1- Les instructions spéciales :

- CMA : complémenter le contenue de l'accumulateur Ac : A ⟶ $\overline{Ac}$c

Exp : MVI A, 55 ; A=55
CMA ; A=AA

- CMC ; complémenter le bit carry : C⟶$\bar{C}$

Si : C=0 ⇒ C⟵ 1
Si : C=1 ⇒ C⟵ 0
Exp : MVI A, FF
ADI A, 15 A=14 et C=1
CMC

- STC ; set bit carry (mettre le bit carry = 1): C⟵ 1

Exp: STC; C⟵ 1
CMC; C⟵ 0
RAL; × 2 ; A⟵20
RAL; × 2; A⟵40
RAL; × 2; A⟵80

- NOP (No Operation): généralement utilisé dans les boucles de temporisation.

Exp : Si : NOP prend un cycle horloge :
Horloge de 1Mhz ⇒ 1 cycle horloge = 1µs
Exp : MVI B, FA
Etiq NOP
DCR B
JNZ etiq

Les sous-programmes en Assembleur :

- Les sous-programmes sont utilisés pour réduire le nombre de lignes dans un programme principal, ils sont détecté par le programmeur durant la mise en structure et développement du programme principal. Généralement on trouve comme sous-programme :
  - Des temporisations.
  - Des traitements de lecture/écriture.
  - Des traitements de calcule.

a) Algorithme d'une application en général:

1- Saisie de données :
- Operateur
- Capteurs (Accele-3D, RC, Temp..)
- Source de données (Base de données)

2- Traitements :
- Trait1
- Trait2
- Trait3..

3- Présentation des résultats :
   - Affichage
   - Stocker les résultats
   - Impression
4- Fin.

b) Les instructions des sous-programmes :

➢ CALL, Adresse ; instruction pour faire appel à un sous-programme.

- Dans un appel à un sous-programme, le µP sauvegarde l'adresse de la prochaine instruction dans la PILE (mémoire pointé par SP).

➢ RET : cette instruction définie la fin du sous-programme et le retour au programme principal. L'adresse de retour est récupérée de la PILE. C'est-à-dire dans l'exécution de l'instruction CALL le contenue du registre PC est stocké en mémoire PILE comme suite :
M(SP – 2) ⟵ PC_L
M(SP – 1) ⟵ PC_H

- Dans l'exécution de l'instruction RET, le registre PC sera chargé automatiquement par le contenue de la mémoire PILE comme suite :
PC_H ⟵ M(SP – 1)
PC_L ⟵ M(SP -2)

```
Exp : MVI A,25          MVI A, 25
      MOV B,A           MOV B, A
      NOP               MVI A, 6
      NOP               CALL Tempo
      NOP               MVI A, 21
      NOP               MOV C, A
      NOP               MVI A, 5
      NOP               CALL Tempo
      MVI A, 21         MVI A, 15
      MOV C, A          MOV D, A
      NOP               MVI A, 8
      NOP               CALL Tempo
      NOP               RST1
      NOP         Tempo NOP
      NOP               DCR A
      MVI A, 15         JNZ Tempo
      MOV D, A          RET
      NOP
      NOP
```

NOP
NOP
NOP
NOP
NOP
NOP
RST 1

- Exercice:
  - Ecrire un programme qui calcule la somme des valeurs dans l'espace mémoire 8000-8010 , il calcule aussi le nombre de valeurs nul (M(i) =0) , calcule aussi le nombre de valeurs positifs et finalement efface le bloc mémoire 8000-8010 . On range les résultats dans les cases mémoires 8020, 8021, et 8022.
- Note : On utilise des sous-programmes.
- Programme principale :

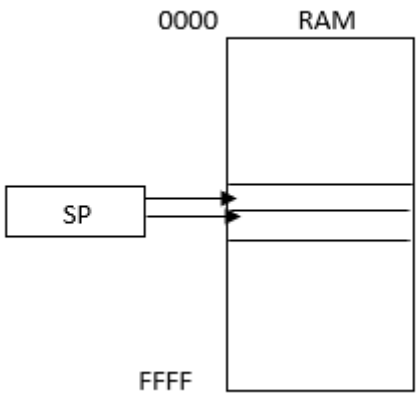
1- Initialisation
2- Appel SP-Sv (7100)
3- Appel SP-VN (7200)
4- Appel SP-VP (7300)
5- Stocker résultats
6- Fin

2- Les instructions sur PILE, Contrôle et I/O :

- La pile est une partie de la mémoire RAM que le programmeur choisi en fixant le contenue du registre pointeur de la PILE 'SP' qui pointe à l'adresse du début de la PILE. La PILE est utilisée automatiquement par le µP dans l'exécution de quelques instructions tel que :

Sans conditionnel, CALL, RET, RSTn, PUSH, POP.

La PILE permet de conserver des données dans la mémoire et de les retirer de cette mémoire d'une mémoire LIFO (Last In First Out).



- RST_n : Reset et saut à un sous-programme de traitement de cette conception.

Exp : RST 1 : fin de programme à exécuter.

- PUSH reg-pair ; mettre dans la PILE le contenue du registre pair.
  Reg pair : BC, DE, HL.

Exp1 : PUSH B

M(SP) ← B
SP ← SP+1
M(SP) ← C
SP ← SP+1

Exp2 : LXI B, 55AA
PUSH B

- POP reg-pair ; retirer de la PILE deux octets consécutives pointée par SP et les ranger dans le registre pair.

Exp : POP D

SP ← SP-1
D ← M(SP)
SP ← SP+1
E ← M(SP)

- PUSH PSW ; mettre dans la PILE les deux registres actifs Accumulateur et Flag.

Exp1 : LXI SP, 6800 ; SP ← 6800
MVI A, 55 ; Ac ← 55
ADI, F1
PUSH PSW
RST 1

M(SP) ← Ac
SP ← SP+1
M(SP) ← F
SP ← SP+1

- POP PSW; retirer de la PILE les deux valeurs à ranger dans l'Acc et Flag.

Exp : POP PSW
RST 1

- XTHL ; changer le contenue de deux cases mémoires consécutives pointé par SP avec le contenue des registres pair H et L.

(L) ← M(SP)
(H) ← M(SP+1)

- SPHL ; stocker le contenue de HL dans le registre SP.

SP ← HL

Exp : LXI H, 6800 HL ← 6800

SPHL SP ← HL=6800

- Exercice:
  - Ecrire un programme qui permute le contenue de 2 cases mémoires en utilisant la PILE.
  - Ecrire un programme qui utilise des appels aux sous-programmes afin de voir ce qui ce passe au niveau de la PILE.

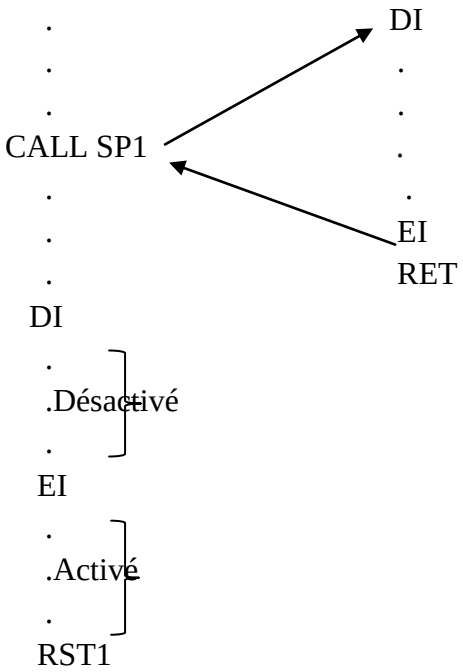
Instruction de contrôle :

- HLT (Halt) :stopper l'exécution d'un programme.
- EI (Enable Interruption) : activer les interruptions.

Exp : EI si je veux travailler avec les interruptions dans mon programme.

- DI (Desable Interruption) : Désactiver les interruptions

Prog Principal



Les Instructions de transfert de données entre microprocesseur et périphérique.

Tout périphérique lié au système à microprocesseur possède une adresse, exemple dans le cas de la carte SDK-8085 le PIO-0 a une adresse de base : 40H et PIO-1 une adresse de base 50H, le PIO-1 est connecté au 8 LED's via le port A et 8 Interrupteurs via le port B.

Il existe deux instructions pour transferer les données entre périphériques et microprocesseur.

- OUT port-adres ; transférer le contenue de l'accumulateur A vers le port de périphérique dot l'adresse est adrs.

Exp : MVI A, 82

OUT 53 ; programmer les ports du PIO-1 : A en entrée et B en sortis , la programmation ce fait dans le registre de contrôle.

MIV A,FF

OUT 50 ; envoie FF sur Port A

Exp : jeux de lumière sur Port A

```
        MVI A,82
        OUT 53
Etiq :  MVI A,FF
        OUT 50
        CALL tempo
        MVI A,00
        OUT 50
        CALL tempo
        JMP  etiq
        RST  1
```

- IN Port-Adres : lire une donnée du présente dans le Port est range la donnée dans l'accumulateur A.
  Acc------ Port-Ardes ;
  Exp : lire une donnée sur port B du PIO-1 et range la donnée dans la case mémoire : 8000.

```
MVI A,82
OUR 53
IN 51
STA 8000
RST  1
```