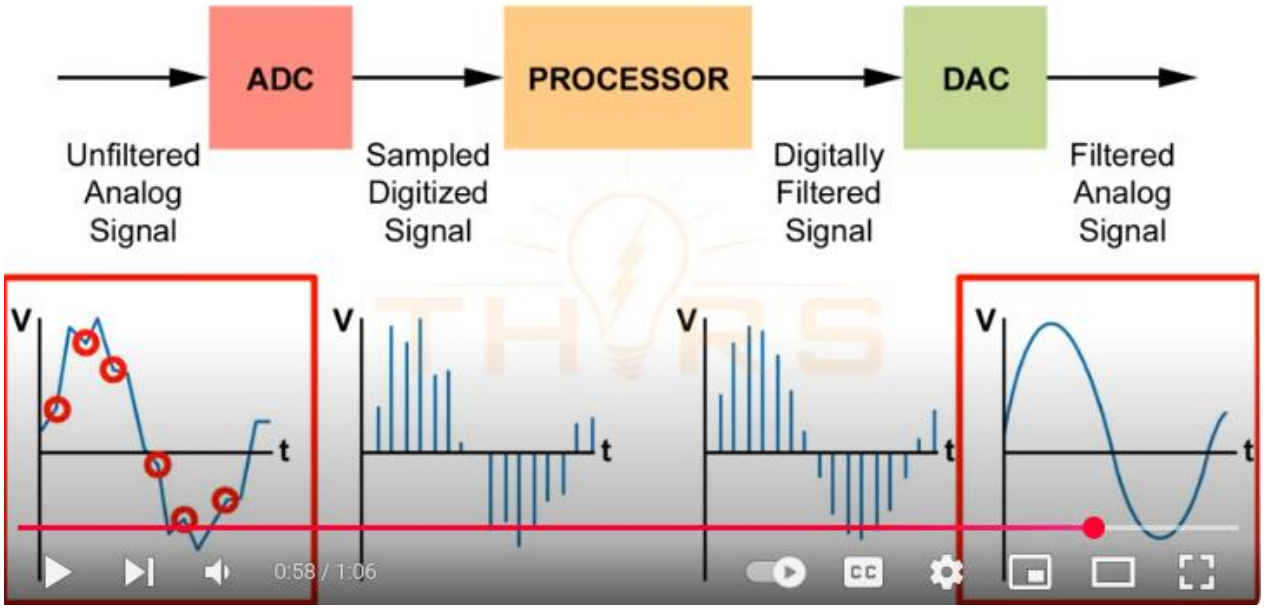
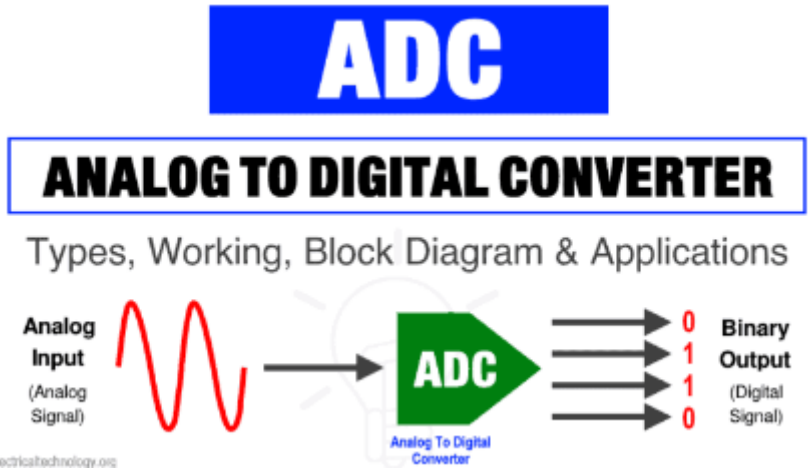
Analog-to-digital converter

M. Fezari

1. Introduction:

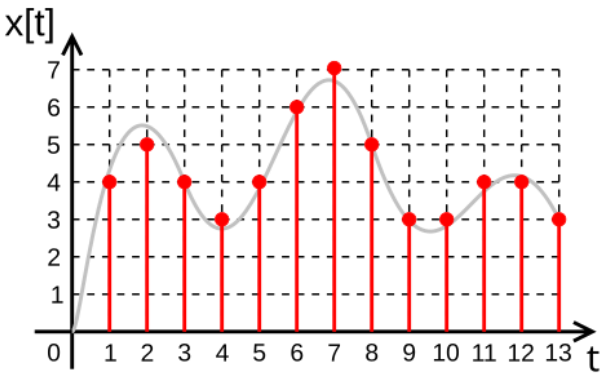
In electronics, an **analog-to-digital converter** (**ADC**, **A/D**, or **A-to-D**) is a system that converts an analog signal, such as a sound picked up by a microphone or light entering a digital camera, into a digital signal. An ADC may also provide an isolated measurement such as an electronic device that converts an analog input voltage or current to a digital number representing the magnitude of the voltage or current. Typically the digital output is a two's complement binary number that is proportional to the input, but there are other possibilities.





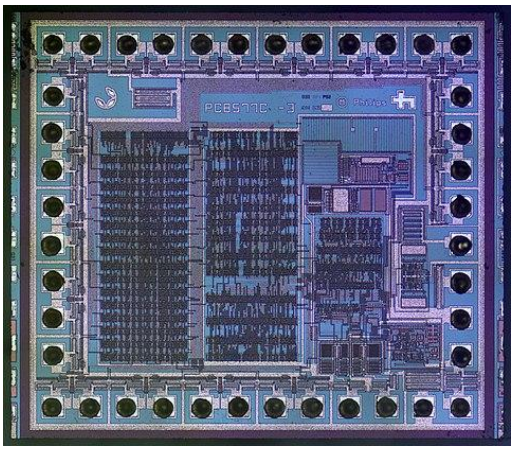
or

to



There are several ADC architectures. Due to the complexity and the need for precisely matched components, all but the most specialized ADCs are implemented as integrated circuits (ICs). These typically take the form of metal–oxide–semiconductor (MOS) mixed-signal integrated circuit chips that integrate both analog and digital circuits.

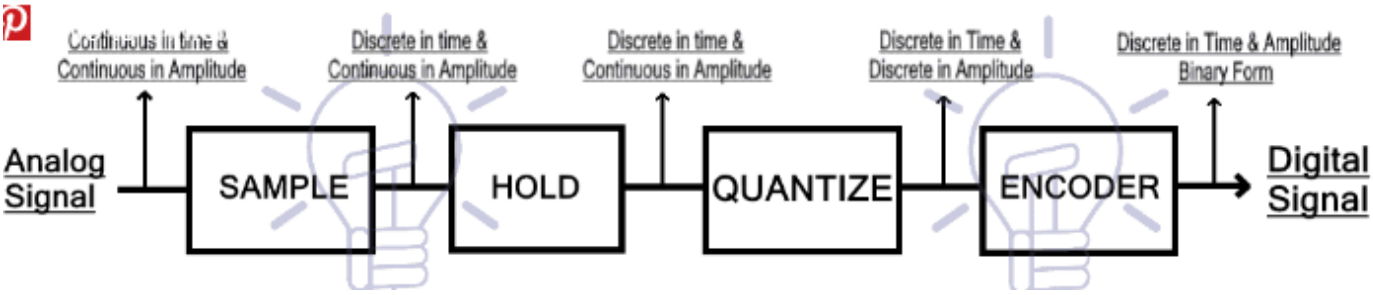
A digital-to-analog converter (DAC) performs the reverse function; it converts a digital signal into an analog signal.



2. ADC how it works:

An ADC converts a continuous-time and continuous-amplitude analog signal to a discrete-time and discrete-amplitude digital signal. The conversion involves quantization of the input, so it necessarily introduces a small amount of quantization error. Furthermore, instead of continuously performing the conversion, an ADC does the conversion periodically, sampling the input, and limiting the allowable bandwidth of the input signal.

The performance of an ADC is primarily characterized by its bandwidth and signal-to-noise and distortion ratio (SNDR). The bandwidth of an ADC is characterized primarily by its sampling rate. The SNDR of an ADC is influenced by many factors, including the resolution, linearity and accuracy (how well the quantization levels match the true analog signal), aliasing and jitter. The SNDR of an ADC is often summarized in terms of its effective number of bits (ENOB), the number of bits of each measure it returns that are on average not noise. An ideal ADC has an ENOB equal to its resolution. ADCs are chosen to match the bandwidth and required SNDR of the signal to be digitized. If an ADC operates at a sampling rate greater than twice the bandwidth of the signal, then per the Nyquist–Shannon sampling theorem, near-perfect reconstruction is possible. The presence of quantization error limits the SNDR of even an ideal ADC. However, if the SNDR of the ADC exceeds that of the input signal, then the effects of quantization error may be neglected, resulting in an essentially perfect digital representation of the bandlimited analog input signal.



Steps to Convert Analog signal to Digital Value:

The analog signal is first applied to the '**sample**' block where it is sampled at a specific sampling frequency. The sample amplitude value is maintained and held in the '**hold**' block. It is an analog value. The hold sample is quantized into discrete value by the '**quantize**' block. At last, the '**encoder**' converts the discrete amplitude into a binary number.

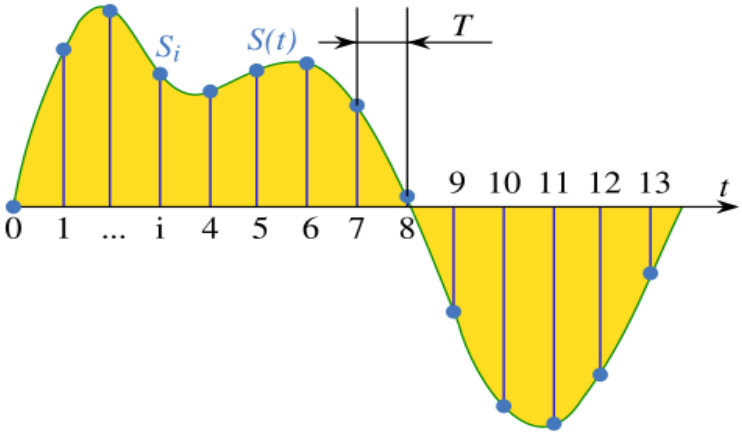
2.1 Analog To Digital Conversion Steps

The conversion from analog signal to a digital signal in an analog to digital converter is explained below using the block diagram given above.

Sample

The **sample** block function is to sample the input analog signal at a specific time interval. The samples are taken in **continuous amplitude** & possess real value but they are **discrete** with respect to **time**.

The sampling frequency plays important role in the conversion. So it is maintained at a specific rate. The sampling rate is set according to the requirement of the system.



Hold

The second block used in ADC is the '**Hold'** block. It has no function. It only holds the sample amplitude until the next sample is taken. The hold value remains unchanged till the next sample.

Quantize

This block is used for **quantization**. It converts the analog or continuous amplitude into discrete amplitude.

The on hold continuous amplitude value in hold block goes through '**quantize'** block & becomes **discrete** in **amplitude**. The signal is now in digital form as it has **discrete time** & **discrete amplitude**.

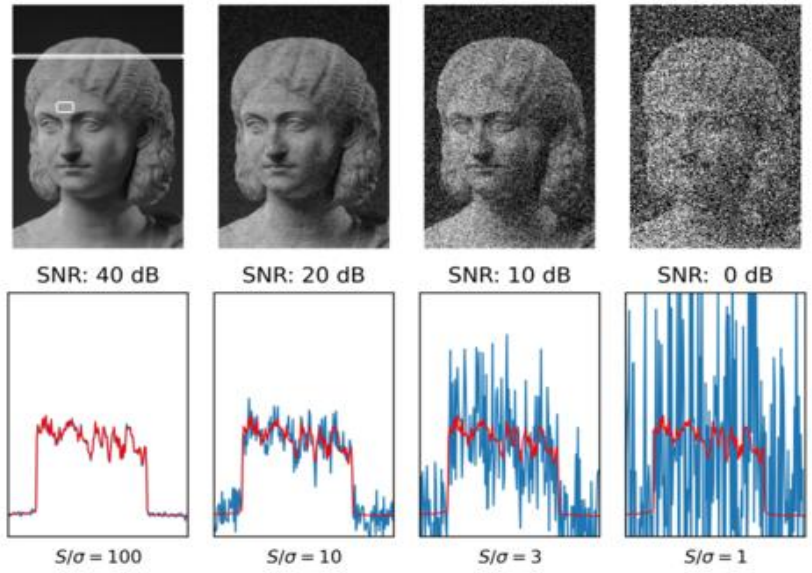
Encoder

The **encoder** block converts the digital signal into **binary form** i.e. into bits.

As we know that the digital devices operate on binary signals so it is necessary to convert the digital signal into the binary form using the Encoder.

This is the whole process of converting an Analog signal into digital form using an **Analog to Digital Converter**. This whole conversion occurs in a microsecond.

- Related Post: Binary Multiplier – Types & Binary Multiplication Calculator



Effect of Noise on signal

Quantification error in Red

3. Parametres:

Resolution

[edit]

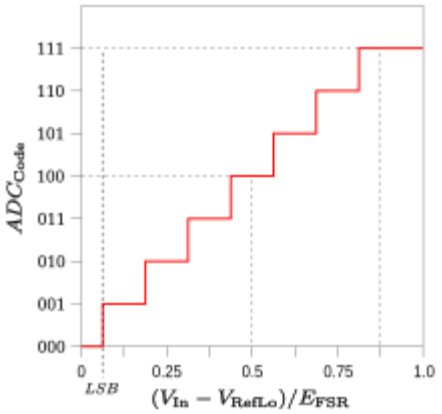


Fig. 1. An 8-level ADC coding scheme

The resolution of the converter indicates the number of different, i.e. discrete, values it can produce over the allowed range of analog input values. Thus a particular resolution determines the magnitude of the quantization error and therefore determines the maximum possible signal-to-noise ratio for an ideal ADC without the use of oversampling. The input samples are usually stored electronically in binary form within the ADC, so the resolution is usually expressed as the audio bit depth. In consequence, the number of discrete values available is usually a power of two. For

example, an ADC with a resolution of 8 bits can encode an analog input to one in 256 different levels ($2^8 = 256$). The values can represent the ranges from 0 to 255 (i.e. as unsigned integers) or from −128 to 127 (i.e. as signed integer), depending on the application.

Resolution can also be defined electrically, and expressed in volts. The change in voltage required to guarantee a change in the output code level is called the least significant bit (LSB) voltage. The resolution Q of the ADC is equal to the LSB voltage. The voltage resolution of an ADC is equal to its overall voltage measurement range divided by the number of intervals:

where M is the ADC's resolution in bits and E_{FSR} is the full-scale voltage range (also called 'span'). E_{FSR} is given by

where V_{RefHi} and V_{RefLow} are the upper and lower extremes, respectively, of the voltages that can be coded.

Normally, the number of voltage intervals is given by

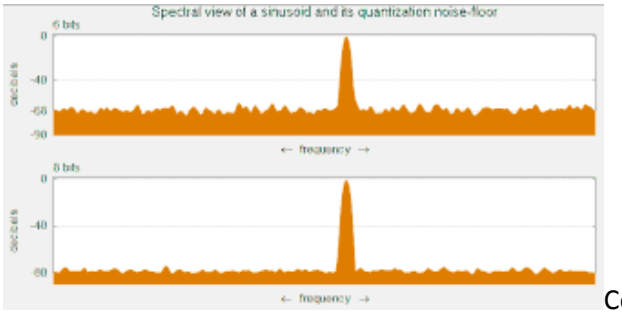
where M is the ADC's resolution in bits.[1]

That is, one voltage interval is assigned in between two consecutive code levels.

Example:

- Coding scheme as in figure 1
- Full scale measurement range = 0 to 1 volt
- ADC resolution is 3 bits: $2^3 = 8$ quantization levels (codes)
- ADC voltage resolution, $Q = 1\ V / (2^3 - 1) = 0.143\ V$ (intervals)

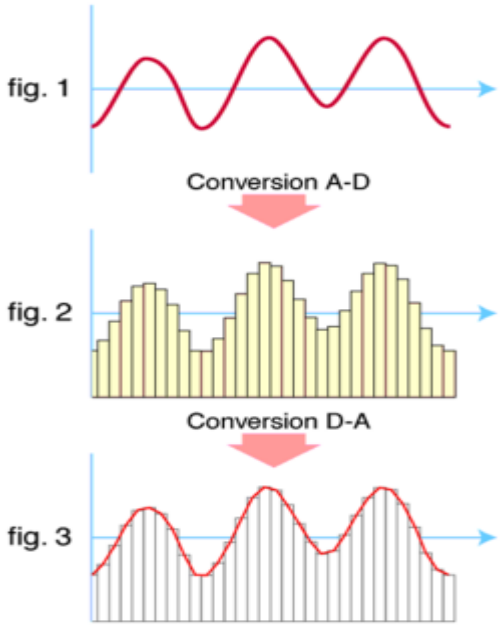
In many cases, the useful resolution of a converter is limited by the signal-to-noise ratio (SNR) and other errors in the overall system expressed as an ENOB.



Comparison of quantizing a sinusoid to 64 levels (6 bits) and 256 levels (8 bits). The additive noise created by 6-bit quantization is 12 dB greater than the noise created by 8-bit quantization. When the spectral distribution is flat, as in this example, the 12 dB difference manifests as a measurable difference in the noise floors.

Quantization error

[edit]



Analog to digital conversion as shown with fig. 1 and fig. 2

Quantization error is introduced by the quantization inherent in an ideal ADC. It is a rounding error between the analog input voltage to the ADC and the output digitized value. The error is nonlinear and signal-dependent. In an ideal ADC, where the quantization error is uniformly distributed between −½ LSB and +½ LSB, and the signal has a uniform distribution covering all quantization levels, the signal-to-quantization-noise ratio (SQNR) is given by

$$SQNR = \frac{\frac{V}{\sqrt{2}}}{2^{N-1} \cdot \sqrt{12}} = \frac{V}{\sqrt{2}} \frac{2^{N-1} \cdot \sqrt{12}}{V} = 2^N \sqrt{\frac{3}{2}}$$

$$SQNR_{dB} = 20 \log_{10}\left(2^N \sqrt{\frac{3}{2}}\right) = 20\left(N \log_{10} 2 + \log_{10} \sqrt{\frac{3}{2}}\right) = 6.02N + 1.7609$$

where Q is the number of quantization bits. For example, for a 16-bit ADC, the quantization error is 96.3 dB below the maximum level.

Quantization error is distributed from DC to the Nyquist frequency. Consequently, if part of the ADC's bandwidth is not used, as is the case with oversampling, some of the quantization error will occur out-of-band, effectively improving the SQNR for the bandwidth in use. In an oversampled system, noise shaping can be used to further increase SQNR by forcing more quantization error out of band.

Dither

In ADCs, performance can usually be improved using dither. This is a very small amount of random noise (e.g. white noise), which is added to the input before conversion. Its effect is to randomize the state of the LSB based on the signal. Rather than the signal simply getting cut off altogether at low levels, it extends the effective range of signals that the ADC can convert, at the expense of a slight increase in noise. Dither can only increase the resolution of a sampler. It cannot improve the linearity, and thus accuracy does not necessarily improve.

Quantization distortion in an audio signal of very low level with respect to the bit depth of the ADC is correlated with the signal and sounds distorted and unpleasant.

With dithering, the distortion is transformed into noise. The undistorted signal may be recovered accurately by averaging over time. Dithering is also used in integrating systems such as electricity meters. Since the values are added together, the dithering produces results that are more exact than the LSB of the analog-to-digital converter.

Dither is often applied when quantizing photographic images to a fewer number of bits per pixel—the image becomes noisier but to the eye looks far more realistic than the quantized image, which otherwise becomes banded. This analogous process may help to visualize the effect of dither on an analog audio signal that is converted to digital.

Accuracy

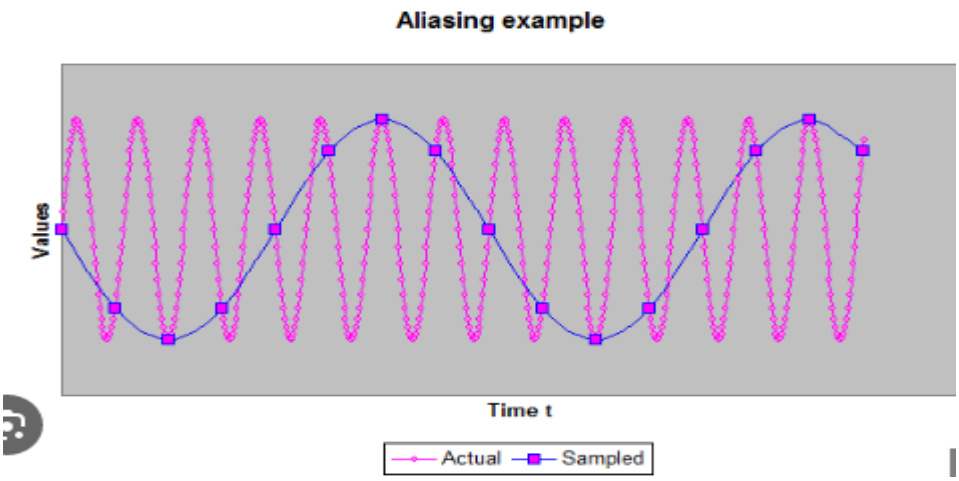
An ADC has several sources of errors. Quantization error and (assuming the ADC is intended to be linear) non-linearity are intrinsic to any analog-to-digital conversion. These errors are measured in a unit called the least significant bit (LSB). In the above example of an **eight-bit** ADC, an error of one LSB is $\frac{1}{256}$ of the full signal range, or about **0.4%.**

Sampling rate

An analog signal is continuous in time and it is necessary to convert this to a flow of digital values. It is therefore required to define the rate at which new digital values are sampled from the analog signal. The rate of new values is called the *sampling rate* or *sampling frequency* of the converter. A continuously varying bandlimited signal can be sampled and then the original signal can be reproduced from the discrete-time values by a reconstruction filter. The Nyquist–Shannon sampling theorem implies that a faithful reproduction of the original signal is only possible if the sampling rate is higher than twice the highest frequency of the signal.

Since a practical ADC cannot make an instantaneous conversion, the input value must necessarily be held constant during the time that the converter performs a conversion (called the *conversion time*). An input circuit called a sample and hold performs this task—in most cases by using a capacitor to store the analog voltage at the input, and using an electronic switch or gate to disconnect the capacitor from the input. Many ADC integrated circuits include the sample and hold subsystem internally.

Aliasing : Aliasing occurs because instantaneously sampling a function at two or fewer times per cycle results in missed cycles, and therefore the appearance of an incorrectly lower frequency. For example, a 2 kHz sine wave being sampled at 1.5 kHz would be reconstructed as a 500 Hz sine wave.



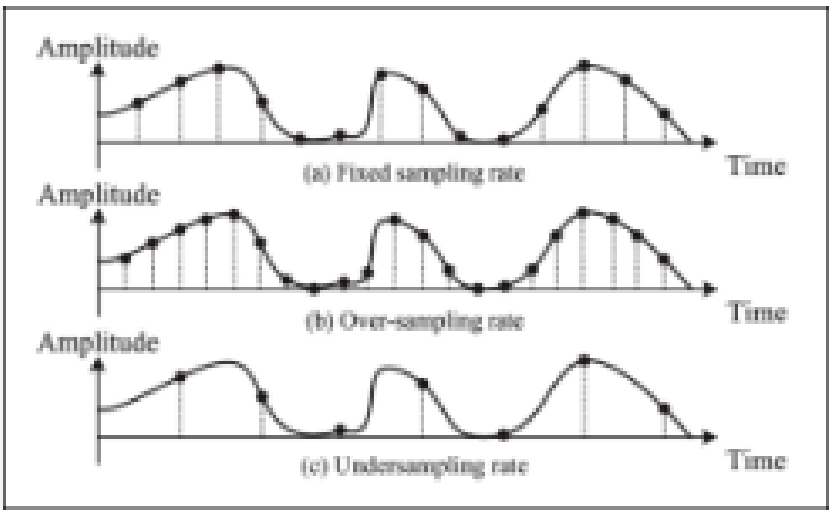
To avoid aliasing, the input to an ADC must be low-pass filtered to remove frequencies above half the sampling rate. This filter is called an *anti-aliasing filter*, and is essential for a practical ADC system that is applied to analog signals with higher frequency content. In applications where protection against aliasing is essential, oversampling may be used to greatly reduce or even eliminate it

Oversampling :

For economy, signals are often sampled at the minimum rate required with the result that the quantization error introduced is white noise spread over the whole passband of the converter. If a signal is sampled at a rate much higher than the Nyquist rate and then digitally filtered to limit it to the signal bandwidth produces the following advantages:

- Oversampling can make it easier to realize analog anti-aliasing filters
- Improved audio bit depth
- Reduced noise, especially when noise shaping is employed in addition to oversampling.

Oversampling is typically used in audio frequency ADCs where the required sampling rate (typically 44.1 or 48 kHz) is very low compared to the clock speed of typical transistor circuits (>1 MHz). In this case, the performance of the ADC can be greatly increased at little or no cost. Furthermore, as any aliased signals are also typically out of band, aliasing can often be eliminated using very low cost filters.



Factors Of ADC

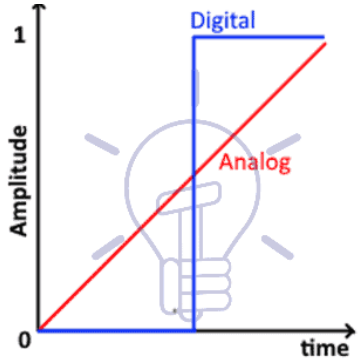
Resolution:

Resolution of an ADC is the **number of bits** that represents the digital signal's **amplitude**.

The analog signal has continuous amplitude. It can have **infinite values** i.e. real, floating basically any value one can imagine. On the other hand, the digital signal has a discrete and finite number of values. These discrete values are represented using **binary numbers** (bits).

To better understand the idea of resolution of **ADC**,

- 1-Bit Resolution



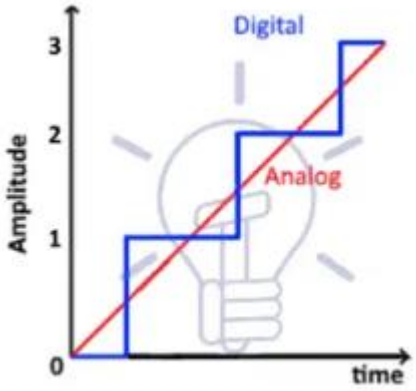
1-bit Resolution

The figure above shows an analog signal represented in a digital form which is either 0 or 1. This is a 1-bit resolution. The **resolution** of **ADC** defines its **number of steps**.

$$\textbf{number of steps} = 2^n$$

Where **n** is the number of bits. Therefore, there are 2 steps in 1-bit resolution.

- 2-Bit Resolution

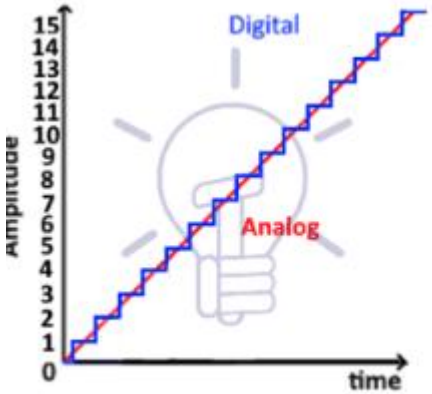


2-bit Resolution

This figure shows the conversion of analog to digital in 2-bit resolution. There are 4 steps or **quantization levels**.

$$\textbf{No of steps} = 2^n = 2^2 = 4$$

- 4-Bit Resolution



4-bit Resolution

This figure shows 4-bit resolution. The number of steps in 4-bit resolution is 16.

$$\textbf{No of steps} = 2^n = 2^4 = 16$$

The number of steps increases **exponentially** with increase in the **bit-resolution**. It also implies that by increasing the bits of resolution the converted digital signal becomes more like the original analog signal. So ideally, we can say that a digital signal with **infinite resolution** is an analog signal.

Related Post: Introduction to Signals, Types, Properties, Operation & Application

Width Of The Step

The voltage difference between two **adjacent steps** is known as the **width of the step**. It is denoted by **Δv**. So a single step represents a **fixed voltage** that is

$$\Delta v = v_{ref}/2^n$$

Where v_{ref} is the maximum voltage being converted & **n** represents the bits of resolution.
For example:

$$v_{ref} = 10.24v \ \& \ n = 10 \text{ bits}$$

Then:

$$\Delta v = 10.24/2^{10}$$
$$\Delta v = 10.24/1024$$
$$\Delta v = 0.01v$$

Thus the step-size or width of the step is **0.01v**. In this ADC, a single bit increase represents a **0.01v** of increase in the analog input. If analog input is increased by **0.01v** then the output is increased by **1 bit.**

Quantization Error

The **ADC** updates its value if the increase or decrease in its input voltage is greater than **Δv/2**. Any change less than **Δv/2** will not be registered. This is known as **Quantization Error**.
In other words, the difference between the input and the digital round-off figure of output is known as quantization error.

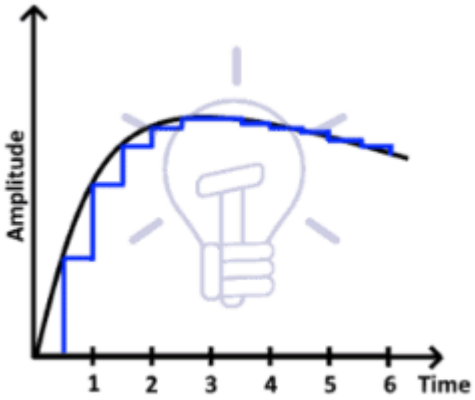
The increase in the **resolution** of the ADC decreases the **step-size** if the v_{ref} remain constant. Consequently, the **quantization error** decreases.

- Related Post: Types of Modulation Techniques used in Communication Systems

Sampling Rate

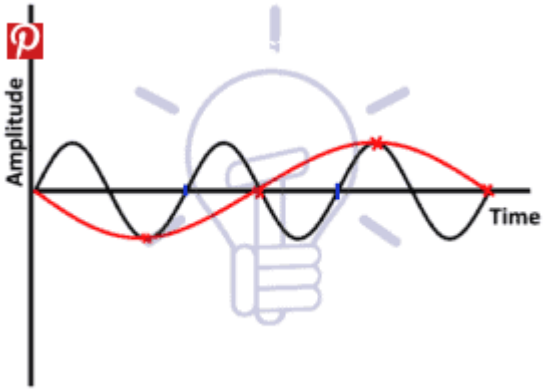
The number of samples taken during a single second is known as **sampling rate** or **sampling frequency**.
The sampling rate should be set according to the input signal. It should not be **very low** or **very high**.

Example of sampling:



This example shows that the sampling rate is 0.5 sec, as it takes 2 samples in one second.

Aliasing



If the sampling rate is **very low** then the resultant signal will not look anything like the original signal. In fact, it will become a different signal after reconstruction. This problem is known as **aliasing**.

To avoid this problem, the sampling rate should be kept **higher** than **twice the frequency** of the input signal. **Anti-aliasing filters** are also used for removing the frequency components higher than one half of the sampling rate. it blocks the aliasing components from being sampled.

Nyquist Criteria

Nyquist criteria suggest the **minimum** possible **sampling rate** for an analog signal which can be **reconstructed** successfully. If the highest frequency of the analog signal is **f**, the signal can be reconstructed successfully from its samples, if the samples are taken at a sampling frequency greater than **2f**.

Offset

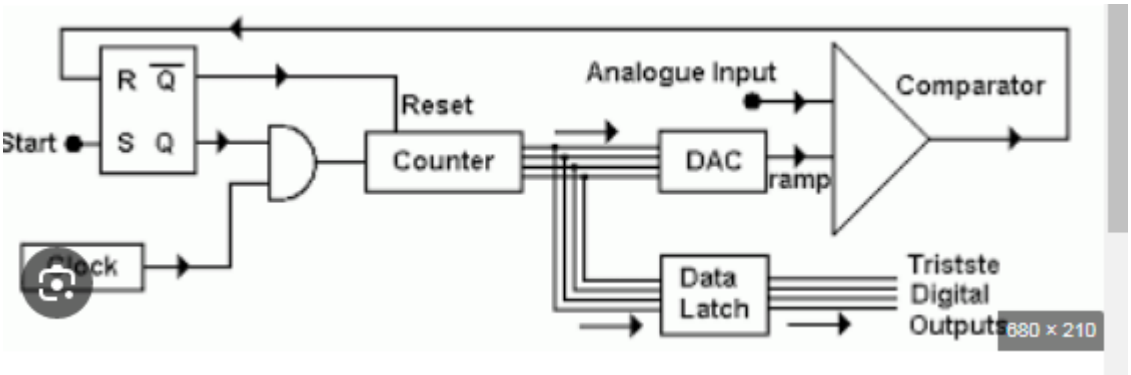
The **offset** in ADC is the **shift** in the digital output. For example, for input $\mathbf{v_{in} = 0}$, the output might not necessarily be digital **0**. It can be digital **5**, which will be the offset of the **ADC**.

Related Post: What is Quantization & Sampling? Types & Laws of Compression

4. Type of ADCs

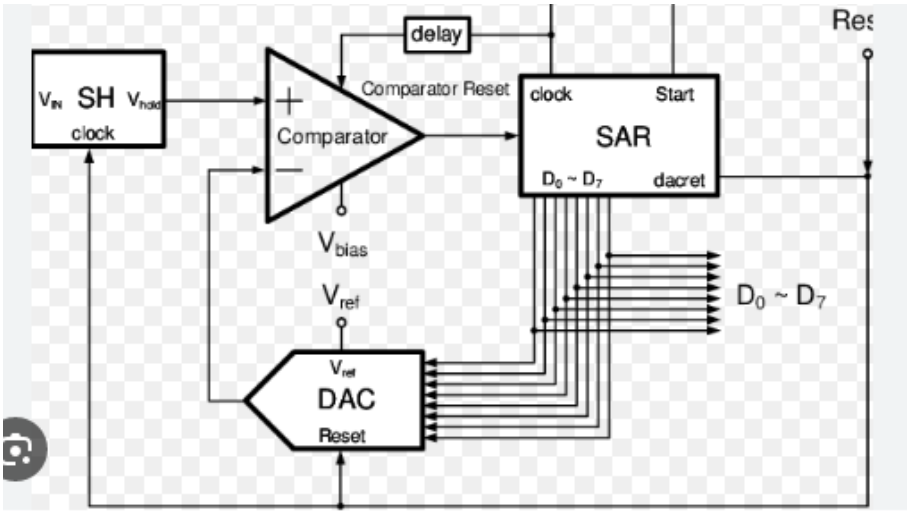
4.1 Ramp-compare

A ramp-compare ADC produces a saw-tooth signal that ramps up or down then quickly returns to zero.[14] When the ramp starts, a timer starts counting. When the ramp voltage matches the input, a comparator fires, and the timer's value is recorded. Timed ramp converters can be implemented economically,[a] however, the ramp time may be sensitive to temperature because the circuit generating the ramp is often a simple analog integrator. A more accurate converter uses a clocked counter driving a DAC. A special advantage of the ramp-compare system is that converting a second signal just requires another comparator and another register to store the timer value. To reduce sensitivity to input changes during conversion, a sample and hold can charge a capacitor with the instantaneous input voltage and the converter can time the time required to discharge with a constant current.



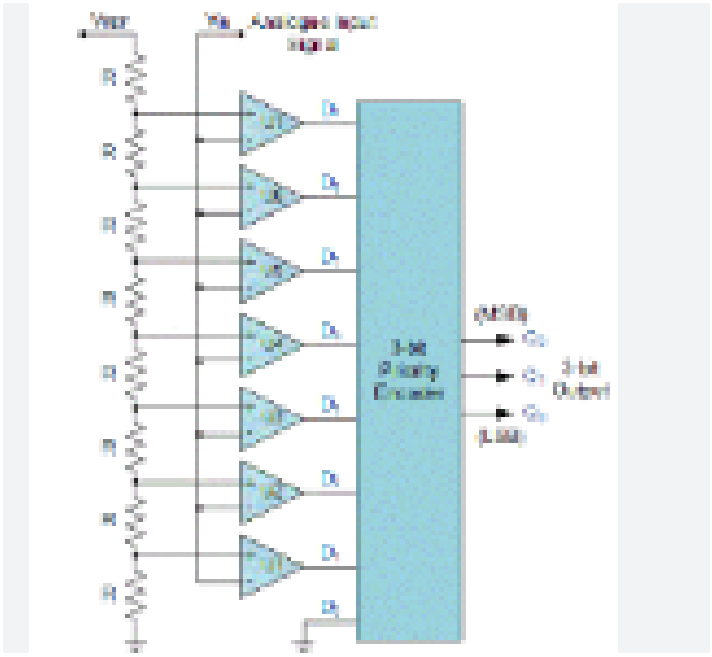
4.2 Successive approximation

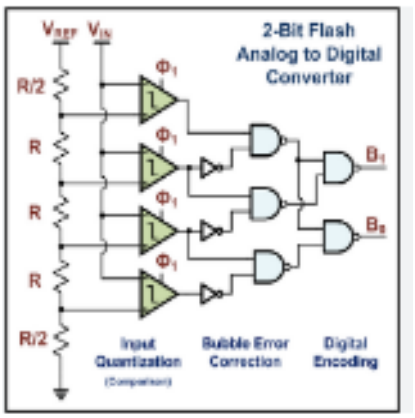
A successive-approximation ADC uses a comparator and a binary search to successively narrow a range that contains the input voltage. At each successive step, the converter compares the input voltage to the output of an internal digital-to-analog converter (DAC) which initially represents the midpoint of the allowed input voltage range. At each step in this process, the approximation is stored in a successive approximation register (SAR) and the output of the digital-to-analog converter is updated for a comparison over a narrower range.



4.3. A **flash ADC**

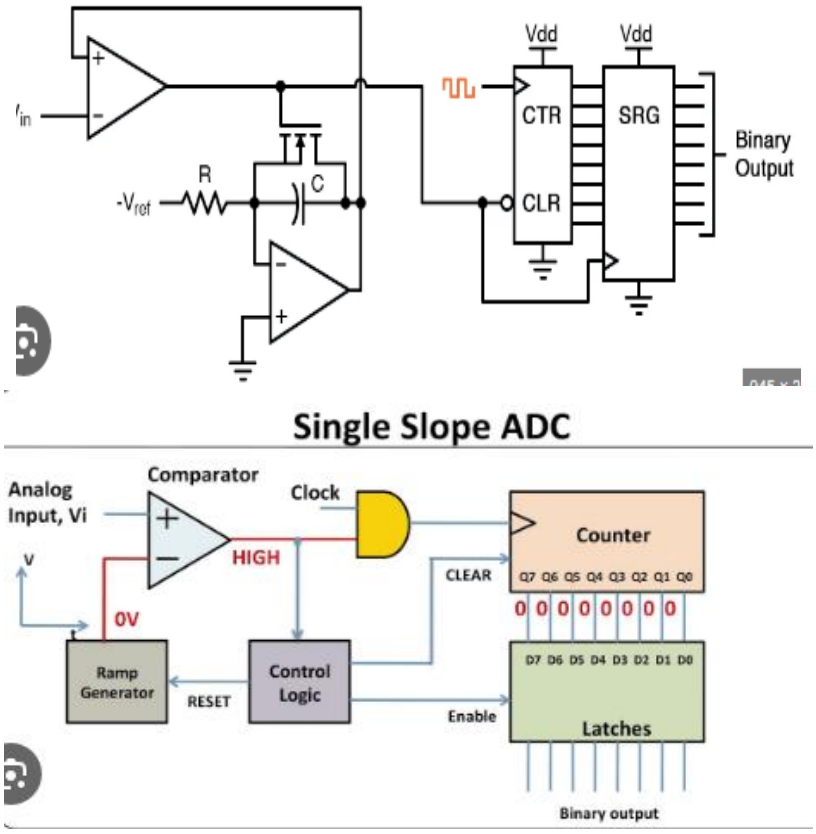
A **flash ADC** (also known as a **direct-conversion ADC**) is a type of analog-to-digital converter that uses a linear voltage ladder with a comparator at each "rung" of the ladder to compare the input voltage to successive reference voltages. Often these reference ladders are constructed of many resistors; however, modern implementations show that capacitive voltage division is also possible. The output of these comparators is generally fed into a digital encoder, which converts the inputs into a binary value (the collected outputs from the comparators can be thought of as a unary value).





4.4 Integrating

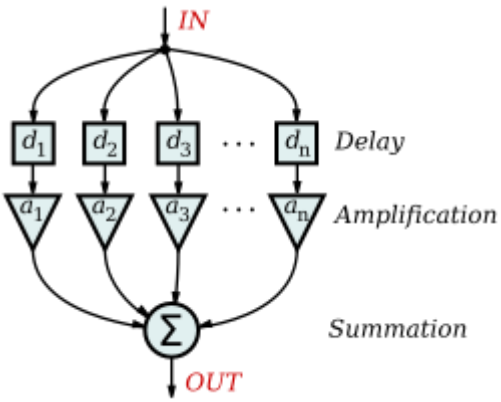
An **integrating ADC** (also **dual-slope** or **multi-slope** ADC) applies the unknown input voltage to the input of an integrator and allows the voltage to ramp for a fixed time period (the run-up period). Then a known reference voltage of opposite polarity is applied to the integrator and is allowed to ramp until the integrator output returns to zero (the run-down period). The input voltage is computed as a function of the reference voltage, the constant run-up time period, and the measured run-down time period. The run-down time measurement is usually made in units of the converter's clock, so longer integration times allow for higher resolutions. Likewise, the speed of the converter can be improved by sacrificing resolution. Converters of this type (or variations on the concept) are used in most digital voltmeters for their linearity and flexibility.



5. Applications:

Digital filtering : In signal processing, a **digital filter** is a system that performs mathematical operations on a sampled, discrete-time signal to reduce or enhance certain aspects of that signal. This is in contrast to the other major type of electronic filter, the analog filter, which is typically an electronic circuit operating on continuous-time analog signals.

A digital filter system usually consists of an analog-to-digital converter (ADC) to sample the input signal, followed by a microprocessor and some peripheral components such as memory to store data and filter coefficients etc. Program Instructions (software) running on the microprocessor implement the digital filter by performing the necessary mathematical operations on the numbers received from the ADC. In some high performance applications, an FPGA or ASIC is used instead of a general purpose microprocessor, or a specialized digital signal processor (DSP) with specific paralleled architecture for expediting operations such as filtering.[1][2]

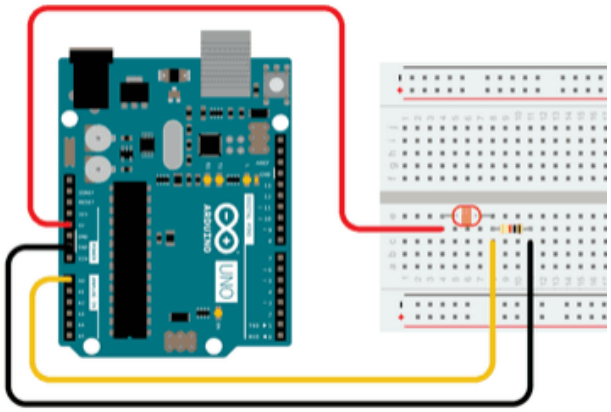


In the modern world of growing technology, we are dependent on digital devices. These digital devices operate on the digital signal. But not every quantity is in digital form instead they are in analog form. So an ADC is used for converting analog signals into digital signals. The **applications** of ADC are limitless. Some of these **applications** given below:

- **Cell phones** operate on the digital voice signal. Originally the voice is in analog form, which is converted through ADC before feeding to the cell phone transmitter.
- **Images** and **videos** captured using camera is stored in any digital device, is also converted into digital form using ADC.
- Medical Imaging like **x-ray** & **MRI** also uses **ADC** to convert images into Digital form before modification. They are then modified for better understanding.
- Music from the **cassette** is also converted into the digital form such as **CDs** and **thumb drives** using **ADC** converters.
- **Digital Oscilloscope** also contains **ADC** for converting Analog signal into a digital signal for display purposes & different other features.
- **Air conditioner** contains **temperature sensors** for maintaining the room temperature. This temperature is converted into digital form using ADC so that onboard controller can read & adjust the cooling effect.

In today's modern world almost every device has become the **digital version** of itself & they need to have ADC in it. Because it has to operate in digital domain which can be only acquired using **analog to digital converter (ADC)**.

6. Example of using ADC of Arduino



- When we interface sensors to the microcontroller, the output of the sensor many of the times is analog in nature. But the microcontroller processes digital signals.
- Hence, we use ADC in between the sensor and microcontroller. It converts an analog signal into a digital and gives it to the microcontroller.
- There are many applications of ADC like in a biometric application, Environment monitoring, Gas leakage detection etc.

Arduino Uno has 6 0n-board ADC channels which can be used to read analog signal in the range 0-5V. It has 10-bit ADC means it will give digital value in the **range of 0 – 1023 (2^10).** This is called as a resolution which indicates the number of discrete values it can produce over the range of analog values.

Digital Output value Calculation

ADC Resolution = Vref / ((2^n) - 1)

Digital Output = Vin / Resolution

Where,

Vref - The reference voltage is the maximum value that the ADC can convert.

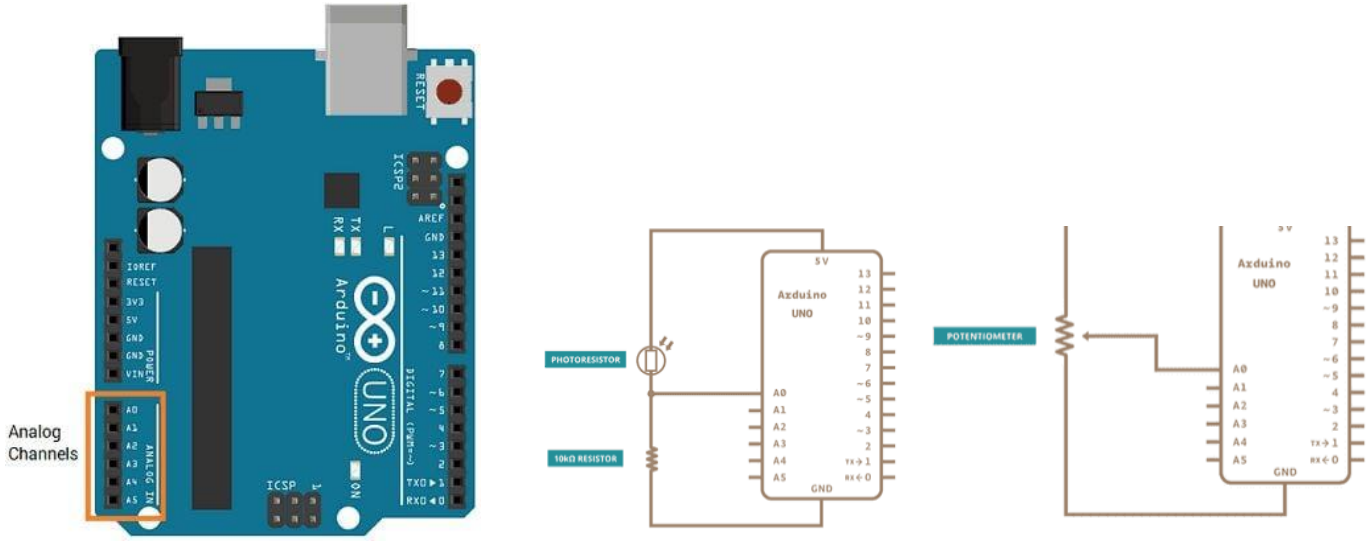
To keep things simple, let us consider that Vref is 5V,

For 0 Vin, digital o/p value = 0

For 5 Vin, digital o/p value = 1023 (10-bit)

For 2.5 Vin, digital o/p value = 512 (10-bit)

ADC Pins of Arduino Uno



Arduino ADC pin Diagram

Analog Functions for Arduino ADC

- **analogRead (pin)**
  This function is used to read analog value from specified analog pin.
  `pin` - number of analog pin which we want to read
  `returns` - digital value 0 – 1023

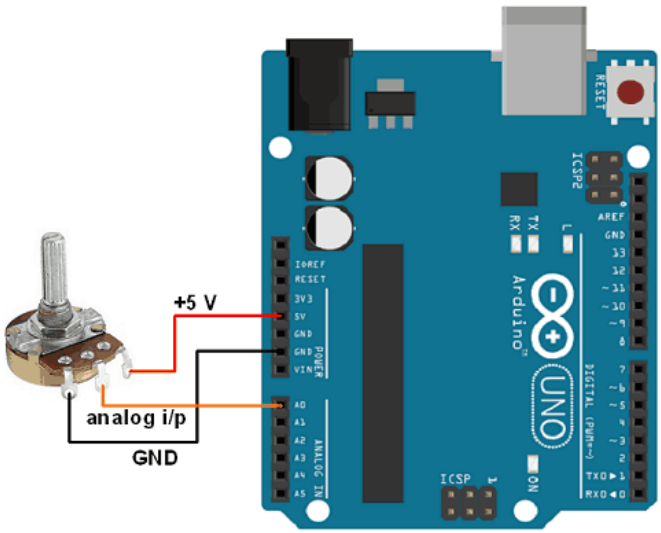
```
e.g. analogRead(A0) //read analog value at A0 channel
```

- **analogReference (type)**
  This function is used for configuring the reference voltage used for analog input.

let's see How to Read Analog values using Arduino

Let’s write a program to read varying analog value generated using potentiometer which is connected to A0 analog channel. Display the digital value on Serial monitor which we got from the Arduino ADC.

Potentiometer Interfacing with Arduino Uno



Potentiometer connected Arduino ADC Channel

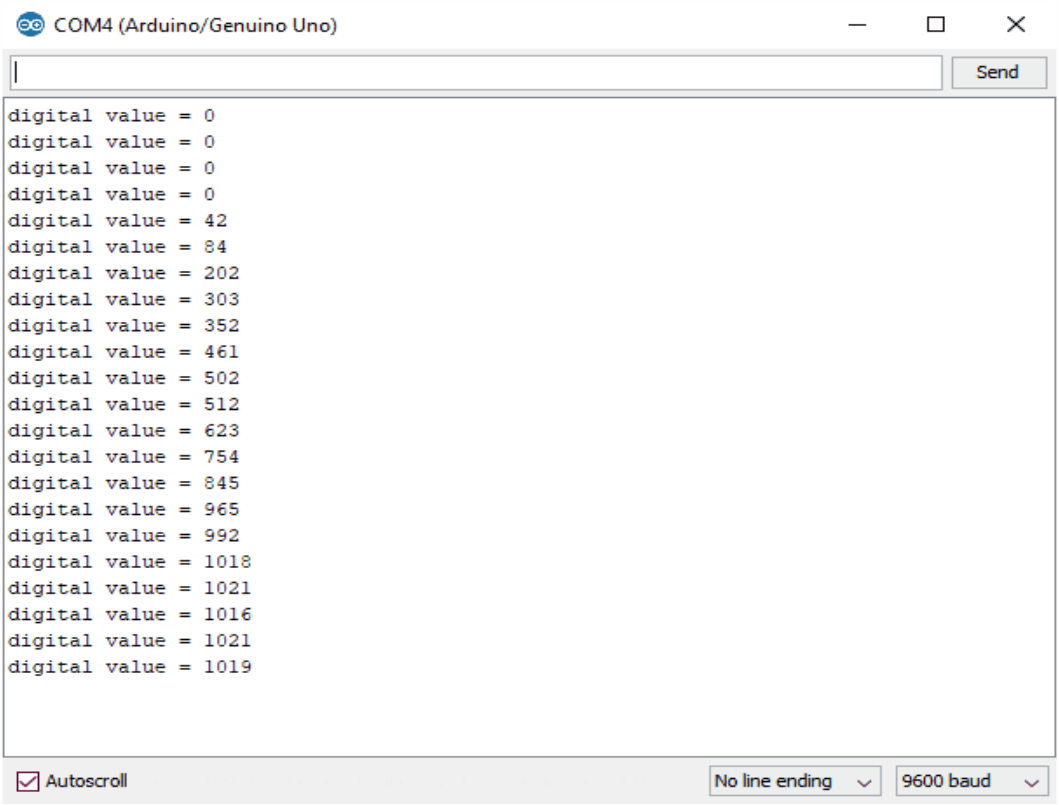
Arduino Code for reading analog value

```
int sensorPin = A0;   // input pin for the potentiometer
int digitalValue = 0;// variable to store the value coming from the sensor

void setup() {
  Serial.begin(9600);
}

void loop() {
  digitalValue = analogRead(sensorPin);// read the value from the analog channel
  Serial.print("digital value = ");
  Serial.println(digitalValue);          //print digital value on serial monitor
  delay(1000);
}
```

Output on Serial Monitor



Note: If nothing is connected to analog input channel then the analogRead () function return the noisy fluctuating value.

Read Analog Voltage using Arduino Uno

As ADC provide digital output which is proportional to analog value. To know what is input analog value, we need to convert this digital value back to analog value through program. To convert this digital value to analog input voltage,

Aout = digital value * (Vref/2^n – 1)

e.g. digital value = 512 and ADC is 10-bit with 5V Vref. But, we want to know that for what analog voltage it is giving respective digital value. Then,

Aout = 512 * (5 V / 1023)
= 2.5 V

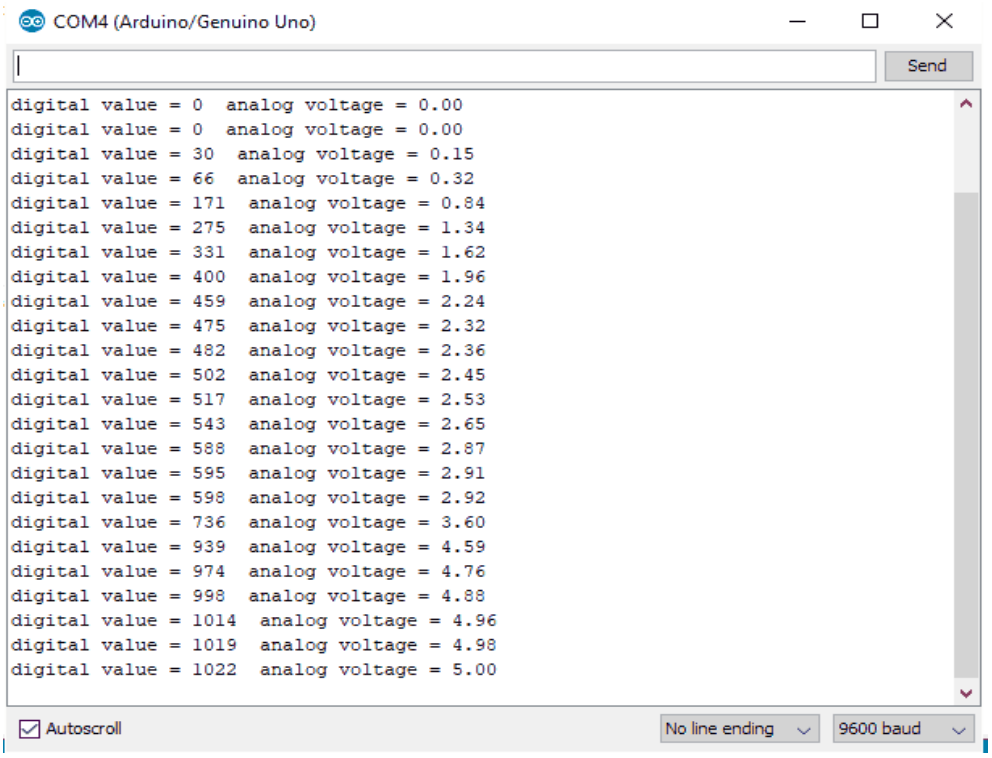
Code for Read Analog Voltage using Arduino

```
int sensorPin = A0;   // select the input pin for the potentiometer
int digitalValue = 0;  // variable to store the value coming from the sensor
float analogVoltage = 0.00;

void setup() {
  Serial.begin(9600);
}

void loop() {
  digitalValue = analogRead(sensorPin);// read the value from the analog channel
  Serial.print("digital value = ");
  Serial.print(digitalValue);        //print digital value on serial monitor
  //convert digital value to analog voltage
  analogVoltage = (digitalValue * 5.00)/1023.00;
  Serial.print("  analog voltage = ");
  Serial.println(analogVoltage);
  delay(1000);
}
```

Output on Serial Window



References

[1] https://www.electronicwings.com/arduino/adc-in-arduino
[2] https://www.knowelectronic.com/digital-to-analog-converter/
[3] https://www.youtube.com/watch?v=OBcCZkCf-OI

- Joystick interfacing with Arduino Uno
- Monitor Room temperature by interfacing LM35 to Arduino UNO
- Accelerometer ADXL335 interfacing with Arduino UNO