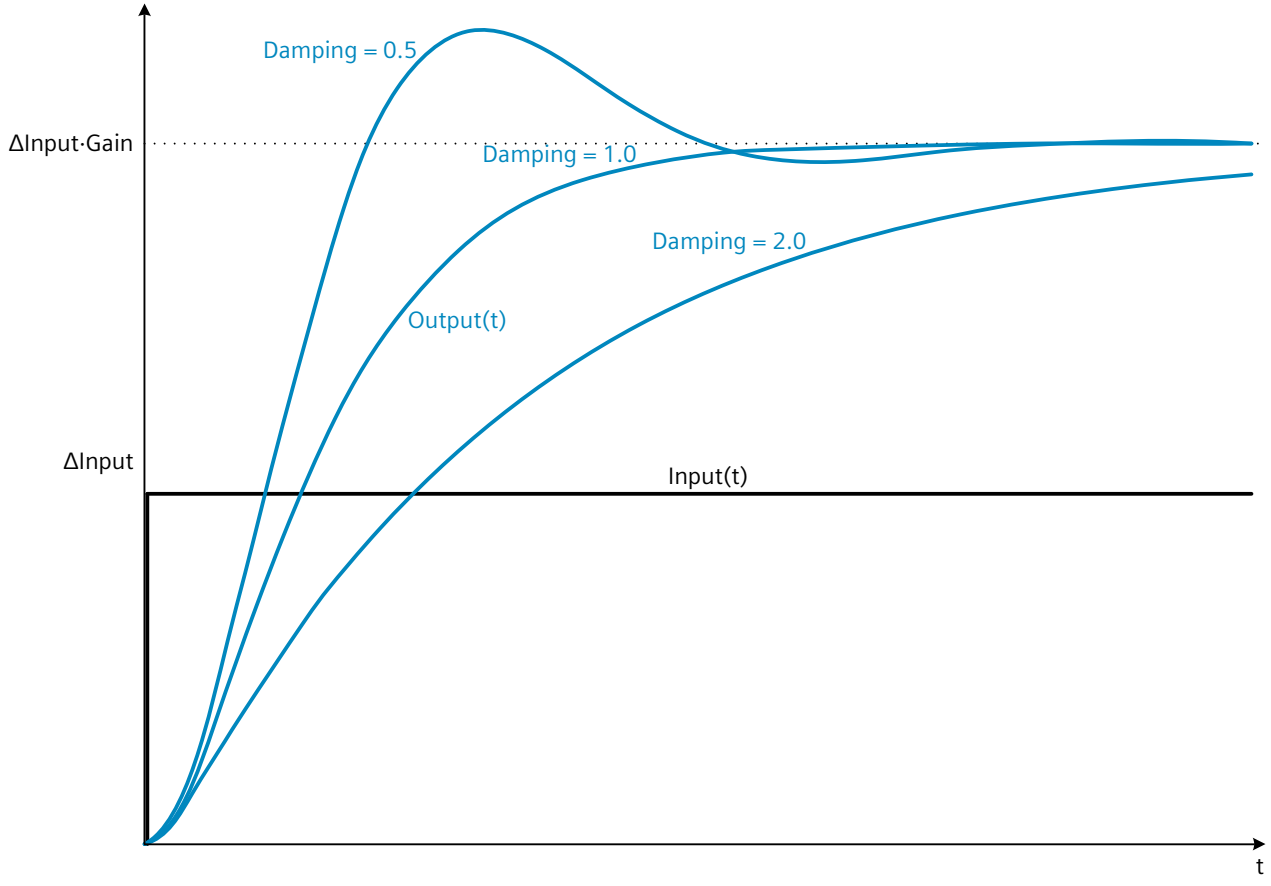
La formule suivante est valable pour Damping = 1 :

$$Output(t) = \Delta Input \cdot Gain \cdot \left(1 - e^{-\frac{t}{TimeConstant}} - \frac{t}{TimeConstant} \cdot e^{-\frac{t}{TimeConstant}}\right)$$

La formule suivante est valable pour Damping > 1 avec Lag_1 et Lag_2 calculés ci-dessus :

$$Output(t) = \Delta Input \cdot Gain \cdot \left(1 - \frac{Lag_1}{Lag_1 - Lag_2} \cdot e^{-\frac{t}{Lag_1}} + \frac{Lag_2}{Lag_1 - Lag_2} \cdot e^{-\frac{t}{Lag_2}}\right)$$

La figure ci-dessous montre la réponse indicielle d'un élément PT2 avec différentes valeurs pour l'atténuation :



Réponse en fréquence d'un élément PT2

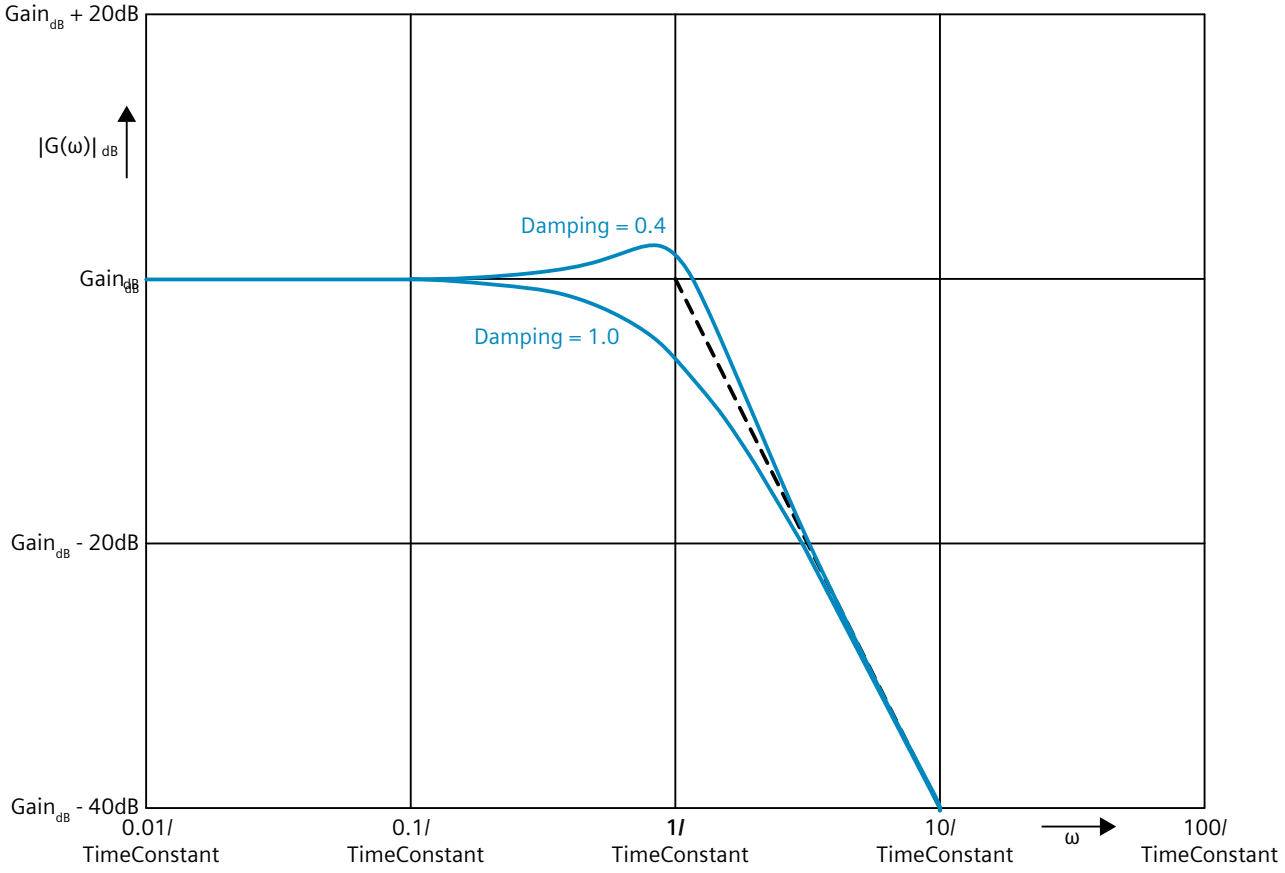
La réponse en fréquence d'un élément de transmission est décrite par la réponse d'amplitude et la réponse de phase.
La réponse d'amplitude décrit l'amplification d'un signal par l'élément de transmission en fonction de la pulsation du signal.
L'équation suivante décrit la réponse d'amplitude d'un élément PT2 :

$$|G(\omega)| = \frac{\text{Gain}}{\sqrt{(1 - [\omega \cdot \text{TimeConstant}]^2)^2 + (\omega \cdot 2 \cdot \text{Damping} \cdot \text{TimeConstant})^2}}$$

|G(ω)| Amplification d'un signal en fonction de la pulsation

ω Pulsation

La figure suivante montre la réponse d'amplitude d'un élément PT2 avec une atténuation de 0,4 et 1,0 :



REMARQUE

Si Damping < 1/√2, une surélévation de résonance apparaît dans la réponse d'amplitude.

La réponse de phase décrit le décalage de phases d'un signal par l'élément de transmission en fonction de la pulsation du signal.

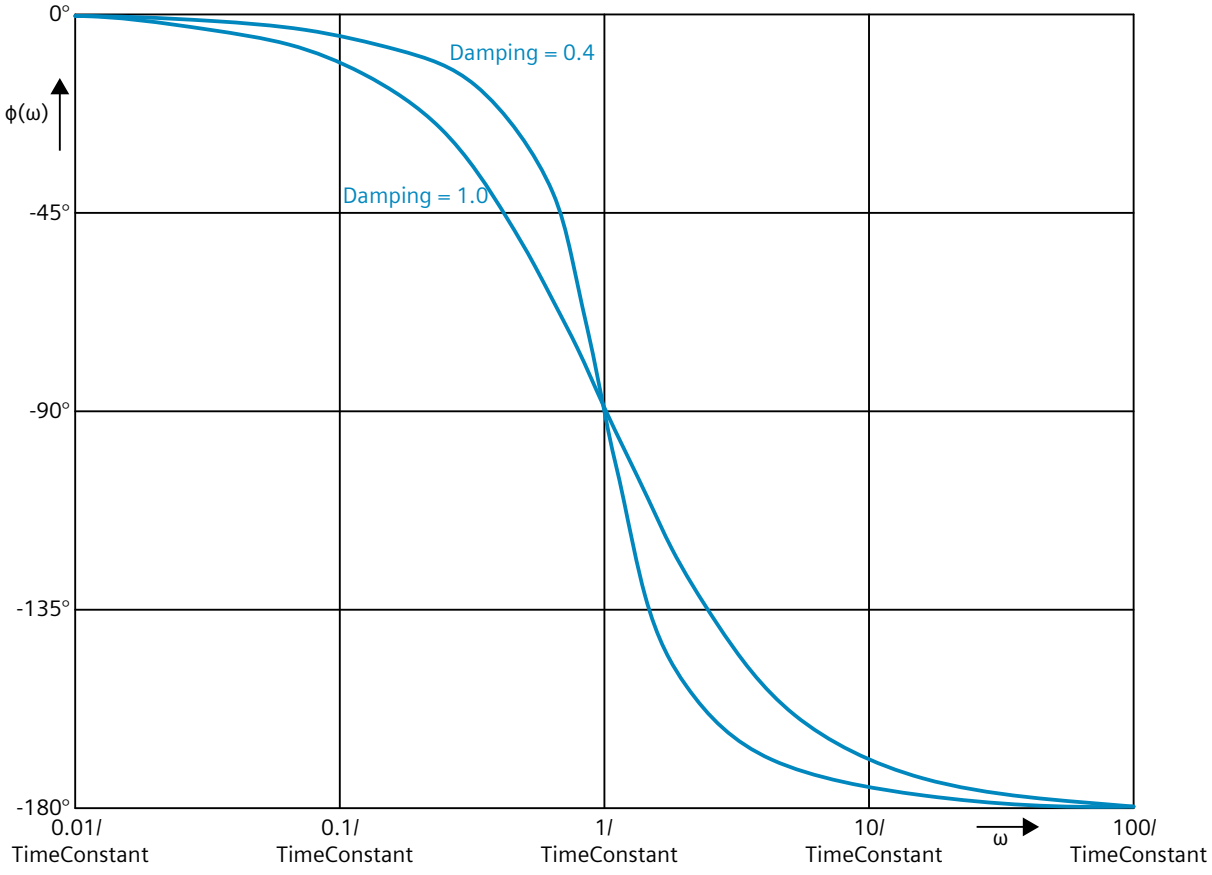
L'équation suivante décrit la réponse de phase d'un élément PT2 :

$$\varphi(\omega) = -\tan^{-1}\left(\frac{\omega \cdot 2 \cdot \text{Damping} \cdot \text{TimeConstant}}{1 - (\omega \cdot \text{TimeConstant})^2}\right)$$

φ(ω) Décalage de phases en fonction de la pulsation

ω Pulsation

La figure suivante montre la réponse de phase d'un élément PT2 :



Appel

Dans un OB ou une FC, l'instruction Filter_PT2 est appelée comme DB d'instance unique. Dans un FB, l'instruction Filter_PT2 peut être appelée aussi bien comme DB d'instance unique que comme DB de multi-instance ou DB d'instance de paramètre.

Aucun objet technologique n'est créé lors de l'appel de l'instruction. Vous ne disposez pas d'une interface de paramétrage et de mise en service. Vous paramétrez Filter_PT2 directement via le DB d'instance et vous mettez en service Filter_PT2 via une table de visualisation du programme utilisateur dans la CPU ou HMI.

Démarrage

Les variables de la plage statique de Filter_PT2 ne sont pas rémanentes. Ces variables sont initialisées à l'aide des valeurs initiales après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN. Si vous modifiez les valeurs actuelles en mode en ligne et que ces valeurs doivent être conservées après le changement d'état de fonctionnement de la CPU, sauvegardez ces valeurs dans les valeurs initiales du bloc de données.

Vous définissez la valeur d'initialisation pour le paramètre Output sur la variable StartMode.

La valeur d'initialisation est émise au premier appel de Filter_PT2 après le

- changement d'état de fonctionnement de la CPU

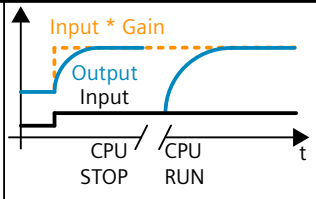
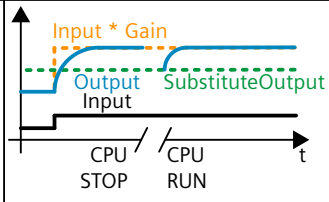
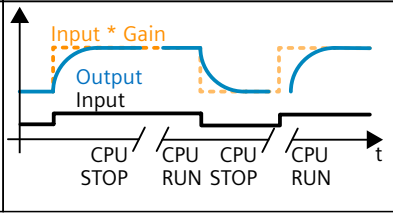
ou

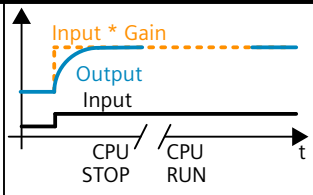
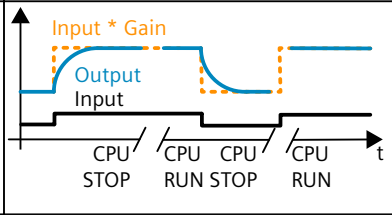
- L'exécution de "Charger les valeurs de départ comme valeurs en cours" (uniquement pour l'option "Toutes les valeurs", et pas pour l'option "Consignes uniquement")

au paramètre Output.

Lors des appels suivants, Filter_PT2 calcule la valeur de réglage, à partir de cette valeur d'initialisation, sur la base de la valeur d'entrée et de la configuration du filtre.

Le tableau suivant présente la dépendance entre la variable StartMode et le paramètre Output. Les valeurs dans la colonne Output sont émises au paramètre Output après le changement d'état de fonctionnement de la CPU :

StartMode	Output	Exemple
0	Valeur du paramètre Input	
1	Valeur du paramètre SsubstituteOutput	
2	Reste inchangé Le paramètre Output est rémanent. Paramètre par défaut Utilisé si StartMode n'est pas dans la plage 0...4	

StartMode	Output	Exemple
3	0.0	
4	Valeur du produit Input * Gain	

Les points suivants s'appliquent en plus pour toutes les valeurs de la variable StartMode :

- La valeur d'initialisation est limitée à la plage de valeurs du type de données REAL. Ce n'est qu'ensuite que la valeur d'initialisation est émise au paramètre Output.
- Si la valeur d'initialisation n'est pas une valeur REAL valide, la valeur de réglage de remplacement est émise au paramètre Output. La valeur de réglage de remplacement est configurée par la variable ErrorMode. La valeur de réglage de remplacement est limitée à la plage de valeurs du type de données REAL avant d'être émise au paramètre Output. Si la valeur de réglage de remplacement n'est pas non plus une valeur REAL valide, 0.0 est émise au paramètre Output. Lors des appels suivants, l'instruction calcule la valeur de réglage à partir de cette valeur de réglage de remplacement.
- La variable StartMode n'est active que lorsque le paramètre Reset = FALSE lors du premier appel de l'instruction et si aucune erreur n'apparaît avec le message d'erreur ErrorBits ≥ 16#0002_0000. Lorsque le paramètre Reset = TRUE, la valeur du paramètre SubstituteOutput est émise au paramètre Output. En présence d'une erreur déclenchant un message d'erreur ErrorBits ≥ 16#0002_0000, la valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output.

Comportement en cas d'erreur

L'instruction Filter_PT2 détecte différentes erreurs pouvant survenir lors du calcul de la valeur de réglage. Le résultat de ce calcul peut être émis en dépit de la présence d'une erreur à la sortie. Si aucun calcul correct de la valeur de réglage n'est possible à cause d'une erreur, une valeur de réglage de remplacement est émise à la sortie.

Vous définissez, pour la variable ErrorMode, la valeur de réglage de remplacement en cas d'erreur rendant impossible tout calcul correct de la valeur de réglage.

Le tableau suivant présente la dépendance entre la valeur de la variable ErrorMode et la valeur de réglage de remplacement que Filter_PT2 émet au paramètre Output :

ErrorMode	Output
0	Valeur du paramètre Input
1	Valeur du paramètre SubstituteOutput
2	La dernière valeur de réglage de filtre valide 0.0, si aucune valeur de réglage de filtre valide n'est disponible Paramètre par défaut Utilisé si ErrorMode n'est pas dans la plage 0...4

ErrorMode	Output
3	0.0
4	Valeur du produit Input * Gain

Les points suivants s'appliquent en plus pour toutes les valeurs de la variable ErrorMode :

- Si la valeur de réglage de remplacement n'a pas de de valeur REAL valide, 0.0 est émis comme valeur de réglage.
- La valeur de réglage de remplacement est limitée à la plage de valeurs du type de données REAL. Ce n'est qu'ensuite que la valeur de réglage de remplacement est émise au paramètre Output.
- La variable ErrorMode n'est active que lorsque le paramètre Reset = FALSE. Lorsque le paramètre Reset = TRUE, la valeur du paramètre SubstituteOutput est émise au paramètre Output.

Le paramètre Error indique si une erreur est actuellement présente. Quand l'erreur a disparu, Error est réglé sur FALSE. Le paramètre ErrorBits indique les erreurs qui se sont produites. ErrorBits est rémanent et n'est réinitialisé que par un front montant au paramètre Reset ou ErrorAck.

10.10.3 Fonctionnement de Filter_PT2

Comportement Reset

Le comportement de Filter_PT2 dépend du paramètre Reset de la manière suivante :

- Lorsque le paramètre Reset =TRUE, la valeur du paramètre SubstituteOutput est émise au paramètre Output.
- Lorsque le paramètre Reset = FALSE, la valeur émise au paramètre Output est calculée par l'algorithme de filtre.
- Si le paramètre Reset passe de FALSE à TRUE, la valeur au paramètre Output devient directement la valeur du paramètre SubstituteOutput. Ce changement peut se faire par à-coups. Le paramètre ErrorBits est également réinitialisé.
- Si le paramètre Reset passe de TRUE à FALSE, l'algorithme de filtre est réglé de manière à ce que le changement se fasse sans à-coups.

Comportement de validation EN/ENO

Si l'une des conditions suivantes est remplie, la sortie de validation ENO est réglée sur FALSE :

- L'entrée de validation EN est réglée sur TRUE et le paramètre Output est défini par une valeur de réglage de remplacement pour les messages d'erreur ErrorBits ≥ 16#0001_0000.
- L'entrée de validation EN est réglée sur FALSE.

Sinon, la sortie de validation ENO est réglée sur TRUE.

Mesure automatique du temps de cycle

Pour le calcul de la valeur de réglage, Filter_PT2 a besoin du temps qui s'est écoulé depuis le dernier appel de Filter_PT2.

Le temps de cycle est automatiquement mesuré par défaut et est émis au niveau de la variable CycleTime.Value à partir du deuxième appel. Filter_PT2 mesure le temps de cycle à chaque appel de l'instruction et est donc utilisable dans des cycles d'appel non équidistants, par ex. dans l'OB1.

Notez que des appels conditionnels, les points d'arrêt actifs ou le chargement d'instantanés comme valeurs en cours rallongent le temps de cycle en cas de mesure automatique du temps de cycle. Un temps de cycle trop élevé est détecté comme erreur avec le message d'erreur ErrorBits = 16#0008_0000.

Si la mesure du temps de cycle ne fournit aucun résultat valide, Filter_PT2 calcule la valeur de réglage actuelle à l'aide du dernier temps de cycle valide. De plus, Filter_PT2 émet un message d'erreur au paramètre ErrorBits.

Si vous désactivez la mesure automatique du temps de cycle en définissant la variable CycleTime.EnableMeasurement = FALSE, vous devez spécifier le temps de cycle manuellement au niveau de la variable CycleTime.Value. Filter_PT2 vérifie la validité de la variable CycleTime.Value à chaque appel.

Mesure automatique du temps de cycle avec points d'arrêt

Si des points d'arrêt sont actifs entre deux appels de Filter_PT2, la mesure automatique du temps de cycle donne le temps réel qui s'est écoulé entre deux appels. Si un point d'arrêt est actif, la CPU se trouve à l'état de fonctionnement ATTENTE.

REMARQUE

Les points d'arrêt actifs allongent l'intervalle de temps entre deux appels de Filter_PT2.

Avec un intervalle de temps plus long entre deux appels, la modification de la valeur de réglage augmente également au paramètre Output.

Il est en outre possible que des intervalles plus longs violent la condition TimeConstant ≥ CycleTime.Value/2 et qu'une erreur avec message d'erreur ErrorBits = 16#0008_0000 soit alors détectée.

Si vous n'avez pas besoin du calcul de la valeur de réglage sur la base du temps réel avec des points d'arrêt actifs, exécutez alors les étapes suivantes :

- Désactivez la mesure automatique du temps de cycle en définissant la variable CycleTime.EnableMeasurement = FALSE.
- Saisissez le temps de cycle manuellement au niveau de la variable CycleTime.Value.

10.10.4 Paramètre d'entrée Filter_PT2

Paramètre	Type de données	Valeur par défaut	Description
Input	REAL	0.0	Valeur d'entrée
SubstituteOutput	REAL	0.0	SubstituteOutput est utilisé comme valeur de réglage de remplacement si • Reset = TRUE ou • si aucun calcul correct de la valeur de réglage n'est possible à cause d'une erreur avec le message d'erreur ErrorBits ≥ 16#0001_0000 et que ErrorMode est configuré à la valeur 1.
ErrorAck	BOOL	FALSE	Supprime les messages d'erreur • Front FALSE -> TRUE ErrorBits est réinitialisé.
Reset	BOOL	FALSE	Exécute un redémarrage de l'instruction • Front FALSE -> TRUE ErrorBits est réinitialisé. • Tant que Reset est réglé sur TRUE, la valeur de réglage de remplacement SubstituteOutput est émise à la sortie. • Tant que Reset est réglé sur FALSE, le calcul de la valeur de réglage est exécuté.

10.10.5 Paramètre de sortie Filter_PT2

Paramètre	Type de données	Valeur par défaut	Description
Output	REAL	0.0	Valeur de réglage La valeur de réglage est rémanente.
ErrorBits	DWORD	DW#16#0	Le paramètre ErrorBits (Page 550) signale quels sont les messages d'erreur présents. ErrorBits est rémanent et réinitialisé à Reset ou ErrorAck en cas de front montant.
Error	BOOL	FALSE	Le réglage de Error sur TRUE indique la présence d'au moins une erreur actuellement.

10.10.6 Variables statiques Filter_PT2

Variable	Type de données	Valeur par défaut	Description
Gain	REAL	1.0	Gain proportionnel
TimeConstant	REAL	25.0	Constante de temps en secondes Plage de valeurs autorisée : TimeConstant ≥ CycleTime.Value/2
Damping	REAL	1.0	Atténuation Plage de valeurs autorisée : Damping > 0.0
ErrorMode	INT	2	Sélection de la valeur de réglage de remplacement en cas d'erreur • 0 = Input • 1 = SubstituteOutput • 2 = dernière valeur de réglage de filtre valide • 3 = 0.0 • 4 = Input * Gain Plage de valeurs autorisée : 0 à 4
StartMode	INT	2	Sélection de la valeur de réglage pour le premier appel de l'instruction • 0 = Input • 1 = SubstituteOutput • 2 = dernière valeur de réglage • 3 = 0.0 • 4 = Input * Gain Plage de valeurs autorisée : 0 à 4
CycleTime	AuxFct_CycleTime	-	Données du temps de cycle
CycleTime.Value	REAL	0.1	Temps de cycle en secondes (intervalle entre 2 appels) Plage de valeurs autorisée : CycleTime.Value > 0.0
CycleTime.EnableMeasurement	BOOL	TRUE	Mesure automatique du temps de cycle • FALSE = désactivée • TRUE = activée

10.10.7 Paramètre ErrorBits

En présence de plusieurs erreurs simultanées, les valeurs des ErrorBits s'affichent comme addition binaire. L'affichage de ErrorBits = 16#0000_0003, par ex., indique la présence simultanée des erreurs 16#0000_0001 et 16#0000_0002.

Avec Filter_PT2, les erreurs émises au paramètre ErrorBits sont réparties en deux catégories :

- les erreurs déclenchant des messages d'erreur ErrorBits < 16#0001_0000
- les erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000

Erreurs déclenchant des messages d'erreur ErrorBits < 16#0001_0000

En présence d'une ou de plusieurs erreurs déclenchant des messages d'erreur ErrorBits < 16#0001_0000, Filter_PT2 réagit de la manière suivante :

- La valeur de réglage est déterminée comme suit en dépit de cette erreur :
  - Si Reset = FALSE, , calcul de la valeur de réglage par l'algorithme de filtre
  - Si Reset = TRUE, sortie de SubstituteOutput
- Le paramètre de sortie Error est mis à 1.
- La sortie de validation ENO n'est pas modifiée.

Le paramètre de sortie Error est supprimé dès que l'erreur a disparu.

ErrorBits (DW#16#...)	Description
0000_0000	Pas d'erreur.
0000_0001	**Cause d'erreur et réaction en cas d'erreur :** Le paramètre Output a été limité à -3.402823e+38 ou +3.402823e+38. **Solution :** Si la valeur déterminée par la fonction de filtre est émise à la sortie (Reset = FALSE et ErrorBits < 16#0001_0000), vérifiez les variables suivantes utilisées dans le calcul de filtre : • Input • Gain • TimeConstant • Damping • CycleTime.Value Si ErrorBits ≥ 16#0001_0000 et Reset = FALSE, la valeur de réglage de remplacement est limitée à sa sortie. Vérifiez ensuite les paramètres suivants en fonction de la valeur paramétrée à la variable ErrorMode : • Input • SubstituteOutput • Le produit de Input et Gain Si Reset = TRUE, vérifiez le paramètre SubstituteOutput.
0000_0002	**Cause d'erreur :** La mesure du temps de cycle ne donne pas de valeur valide, tandis que le calcul de la valeur de réglage est exécuté (Reset = FALSE). **Réaction en cas d'erreur :** Si une valeur valide du temps de cycle a déjà été mesurée, Filter_PT2 calcule la valeur de réglage à partir de cette dernière valeur de la variable CycleTime.Value. Si aucune valeur valide du temps de cycle n'a été mesurée préalablement, Filter_PT2 continue d'émettre la valeur de réglage configurée avec la variable StartMode au paramètre Output.

Erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000

En présence d'une ou de plusieurs erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000, Filter_PT2 réagit de la manière suivante :

- La valeur de réglage ne peut pas être déterminée comme prévu. À la place, c'est la valeur de réglage de remplacement qui est émise.
- Le paramètre de sortie Error est mis à 1.
- La sortie de validation ENO est réglée sur FALSE.

Dès que les erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000 ont disparu, Filter_PT2 réagit de la manière suivante :

- La valeur de réglage est déterminée comme suit :
  - Calcul de la valeur de réglage par l'algorithme de filtre si Reset = FALSE
  - Sortie de SubstituteOutput si Reset = TRUE
- La sortie de validation ENO est réglée sur TRUE.

Le paramètre de sortie Error est supprimé dès que l'erreur a disparu.

<table>
<tr><th>ErrorBits (DW#16#...)</th><th colspan="3">Description</th></tr>
<tr><td>0001_0000</td><td colspan="3">Cause d'erreur :
Le paramètre SubstituteOutput ou une autre variable qui est utilisée comme valeur de réglage n'a pas de valeur REAL valide.
Réaction en cas d'erreur :
La sortie est réglée sur 0.0.
Solution :
Vérifiez que la variable utilisée comme valeur de réglage est une valeur REAL valide (≠NaN p. ex. 16#7FFF_FFFF). La variable utilisée comme valeur de réglage dépend de Reset et de ErrorMode :</td></tr>
<tr><td></td><td>Reset</td><td>ErrorMode</td><td>Valeur de réglage</td></tr>
<tr><td></td><td>FALSE</td><td>0</td><td>Input</td></tr>
<tr><td></td><td>FALSE</td><td>1</td><td>SubstituteOutput</td></tr>
<tr><td></td><td>FALSE</td><td>4</td><td>Produit de Input et Gain</td></tr>
<tr><td></td><td>TRUE</td><td>-</td><td>SubstituteOutput</td></tr>
<tr><td>0002_0000</td><td colspan="3">Cause d'erreur :
Le paramètre Input n'a pas de valeur REAL valide, tandis que le calcul de la valeur de réglage est exécuté (Reset = FALSE).
Réaction en cas d'erreur :
La valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output.
0.0 est émis comme valeur de réglage si ErrorMode = 0.
Solution :
Vérifiez que le paramètre Input est une valeur REAL valide (≠NaN p. ex. 16#7FFF_FFFF).</td></tr>
<tr><td>0004_0000</td><td colspan="3">Cause d'erreur :
Le calcul de la valeur de réglage donne une valeur REAL non valide pour le paramètre Output.
Réaction en cas d'erreur :
La valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output.
Solution :
Vérifiez toutes les variables utilisées pour le calcul de la valeur de réglage :
• Input
• Gain
• TimeConstant
• Damping
• CycleTime.Value</td></tr>
</table>

ErrorBits (DW#16#...)	Description
	Les valeurs de ces variables sont valides. Le calcul de la valeur de réglage dans cette combinaison de variables échoue.
0008_0000	**Cause d'erreur :** La variable Gain, TimeConstant ou Damping n'a pas de valeur valide, tandis que le calcul de la valeur de réglage est exécuté (Reset = FALSE). **Réaction en cas d'erreur :** La valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output. **Solution :** Vérifiez que les conditions suivantes pour les valeurs des variables Gain, TimeConstant et Damping sont remplies : • -3.402823e+38 ≤ Gain ≤ 3.402823e+38 • CycleTime.Value/2 ≤ TimeConstant ≤ 3.402823e+38 • 0.0 < Damping ≤ 3.402823e+38 • Les valeurs sont des valeurs REAL valides (≠ NaN par ex. 16#7FFF_FFFF) **Informations complémentaires :** Tenez compte du fait que la condition CycleTime.Value/2 ≤ TimeConstant n'est pas respectée dans les scénarios suivants : • L'intervalle entre deux appels de Filter_PT2 est plus long que 2 * TimeConstant, p. ex. en raison d'appels conditionnels dans l'exécution du programme ou de points d'arrêt actifs. • Un instantané du DB d'instance Filter_PT2 est chargé comme valeurs actuelles dans la CPU et la création de l'instantané remonte à plus de 2 * TimeConstant. Dans ces scénarios, une erreur avec message d'erreur ErrorBits = 16#0008_0000 est détectée en cas de mesure automatique du temps de cycle.
0020_0000	**Cause d'erreur :** La variable (configurée avec StartMode) pour l'initialisation du paramètre Output lors du premier appel de l'instruction n'a pas de valeur REAL valide. **Réaction en cas d'erreur :** Lors du premier appel de l'instruction, la valeur de réglage de remplacement qui est configurée au niveau de la variable ErrorMode est émise au paramètre Output. Lors des appels suivants, Filter_PT2 calcule la valeur de réglage à partir de cette valeur de réglage de remplacement. **Solution :** Vérifiez que la variable pour l'initialisation du paramètre Output est une valeur REAL valide (≠ NaN p. ex. 16#7FFF_FFFF). Si Reset = FALSE, l'initialisation n'est active lors du premier appel de l'instruction qu'après le passage de la CPU de l'état de fonctionnement STOP à l'état RUN. La variable utilisée pour l'initialisation du paramètre Output dépend de StartMode : • StartMode = 1: SubstituteOutput • StartMode = 2: Output • Produit StartMode = 4: de Input et Gain
0040_0000	**Cause d'erreur :** La variable CycleTime.Value n'a pas de valeur valide, tandis que le calcul de la valeur de réglage est exécuté (Reset = FALSE). **Réaction en cas d'erreur :** La valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output. **Solution :** Vérifiez que les conditions suivantes sont remplies : • 0.0 < CycleTime.Value ≤ 3.402823e+38 • CycleTime.Value est une valeur REAL valide (≠ NaN, p. ex. 16#7FFF_FFFF) **Informations complémentaires :** Pour un calcul automatique de la valeur de la variable CycleTime.Value, réglez la variable CycleTime.EnableMeasurement sur TRUE.

10.11 Filter_DT1

10.11.1 Compatibilité avec CPU et FW

Le tableau suivant montre les CPU et les versions de Filter_DT1 compatibles.

CPU	FW	Filter_DT1
S7-1200	à partir de V4.2	V1.0
CPU basées sur S7-1500	à partir de V2.0	V1.0

10.11.2 Description Filter_DT1

Description

L'instruction Filter_DT1 est un élément de dérivation proportionnel avec un retard de premier ordre, également appelé élément DT1.

Vous pouvez utiliser Filter_DT1 aux fins suivantes :

- Filtre passe-haut pour atténuer les contenus à basse fréquence dans un signal.
- Élément de dérivation pour calculer la dérivation d'un signal, par exemple la vitesse depuis des valeurs de positionnement.
- Action anticipatrice pour atténuer l'effet des perturbations mesurables sur le processus.

Vous pouvez régler les paramètres de filtre suivants :

- Temps de dérivation (Td)
- Constante de temps de retard (Lag)

REMARQUE

Différences entre un élément DT1 à action continue et Filter_DT1

Filter_DT1 Étant exécuté dans un programme API, Filter_DT1 est une mise en œuvre à temps discret d'un élément DT1. Les systèmes à temps discret n'ont pas les mêmes propriétés que le modèle à action continue correspondant. Les systèmes à temps discret peuvent cependant reproduire correctement un système à action continue en fonction du temps de cycle : Plus le temps de cycle est petit et constant, plus l'écart entre les propriétés de Filter_DT1 et les propriétés de l'élément DT1 à action continue est petit. Les propriétés d'un élément DT1 à action continue sont la fonction de transfert, le comportement temporel et la réponse en fréquence décrits ci-dessous.

Pour une reproduction correcte de la réponse en fréquence, il est recommandé de régler un temps de cycle maximal d'un dixième de la durée de période la plus courte des composants de signal d'entrée. Par exemple, un signal avec éléments de fréquence de jusqu'à 50 Hz a une durée de période la plus courte de 20 ms. Pour obtenir reproduction correcte de la réponse en fréquence, il est par exemple recommandé de régler un temps de cycle maximal de 2 ms.

Fonction de transmission d'un élément DT1

La formule suivante indique la fonction de transfert d'un élément DT1, avec s équivalent à l'opérateur Laplace :

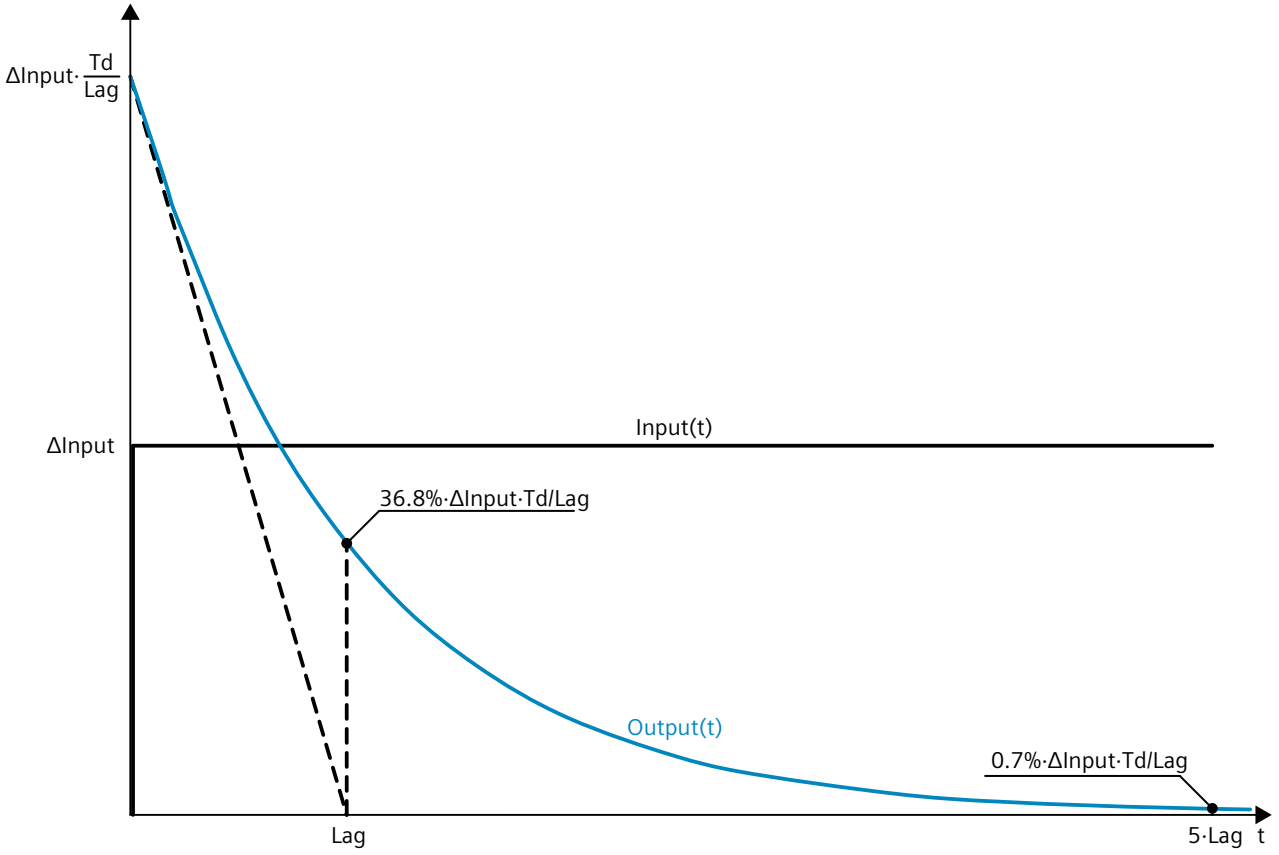
$$G(s) = \frac{\text{Output(s)}}{\text{Input(s)}} = \frac{\text{Td} \cdot s}{1 + \text{Lag} \cdot s}$$

Comportement temporel d'un élément DT1

La réponse indicielle est la réaction de la valeur de réglage à un saut de la valeur d'entrée. La réponse indicielle pour un saut de la valeur d'entrée de 0 à ΔInput peut être calculée avec la formule suivante :

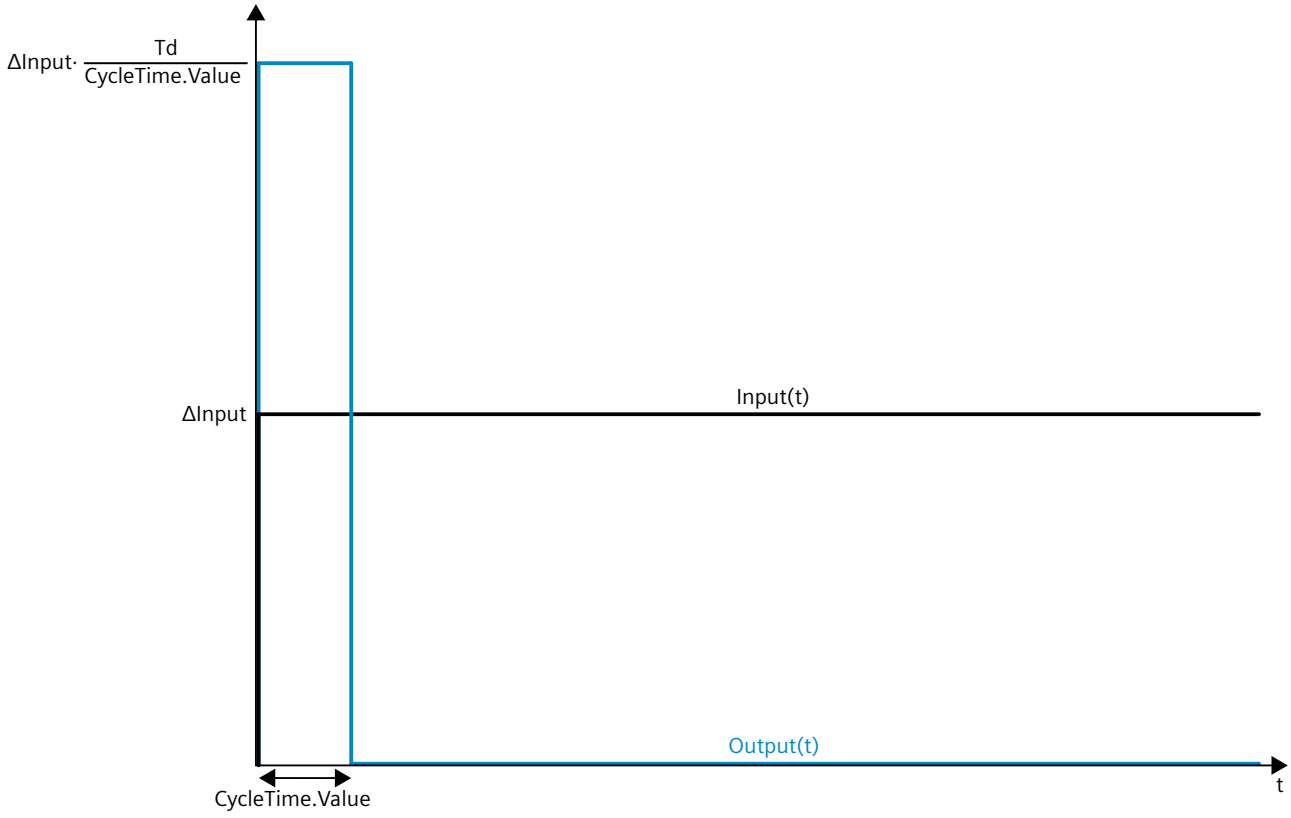
$$\text{Output(t)} = \Delta\text{Input} \cdot \frac{\text{Td}}{\text{Lag}} \cdot e^{-\frac{t}{\text{Lag}}}$$

La figure ci-dessous montre la réponse indicielle d'un élément PT1 :



Pour désactiver le retard, réglez le paramètre Lag à la valeur minimale CycleTime.Value / 2. Les modifications de la valeur d'entrée sont dans ce cas multipliées par Td / CycleTime.Value et émises au paramètre Output. Après un cycle, la valeur de réglage est 0.0.

La figure ci-dessous montre la réponse indicielle pour le cas Lag = CycleTime.Value / 2 :



Réponse en fréquence d'un élément DT1

La réponse en fréquence d'un élément de transmission est décrite par la réponse d'amplitude et la réponse de phase.

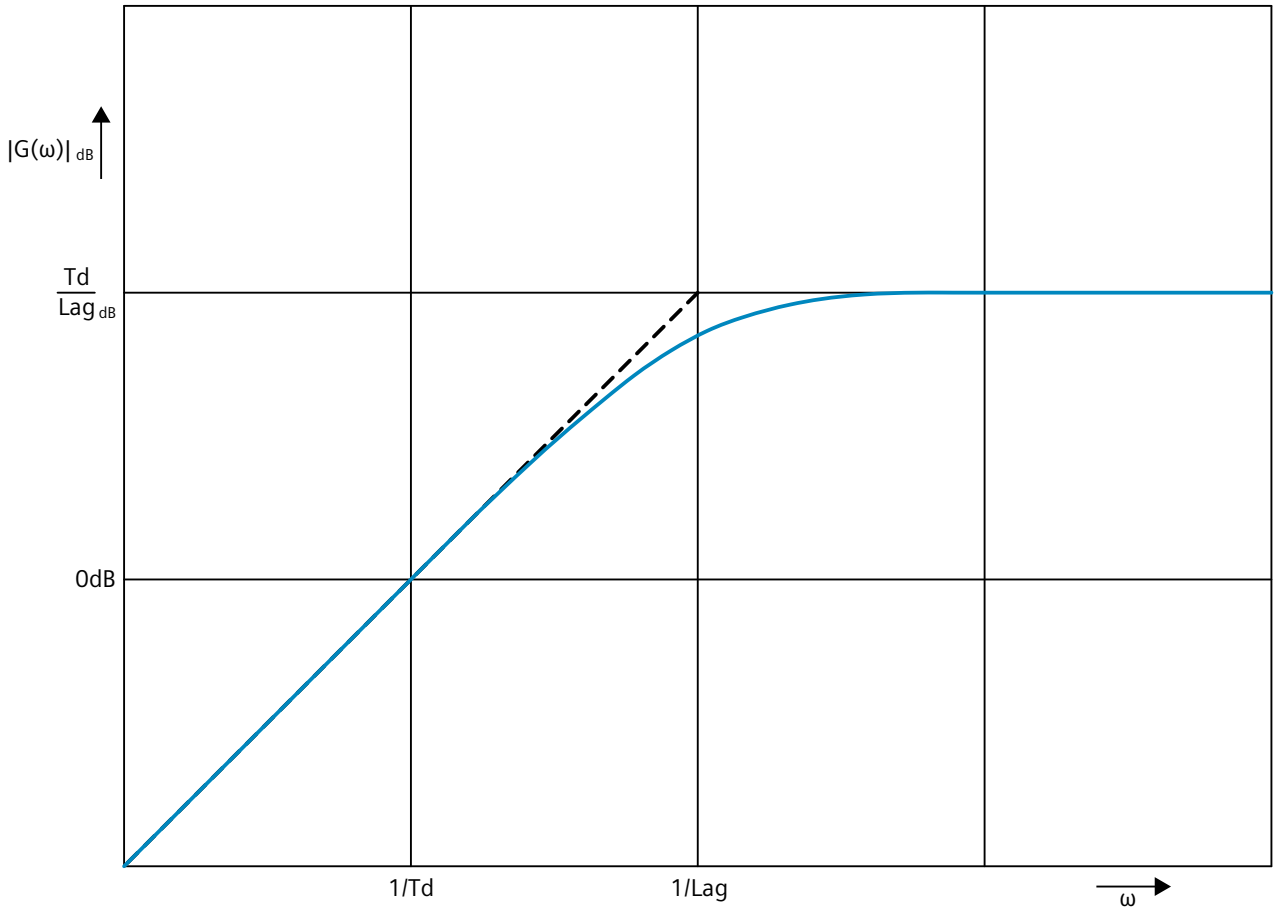
La réponse d'amplitude décrit l'amplification d'un signal par l'élément de transmission en fonction de la pulsation du signal.

L'équation suivante décrit la réponse d'amplitude d'un élément DT1 :

$$|G(\omega)| = \frac{\omega \cdot Td}{\sqrt{1 + (\omega \cdot Lag)^2}}$$

|G(ω)|	Amplification d'un signal en fonction de la pulsation
ω	Pulsation

La figure suivante montre la réponse d'amplitude d'un élément DT1 :

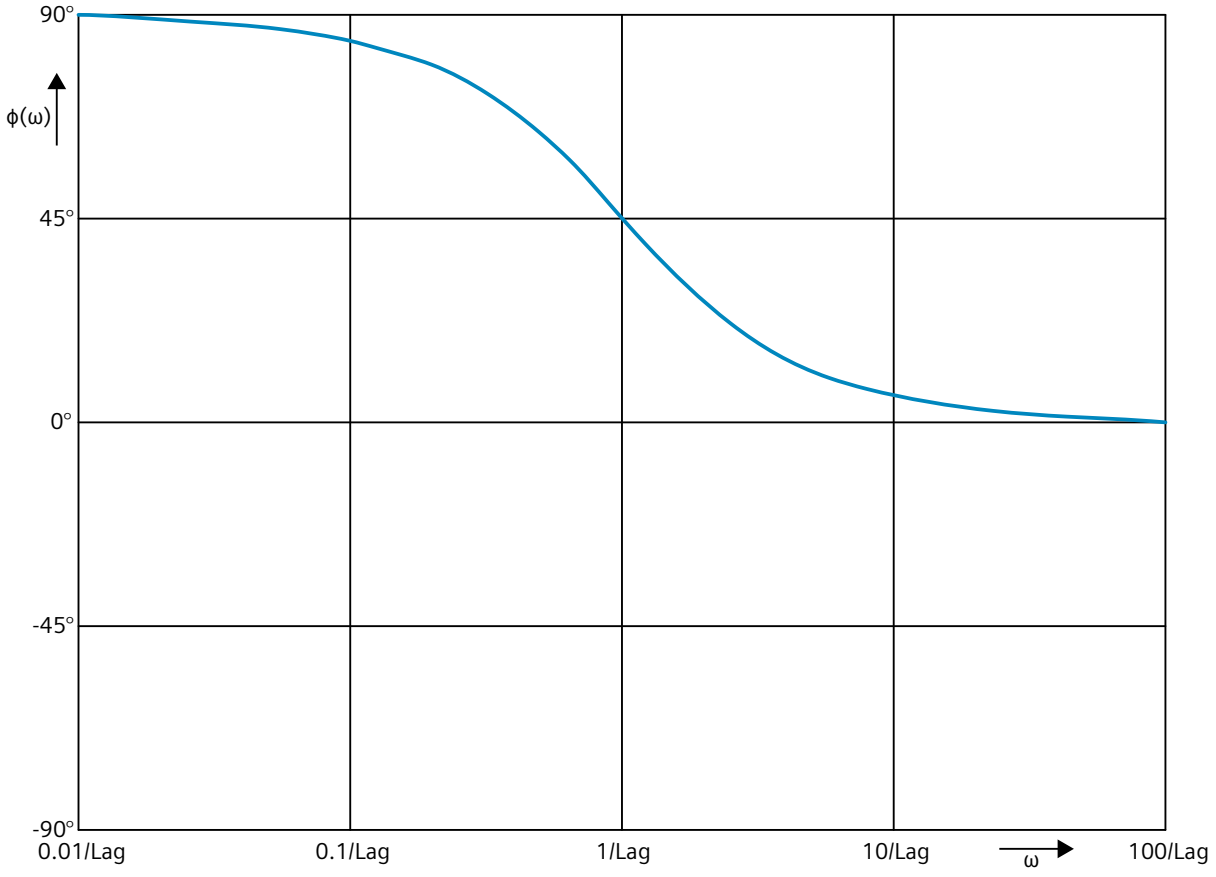


La réponse de phase décrit le décalage de phases d'un signal par l'élément de transmission en fonction de la pulsation du signal.
L'équation suivante décrit la réponse de phase d'un élément DT1 :

$$\varphi(\omega) = \tan^{-1}(\omega \cdot \mathrm{Lag})$$

φ(ω) Décalage de phases en fonction de la pulsation
ω Pulsation

La figure suivante montre la réponse de phase d'un élément DT1 :



Appel

Dans un OB ou une FC, l'instruction Filter_DT1 est appelée comme DB d'instance unique. Dans un FB, l'instruction Filter_DT1 peut être appelée aussi bien comme DB d'instance unique que comme DB de multi-instance ou DB d'instance de paramètre.
Aucun objet technologique n'est créé lors de l'appel de l'instruction. Vous ne disposez pas d'une interface de paramétrage et de mise en service. Vous paramétrez Filter_DT1 directement via le DB d'instance et vous mettez en service Filter_DT1 via une table de visualisation du programme utilisateur dans la CPU ou HMI.

Démarrage

Les variables de la plage statique de Filter_DT1 ne sont pas rémanentes. Ces variables sont initialisées à l'aide des valeurs initiales après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN. Si vous modifiez les valeurs actuelles en mode en ligne et que ces valeurs doivent être conservées après le changement d'état de fonctionnement de la CPU, sauvegardez ces valeurs dans les valeurs initiales du bloc de données.

Vous définissez la valeur d'initialisation pour le paramètre Output sur la variable StartMode.

La valeur d'initialisation est émise au premier appel de Filter_DT1 après le

- changement d'état de fonctionnement de la CPU

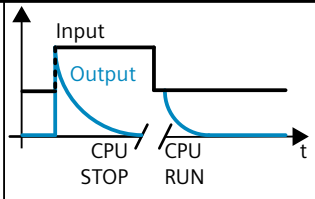
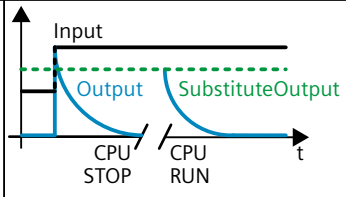
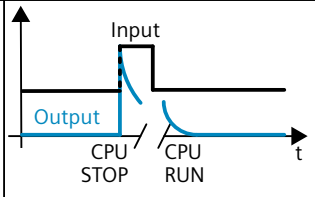
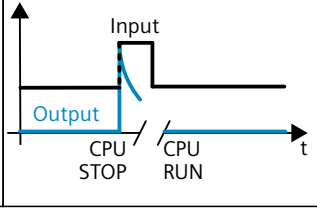
ou

- l'exécution de "Charger les valeurs de départ comme valeur en cours" (uniquement pour l'option "Toutes les valeurs", et pas pour l'option "Consignes uniquement")

au paramètre Output.

Lors des appels suivants, Filter_DT1 calcule la valeur de réglage, à partir de cette valeur d'initialisation, sur la base de la valeur d'entrée et de la configuration du filtre.

Le tableau suivant présente la dépendance entre la variable StartMode et le paramètre Output. Les valeurs dans la colonne Output sont émises au paramètre Output après le changement d'état de fonctionnement de la CPU :

StartMode	Output	Exemple
0	Valeur du paramètre Input	
1	Valeur du paramètre SubstituteOutput	
2	Reste inchangé Le paramètre Output est rémanent. Paramètre par défaut Utilisé si StartMode n'est pas dans la plage 0...3	
3	0.0	

Les points suivants s'appliquent en plus pour toutes les valeurs de la variable StartMode :

- La valeur d'initialisation est limitée à la plage de valeurs du type de données REAL. Ce n'est qu'ensuite que la valeur d'initialisation est émise au paramètre Output.
- Si la valeur d'initialisation n'est pas une valeur REALvalide, la valeur de réglage de remplacement est émise au paramètre Output. La valeur de réglage de remplacement est configurée par la variable ErrorMode. La valeur de réglage de remplacement est limitée à la plage de valeurs du type de données REAL avant d'être émise au paramètre Output. Si la valeur de réglage de remplacement n'est pas non plus une valeur REAL valide, 0.0 est émise au paramètre Output. Lors des appels suivants, l'instruction calcule la valeur de réglage à partir de cette valeur de réglage de remplacement.
- La variable StartMode n'est active que lorsque le paramètre Reset = FALSE lors du premier appel de l'instruction et si aucune erreur n'apparaît avec le message d'erreur ErrorBits ≥ 16#0002_0000. Lorsque le paramètre Reset = TRUE, la valeur du paramètre SubstituteOutput est émise au paramètre Output. En présence d'une erreur déclenchant un message d'erreur ErrorBits ≥ 16#0002_0000, la valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output.

Comportement en cas d'erreur

L'instruction Filter_DT1 détecte différentes erreurs pouvant survenir lors du calcul de la valeur de réglage. Le résultat de ce calcul peut être émis en dépit de la présence d'une erreur à la sortie. Si aucun calcul correct de la valeur de réglage n'est possible à cause d'une erreur, une valeur de réglage de remplacement est émise à la sortie.

Vous définissez, pour la variable ErrorMode, la valeur de réglage de remplacement en cas d'erreur rendant impossible tout calcul correct de la valeur de réglage.

Le tableau suivant présente la dépendance entre la valeur de la variable ErrorMode et la valeur de réglage de remplacement que Filter_DT1 émet au paramètre Output :

ErrorMode	Output
0	Valeur du paramètre Input
1	Valeur du paramètre SubstituteOutput
2	Dernière valeur de réglage de filtre valide 0.0, si aucune valeur de réglage de filtre valide n'est disponible Paramètre par défaut Utilisé si ErrorMode n'est pas dans la plage 0...3
3	0.0

Les points suivants s'appliquent en plus pour toutes les valeurs de la variable ErrorMode :

- Si la valeur de réglage de remplacement n'a pas de de valeur REAL valide, 0.0 est émis comme valeur de réglage.
- La valeur de réglage de remplacement est limitée à la plage de valeurs du type de données REAL. Ce n'est qu'ensuite que la valeur de réglage de remplacement est émise au paramètre Output.
- La variable ErrorMode n'est active que lorsque le paramètre Reset = FALSE. Lorsque le paramètre Reset = TRUE, la valeur du paramètre SubstituteOutput est émise au paramètre Output.

Le paramètre Error indique si une erreur est actuellement présente. Quand l'erreur a disparu, Error est réglé sur FALSE. Le paramètre ErrorBits indique les erreurs qui se sont produites.

ErrorBits est rémanent et n'est réinitialisé que par un front montant au paramètre Reset ou ErrorAck.

10.11.3 Fonctionnement de Filter_DT1

Comportement Reset

Le comportement de Filter_DT1 dépend du paramètre Reset de la manière suivante :

- Lorsque le paramètre Reset =TRUE, la valeur du paramètre SubstituteOutput est émise au paramètre Output.
- Lorsque le paramètre Reset = FALSE, la valeur émise au paramètre Output est calculée par l'algorithme de filtre.
- Si le paramètre Reset passe de FALSE à TRUE, la valeur au paramètre Output devient directement la valeur du paramètre SubstituteOutput. Ce changement peut se faire par à-coups. Le paramètre ErrorBits est également réinitialisé.
- Si le paramètre Reset passe de TRUE à FALSE, l'algorithme de filtre est réglé de manière à ce que le changement se fasse sans à-coups.

Comportement de validation EN/ENO

Si l'une des conditions suivantes est remplie, la sortie de validation ENO est réglée sur FALSE :

- L'entrée de validation EN est réglée sur TRUE et le paramètre Output est défini par une valeur de réglage de remplacement pour les messages d'erreur ErrorBits ≥ 16#0001_0000
- L'entrée de validation EN est réglée sur FALSE.

Sinon, la sortie de validation ENO est réglée sur TRUE.

Mesure automatique du temps de cycle

Pour le calcul de la valeur de réglage, Filter_DT1 a besoin du temps qui s'est écoulé depuis le dernier appel de Filter_DT1.

Le temps de cycle est automatiquement mesuré par défaut et est émis au niveau de la variable CycleTime.Value à partir du deuxième appel. Filter_DT1 mesure le temps de cycle à chaque appel de l'instruction et est donc utilisable dans des cycles d'appel non équidistants, par ex. dans l'OB1.

Notez que des appels conditionnels, les points d'arrêt actifs ou le chargement d'instantanés comme valeurs en cours rallongent le temps de cycle en cas de mesure automatique du temps de cycle. Un temps de cycle trop élevé est détecté comme erreur avec le message d'erreur ErrorBits = 16#0008_0000.

Si la mesure du temps de cycle ne fournit aucun résultat valide, Filter_DT1 calcule la valeur de réglage actuelle à l'aide du dernier temps de cycle valide. De plus, Filter_DT1 émet un message d'erreur au paramètre ErrorBits.

Si vous désactivez la mesure automatique du temps de cycle en définissant la variable CycleTime.EnableMeasurement = FALSE, vous devez spécifier le temps de cycle manuellement au niveau de la variable CycleTime.Value. Filter_DT1 vérifie la validité de la variable CycleTime.Value à chaque appel.

Mesure automatique du temps de cycle avec points d'arrêt

Si des points d'arrêt sont actifs entre deux appels de Filter_DT1, la mesure automatique du temps de cycle donne le temps réel qui s'est écoulé entre deux appels. Si un point d'arrêt est actif, la CPU se trouve à l'état de fonctionnement ATTENTE.

REMARQUE

Les points d'arrêt actifs allongent l'intervalle de temps entre deux appels de Filter_DT1.

Avec un intervalle de temps plus long entre deux appels, la modification de la valeur de réglage augmente également au paramètre Output.

Il est en outre possible que des intervalles plus longs violent les conditions Lag ≥ CycleTime.Value/2 ou Td ≥ CycleTime.Value. Une erreur est alors ensuite détectée avec le message d'erreur ErrorBits = 16#0008_0000 .

Si vous n'avez pas besoin du calcul de la valeur de réglage sur la base du temps réel avec des points d'arrêt actifs, exécutez alors les étapes suivantes :

- Désactivez la mesure automatique du temps de cycle en définissant la variable CycleTime.EnableMeasurement = FALSE.
- Saisissez le temps de cycle manuellement au niveau de la variable CycleTime.Value.

10.11.4 Paramètre d'entrée Filter_DT1

Paramètre	Type de données	Valeur par défaut	Description
Input	REAL	0.0	Valeur d'entrée
SubstituteOutput	REAL	0.0	SubstituteOutput est utilisé comme valeur de réglage de remplacement si • Reset = TRUE ou • si aucun calcul correct de la valeur de réglage n'est possible à cause d'une erreur avec le message d'erreur ErrorBits ≥ 16#0001_0000 et que ErrorMode est configuré à la valeur 1.
ErrorAck	BOOL	FALSE	Supprime les messages d'erreur • Front FALSE -> TRUE ErrorBits est réinitialisé.
Reset	BOOL	FALSE	Exécute un redémarrage de l'instruction • Front FALSE -> TRUE ErrorBits est réinitialisé. • Tant que Reset est réglé sur TRUE, la valeur de réglage de remplacement SubstituteOutput est émise à la sortie. • Tant que Reset est réglé sur FALSE, le calcul de la valeur de réglage est exécuté.

10.11.5 Paramètre de sortie Filter_DT1

Paramètre	Type de données	Valeur par défaut	Description
Output	REAL	0.0	Valeur de réglage La valeur de réglage est rémanente.
ErrorBits	DWORD	DW#16#0	Le paramètre ErrorBits (Page 563) signale quels sont les messages d'erreur présents. ErrorBits est rémanent et réinitialisé à Reset ou ErrorAck en cas de front montant.
Error	BOOL	FALSE	Le réglage de Error sur TRUE indique la présence d'au moins une erreur actuellement.

10.11.6 Variables statiques Filter_DT1

Variable	Type de données	Valeur par défaut	Description
Td	REAL	25.0	Temps de dérivation en secondes Plage de valeurs autorisée : Td ≥ CycleTime.Value
Lag	REAL	5.0	Constante de temps de retard en secondes Plage de valeurs autorisée : Lag ≥ CycleTime.Value/2
ErrorMode	INT	2	Sélection de la valeur de réglage de remplacement en cas d'erreur • 0 = Input • 1 = SubstituteOutput • 2 = dernière valeur de réglage de filtre valide • 3 = 0.0 Plage de valeurs autorisée : 0 à 3
StartMode	INT	2	Sélection de la valeur de réglage pour le premier appel de l'instruction • 0 = Input • 1 = SubstituteOutput • 2 = dernière valeur de réglage • 3 = 0.0 Plage de valeurs autorisée : 0 à 3
CycleTime	AuxFct_CycleTime	-	Données du temps de cycle
CycleTime.Value	REAL	0.1	Temps de cycle en secondes (intervalle entre 2 appels) Plage de valeurs autorisée : CycleTime.Value > 0.0
CycleTime.EnableMeasurement	BOOL	TRUE	Mesure automatique du temps de cycle • FALSE = désactivée • TRUE = activée

10.11.7 Paramètre ErrorBits

En présence de plusieurs erreurs simultanées, les valeurs des ErrorBits s'affichent comme addition binaire. L'affichage de ErrorBits = 16#0000_0003, par ex., indique la présence simultanée des erreurs 16#0000_0001 et 16#0000_0002.

Avec Filter_DT1, les erreurs émises au paramètre ErrorBits sont réparties en deux catégories :

- les erreurs déclenchant des messages d'erreur ErrorBits < 16#0001_0000
- les erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000

Erreurs déclenchant des messages d'erreur ErrorBits < 16#0001_0000

En présence d'une ou de plusieurs erreurs déclenchant des messages d'erreur ErrorBits < 16#0001_0000, Filter_DT1 réagit de la manière suivante :

- La valeur de réglage est déterminée comme suit en dépit de cette erreur :
  - Calcul de la valeur de réglage par l'algorithme de filtre si Reset = FALSE
  - Sortie de SubstituteOutput si Reset = TRUE
- Le paramètre de sortie Error est mis à 1.
- La sortie de validation ENO n'est pas modifiée.

Le paramètre de sortie Error est supprimé dès que l'erreur a disparu.

ErrorBits (DW#16#...)	**Description**
0000_0000	Pas d'erreur.
0000_0001	**Cause d'erreur et réaction en cas d'erreur :** Le paramètre Output a été limité à -3.402823e+38 ou +3.402823e+38. **Solution :** Si la valeur déterminée par la fonction de filtre est émise à la sortie (Reset = FALSE et ErrorBits < 16#0001_0000), vérifiez les variables suivantes utilisées dans le calcul de filtre : • Input • Td • Lag • CycleTime.Value Si ErrorBits≥ 16#0001_0000 et Reset = FALSE, la valeur de réglage de remplacement est limitée à sa sortie. Vérifiez ensuite les paramètres suivants en fonction de la valeur paramétrée à la variable ErrorMode: • Input • SubstituteOutput Si Reset = TRUE, vérifiez le paramètre SubstituteOutput.
0000_0002	**Cause d'erreur :** La mesure du temps de cycle ne donne pas de valeur valide, tandis que le calcul de la valeur de réglage est exécuté (Reset = FALSE). **Réaction en cas d'erreur :** Si une valeur valide du temps de cycle a déjà été mesurée, Filter_DT1 calcule la valeur de réglage à partir de cette dernière valeur de la variable CycleTime.Value. Si aucune valeur valide du temps de cycle n'a été mesurée préalablement, Filter_DT1 continue d'émettre la valeur de réglage configurée avec la variable StartMode au paramètre Output.

Erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000

En présence d'une ou de plusieurs erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000, Filter_DT1 réagit de la manière suivante :

- La valeur de réglage ne peut pas être déterminée comme prévu. À la place, c'est la valeur de réglage de remplacement qui est émise.
- Le paramètre de sortie Error est mis à 1.
- La sortie de validation ENO est réglée sur FALSE.

Dès que les erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000 ont disparu, Filter_DT1 réagit de la manière suivante :

- La valeur de réglage est déterminée comme suit :
  - Calcul de la valeur de réglage par l'algorithme de filtre si Reset = FALSE
  - Sortie de SubstituteOutput si Reset = TRUE
- La sortie de validation ENO est réglée sur TRUE.

Le paramètre de sortie Error est supprimé dès que l'erreur a disparu.

<table>
<tr><th>ErrorBits (DW#16#...)</th><th>Description</th></tr>
<tr><td>0001_0000</td><td>Cause d'erreur :
Le paramètre SubstituteOutput ou une autre variable qui est utilisée comme valeur de réglage n'a pas de valeur REAL valide.
Réaction en cas d'erreur :
La sortie est réglée sur 0.0.
Solution :
Vérifiez que la variable utilisée comme valeur de réglage est une valeur REAL valide (≠NaN p. ex. 16#7FFF_FFFF). La variable utilisée comme valeur de réglage dépend de Reset et de ErrorMode :
<table>
<tr><th>Reset</th><th>ErrorMode</th><th>Valeur de réglage</th></tr>
<tr><td>FALSE</td><td>0</td><td>Input</td></tr>
<tr><td>FALSE</td><td>1</td><td>SubstituteOutput</td></tr>
<tr><td>TRUE</td><td>-</td><td>SubstituteOutput</td></tr>
</table></td></tr>
<tr><td>0002_0000</td><td>Cause d'erreur :
Le paramètre Input n'a pas de valeur REAL valide, tandis que le calcul de la valeur de réglage est exécuté (Reset = FALSE).
Réaction en cas d'erreur :
La valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output.
0.0 est émis comme valeur de réglage si ErrorMode = 0.
Solution :
Vérifiez que le paramètre Input est une valeur REAL valide (≠NaN p. ex. 16#7FFF_FFFF).</td></tr>
<tr><td>0004_0000</td><td>Cause d'erreur :
Le calcul de la valeur de réglage donne une valeur REAL non valide pour le paramètre Output.
Réaction en cas d'erreur :
La valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output.
Solution :
Vérifiez toutes les variables utilisées pour le calcul de la valeur de réglage :
• Input
• Td
• Lag
• CycleTime.Value
Les valeurs de ces variables sont valides. Le calcul de la valeur de réglage dans cette combinaison de variables échoue.</td></tr>
</table>

ErrorBits (DW#16#...)	Description
0008_0000	**Cause d'erreur :** La variable Td ou Lag a une valeur valide, tandis que le calcul de la valeur de réglage est exécuté (Reset = FALSE). **Réaction en cas d'erreur :** La valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output. **Solution :** Vérifiez que les conditions suivantes pour les valeurs des variables Td et Lag sont remplies : • CycleTime.Value ≤ Td ≤ 3.402823e+38 • CycleTime.Value/2 ≤ Lag ≤ 3.402823e+38 • Les valeurs sont des valeurs REAL valides (≠ NaN par ex. 16#7FFF_FFFF) Informations complémentaires : Tenez compte du fait que les conditions CycleTime.Value ≤ Td et CycleTime.Value/2 ≤ Lag peuvent ne pas être respectées dans les scénarios suivants : • L'intervalle entre deux appels de Filter_DT1 est plus long que 2 * Lag ou Td, par ex. en raison d'appels conditionnels dans l'exécution du programme ou de points d'arrêt actifs. • Un instantané du DB d'instance Filter_DT1 est chargé comme valeurs actuelles dans la CPU et la création de l'instantané remonte à plus de 2 * Lag ou Td. Dans ces scénarios, une erreur avec message d'erreur ErrorBits = 16#0008_0000 est détectée en cas de mesure automatique du temps de cycle.
0020_0000	**Cause d'erreur :** La variable (configurée avec StartMode) pour l'initialisation du paramètre Output lors du premier appel de l'instruction n'a pas de valeur REAL valide. **Réaction en cas d'erreur :** Lors du premier appel de l'instruction, la valeur de réglage de remplacement qui est configurée au niveau de la variable ErrorMode est émise au paramètre Output. Lors des appels suivants, Filter_DT1 calcule la valeur de réglage à partir de cette valeur de réglage de remplacement. **Solution :** Vérifiez que la variable pour l'initialisation du paramètre Output est une valeur REAL valide (≠ NaN p. ex. 16#7FFF_FFFF). Si Reset = FALSE, l'initialisation n'est active lors du premier appel de l'instruction qu'après le passage de la CPU de l'état de fonctionnement STOP à l'état RUN. La variable utilisée pour l'initialisation du paramètre Output dépend de StartMode : • StartMode = 1: SubstituteOutput • StartMode = 2: Output
0040_0000	**Cause d'erreur :** La variable CycleTime.Value n'a pas de valeur valide, tandis que le calcul de la valeur de réglage est exécuté (Reset = FALSE). **Réaction en cas d'erreur :** La valeur de réglage de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output. **Solution :** Vérifiez que les conditions suivantes sont remplies : • 0.0 < CycleTime.Value ≤ 3.402823e+38 • CycleTime.Value est une valeur REAL valide (≠ NaN, p. ex. 16#7FFF_FFFF) **Informations complémentaires :** Pour un calcul automatique de la valeur de la variable CycleTime.Value, réglez la variable CycleTime.EnableMeasurement sur TRUE.

10.12 Filter_Universal

10.12.1 Compatibilité avec CPU et FW

Le tableau suivant montre les versions de Filter_Universal compatibles avec les CPU.

CPU	FW	Filter_Universal
CPU basées sur S7-1500	à partir de V2.0	V1.0

10.12.2 Description Filter_Universal

Description

L'instruction Filter_Universal est un filtre configurable dans l'ordre 1 à 10.
Il est utilisé pour manipuler un signal de manière à transmettre ou amortir des composantes de fréquence spécifiques de ce signal.
Filter_Universal est utilisable aux fins suivantes :

- Filtre passe-haut
- Filtre passe-bas
- Filtre passe-bande
- Filtre coupe-bande

Les paramètres de filtre suivants peuvent être spécifiés avec les variables correspondantes pour activer le comportement de filtre souhaité :

- Type (variable Type)
- Fréquence (variable Frequency)
- Bande passante (variable Bandwidth)
- Ordre (variable Order)
- Caractéristique (variable Characteristic)

Une description détaillée des paramètres de filtre et des variables correspondantes est disponible au chapitre Paramètres de filtre (Page 568).

Appel

Filter_Universal requiert un temps de cycle constant et doit donc être appelé dans un OB d'alarme cyclique.
L'instruction Filter_Universal est appelée comme DB d'instance unique dans un OB ou une FC. Dans un FB, l'instruction Filter_Universal peut être appelée comme DB d'instance unique, comme DB multi-instance ou comme DB d'instance de paramètres.
Aucun objet technologique n'est créé en cas d'appel de l'instruction. Vous ne disposez pas d'une interface de paramétrage et de mise en service. Vous paramétrez Filter_Universal directement via le DB d'instance et vous mettez en service Filter_Universal via une table de visualisation du programme utilisateur dans la CPU ou HMI.

Démarrage

Les variables de la plage statique de Filter_Universal ne sont pas rémanentes. Ces variables sont initialisées avec les valeurs de départ après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN. Si vous modifiez les valeurs actuelles en mode en ligne et que ces valeurs doivent être conservées après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN, sauvegardez-les dans les valeurs de départ du bloc de données.

Vous pouvez utiliser la variable StartMode (Page 574) pour spécifier le comportement de démarrage de l'instruction Filter_Universal lors du premier appel après le passage de la CPU de l'état de fonctionnement STOP à l'état RUN.

Comportement en cas d'erreur

Si la valeur de sortie ne peut pas être calculée correctement, l'instruction Filter_Universal renvoie une valeur de sortie de remplacement et une erreur avec un message d'erreur ErrorBits >= 16#0002_0000 à la place. Vous pouvez utiliser la variable ErrorMode (Page 581) pour définir la valeur de sortie de remplacement comme suit :

ErrorMode	Output
0	Valeur du paramètre Input
1	Valeur du paramètre SubstituteOutput
2	Dernière valeur de sortie de filtre valide 0.0, si aucune valeur de sortie de filtre valide n'est disponible.
3	0.0

Les points suivants s'appliquent en plus pour toutes les valeurs de la variable ErrorMode :

- Si la valeur de sortie de remplacement n'est pas une valeur REAL valide, 0.0 est sorti comme valeur de sortie.
- La valeur de sortie de remplacement est limitée à la plage de valeurs -3.402823e+38 .. +3.402823e+38 du type de données REAL. Ce n'est qu'alors que la valeur de sortie de remplacement est sortie au paramètre Output.
- La variable ErrorMode n'est active que lorsque le paramètre Reset = FALSE. Si le paramètre Reset = TRUE est défini, la valeur du paramètre SubstituteOutput ou 0.0 est sortie au paramètre Output.

Le paramètre Error indique si une erreur est actuellement présente. Quand l'erreur a disparu, Error est réglé sur FALSE. Le paramètre ErrorBits indique les erreurs qui se sont produites. ErrorBits est rémanent et n'est réinitialisé que par un front montant au paramètre Reset ou ErrorAck.

Filter_Universal revient au calcul de calcul de valeur de sortie par l'algorithme de filtre si plus aucune erreur avec messages d'erreur ErrorBits ≥ 16 0002_0000 n'est présente. La commutation dépend du type de filtre :

- Pour le filtre passe-haut et le filtre passe-bande (Type = 1 ou 2), l'algorithme de filtre est appliqué comme si celui-ci se trouvait dans un état stationnaire avec Output = 0.0. Si le paramètre Input reste constant, la valeur de sortie saute à Output = 0.0. Si le paramètre Input est modifié, la valeur de sortie saute à une valeur correspondante.
- Pour le filtre passe-bas et le filtre coupe-bande (Type = 0 ou 3), l'algorithme de filtre est appliqué comme si celui-ci se trouvait dans un état stationnaire avec Output = SubstituteOutput. La commutation s'effectue sans à-coup.

10.12.3 Fonctionnement Filter_Universal

10.12.3.1 Paramètres de filtre

Les types de paramètres de filtre. la fréquence, la bande passante, la caractéristique et l'ordre peuvent être prédéfinis avec les variables correspondantes pour activer le comportement de filtre souhaité :

Type de filtre

Le type de filtre détermine le comportement de transfert général pour les différentes composantes de fréquence du signal d'entrée. Il est déterminé par la variable Type.
Le tableau suivant décrit les types de filtre suivants :

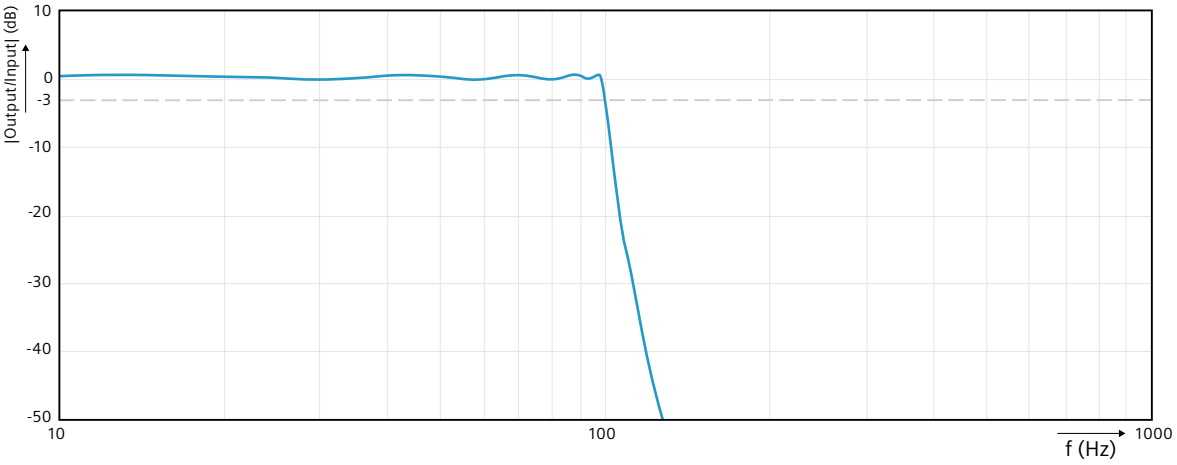
Type	Description	Exemple d'application
0	Filtre passe-bas : Le filtre transmet des composantes de fréquence inférieures à la fréquence de coupure et amortit des composantes de fréquence supérieures à la fréquence de coupure.	Réduction du bruit pour une valeur d'entrée mesurée afin de lisser l'allure des signaux
1	Filtre passe-haut Le filtre transmet des composantes de fréquence supérieures à la fréquence de coupure et amortit des composantes de fréquence inférieures à la fréquence de coupure.	Réjection de composantes de fréquence continues et basses, p. ex. une composante continue dans le signal
2	Filtre passe-bande Le filtre transmet des composantes de fréquence au sein d'une plage donnée autour de la fréquence centrale et amortit des composantes de fréquence en-dehors de la plage.	Détermination du signal utile avec une plage de fréquence donnée à partir d'un signal contenant des composantes de fréquence supplémentaires
3	Filtre coupe-bande Le filtre amortit des composantes de fréquence au sein d'une plage donnée autour de la fréquence centrale et transmet des composantes de fréquence en-dehors de la plage.	L'atténuation de perturbations dans une plage de fréquence donnée, p. ex. perturbations par la fréquence réseau

Fréquence et bande passante

Pour le filtre passe-bas et le filtre passe-haut, la fréquence de coupure est déterminée par la variable Frequency. La fréquence de coupure est la fréquence à laquelle le gain est réduit à $1/\sqrt{2} \approx 0.707 \approx -3$dB. Un signal d'entrée sinusoïdal avec une amplitude de 1.0 et une fréquence égale à la fréquence de coupure entraîne un signal de sortie sinusoïdal avec une amplitude de 0.707.

Le rapport de la valeur de sortie à la valeur d'entrée (gain) peut être représenté en fonction de la fréquence dans la réponse d'amplitude. La valeur 0 dB correspond à un gain = 1.0.

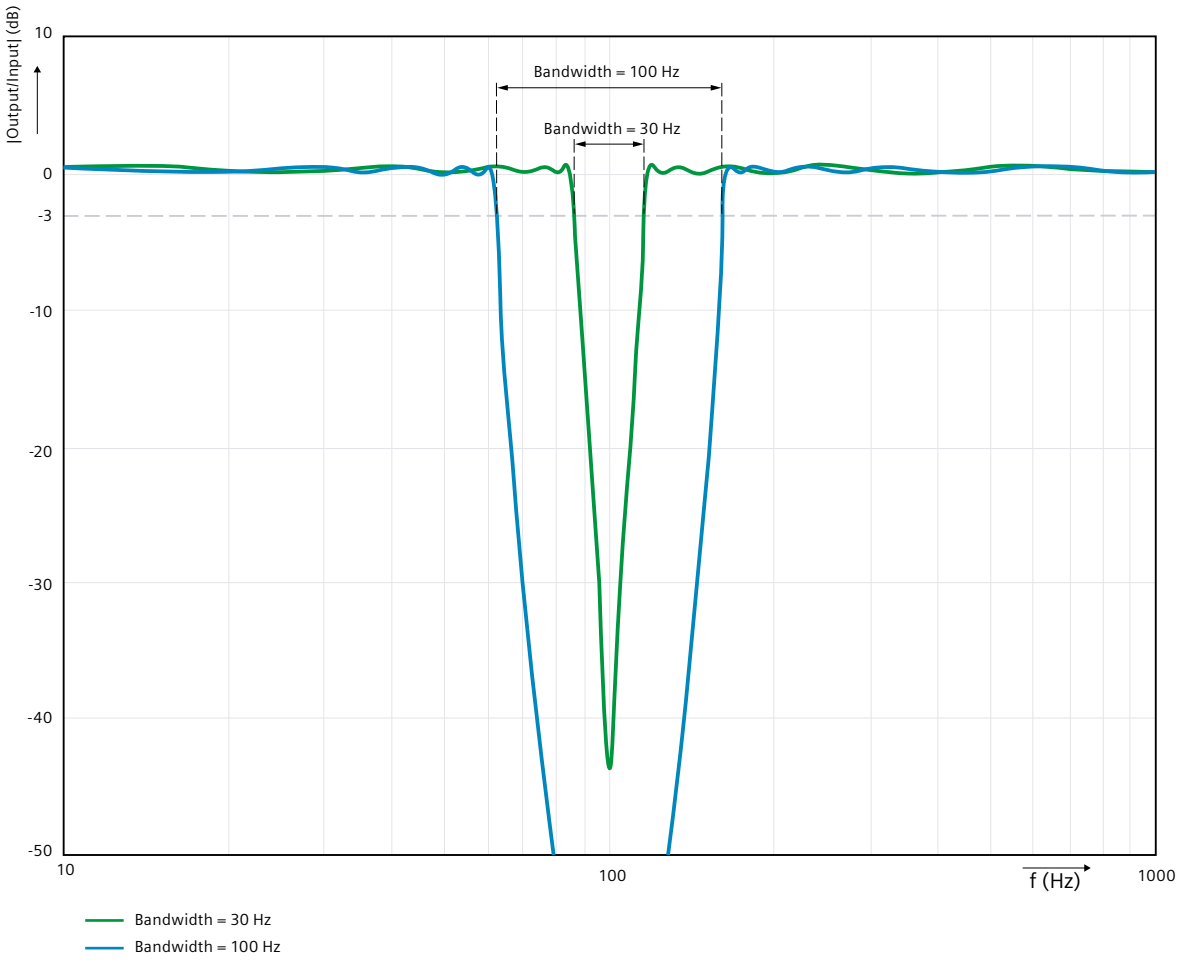
La figure suivante montre la réponse d'amplitude d'un filtre passe-bas avec Frequency = 100 Hz (Order = 10, Characteristic = 2) :



Le filtre passe-bande et le filtre coupe-bande ont une fréquence de coupure inférieure et une fréquence de coupure supérieure dont la position est déterminée par la fréquence centrale et la bande passante. Le tableau suivant montre les variables correspondantes configurées pour de tels filtres :

Variable	Description
Frequency	Détermine la fréquence centrale qui représente le centre géométrique de la fréquence de coupure inférieure et de la fréquence de coupure. La fréquence centrale se trouve au milieu de la fréquence de coupure inférieure et de la fréquence de coupure supérieure en cas d'échelle logarithmique de la fréquence.
Bandwidth	La bande passante détermine la différence entre la fréquence de coupure inférieure et de la fréquence de coupure supérieure. Elle définit la largeur de la plage de fréquence transmise ou amortie.

La figure suivante montre la réponse d'amplitude d'un filtre coupe-bande avec Frequency = 100 Hz et différentes valeurs pour la variable Bandwidth (Order = 10, Characteristic = 2) :



La fréquence de coupure ou la fréquence centrale maximale utilisable dépend du temps de cycle. La plage de valeurs admissible est Frequency < 0.5 / CycleTime.Value.

La bande passante maximale utilisable dépend du temps de cycle et de la fréquence centrale. La plage de valeurs admissible est Bandwidth < 0.5 / CycleTime.Value - Frequency..

REMARQUE

Afin d'éviter un repliement de spectre, la cadence d'échantillonnage pour Filter_Universal (= 1 / temps de cycle) doit être d'au moins le double de la fréquence maximale des signaux ou composantes de signal traités. Il est recommandé de régler un temps de cycle inférieur à cette valeur limite.

Exemple : un signal avec des composantes de fréquence jusqu'à 100 Hz requiert une cadence d'échantillonnage supérieure à 200 Hz. Cela correspond à un temps de cycle maximal de 5 ms, mais il est cependant recommandé de régler une valeur inférieure.

Ordre

L'ordre des filtres détermine la rapidité avec laquelle l'amortissement augmente au-delà de la fréquence de coupure. Cela correspond à la pente de la réponse d'amplitude au-delà de la fréquence de coupure. L'ordre de filtre est déterminé par la variable Order.
Avec un filtre d'ordre supérieur :

- la même fréquence est amortie plus fortement au-delà de la fréquence de coupure. La réponse d'amplitude indique une pente plus élevée.
- le temps d'exécution de Filter_Universal. augmente
- la suroscillation de la réponse indicielle augmente après un échelon d'entrée pour caractéristique de filtre Butterworth et Chebyshev (variable Characteristic = 1 ou 2).

Pour le filtre passe-bande et le filtre coupe-bande, il est recommandé de n'utiliser que des filtres d'ordre supérieur, sinon il se peut que l'effet de filtre souhaité dans la plage de fréquence autour de la fréquence centrale ne soit pas atteint.
Pour l'ordre de filtre, il est possible de configurer des valeurs de 0 à 10 sur la variable Order.
En cas de réglage Order = 0, le filtre est sans effet et Output = Input.

Caractéristique

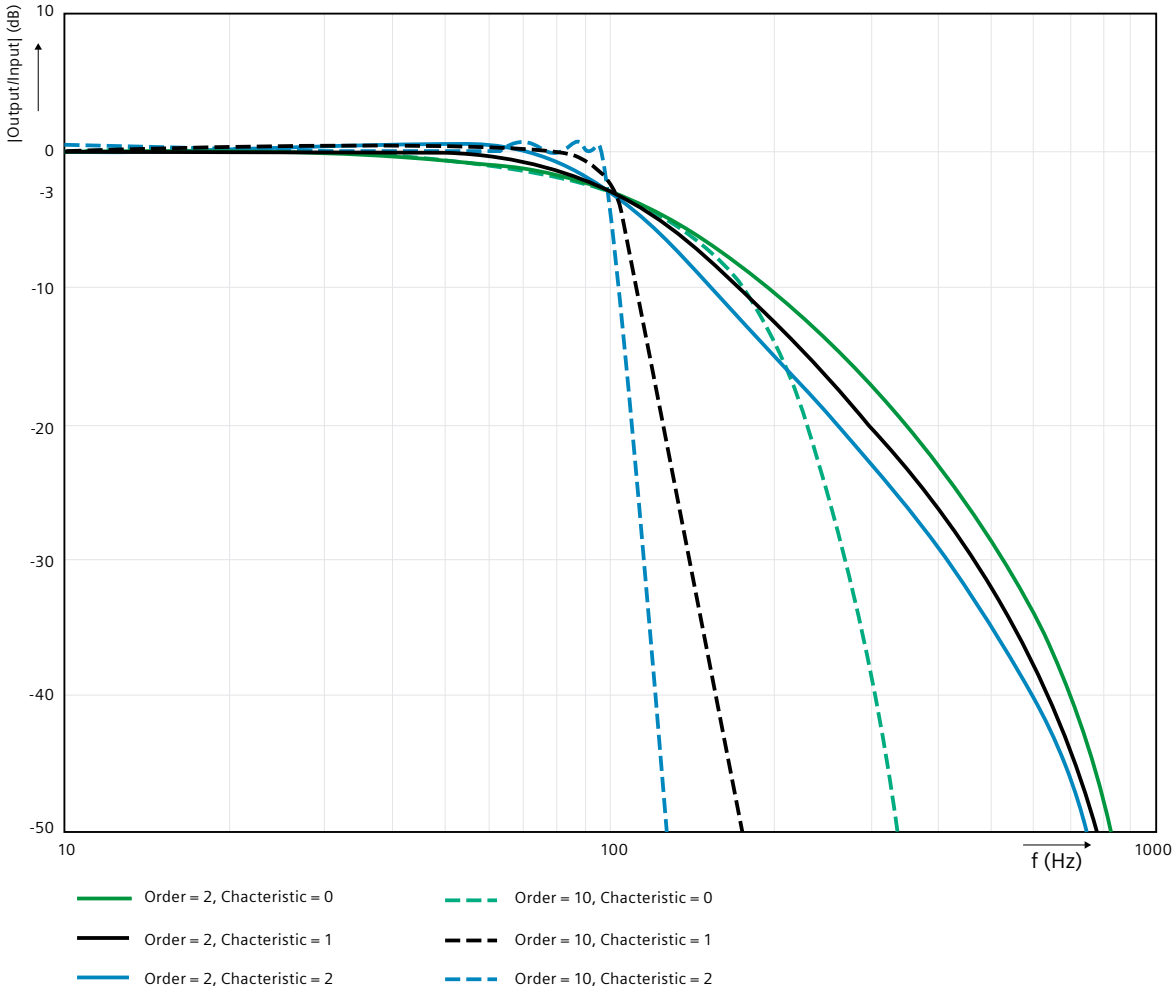
La caractéristique de filtre est déterminée par la variable Characteristic.
Elle a un effet sur :

- l'ondulation de la réponse d'amplitude dans la plage de transmission
- la pente de la réponse d'amplitude au-delà de la fréquence de coupure (vitesse à laquelle l'amortissement augmente).
- la suroscillation dans la réponse indicielle après un échelon d'entrée

Pour Filter_Universal, trois caractéristiques peuvent être configurées sur la base de la variable Characteristic :

Caractéristique	Description
0	Bessel : Ce filtre a une réponse d'amplitude plate dans la plage de transmission. La pente de la réponse d'amplitude au-delà de la fréquence de coupure est inférieure en comparaison avec le filtre Butterworth et le filtre Chebyshev. La réponse indicielle indique seulement une petite suroscillation.
1	Butterworth : Ce filtre a une réponse d'amplitude plate dans la plage de transmission. La pente de la réponse d'amplitude au-delà de la fréquence de coupure est supérieure en comparaison avec le filtre Bessel inférieure en comparaison avec le filtre Chebyshev. La suroscillation de la réponse indicielle est supérieure au filtre Bessel et inférieure au filtre Chebyshev.
2	Chebyshev type I : Ce filtre a une ondulation de 0.5 dB de la réponse d'amplitude dans la plage de transmission. La pente de la réponse d'amplitude au-delà de la fréquence de coupure est supérieure en comparaison avec le filtre Bessel et avec le filtre Butterworth. La réponse indicielle indique une suroscillation plus élevée que le filtre Bessel et le filtre Butterworth.

La figure suivante montre l'effet de différentes valeurs d'ordre et caractéristiques sur la réponse d'amplitude du filtre passe-bas :



Modifier des paramètres de filtre

Avant que l'algorithme de filtre ne calcule la valeur de sortie, Filter_Universal doit déterminer une fois les coefficients de filtre en fonction des paramètres de filtre. La détermination est lancée dans les situations suivantes :

- Pendant la première exécution après le passage de la CPU de l'état de fonctionnement STOP à RUN
- À chaque modification des paramètres de filtre
- Avec l'exécution de "Charger les valeurs de départ comme valeurs actuelles"

Les conditions suivantes sont vérifiées pour les paramètres de filtre dans le cadre de la détermination :

- 0.0 < Frequency < 0.5 / CycleTime.Value
- 0.0 ≤ Bandwidth < 0.5 / CycleTime.Value - Frequency
- 0 ≤ Type ≤ 3
- 0 ≤ Characteristic ≤ 2
- 0 ≤ Order ≤ 10

Un calcul correct de la valeur de sortie par l'algorithme de filtre n'est pas possible si l'une de ces conditions n'est pas remplie alors que Reset = FALSE. Dans ce cas, un message d'erreur est généré et une valeur de sortie de remplacement est générée au paramètre Output jusqu'à ce que tous les paramètres de filtre aient une valeur valide.

Si toutes les valeurs sont valides, les coefficients de filtre sont déterminés une fois et enregistrés en interne pour le calcul d'algorithme de filtre.

La réaction de la valeur de sortie à des modifications valides des paramètres de filtre dépend du type de filtre :

- Pour le filtre passe-haut et le filtre passe-bande (Type = 1 ou 2), l'algorithme de filtre est appliqué comme si celui-ci se trouvait dans un état stationnaire avec Output = 0.0. Si le paramètre Input reste constant, la valeur de sortie saute à Output = 0.0. Si le paramètre Input est modifié, la valeur de sortie saute à une valeur correspondante.
- Pour le filtre passe-bas et le filtre coupe-bande (Type = 0 ou 3), l'algorithme de filtre est appliqué comme si celui-ci se trouvait dans un état stationnaire avec Output = SubstituteOutput. La commutation s'effectue sans à-coup.

Pour des applications à temps critique, il faut tenir compte du fait que la détermination des coefficients de filtre requiert un multiple du temps d'exécution pour le calcul de l'algorithme de filtre.

10.12.3.2 Initialiser les valeurs de sortie

La première valeur du paramètre Output est initialisée après les actions suivantes pour la première exécution :

- Passage du mode de fonctionnement de la CPU de STOP à RUN
- Exécution de "Charger les valeurs de départ comme valeurs actuelles" avec l'option "Toutes les valeurs"

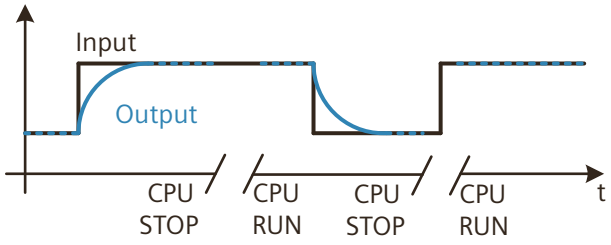
La première valeur du paramètre Output dépend du type de filtre :

- Pour les filtres passe-haut et les filtres passe-bande (Type = 1 ou 2), la première valeur du paramètre Output = 0.0.
- Pour les filtres passe-bas et les filtres coupe-bande (Type = 0 ou 3), la première valeur du paramètre Output peut être configurée via la variable StartMode.

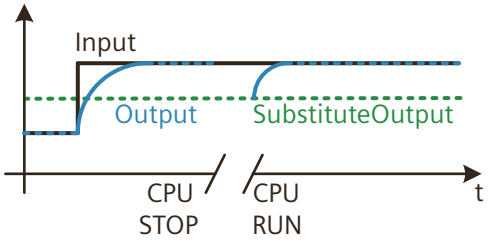
Lors des appels suivants, Filter_Universal calcule la valeur de sortie, à partir de cette valeur d'initialisation et avec prise en compte de la valeur d'entrée et des paramètres de filtre.

Vous avez le choix entre les réglages suivants de la variable StartMode pour les filtres passe-bas et les filtres coupe-bande :

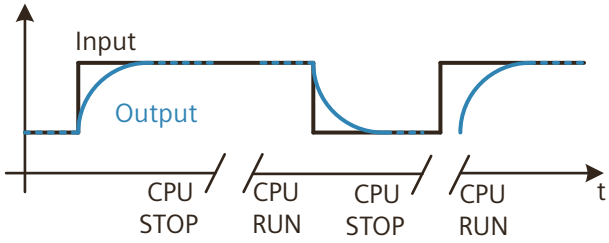
- StartMode = 0
  Le paramètre Output prend la valeur du paramètre Input.



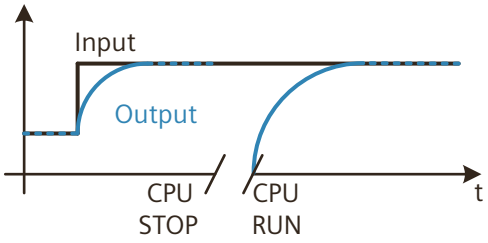
- StartMode = 1
  Le paramètre Output prend la valeur du paramètre SubstituteOutput.



- StartMode = 2
  Le paramètre Output reste inchangé.



- StartMode = 3
  Le paramètre Output prend la valeur 0.0.



Les points suivants s'appliquent en plus pour toutes les valeurs de la variable StartMode :

- La variable StartMode et les paramètres de filtre ne sont pas rémanents. Ces variables sont initialisées avec les valeurs initiales après chaque passage de la CPU de l'état de fonctionnement STOP à l'état RUN. Vérifiez que ces variables ont des valeurs appropriées lors du premier appel de l'instruction Filter_Universal après le passage de la CPU de l'état de fonctionnement STOP à l'état RUN afin d'obtenir le comportement souhaité.
- La valeur sélectionnée par StartMode est limitée à la plage de valeurs du type de données REAL. Ce n'est qu'alors que la valeur est sortie au paramètre Output.
- Si la valeur sélectionnée par StartMode n'est pas une valeur REAL valide, la valeur de sortie de remplacement est sortie au paramètre Output. La valeur de sortie de remplacement est configurée par la variable ErrorMode et est limitée à la plage de valeurs du type de données REAL. Si la valeur de sortie de remplacement n'est pas non plus une valeur REAL valide, 0.0 est émis au paramètre Output. Lors des appels suivants, l'instruction calcule la valeur de sortie à partir de cette valeur de sortie de remplacement.
- La variable StartMode n'affecte le paramètre Output que si le paramètre Reset = FALSE est défini et qu'aucune erreur n'apparaît avec le message d'erreur ErrorBits ≥ 16#0002_0000 en même temps. Si le paramètre Reset = TRUE, la valeur du paramètre SubstituteOutput est sortie au paramètre Output. En présence d'une erreur avec message d'erreur ErrorBits ≥ 16#0002_0000, la valeur de sortie de remplacement qui est configurée dans la variable ErrorMode est sortie au paramètre Output.

10.12.3.3 Valeur finale en état stationnaire

Si la valeur d'entrée est constante, la valeur de sortie de Filter_Universal devrait également atteindre une valeur finale constante après un certain temps.

- Output = 0.0 pour les filtres passe-haut et les filtres passe-bande (Type = 1 ou 2)
- Output = Input pour les filtres passe-bas et les filtres coupe-bande (Type = 0 ou 3)

La précision limitée de l'arithmétique à virgule flottante peut avoir pour conséquence que cette valeur finale n'est pas exactement atteinte.

On l'observe plus souvent avec des filtres d'ordre impair (variable Order = 1, 3, 5, 7 ou 9) qu'avec des filtres d'ordre pair.

Outre le choix d'un ordre de filtre pair, mettre la variable FinalValueMode à 1 peut aider à atteindre la valeur finale. Si la valeur absolue de la valeur de sortie ne change pas pendant plusieurs cycles, ce réglage commute la valeur de sortie sur la valeur définitive. Cette option n'est effective que si la valeur d'entrée est constante.

L'utilisation de l'option FinalValueMode = 1 peut presque doubler le temps de calcul de l'algorithme de filtre. L'effet sur le temps d'exécution dépend des paramètres de filtre, de la valeur d'entrée et du temps de cycle. Pour les applications à temps critique, il est possible de vérifier si l'utilisation de la variable FinalValueMode = 1 est nécessaire ou si le comportement avec FinalValueMode = 0 est suffisant.

En fonction des paramètres de filtre, de la valeur d'entrée et du temps de cycle, il est possible que la valeur finale soit déjà exactement atteinte avec l'option FinalValueMode = 0 et que l'option FinalValueMode = 1 ne soit pas nécessaire.

Exemple :

Le tableau suivant montre l'effet de la variable FinalValueMode sur la valeur de sortie d'un filtre passe-bas avec Frequency = 120.0, Order = 1, Characteristic = 2 et CycleTime.Value = 0.001, avec un échelon d'entrée de 1.0 à 0.0 :

Temps en secondes	Input	Output avec FinalValueMode = 0	Output avec FinalValueMode = 1
-0.001	1.0	1.0000000	1.0000000
0.000	0.0	0.7163693	0.7163693
0.001	0.0	0.3100007	0.3100007
...	0.0	...	...
0.075	0.0	-4.597156E-17	-4.597156E-17
0.076	0.0	4.597156E-17	4.597156E-17
0.077	0.0	-4.597156E-17	-4.597156E-17
0.078	0.0	4.597156E-17	0.0000000
0.079	0.0	-4.597156E-17	0.0000000
...	0.0	+/-4.597156E-17	0.0000000

10.12.3.4 Utilisation dans des applications à temps critique

Le temps d'exécution de Filter_Universal dépend fortement de sa configuration. En cas de configuration standard (CycleTime.EnableDetection = TRUE, FinalValueMode = 1), il n'est pas possible d'exécuter un filtre d'ordre élevé dans le temps de cycle le plus rapide possible pour tous les types de CPU. En cas d'utilisation de Filter_Universal pour des fréquences de signal élevées, de tels temps de cycle rapides peuvent être nécessaires. Les fréquences de signal jusqu'à 2 kHz requièrent par exemple un temps de cycle maximal de 250 µsec.

Les modifications suivantes de la configuration peuvent aider à réduire le temps d'exécution de Filter_Universal pour une application à temps critique :

- Réduction de l'ordre de filtres (variable Order) si le comportement de filtre souhaité peut encore être atteint.
- Mise de FinalValueMode à 0 si cela n'entraîne pas de différente significative dans le comportement de filtre pour des valeurs d'entrée constantes par rapport à FinalValueMode = 1.
- Réglage de CycleTime.EnableDetection sur FALSE et spécification manuelle de la variable CycleTime.Value. Cela influe uniquement sur le temps d'exécution du premier appel ou après une modification des paramètres de filtre.

L'état de fonctionnement actuel de Filter_Universal a également une influence sur le temps d'exécution. La détermination des coefficients de filtre lors de la première exécution ou en cas de modification des paramètres de filtre requiert un multiple du temps d'exécution nécessaire au calcul de l'algorithme de filtre avec paramètres de filtre constants.

10.12.3.5 Environnement d'appel et détection automatique du temps de cycle

Filter_Universal requiert un temps de cycle constant pour le calcul de la valeur de sortie et doit donc être appelé dans un OB d'alarme cyclique. Avec la configuration standard, Filter_Universal détecte automatiquement le temps de cycle de l'OB et l'enregistre dans la variable CycleTime.Value. Cela se produit dans les situations suivantes :

- Pendant la première exécution après le passage de la CPU de l'état de fonctionnement STOP à RUN
- À chaque modification des paramètres de filtre
- Avec l'exécution de "Charger les valeurs de départ comme valeurs actuelles"

Un calcul correct du paramètre Output n'est pas possible si la détection automatique du temps de cycle ne fournit aucun résultat valide ou si Filter_Universal n'est pas appelé dans un OB d'alarme cyclique. Dans ce cas, un message d'erreur est généré et une valeur de sortie de remplacement est générée au paramètre Output jusqu'à ce qu'un temps de cycle valide soit détecté par un OB d'alarme cyclique.

Tenez compte du fait que des modifications par des appels conditionnels de Filter_Universal peut entraîner des écarts entre le temps de cycle détecté et le temps de cycle réel, ce qui influe sur le comportement de filtre. Pour cette raison, évitez des appels conditionnels de Filter_Universal.

Les points d'arrêt actifs ou le chargement d'instantanés comme valeurs actuelles n'ont pas d'influence sur la détection automatique du temps de cycle.

Si vous désactivez la détection automatique du temps de cycle en définissant la variable CycleTime.EnableDetection = FALSE, vous devez spécifier le temps de cycle manuellement dans la variable CycleTime.Value. Filter_Universal vérifie la validité de la variable Variable CycleTime.Value à chaque appel.

La désactivation de la détection automatique du temps de cycle réduit le temps d'exécution de Filter_Universal, ce qui peut être utile pour des applications à temps critique. Les appels

en-dehors d'un OB d'alarme cyclique peuvent avoir un effet négatif sur le comportement de filtre car le temps de cycle réel n'est alors pas constant.

10.12.3.6 Comportement reset

Le comportement de l'instruction Filter_Universal dépend du paramètre Reset de la manière suivante :

- Si le paramètre Reset = TRUE, la valeur du paramètre SubstituteOutput est sortie au paramètre Output.
- Lorsque le paramètre Reset = FALSE, la valeur émise au paramètre Output est calculée par l'algorithme de filtre.
- Si le paramètre Reset passe de FALSE à TRUE, la valeur au paramètre Output devient directement la valeur du paramètre SubstituteOutput. Ce changement peut se faire avec à-coup. Le paramètre ErrorBits est également réinitialisé.
- Si le paramètre Reset passe de TRUE à FALSE, le comportement dépend du type de filtre.
  - Pour le filtre passe-haut et le filtre passe-bande (Type = 1 ou 2), l'algorithme de filtre est appliqué comme si celui-ci se trouvait dans un état stationnaire avec Output = 0.0. Si le paramètre Input reste constant, la valeur de sortie saute à Output = 0.0. Si le paramètre Input est modifié, la valeur de sortie saute à une valeur correspondante.
  - Pour le filtre passe-bas et le filtre coupe-bande (Type = 0 ou 3), l'algorithme de filtre est appliqué comme si celui-ci se trouvait dans un état stationnaire avec Output = SubstituteOutput. La commutation s'effectue sans à-coup.

10.12.3.7 Comportement de validation EN/ENO

Si l'une des conditions suivantes est remplie, la sortie de validation ENO prend la valeur FALSE :

- L'entrée de validation EN est réglée sur TRUE et le paramètre Output est défini par une valeur de sortie de remplacement pour les messages d'erreur ErrorBits ≥ 16#0001_0000
- L'entrée de validation EN prend la valeur FALSE.

Sinon, la sortie de validation ENO est réglée sur TRUE.

10.12.4 Paramètre d'entrée Filter_Universal

Paramètre	Type de données	Valeur par défaut	Description
Input	REAL	0.0	Valeur d'entrée
SubstituteOutput	REAL	0.0	SubstituteOutput est utilisé comme valeur de sortie de remplacement si Reset = TRUE ou l'un des modes suivants est actuellement en vigueur : • ErrorMode = 1 • StartMode = 1
ErrorAck	BOOL	FALSE	Supprime les messages d'erreur. • Front FALSE -> TRUE ErrorBits est remis à 0.
Reset	BOOL	FALSE	Réinitialise l'instruction. • Front FALSE -> TRUE ErrorBits est réinitialisé. • Tant que Reset a la valeur TRUE, la valeur de sortie de remplacement SubstituteOutput est fournie. • Tant que Reset est réglé sur FALSE, le calcul de la valeur de sortie est exécuté. • Front TRUE -> FALSE – Pour le filtre passe-haut et le filtre passe-bande (type 1 ou 2), l'algorithme de filtre est appliqué comme si celui-ci se trouvait dans un état stationnaire avec Output = 0.0. – Pour le filtre passe-bas et le filtre coupe-bande (type 0 ou 3), l'algorithme de filtre est appliqué comme si celui-ci se trouvait dans un état stationnaire avec Output = SubstituteOutput.

10.12.5 Paramètre de sortie Filter_Universal

Paramètre	Type de données	Valeur par défaut	Description
Output	REAL	0.0	Valeur de sortie La valeur de sortie est rémanente.
ErrorBits	DWORD	16#0	Le paramètre ErrorBits (Page 581) signale quels sont les messages d'erreur présents. ErrorBits est rémanent et réinitialisé à Reset ou ErrorAck en cas de front montant.
Error	BOOL	FALSE	Le réglage de Error sur TRUE indique la présence d'au moins une erreur actuellement.

10.12.6 Variables statiques Filter_Universal

Variable	Type de données	Valeur par défaut	Description
Frequency	REAL	50.0	Fréquence de coupure de passe-bas et passe-haut ou fréquence centrale de passe-bande et coupe-bande en Hz Plage de valeurs admissible : 0.5/CycleTime.Value > Frequency > 0.0
Bandwidth	REAL	0.0	Bande passante de passe-bande et coupe-bande en Hz Plage de valeurs admissible : 0.5/CycleTime.Value - Frequency > Bandwidth ≥ 0.0
Type	INT	0	Type de filtre • 0 = filtre passe-bas • 1 = filtre passe-haut • 2 = filtre passe-bande • 3 = filtre coupe-bande Plage de valeurs admissible : 0 à 3
Characteristic	INT	0	Caractéristique de filtre • 0 = Bessel • 1 = Butterworth • 2 = Chebyshev avec ondulation 0.5 dB dans la plage de transmission Plage de valeurs admissible : 0 à 2
Order	INT	2	Ordre de filtre (Output = Input si Order = 0) Plage de valeurs admissible : 0 à 10
ErrorMode	INT	2	Sélection de la valeur de sortie de remplacement en cas d'erreur • 0 = Input • 1 = SubstituteOutput • 2 = dernière valeur de sortie de filtre valide • 3 = 0.0 Plage de valeurs admissible : 0 à 3 Si la valeur de ErrorMode ne correspond pas à la plage de valeurs admissible, alors ErrorMode = 2.
StartMode	INT	2	Sélection de la première valeur de sortie pour filtre passe-bas et filtre coupe-bande • 0 = Input • 1 = SubstituteOutput • 2 = dernière valeur de sortie • 3 = 0.0 Plage de valeurs admissible : 0 à 3 Si la valeur de StartMode ne correspond pas à la plage de valeurs admissible, alors StartMode = 2.
FinalValueMode	INT	1	Sélection du comportement en état stationnaire • 0 = écarts possibles entre valeur de sortie et valeur finale • 1 = la valeur de sortie atteint exactement la valeur finale Plage de valeurs admissible : 0 à 1 Si la valeur de FinalValueMode ne correspond pas à la plage de valeurs admissible, alors FinalValueMode = 1.

Variable	Type de données	Valeur par défaut	Description
CycleTime	AuxFct_CycleTimeDetection	-	Données du temps de cycle
CycleTime.Value	REAL	0.001	Temps de cycle en secondes (intervalle entre deux appels) Plage de valeurs admissible : CycleTime.Value > 0.0
CycleTime.EnableDetection	BOOL	TRUE	Détection automatique du temps de cycle • FALSE = désactivée • TRUE = activée

10.12.7 Paramètre ErrorBits

En présence de plusieurs erreurs simultanées, les valeurs des ErrorBits s'affichent comme addition binaire. L'affichage de ErrorBits = 16#0000_0003, par ex., indique la présence simultanée des erreurs 16#0000_0001 et 16#0000_0002.
Avec Filter_Universal, les erreurs signalées au paramètre ErrorBits sont réparties en deux catégories :

- Erreurs avec messages d'erreur ErrorBits < 16#0001_0000
  La valeur de sortie peut être calculée malgré l'erreur.
- Erreurs avec messages d'erreur ErrorBits ≥ 16#0001_0000
  L'erreur empêche le calcul de la valeur de sortie. Une valeur de sortie de remplacement est sortie.

Erreurs déclenchant des messages d'erreur ErrorBits < 16#0001_0000

En présence d'une ou de plusieurs erreurs avec messages d'erreur ErrorBits < 16#0001_0000, l'instruction Filter_Universal réagit de la manière suivante :

- La valeur de sortie est déterminée comme suit en dépit de cette erreur :
  - Calcul de la valeur de sortie par l'algorithme de filtre si Reset = FALSE
  - Sortie de SubstituteOutput si Reset = TRUE
- Le paramètre de sortie Error est mis à 1.
- La sortie de validation ENO n'est pas modifiée.

Le paramètre de sortie Error est supprimé dès que l'erreur a disparu.

ErrorBits (DW#16#...)	Description
0000_0000	Pas d'erreur.
0000_0001	**Cause d'erreur et réaction en cas d'erreur :** Le paramètre Output a été limité à -3.402823e+38 ou +3.402823e+38. **Solution :** Si la valeur déterminée par la fonction de filtre est émise à la sortie (Reset = FALSE et ErrorBits < 16#0001_0000), vérifiez le paramètre Input. Si ErrorBits≥ 16#0001_0000 et Reset = FALSE, la valeur de sortie de remplacement est limitée à sa sortie. Vérifiez ensuite les paramètres suivants en fonction de la valeur paramétrée à la variable ErrorMode: • Input • SubstituteOutput Si Reset = TRUE, vérifiez le paramètre SubstituteOutput.

Erreurs déclenchant des messages d'erreur ErrorBits ≥ 16#0001_0000

En présence d'une ou de plusieurs erreurs avec messages d'erreur ErrorBits ≥ 16#0001_0000, l'instruction Filter_Universal réagit de la manière suivante :

- La valeur de sortie ne peut pas être déterminée comme prévu. À la place, c'est la valeur de sortie de remplacement qui est émise.
- Le paramètre de sortie Error est mis à 1.
- La sortie de validation ENO est réglée sur FALSE.

Dès que les erreurs avec messages d'erreur ErrorBits ≥ 16#0001_0000 ont disparu, Filter_Universal réagit de la manière suivante :

- La valeur de sortie est déterminée comme suit :
  - Calcul de la valeur de sortie par l'algorithme de filtre si Reset = FALSE
  - Sortie de SubstituteOutput si Reset = TRUE
- La sortie de validation ENO est réglée sur TRUE.

Le paramètre de sortie Error est supprimé dès que l'erreur a disparu.

<table>
<tr><th>ErrorBits (DW#16#...)</th><th>Description</th></tr>
<tr><td>0001_0000</td><td>Cause de l'erreur :
Le paramètre SubstituteOutput ou une autre variable qui est utilisée comme valeur de sortie n'a pas de valeur REAL valide.
Réaction à l'erreur :
La sortie est réglée sur 0.0.
Solution :
Vérifiez que la variable utilisée comme valeur de sortie est une valeur REAL valide (≠NaN p. ex. 16#7FFF_FFFF). La variable utilisée comme valeur de sortie dépend de Reset et de ErrorMode :
<table>
<tr><th>Reset</th><th>ErrorMode</th><th>Valeur de sortie</th></tr>
<tr><td>FALSE</td><td>0</td><td>Input</td></tr>
<tr><td>FALSE</td><td>1</td><td>SubstituteOutput</td></tr>
<tr><td>TRUE</td><td>-</td><td>SubstituteOutput</td></tr>
</table></td></tr>
<tr><td>0002_0000</td><td>Cause de l'erreur :
Le paramètre Input n'a pas de valeur REAL valide, tandis que le calcul de la valeur de sortie est exécuté (Reset = FALSE).
Réaction à l'erreur :
La valeur de sortie de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output.
0.0 est émis comme valeur de sortie si ErrorMode = 0.
Solution :
Vérifiez que le paramètre Input est une valeur REAL valide (≠NaN p. ex. 16#7FFF_FFFF).</td></tr>
<tr><td>0004_0000</td><td>Cause de l'erreur :
Le calcul de la valeur de sortie donne une valeur REAL non valide pour le paramètre Output.
Réaction à l'erreur :
La valeur de sortie de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output.
Solution :</td></tr>
</table>

ErrorBits (DW#16#...)	Description
	Vérifiez toutes les variables utilisées pour le calcul de la valeur de sortie : • Input • Frequency • Bandwidth • Type • Characteristic • Order • CycleTime.Value Les valeurs de ces variables sont valides. Le calcul de la valeur de sortie dans cette combinaison de variables échoue.
0008_0000	**Cause de l'erreur :** Un ou plusieurs paramètres de filtre n'ont pas de valeur valide, tandis que le calcul de la valeur de sortie est exécuté (Reset = FALSE). **Réaction à l'erreur :** La valeur de sortie de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output. **Solution :** Vérifiez que les conditions suivantes pour les valeurs des paramètres de filtre sont remplies : • 0.0 < Frequency < 0.5 / CycleTime.Value • 0.0 ≤ Bandwidth < 0.5 / CycleTime.Value - Frequency • 0 ≤ Type ≤ 3 • 0 ≤ Characteristic ≤ 2 • 0 ≤ Order ≤ 10
0010_0000	**Cause de l'erreur :** échec de la détection automatique du temps de cycle car Filter_Universal n'est pas appelé dans un OB d'alarme cyclique. **Réaction à l'erreur :** La valeur de sortie de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output. **Solution :** Assurez-vous que Filter_Universal est appelé dans un OB d'alarme cyclique. **Informations complémentaires :** La détection automatique du temps de cycle peut être désactivée en définissant la variable CycleTime.EnableDetection = FALSE. Vous devez ensuite indiquer le temps de cycle manuellement dans la variable CycleTime.Value. L'appel de Filter_Universal en-dehors d'un OB d'alarme cyclique peut avoir un effet négatif sur le comportement de filtre car le temps de cycle réel n'est alors pas constant.
0020_0000	**Cause de l'erreur :** La variable (configurée avec StartMode) pour l'initialisation du paramètre Output lors du premier appel de l'instruction n'a pas de valeur REAL valide. **Réaction à l'erreur :** Lors du premier appel de l'instruction, la valeur de sortie de remplacement qui est configurée au niveau de la variable ErrorMode est émise au paramètre Output. Lors des appels suivants, Filter_Universal calcule la valeur de sortie à partir de cette valeur de sortie de remplacement. **Solution :** Vérifiez que la variable pour l'initialisation du paramètre Output est une valeur REAL valide (≠ NaN p. ex. 16#7FFF_FFFF). Si Reset = FALSE, l'initialisation n'est active lors du premier appel de l'instruction qu'après le passage de la CPU de l'état de fonctionnement STOP à l'état RUN. La variable utilisée pour l'initialisation du paramètre Output dépend de StartMode : • StartMode = 1: SubstituteOutput • StartMode = 2: Output

ErrorBits (DW#16#...)	Description
0040_0000	**Cause de l'erreur :** La variable CycleTime.Value n'a pas de valeur valide, tandis que le calcul de la valeur de sortie est exécuté (Reset = FALSE). **Réaction à l'erreur :** La valeur de sortie de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output. **Solution :** Vérifiez que les conditions suivantes sont remplies : • 0.0 < CycleTime.Value ≤ 3.402823e+38 • CycleTime.Value est une valeur REAL valide (≠ NaN, p. ex. 16#7FFF_FFFF) **Informations complémentaires :** Pour un calcul automatique de la valeur de la variable CycleTime.Value, réglez la variable CycleTime.EnableDetection sur TRUE.
0080_0000	**Cause de l'erreur :** Une erreur interne s'est produite lors de la détection automatique du temps de cycle. **Réaction à l'erreur :** La valeur de sortie de remplacement qui est configurée avec la variable ErrorMode est émise au paramètre Output. **Solution :** Assurez-vous que Filter_Universal est appelé dans un OB d'alarme cyclique. Si l'erreur persiste, contactez le support client SIMATIC. **Informations complémentaires :** La détection automatique du temps de cycle peut être désactivée en définissant la variable CycleTime.EnableDetection = FALSE. Vous devez ensuite indiquer le temps de cycle manuellement dans la variable CycleTime.Value.

Index

S

T

V