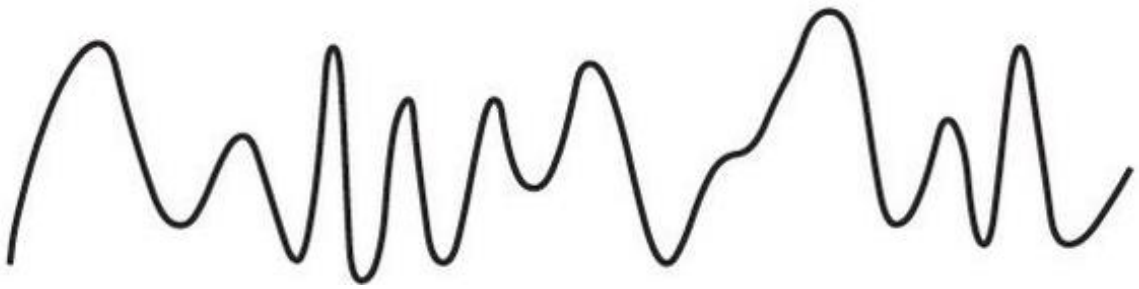
Introduction To Analog World

We live in an ***analog world***. Every parameter in our life can have infinite possibilities of values. Temperature, pressure, colors , etc. … even between the value 0 and the value 1 there are ***infinite values***.



But in the ***Embedded System world***, we can process only digital values. Therefor, we need a ***translation unit*** that can convert any signal from its original analog form, to a digital form that can be processed by the processor.

ADC Definition

The Analog To Digital Converter (ADC) is an electronic circuit which converts a signal from ***analog form*** (continuous) to ***digital form*** (discrete) to allow it to be processed by the processor.

There are many types of ADC circuits, how ever the most commonly used one in Embedded Systems applications called ***Successive Approximation ADC*** (SAR ADC).

Its basic idea depends on a counter that counts on a clock signal called ***ADC Clock***, and every count it increases a ***compare analog signal voltage value*** that is compared to the input signal, the counter stop counting when the compare signal is approximately equals to the input signal. Then the counter value represent the digital value of the input signal.

ADC Terminology

ADC Resolution

Number of the bits that represent the output digital value.

Reference Voltage

Maximum Voltage could be measured by the ADC, at this voltage all bits of the output value shall be 1

ADC Step

The analog value needed to increase the digital value by 1

$$\text{Step} = \frac{\text{Reference Voltage}}{2^{\text{ADC Resolution}}}$$

Conversion Time

Time taken by ADC to convert analog signal to digital signal

ADC Clock

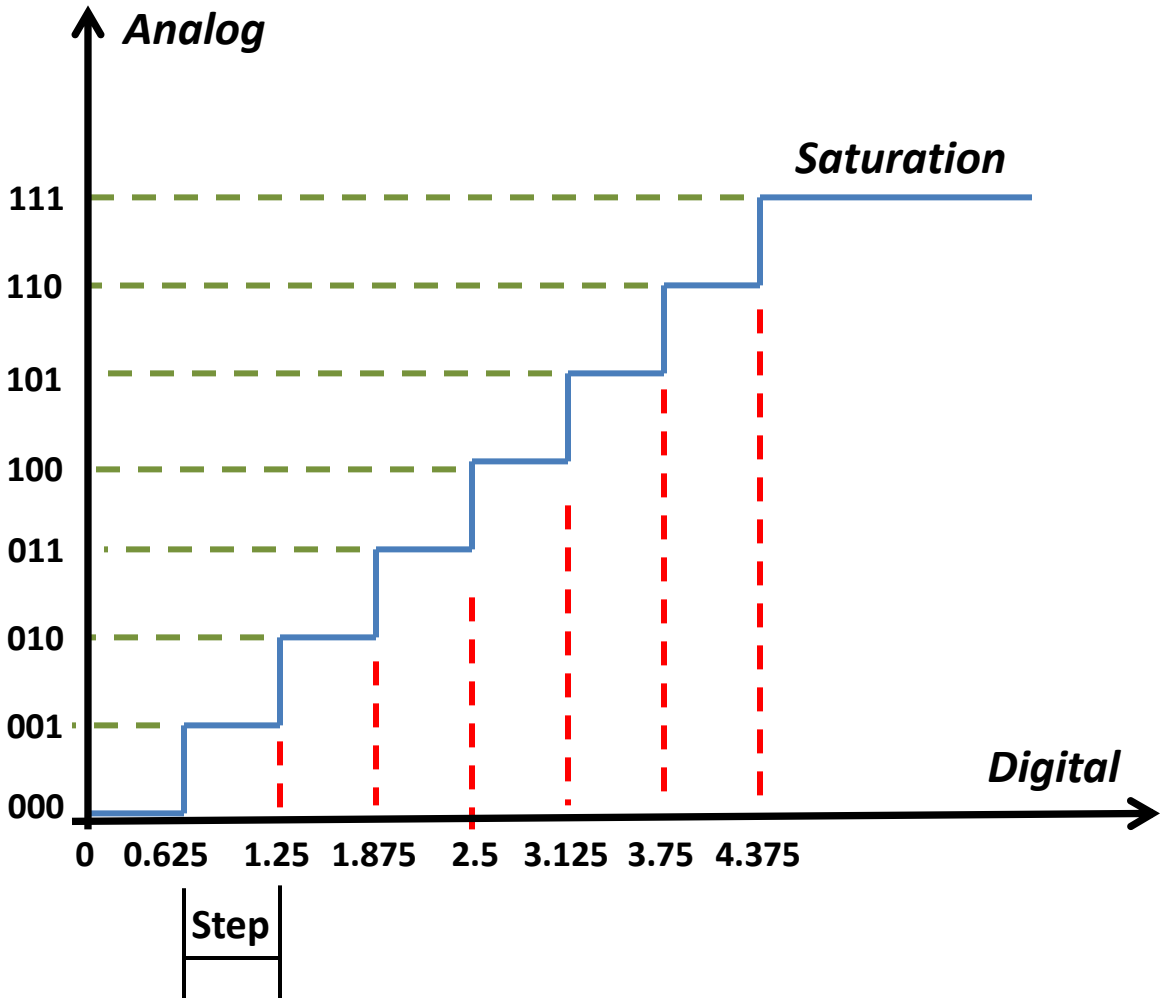
The input clock to the counter inside the ADC. The **higher** frequency of the ADC clock, the **lower** conversion time and **higher** power consumption.

3 Bit ADC Example

This graph shows the mapping between the analog values and digital values for ***3 bit ADC*** with reference voltage = 5V.

The 3 bit ADC has 8 possible digital values. So, simply the step = 8 / 5 = **0.625v**.

Important Note:
The ADC saturates at voltage =
Reference Voltage – 1 step.



Converting Digital Value to Analog Value

The ADC converts the analog value to digital value. In the code we need to make get back the original analog value to act on it.

$$\text{Analog Value} = \text{Digital Value} \times \text{Step} = \text{Digital Value} \times \frac{\text{Reference Voltage}}{2^{\text{ADC Resolution}}}$$

Important Noes:

1- In C programming, when you have an equation that contains multiplication and division, do the multiplication first

Important Notes

1- In C programming, when you have an equation that contains ***multiplication*** and ***division***, **do the multiplication first** in brackets ( ), then apply the division. For ex:

Assuming x = 10, y = 20 and z = 30.
Then: **result = (x/y) * z = 0**
while: **result = (x*z) / y = 15**

Mathematically, the two previous equations are the same. But in Programing **division int by int shall result in int too**. So x/y which mathematically = 0.5 in programming it gives 0. To avoid this problem do the multiplication first to increase the numerator and then do the division.

Important Notes

2- To increase the precision, using ***small units like milli volt*** in calculations ***is preferred*** when comparing to volt. For ex:

Assuming 8 bit ADC, and reference voltage is 5V. To get the analog value for a certain digital value we use the formula:

Analog value = (Digital Value * 5) / 256

The possible values could be obtained from this equation are 0, 1, 2, 3 and 4v although we have 256 digital possible value !
It means that many each 50 digital value will give the same analog Value.

But if we rewrite the equation in terms of milli volt, then:

Analog Value = (Digital Value *5000) / 256.

The output analog value will be in the range from 0 to 4980 approximately.

ADC in AVR Atmega 32

ATmega32 ADC features

The ADC peripheral of the ATmega32 has the following characteristics:

(a) It is a 10-bit ADC.

(b) It has 8 analog input channels, 7 differential input channels, and 2 differential input channels with optional gain of 10x and 200x.

(c) The converted output binary data is held by two special function registers called ADCL (A/D Result Low) and ADCH (A/D Result High).

(d) Because the ADCH:ADCL registers give us 16 bits and the ADC data out is only 10 bits wide, 6 bits of the 16 are unused. We have the option of making either the upper 6 bits or the lower 6 bits unused.

(e) We have three options for V_{ref}. V_{ref} can be connected to AVCC (Analog V_{cc}), internal 2.56 V reference, or external AREF pin.

(f) The conversion time is dictated by the crystal frequency connected to the XTAL pins (Fosc) and ADPS0:2 bits.

Programming ADC in atmega32

Steps in programming the A/D converter using polling

To program the A/D converter of the AVR, the following steps must be taken:

1. Make the pin for the selected ADC channel an input pin.
2. Turn on the ADC module of the AVR because it is disabled upon power-on reset to save power.
3. Select the conversion speed. We use registers ADPS2:0 to select the conversion speed.
4. Select voltage reference and ADC input channels. We use the REFS0 and REFS1 bits in the ADMUX register to select voltage reference and the MUX4:0 bits in ADMUX to select the ADC input channel.
5. Activate the start conversion bit by writing a one to the ADSC bit of ADCSRA.
6. Wait for the conversion to be completed by polling the ADIF bit in the ADCSRA register.
7. After the ADIF bit has gone HIGH, read the ADCL and ADCH registers to get the digital data output. Notice that you have to read ADCL before ADCH; otherwise, the result will not be valid.
8. If you want to read the selected channel again, go back to step 5.
9. If you want to select another V_{ref} source or input channel, go back to step 4.

LAB1

Implement a system that uses a potentiometer to switch LEDs on and off following these conditoins:

1- If the input voltage is more than 0 and less than 1.5 v, turn Red LED on.

2- If the input voltage is more than 1.5 and less than 3 v, turn Yellow LED on.

3- If the input voltage is more than 3v turn Geen LED on.