Quantifying the Performance of the ATmega328P's ADC

Introduction

Quantifying the performance of the ATmega328P's ADC is essential for ensuring the accuracy and reliability of analog-to-digital conversions in real-world systems. ADCs are commonly used in real-time applications, where conversion errors can lead to incorrect outputs or behaviour. By evaluating parameters such as gain error, we can determine the severity of deviations from ideal ADC behaviour. This enables better design decisions, including the selection of an appropriate prescaler. Ultimately, assessing ADC performance supports the development of robust and precise systems.

Methods

Setup

The experimental setup utilised an Arduino .ino file, an Arduino UNO circuit (equipped with the ATmega328P), and a digital oscilloscope used as a signal generator. The Arduino UNO was programmed as a finite-state machine with two push-button-controlled states:

- State 1: the ADC samples analog input continuously at 200 Sa/s and stores the data in a 1000-value ring buffer.
- State 2: sampling pauses and stored values are transmitted via serial connection.

The ADC was configured for 8-bit resolution (i.e., using only the ADCH register). A 2.5V ADC reference voltage was applied.

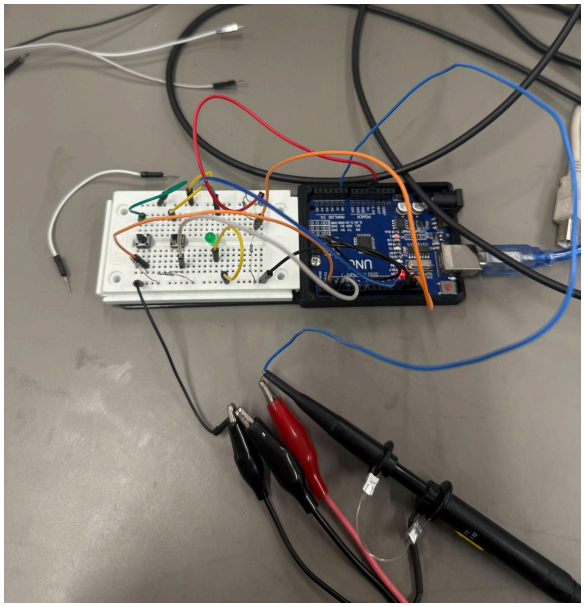

Figure 1: Arduino UNO hardware setup

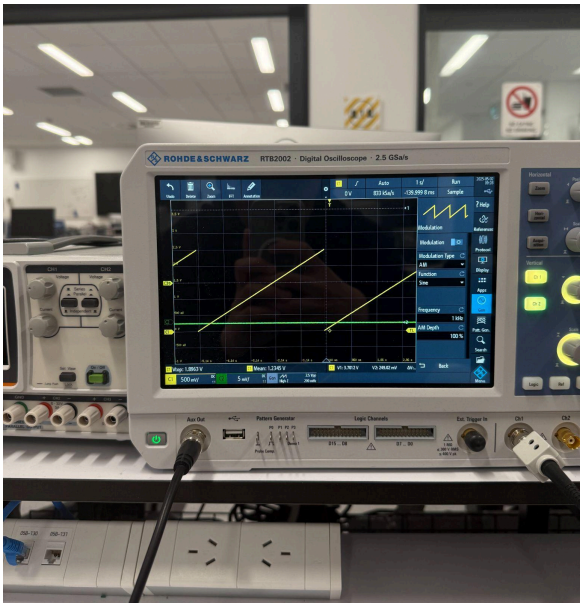

Figure 2: Sawtooth Waveform generated using a Digital Oscilloscope

A 5-second period sawtooth waveform (0–2.5V) was generated using a digital oscilloscope connected to A1. This period was chosen to fully populate the 1000-sample ring buffer at

200 Sa/s. The generator shared a common ground with the Arduino to prevent ground loop issues.

During each cycle, data was collected in State 1 and exported in State 2. Serial output was logged using PuTTY, transferred to Excel, and reorganized in ascending order (ADC code 0–255). The ideal voltage at each sample index n was computed as:

$$V_{ideal} = \frac{n-1}{1000} \times 2.5 \text{ , for } 1 \leq n \leq 1000.$$

Gain Error Measurement

ADC gain error may be evaluated in different ways. We chose to quantify it in terms of Least Significant Bits (LSBs); i.e., the difference between the ideal and actual final ADC transitions from 0x3FE to 0X3FF (Atmel Corporation., 2015). Given an 8-bit ADC and a 2.5V reference voltage, each LSB corresponds to:

$$1\ LSB = \frac{V_{ref}}{2^8} = \frac{2.5V}{256} = 9.77\ mV$$

According to the ATmega328P datasheet, the ideal transition paoint, from ADC output code 254 to 255, should occur at 1.5 LSBs below full scale. This corresponded to approximately 2.485V. We then examined the sampled data to determine the actual voltage at which this transition occurred and calculated the deviation from the ideal value. This voltage was then expressed in terms of LSBs:

$$Gain\ Error(LSBs) = \frac{V_{actual} - V_{ideal}}{1\ LSB}$$

This process was repeated for all possible prescaler values for our ADC (2–128), with results logged and plotted in Excel.

Results

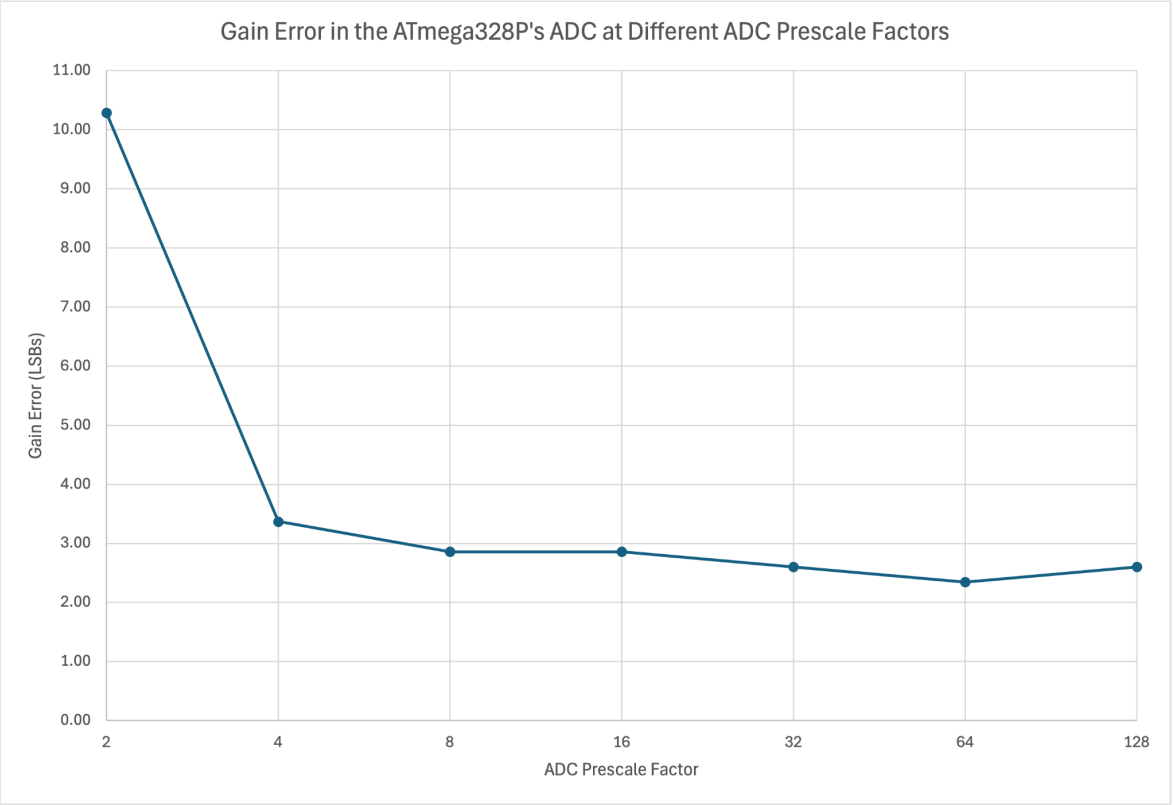

Figure 3: Experimental results observing Gain Error as a function of ADC Prescale Factor

ADC Prescale Factor	Gain Error (LSBs)
2	10.29
4	3.37
8	2.86
16	2.86
32	2.60
64	2.34
128	2.60

Figure 4: Experimental results tabulated

The results present an exponential downward trend for gain error as the prescale factor is increased. When the prescaler is set to 2, the gain error is significantly larger than all other prescale options. Between 4-128, the difference in gain errors is less pronounced but still adheres to this trend.

Discussion

Impact of Prescaler on ADC Accuracy

The results demonstrate that gain error increases significantly at low prescaler values (e.g., 10.29 LSBs at a prescaler of 2). This outcome is consistent with the expectations; lower prescaler values result in a higher ADC clock frequency, which deviates significantly from the ADC's recommended clock range (typically 50 kHz to 200 kHz), leading to inaccurate conversions. Based on these findings, we would predict that other ADC parameters also deviate from the ideal when the prescale factor is very low.

A prescaler of 128 can achieve an ideal clock speed (125kHz), but the further the clock speed is increased away from this, the more inaccurate the ADC will be as illustrated through the gain error readings. At high clock speeds, the ADC may not have sufficient time to complete a sample before the next one is started.

From this, we gather that it is not reasonable to use low prescaler values in real-time systems where accuracy is important. However, we must acknowledge that there will be some error present at all prescale factors. There may be some rare, exceptional applications however where a low prescaler is used if the conversion frequency is much more important than accuracy.

Software Compensation Techniques

To mitigate gain and offset errors, software-based calibration can be applied after conversions. These adjustments can be hardcoded for each prescaler to ensure precision and reliability.

To mitigate offset error only, we can identify the DC offset by measuring the output for a known zero-input condition and adding/subtracting that value from the raw voltage:

$$V_{corrected,\,offset} = V_{raw} \pm V_{offset}$$

For gain error only, we can express it as a percentage and apply an inverse scaling factor. For example, if the gain error is +5%, scale the output voltage by 0.95:

$$V_{corrected,\,gain} = V_{raw} \times (1 - Gain\ Error)$$

Accounting for both errors:

$$V_{corrected} = (1 - Gain\ Error)V_{raw} \pm V_{offset}$$

Experimental Limitations

Two assumptions were made to collect and display the experimental results:

- The reference voltage applied to the ADC was exactly 2.5V; in reality, this was unlikely. This voltage was achieved with a simple voltage divider circuit with two 1k

ohm resistors with tolerances. An improved hardware setup would use some form of voltage regulation.
- The voltages applied via the signal generator were ideal and linearly increased from 0-2.5V; this does not account for signal noise. Using a buffer OPAMP may provide more accuracy.

Other experimental limitations and improvements include:

- Using a larger ring buffer to store data; capturing more values reduces noise effects.
- Using the full 10-bit ADC resolution; a higher resolution would reduce the impact of quantisation error and improve the accuracy of the gain error calculations.
- Repeating the data collection several times and averaging out collected data would account for anomalies.

References

Atmel Corporation. 2015. *8-bit AVR Microcontroller with 32K Bytes In-System Programmable Flash* (Rev. 7810D-AVR-01/15) [Data sheet].