Performance of the ATmega328p's ADC

Introduction

Analog to digital converter, or ADC, is an electronic circuit that converts analog signals to digital signals. ADCs are vital components of real-time systems. For example, an ADC maybe be used to implement a parking sensor system by converting signals produced by a proximity sensor. Analog signals are voltages or currents that vary over time and are affected by the electrical circuit or environment. These signals usually represent physical quantities such as temperature or pressure. Through the ADC, these analog signals are then converted into digital signals. Digital signals are sequences of numbers representing the magnitude of that analog signal swamped at that specific time.

In this report, the performance of the in-built ADC of the ATmega328p will be assessed by calculating the offset and gain error of the ADC.

Method

The Arduino UNO, which contains the ATMega328p microcontroller, is connected to a breadboard with a circuit set up to receive a ramp input from 0V to 5V with a frequency of 200mHz from a wave generator. This signal will then be sent into the Analog-in pins of the Arduino to be converted by the ADC. The converted values will then be stored in a ring buffer with a max size of 1500 samples.

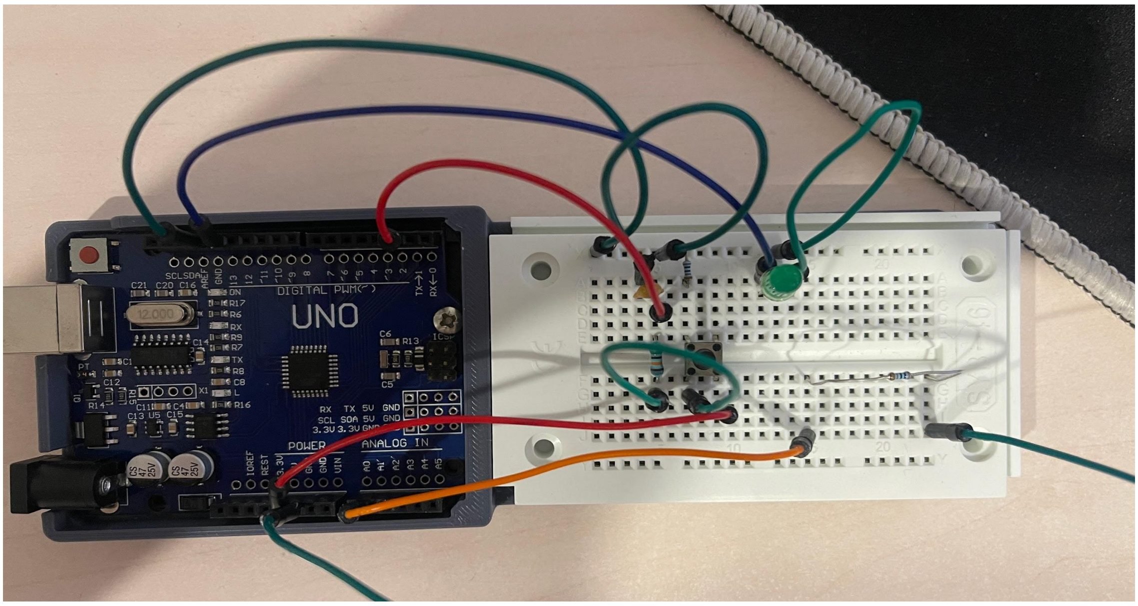


Figure 1: circuit to receive the ramp input

The circuit above is made so that the Arduino can receive the ramp input through Analog port A0, as seen from the orange wire. A button is also implemented so that when pressed, the program will switch states, recording or printing the data. Both states can be interrupted with the button regardless of the task's progress.

The Arduino is programmed using timers, the ADC, and interrupts, written all in low-level C. The ADC is configured to be left-adjusted and only uses 8 bits. The pre-scalar of the ADC will change as we progress through the analysis. A sampling rate of 200 samples per second is required, so the Timer is adjusted to interrupt every 5ms. Each timer interrupt will signal the ADC to convert the voltage to a digital value and then be inserted into the ring buffer. When the ring buffer is full, it will simply start to overwrite the previous recordings.

To quantify the performance of the ATMega328P's ADC, the gain error of each pre-scalar value will be calculated. To calculate the gain error, the following steps will be followed

GAIN VOLTAGE

1. Record down the ring buffer.
2. Cut out the section where the ramp cycle is repeated(where ADC values go from 0 to 255)
3. Plot the ring buffer data alongside a plot of the actual voltage.
4. Find the difference between the ideal ADC value of 5V (255) and the measured value
5. Calculate the LSB of the ADC
6. Calculate gain error for corresponding ADC pre-scalar
7. Repeat for pre-scalar factors from 2 to 128.

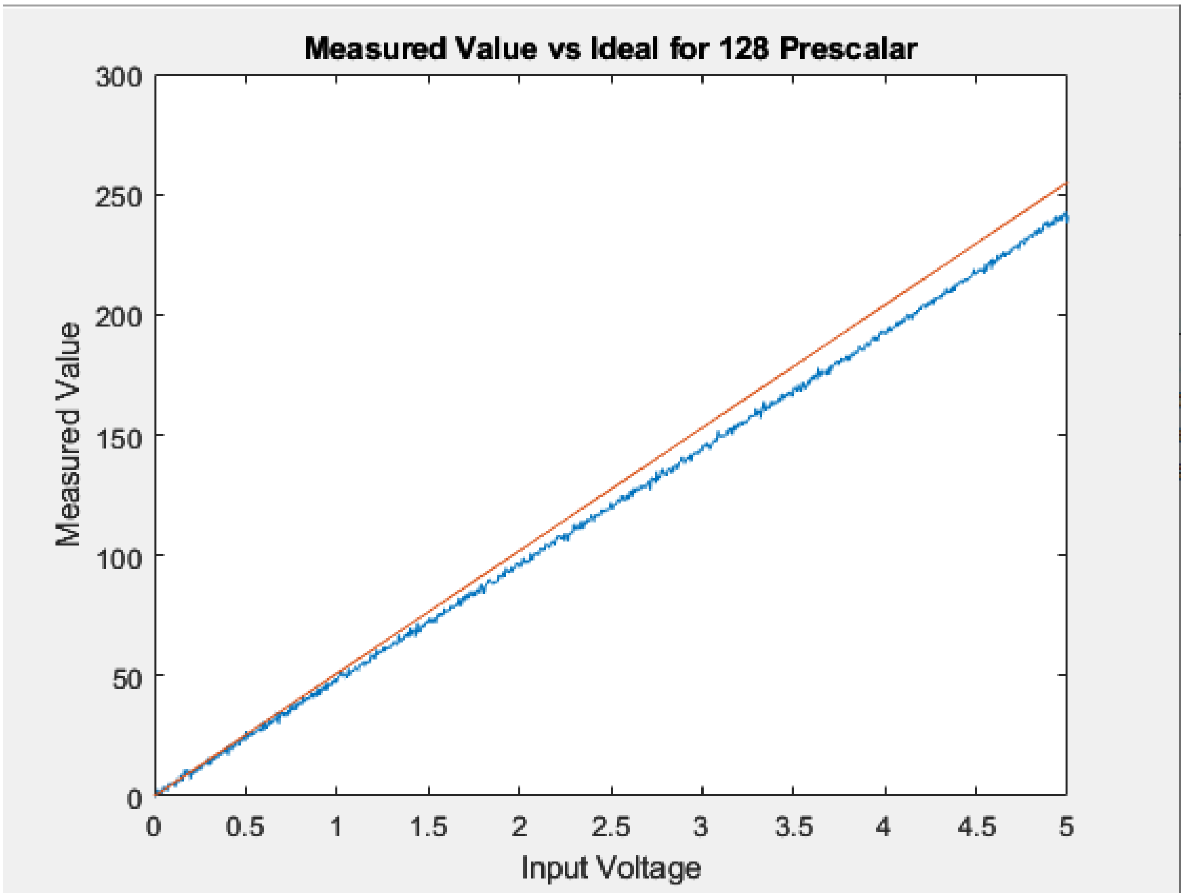
OFFSET VOLTAGE

1. Programming the Arduino to print out the ADC values in volts
2. Connecting the signal generator to the circuit
3. Outputting a constant DC signal from 0.00V
4. Increment by 1mV and monitor the ADC value
5. When the ADC consistently outputs a non-zero value(0.02V smallest detectable voltage)
6. Record down
7. Repeat for factors from 2 to 128.

<u>RESULTS</u>

Gain Error:

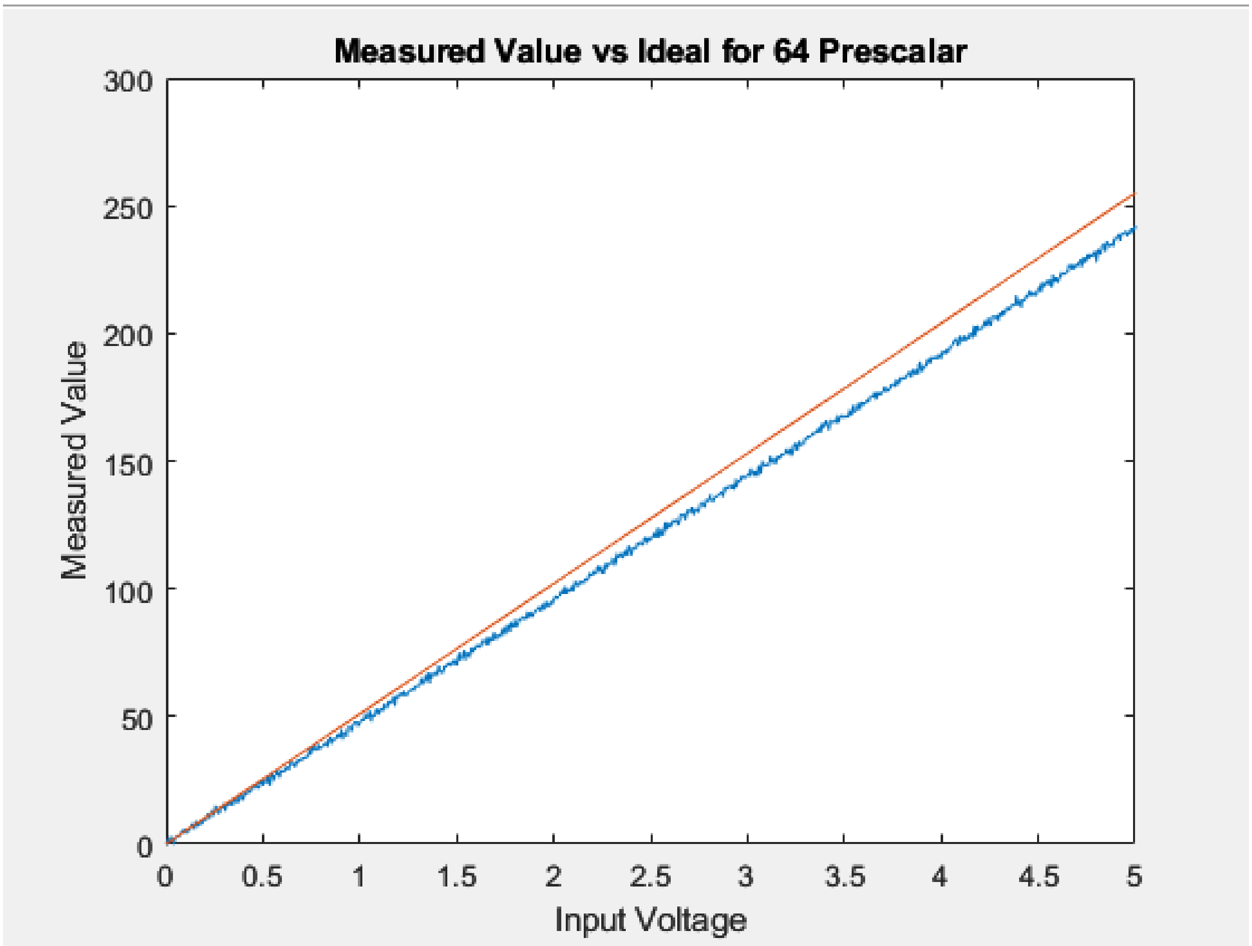
128 Prescalar:



Expected Value for 5V = 255
Measured 243

(255-243)/255*100 = 4.7%
(4.7/100)*256 = **12.04 LSB**

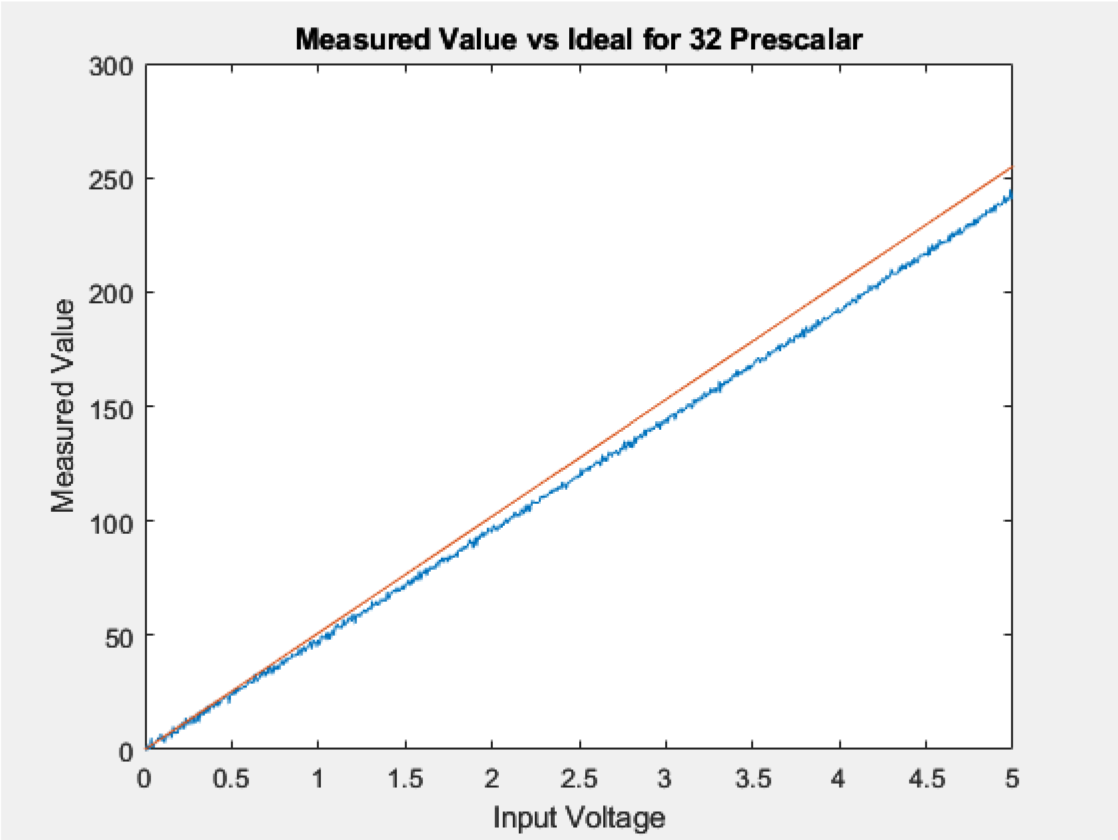
64 Prescalar:



Expected Value for 5V = 255
Measured 241

(255-241)/255*100 = 5.49%
(5.49/100) * 256 = **14.05LSB**

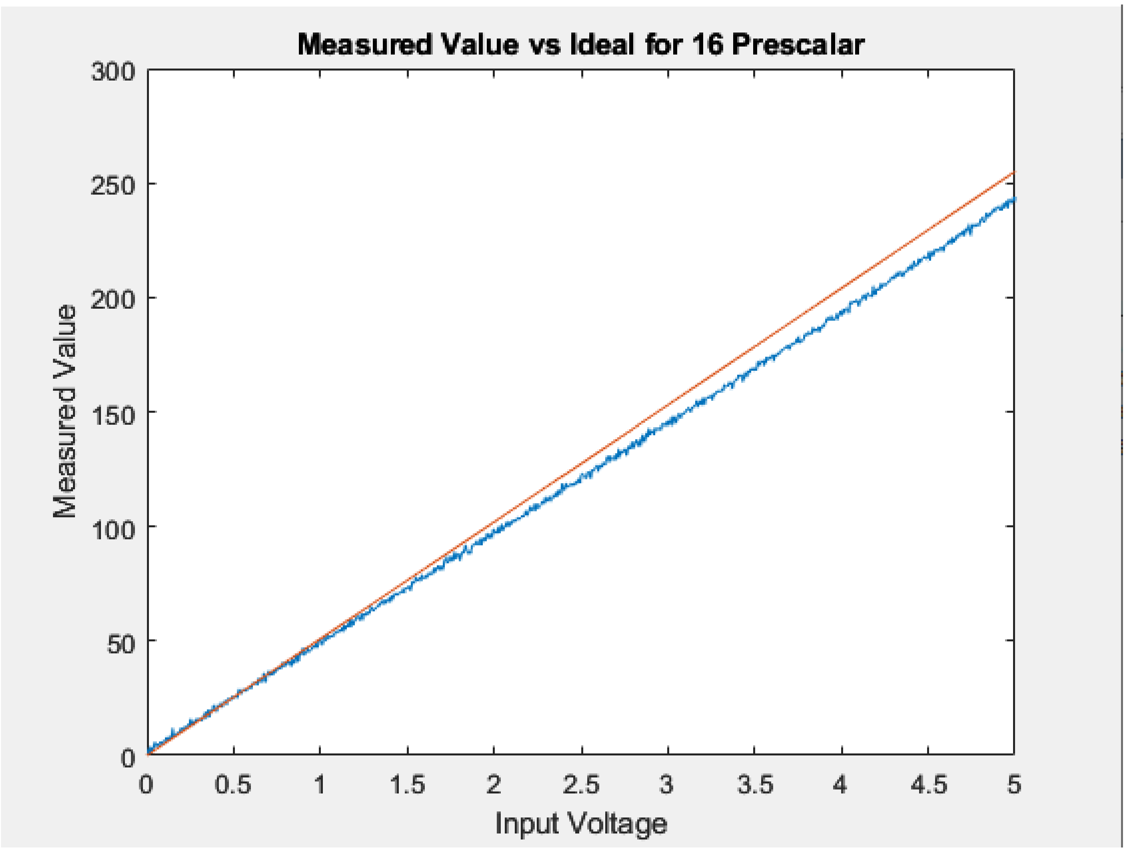
32 Prescaler



Expected Value for 5V = 255
Measured 241

(255-245)/255*100 = 3.9%
(3.9/100) * 256 = **10.03LSB**

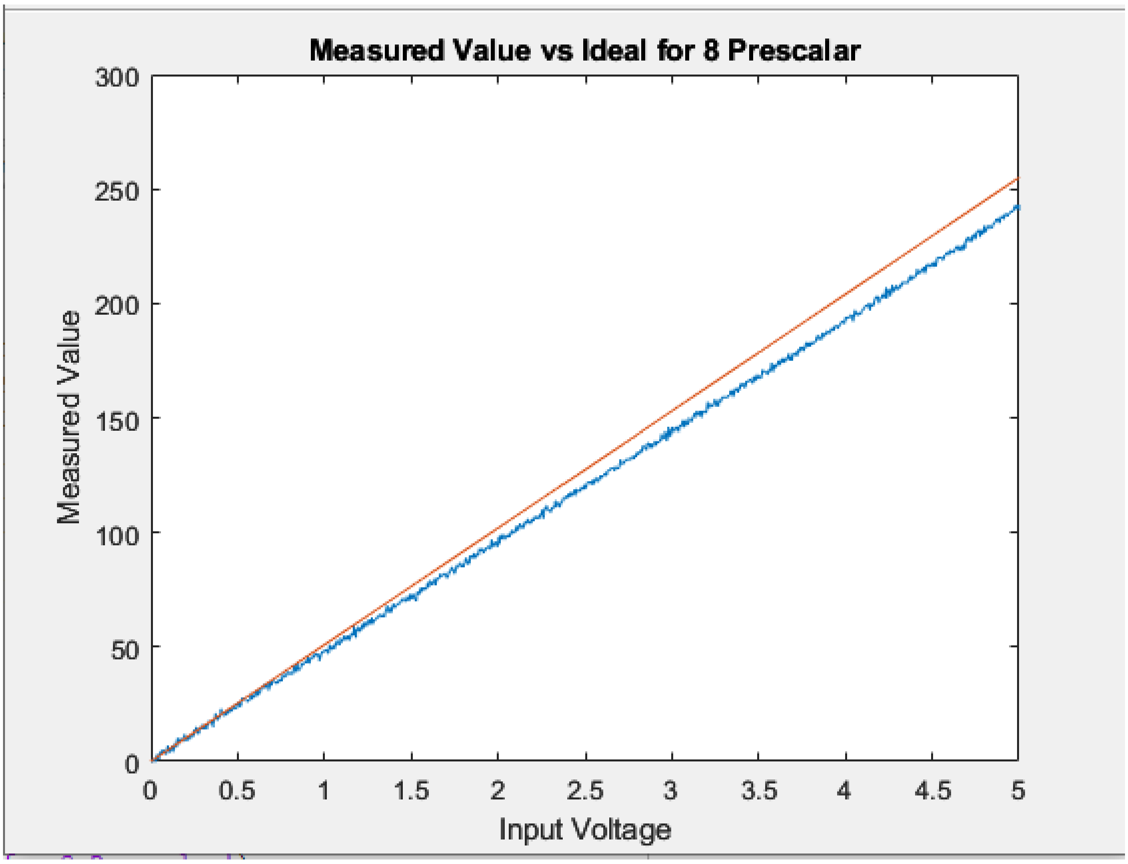
16 Prescaler



Expected Value for 5V = 255
Measured 241

(255-242)/255*100 = 5.09%
(5.09/100) * 256 = **13.05LSB**

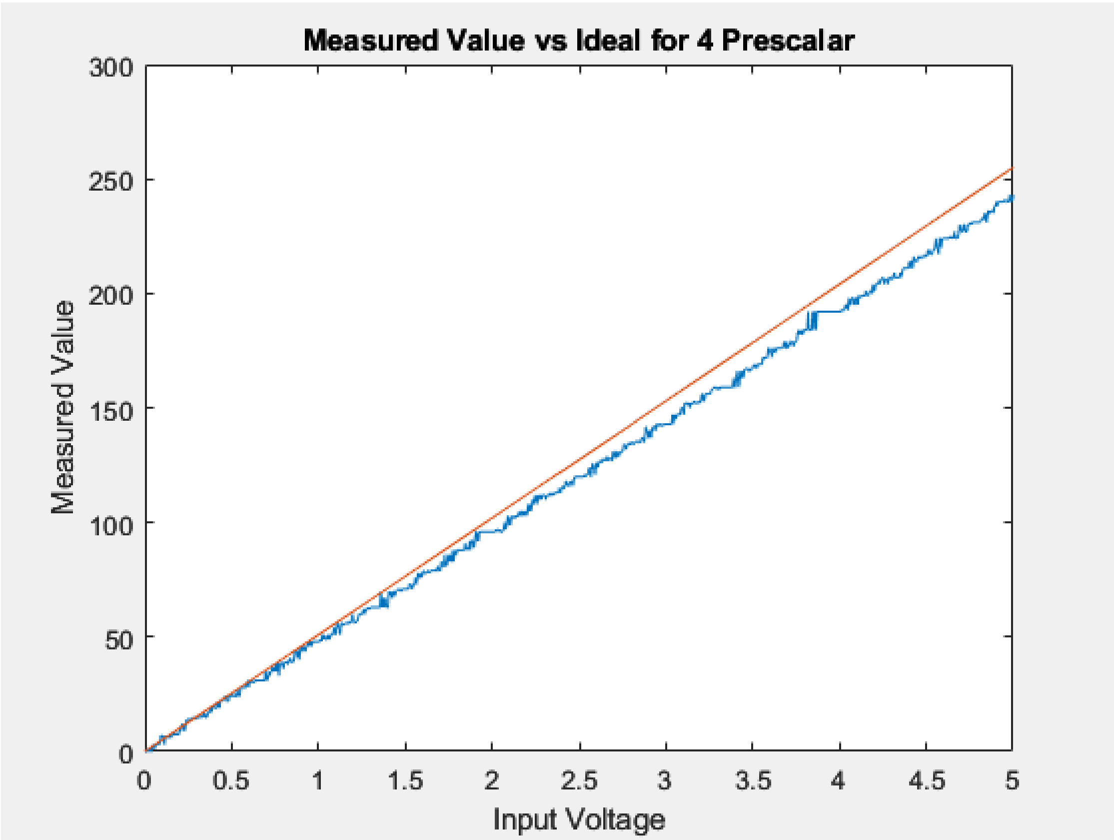
8 Prescaler



Expected Value for 5V = 255
Measured 241

(255-243)/255*100 = 4.71%
(4.71/100) * 256 = **12.04LSB**

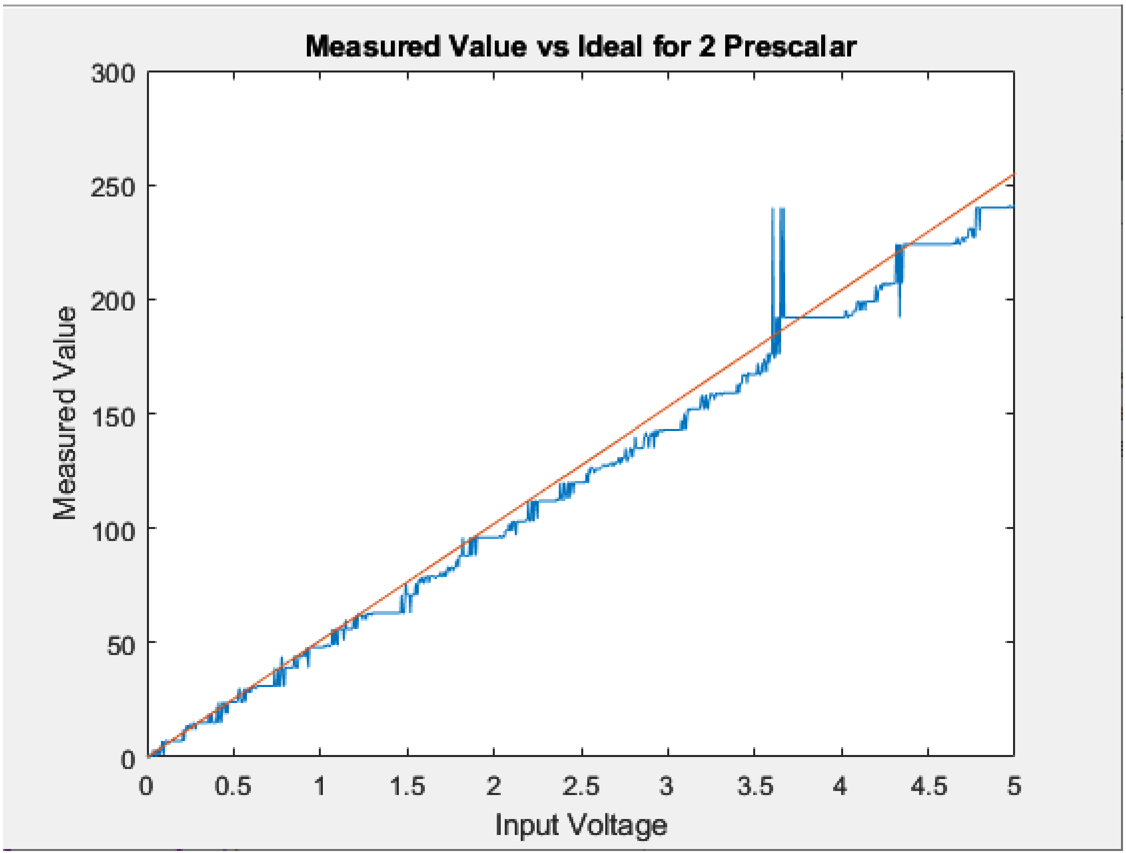
4 Prescaler



Expected Value for 5V = 255
Measured 241

(255-240)/255*100 = 5.88%
(5.88/100) * 256 = **15.05LSB**

Prescaler 2



Expected Value for 5V = 255
Measured 241

(255-241)/255*100 = 5.49%
(5.49/100) * 256 = **14.05LSB**

Offset Error:

Pre-scaler	2	4	8	16	32	64	128
Offset V	8.6mV	10.6mV	9.3mV	7.4mV	8.1mV	11mV	13.2mV
Offset LSB	0.44LSB	0.54LSB	0.47LSB	0.38LSB	0.41LSB	0.56LSB	0.67LSB

Discussion:

After reviewing the results of the gain and offset errors of the different pre-scalar values, it is clear from the gain error that as the pre-scalar increases, the ADC outputs more consistent smoother readings, whereas the smaller pre-scalers have more discontinuity. Throughout all prescaler values, the gain error seems rather consistent, ranging from around 15LSB to 10 LSB. The offset errors also seem to hover around 0.4 LSB to 0.67LSB, with the offset error of 128 having a slightly higher error.

The ramp signal reaches a maximum of 5V. Ideally, the 8-bit ADC should be reading 255; however, in all readings, regardless of the pre-scaler, the ADC never reads past 245. This could be caused by imperfections in the components used in the analysis, such as electrical components on the breadboard(wires, resistors) or the wave generator itself.

Comparing the smoothness of the data from the prescaler values, for pre-scalars above 8 (subjectively), we can say that we only need a minimum of 8 for a reliable reading. For this analysis, a sampling rate of 200Sa/s is used. Therefore we do not require such high speeds of the ADC as higher speeds of the ADC may lead to unreliable readings.

In conclusion, with an average offset LSB of around 0.5, we can say there aren't any problems with the ADC performance at lower voltage readings. The typical LSB from the gain was around 14, much larger than the allowable LSB(1.5) from the datasheet. Therefore we can say that at higher input voltages, the readings become more unreliable and may stem from voltage losses across the components. To produce a more accurate analysis, higher-quality components could be considered.