TECHNOCRATS ROBOTICS
leave the thinking to us...

ANALOG TO DIGITAL CONVERTERS SIGNIFICANCE

SET-3

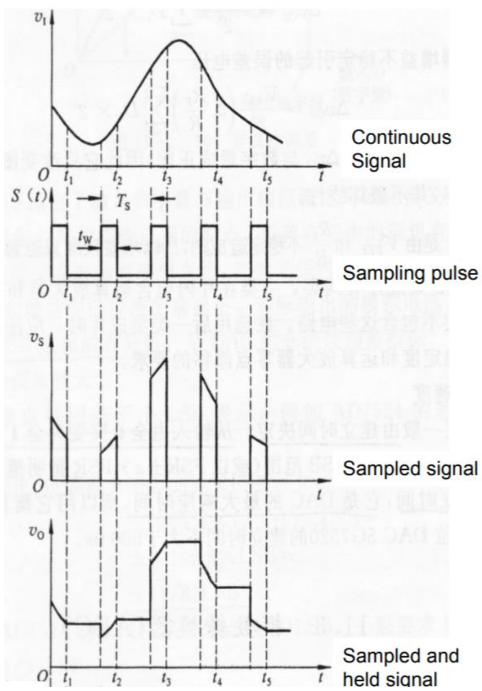

Figure 1. Representation of Analog to Digital Conversion

By

AAYUSH KUMAR (Electrical)

What is Analog to Digital Converter?

An electronic integrated circuit which transforms a signal from analog (continuous) to digital (discrete) form.

Analog signals are directly measurable quantities.

Digital signals only have two states. For digital computer, we refer to binary states, 0 and 1.

Why are they needed?

Microprocessors can only perform complex processing on digitized signals.

When signals are in digital form they are less susceptible to the deleterious effects of additive noise.

ADC Provides a link between the analog world of transducers and the digital world of signal processing and data handling.

Applications of Analog to Digital Converter

ADC are used virtually everywhere where an analog signal has to be processed, stored, or transported in digital form.

Some examples of ADC usage are digital volt meters, cell phone, thermocouples, and digital oscilloscope.

Microcontrollers commonly use 8, 10, 12, or 16 bit ADCs, our micro controller uses an 8 or 10 bit ADC.

Process

It includes 2 steps:

1) Sampling and Holding

2) Quantizing and Encoding

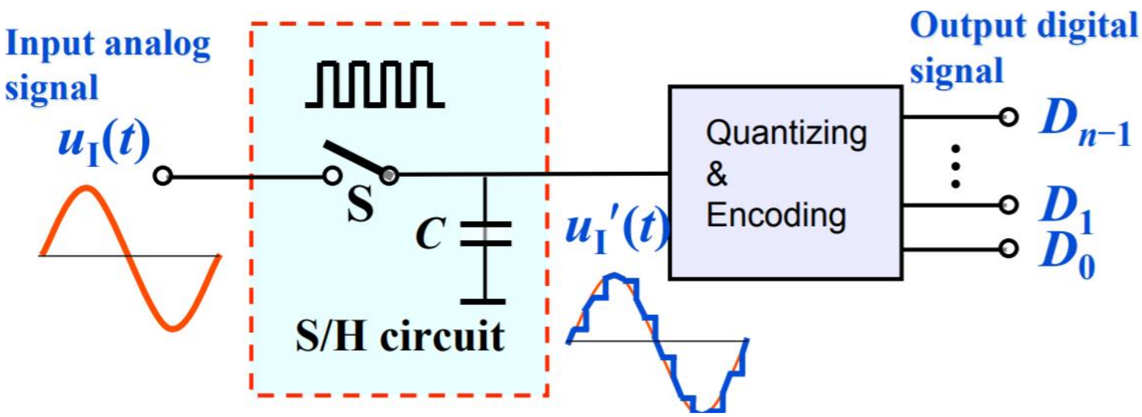

Figure 2. Process of conversion

SAMPLING AND HOLDING

Holding signal benefits, the accuracy of the A/D conversion

Minimum sampling rate should be at least twice the highest data frequency of the analog signal

QUANTIZING AND ENCODING

Quantizing:

Partitioning the reference signal range into a number of discrete quanta, then matching the input signal to the correct quantum.

Encoding:

Assigning a unique digital code to each quantum, then allocating the digital code to the input signal.

Analog to Digital Converter in Arduino

Figure 3. Analog pins in Arduino

Not every pin on a microcontroller has the ability to do analog to digital conversions. On the Arduino board, these pins have an 'A' in front of their label (A0 through A5) to indicate these pins can read analog voltages.

ADCs can vary greatly between microcontroller. The ADC on the Arduino is a 10-bit ADC meaning it has the ability to detect 1,024 (2^{10}) discrete analog levels. Some microcontrollers have 8-bit ADCs (2^8 = 256 discrete levels) and some have 16-bit ADCs (2^{16} = 65,536 discrete levels).

The way an ADC works is fairly complex. There are a few different ways to achieve this feat (see Wikipedia for a list), but one of the most common technique uses the analog voltage to charge up an internal capacitor and then measure the time it takes to discharge across an internal resistor. The microcontroller monitors the number of clock cycles that pass before the capacitor is discharged. This number of cycles is the number that is returned once the ADC is complete.

Relating ADC Value to Voltage

The ADC reports a ratiometric value. This means that the ADC assumes 5V is 1023 and anything less than 5V will be a ratio between 5V and 1023.

$$\frac{Resolution\ of\ the\ ADC}{System\ Voltage} = \frac{ADC\ Reading}{Analog\ Voltage\ Measured}$$

Analog to digital conversions are dependent on the system voltage. Because we predominantly use the 10-bit ADC of the Arduino on a 5V system, we can simplify this equation slightly:

$$\frac{1023}{5} = \frac{ADC\ Reading}{Analog\ Voltage\ Measured}$$

If your system is 3.3V, you simply change 5V out with 3.3V in the equation. If your system is 3.3V and your ADC is reporting 512, what is the voltage measured? It is approximately 1.65V.

If the analog voltage is 2.12V what will the ADC report as a value?

$$\frac{1023}{5.00V} = \frac{x}{2.12V}$$

Rearrange things a bit and we get:

$$\frac{1023}{5.00V} * 2.12V = x$$

$$x = 434$$

Therefore, the ADC should report 434.

Arduino ADC Example

To show this in the real world let's use the Arduino to detect an analog voltage. Use a trimpot, or light sensor, or simple voltage divider to create a voltage. Let's setup a simple trimpot circuit for this example:

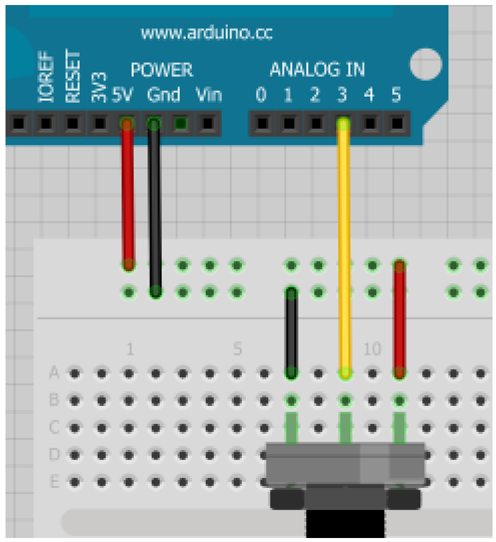

Figure 4. Example using Arduino

To start, we need to define the pin as an input. To match the circuit diagram, we will use A3:

```
pinMode(A3, INPUT);
int x = analogRead(A3); //Reads the analog value on pin A3 into x
Serial.print("Analog value: ");
Serial.println(x);
```

Significance

Analog to digital conversion plays a crucial role in the world we live in today. Most signals (e.g. a song or speech) are analog in nature. However, if you want to transmit them all over the world to multiple people at once, or if you want to enhance its quality by removing background noise, you will need to convert them into digital form.

This need to convert analog data into its digital equivalent stems from the fact that our computers and microprocessors can only handle digital data. By converting analog data into its digital equivalent, you open the door to a million different possibilities. Digital data can be read, stored, transmitted and even manipulated (if need be) by computers.

However, just like your voice (an analog signal) needs a medium (in most cases, air) to travel to your audience, digital media also requires a communication channel or medium.

Unlike analog data, digital data cannot be transmitted through the air as its binary data that holds no physical form! Digital data can however be sent to and from devices by using Digital to Analog Convertors (DACs) and Analog to Digital Convertors (ADCs).

The Role of Analog to Digital Convertor (ADC)

The ADC plays an important role in the transmission of data from one device to another. For example, if you're sending a recording of your voice to a friend, you will need both the ADC and the DAC.

When you press record on your phone, you input analog information to the recorder (in this case, your voice). The phone makes use of an ADC to immediately convert this data to digital form so that it can be processed and edited. The microcomputer then sends this digital data to your target audience using a digital channel of communication like the internet.

However, digital data holds no meaning to your audience. They do not want to hear or see a bunch of 0s and 1s! Therefore, this digital data must be converted back into its original analog form. This is where the DAC comes into play. By transforming this data into its analog form, your audience can play and listen to the recording you sent them with ease! The entire process can be summarized by the following flowchart:

Figure 5. Flowchart of the process

Selecting an ADC

Unfortunately, the process of selecting an ADC is a complicated task. We need to carefully analyse a plethora of different aspects like power consumption, energy rating, resolution, and frequency response, etc. when making your choice.

That being said, the most important factor when choosing an ADC is its sampling rate. Simply put, if the sampling rate of your ADC violates the Nyquist criteria, our system will suffer from aliasing and loss of information.

The Nyquist Theorem states that:

"In order to reproduce a signal without any distortion or loss of data, the sampling frequency must be greater than twice the maximum signal frequency or twice the bandwidth."

Mathematically,

$$fs \geq 2\ fm$$

Where, fm = Maximum frequency in a sample

And, fs = The sampling frequency of the ADC/DAC