**Microprocessor Systems Design**

**EEE42101**

**Experiment 4: Analog Digital Conversion ADC**

# Objectives:

* Get experience with analog digital conversion peripheral.
* Apply analog processing in thermistors.
* Measure temperature.

# Tools:

1. PC
2. Arduino Nano board
3. Testing board
4. MiniB-USB cable

# Components:

1. NTC thermistor
2. 10kΩ resistor.

Note: all material and sources of this course will be available on:

<https://github.com/ashrafmalraheem/Microprocessor_Course>

Feel free to download, study and modify for your own projects.

# Analog to digital converter:

Introduction:

In the previous experiments, we have been dealing with digital signals. However, real world applications don’t have digital signals only. Physical systems can generate values that have infinite value range. Therefore, it is necessary for a microcontroller or any embedded system to be able to convert physical values into digital form. That job is performed by analog to digital conversion module.

The ADC reads analog values in range of 0 – Vcc. Any signal out of this range will cause the ADC to malfunction. The converted values in between (0)b – (2n-1)b where n is the number of digits.

## Analog to digital conversion:

In order to perform analog to digital conversion, there are three steps:

1. Sampling (sample and hold): take a sample of the analog input signal, because it is continuously varying.
2. Quantization: Convert the sampled signal into digital representation. Usually this is done by comparator, counter and a DAC which compare the result of DAC to the input sample.

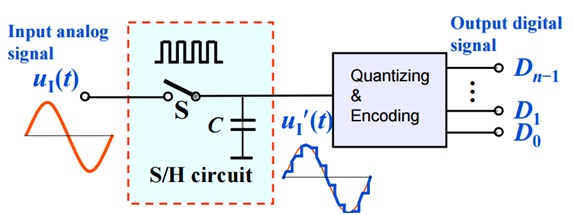


Figure 1 ADC steps: sampling and quantization[[1]](#footnote-1)

1. Conversion result: convert the quantized value into its analog value. The ADC will return a value between 0b – (2n-1)b. To obtain the actual reading it needs to be converted to corresponding voltage:

|  |  |
| --- | --- |
|  | (1) |
|  |  |

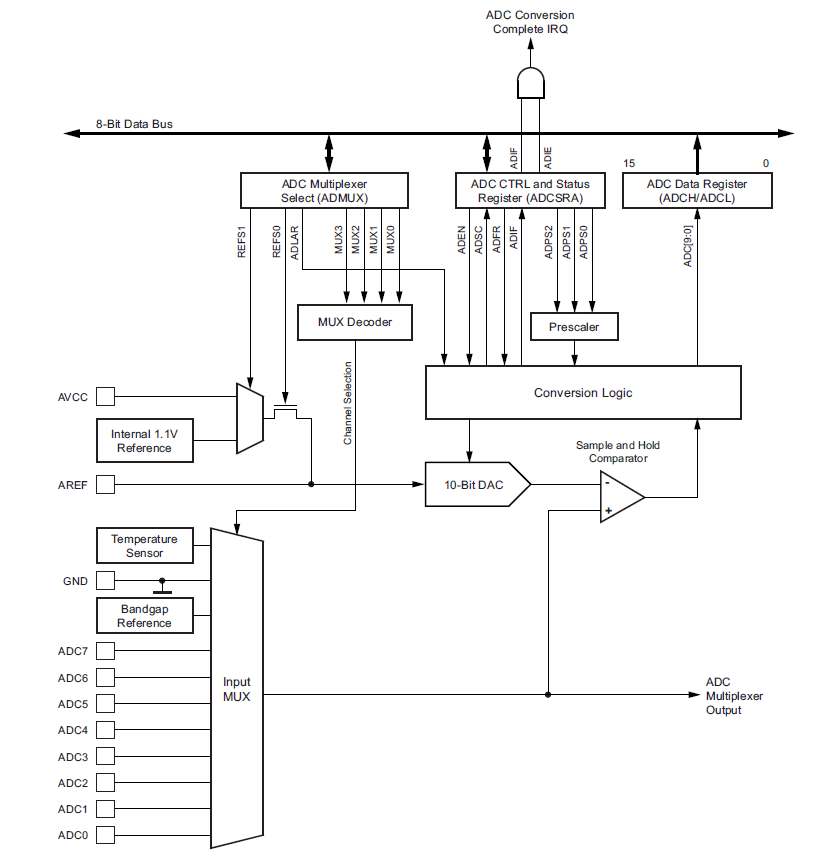


Figure 2 ADC block diagram ATmega328P[[2]](#footnote-2)

# Practice 1: Configure the ADC and get data.

To configure the ADC, there are many registers should be configured in order to work correctly. These steps are guided:

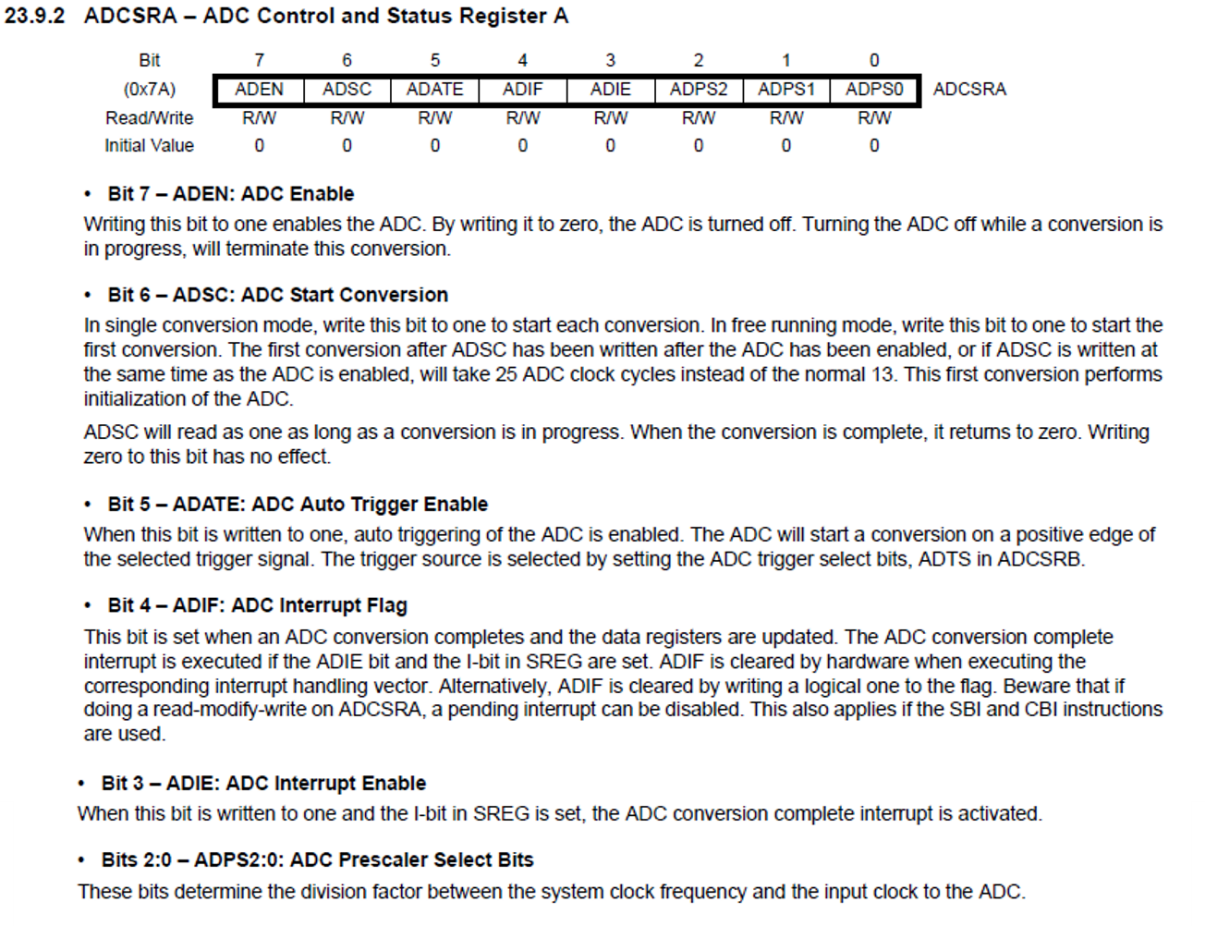
1. Enable the ADC using **ADEN** bit in **ADCSRA** register**.**
2. Set the prescaler so that the ADC clock is between 50-200Khz. Note that the CPU clock speed is 16MHz.

Figure 3 ADC Control and Status Register A

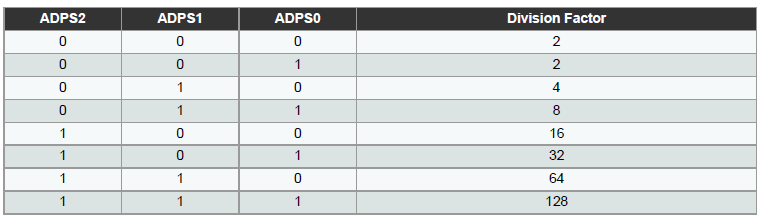


Figure 4 Prescaler Selection

1. Select the reference voltage to be **AVCC** in **ADMUX** register**.**

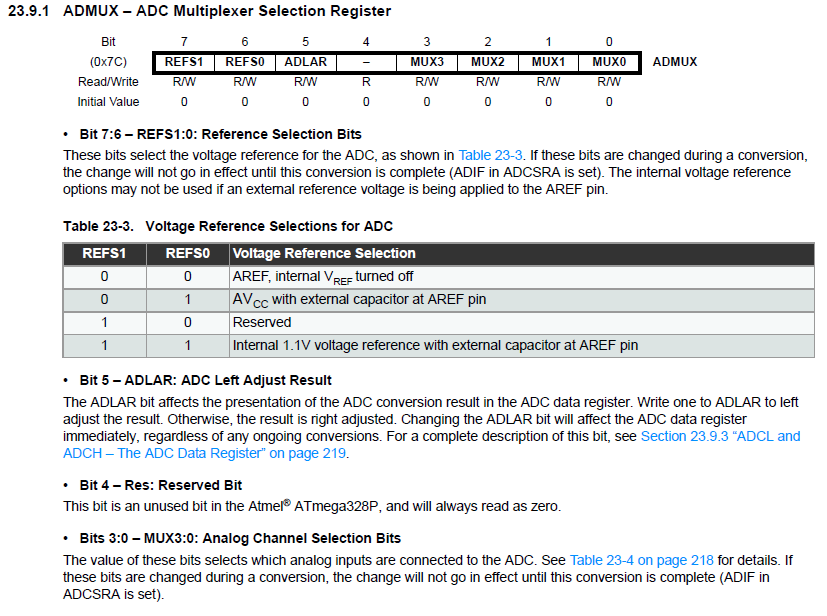


Figure 5 ADC Multiplexer Selection Register

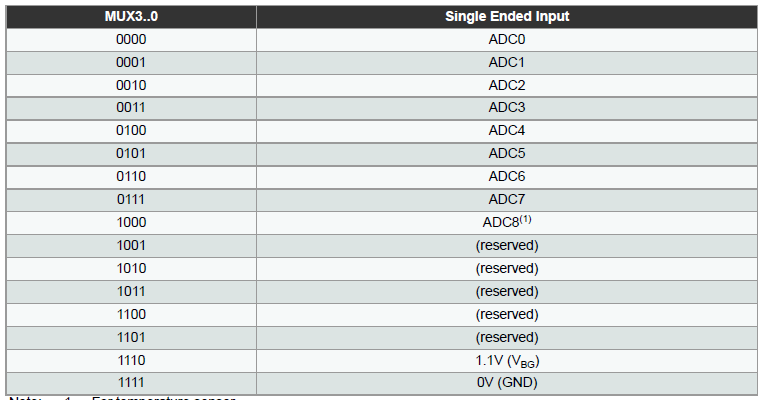


Figure 6 Analog Multiplexer selection bits

|  |
| --- |
| ***Note:*** in your Arduino nano board you should connect on of the analog inputs either to Vcc or GND using one of the resistors you have. This should give you the readings either 1023 or 0. |

The previous steps should be performed once to setup the ADC. The following is to read from the ADC:

1. Select the ADC input channel where the analog signal is connected to. This is done by configuring the ADMUX register bits MUX3-0 as in Figure 6 above.
2. Start the conversion by setting the **ADSC** bit in **ADCSRA** register.
3. Read the results from ADC register.
4. Convert the result by equation (1).

Print the reading using the **logging** function.

# Practice 2: Measure the temperature.

To measure the temperature, one of the most common methods used in sensors is the Negative Coefficient Temperature Resistor (NTC thermistor). This resistor from its name exhibits resistor decline with temperature increase. *NTC* has a non-linear relationship between the resistors and the temperature. The next formula demonstrates the relationship between temperature and resistor:

|  |  |
| --- | --- |
|  | (2) |

Where:

You are going to use **MF52A2103J3470** NTC thermistor, has the following characteristics:

To measure the resistor, you should connect the NTC thermistor with another resistor that has very small drift/sensitivity to temperature. You can construct either of the following circuits:

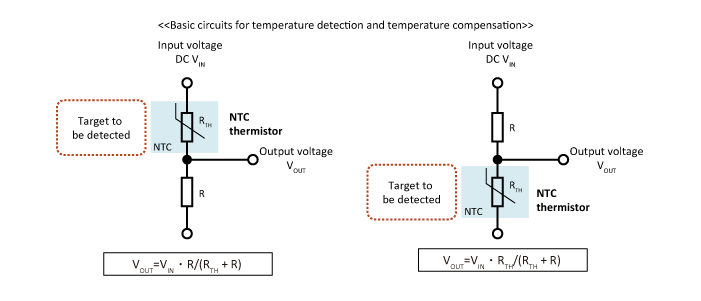


Figure 7 NTC measurement circuit[[3]](#footnote-3)

Solve the potential divider equation. Then solve NTC equation and obtain the temperature reading using logging function.

***Hint:*** You need to use **ln()** function in your code. Ln function in math.h is Log()

1. <https://microcontrollerslab.com/analog-to-digital-adc-converter-working/> [↑](#footnote-ref-1)
2. ATmega328P datasheet. [↑](#footnote-ref-2)
3. <https://product.tdk.com/info/en/products/protection/temperature/chip-ntc-thermistor/technote/apn-chip-ntc-thermistor.html> [↑](#footnote-ref-3)