Semester Project

Thobias Høivik
Western Norway University of Applied Sciences
ELE201: Microcontrollers and Data Networks

Contents

1 Introduction

In this project we use the STM32 microcontroller along with analog sensors to gather data that we analyze for statistical and informational randomness, with the goal of evaluating how feasible such a setup is as a hardware-based true random number generator (TRNG). The data will be sent via serial connection to a computer to analyze the quality of the data.

We will combine theory from information theory, probability, signal processing, and networking to assess both the quality of randomness and how such a system could integrate into a distributed architecture.

We will discuss:

- Theoretical background on randomness and entropy.
- Implementation and data transfer.
- Statistical analysis and randomness testing.
- Viability discussion and possible improvements.
- Network design for scaling such systems.

2 Theory

2.1 Randomness and Entropy

Randomness can be defined both in a statistical and algorithmic sense. A sequence is considered random if it is unpredictable and lacks compressible structure.

Definition 2.1: Shannon Entropy

Given a discrete random variable X that takes values in χ with probability distribution $p : \chi \to [0, 1]$, its entropy is

$$H(X) = -\sum_{x \in \chi} p(x) \log_2 p(x).$$

(as introduced by Shannon [1] and discussed in [2, 4]). Entropy is a central meassure in information theory pertaining to the randomness of data and it will be referenced extensively throughout this text.

2.2 True vs. Pseudo Randomness

A pseudo-random number generator (PRNG) produces deterministic sequences from an initial seed, while a true random number generator (TRNG) relies on physical entropy [3]. For hardware-based RNGs, ensuring unbiased and unpredictable output requires:

- High-quality analog entropy sources.
- Statistical post-processing.

2.3 Hypothesis

We believe that due to the inherit noise of electrical circuits that dominate at low frequencies, the least significant bits (LSBs) of our signal should be a source of high entropy. Then, with further post-processing we should get a high-quality source of random numbers which can be applied in processes which require randomness.

2.4 Theoretical Modeling of Analog Noise

To justify that the least significant bits of our ADC contains randomness, we model the analog sensor output as a combination of deterministic signal and noise:

$$V_{\text{ADC}}(t) = V_{\text{Signal}}(t) + V_{\text{Noise}}(t)$$

- V_{Signal} represents the slowly varying, predictable component (e.g. ambient light level in the case of photoresistor).
- V_{Noise} represents the unpredictable physical noise (thermal-, shot noise, quantization error, sensor-specific noise).

The noise, which is the sum of thermal-, shot-, and other electronic noise is typically modeled as a Gaussian random variable [5]:

$$V_{\text{Noise}}(t) \sim \mathcal{N}(0, \sigma^2_{\text{Noise}})$$

The **ADC quantization** transforms the continuous voltage into discrete levels:

$$X = \text{ADC}(V_{\text{ADC}}) \in \{0, 1, \ldots, 2^{12} - 1\}$$

The voltage corresponding to one LSB (least significant bit) step is

$$\Delta V = \frac{V_{\text{ref}}}{2^{12}}$$

for a 12-bit ADC with reference voltage V_{ref}.

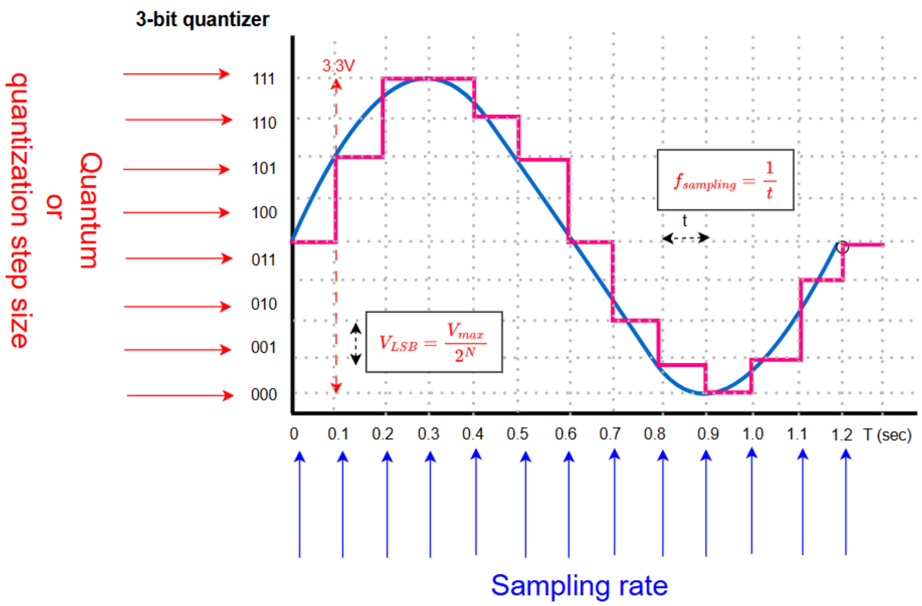

Figure 1: Quantization of analog signal.

The randomness of the extracted bits is meassured by the Shannon Entropy. For a discrete random variable X with k possible outcomes, the maximum possible entropy $H_{\text{MAX}} = \log_2(k)$ bits.

We extract the 4 LSBs, which gives $2^4 = 16$ possible outcomes from $0000_b \to 1111_b$. The maximum entropy then is

$$H_{\text{MAX}} = 4 \text{ bits}$$

Maximum entropy is achieved if the the probability of observing any of the 16 outcomes is uniform [2, 4]:

$$P(D_{\text{LSB}} = i) = \frac{1}{16} \text{ for } 0 \leq i \leq 15$$

To achieve a near-uniform distribution across the 16 LSB states, the standard deviation of the noise σ_{Noise} must be large enough to span multiple quantization steps.

Let $\Delta V_{4-\text{bit}}$ be the voltage corresponding to the 4 LSBs:

$$\Delta V_{4-\text{bit}} = 16\Delta V$$

Consider an input voltage V_{ADC} that falls within a $\Delta V_{4-\text{bit}}$ window. The probability that the resulting digital word D falls into a specific 4-bit LSB state i depends on the area under the Gaussian Probability Density function of the noise that falls into the corresponding voltage interval I_i.

The critical condition for near-maximum entropy is then:

$$\sigma_{\text{Noise}} \gg \Delta V$$

If the noise standard deviation is significantly larger than the LSB step size then the Gaussian noise distribution becomes "smeared" across multiple LSB intervals. Since the noise is zero-mean and σ_{Noise} is large, the probability of the total input V_{ADC} falling into any one LSB interval I_i is nearly equal to the probability of it falling into an adjacent $I_{i\pm1}$.

NOTE: Maybe do FFT on signal to show how noise dominates high-freq components.

3 Implementation

3.1 Hardware Setup

We will use the STM32f767zi microcontroller, connected to a photoresistor.

The microcontroller samples the analog voltage via the ADC and stores or streams the digital values over a serial or network interface. We will be sampling at 12-bits, keeping the 4 least significant bits as we believe these will be most susceptible to noise.

NOTE: REMEMBER TO PUT WIRING DIAGRAM HERE.

3.2 Data Transmission

Data is transmitted from the STM32 to a computer for analysis. In our experimentation we use USART/Serial Connection using a micro-usb connection from the microcontroller to a computer.

4 Results and Analysis

4.1 Results from the Raw Signal

We begin by analyzing the apparent randomness of the raw signal.

In our experimentation we first use Python with the serial library to read the incoming data and matplotlib to plot the signal over time.

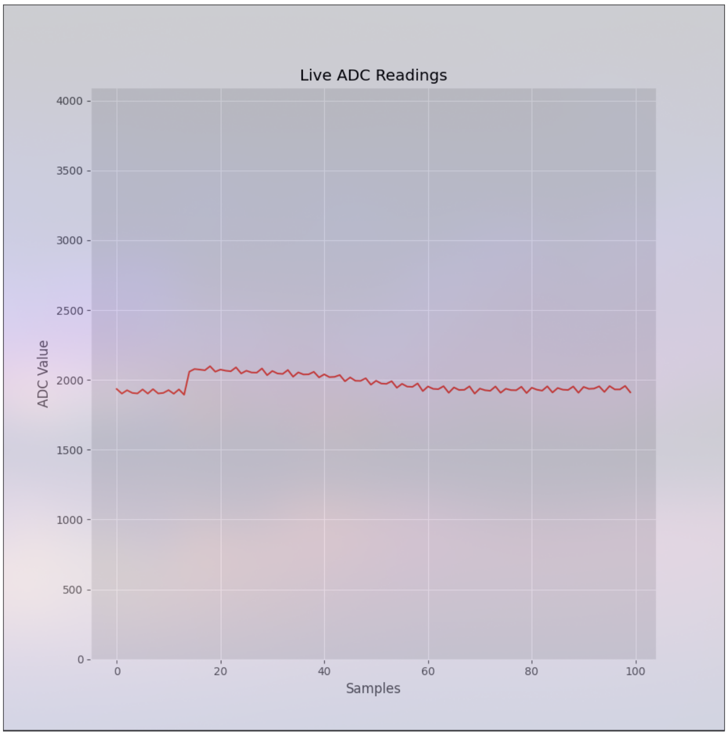

Figure 2: Plot of the raw signal over time

As we can see in this plot, while the signal remains relatively stable, reflecting the stable light level we were testing against, there are definite micro-jitters present. This is a good sign and is almost ceirtainly due to the noise which we predicted would dominate at small levels.

We do have to take into account the fact that the micro-jitters might be due to slight fluctuations of the ambient light-level in our tests. To mitigate this possibility we will from this time forward be testing with the photoresistor covered with electrical tape.

Now we look at the observed probabilities pertaining to the 4 LSBs. We want a roughly equal probability that a given bit in the 4 LSBs is 0 or 1.

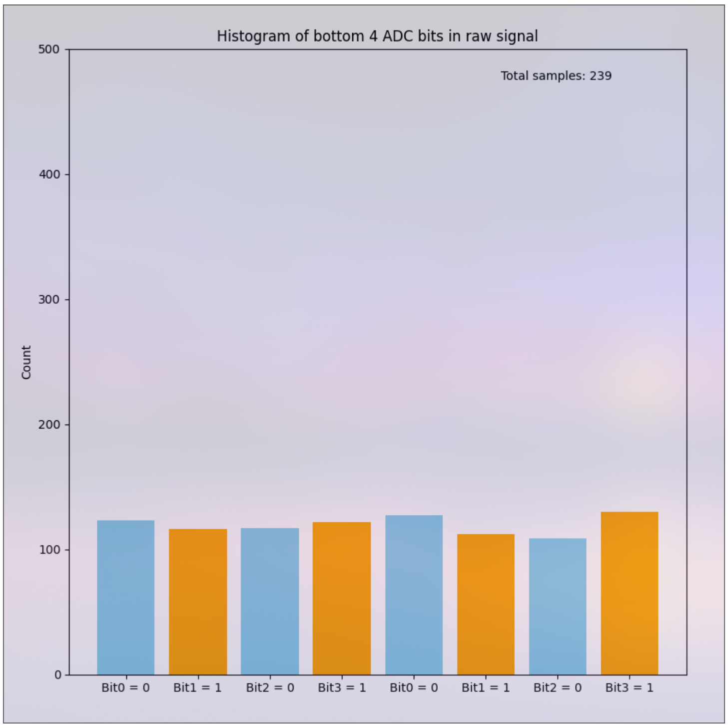

Figure 3: Histogram of 4 LSBs with photoresistor covered

NOTE: Remember to redo this histogram with more samples.

As we can see there appears to be a roughly equal chance that a given bit in the 4 LSBs is 0 or 1, pointing towards high entropy in the LSBs.

4.2 Processing

Our goal is to transform the raw data stream (which is **biased** and **predictable**) into a shorter, statistically perfect random stream.

1. Raw Bit Extraction
    - **Technique:** Only use the least significant bits of the ADC output.
    - The LSBs are dominated by the desired random noise sources and are less influenced by the much larger and more predictable signal (e.g. light level in the case of a light sensor). Thus the MSBs should be discarded entirely as they should exibit fairly low entropy.
2. Entropy Conditioning (Post-Processing) Since the LSBs will more than likely still contain residual bias and correlation we require a powerful **Entropy Extractor** for cryptographic quality.
    - **XOR Folding:** XOR a bit with one a few steps behind. E.g. $\text{Bit}_{\text{Out}} = \text{Bit}_i \oplus \text{Bit}_{i-N}$, where N is chosen to be slightly larger than the observed correlation length.
    - **Cryptographic Hash Extractor (NIST SP 800-90B):** Collect a large buffer of L raw bits. Estimate H_{min} per bit. Take the buffer, and hash it (e.g. SHA-256) to produce a fixed-length output.
      Ths downside with this approach is that we have to sample many times to get a sufficiently large buffer.

4.3 Network Design

A possible setup:

- Each microcontroller is connected to a local router.
- A dedicated subnet for sensor nodes (e.g., 192.168.10.0/24).
- Central analysis server on a separate subnet.

4.4 Subnetting and Addressing

We show how to subnet the network efficiently:

- Example: dividing a /24 into four /26 subnets.
- Assigning IP ranges for sensors, analysis nodes, and administration.

4.5 Data Security and Transmission Integrity

We briefly discuss:

- Packet integrity verification (CRC or checksum).
- Optional encryption for data in transit.
- Synchronization and time-stamping for accurate sampling.

5 Conclusion

We summarize:Fundamentals of Precision ADC Noise Analysis

- Theoretical feasibility of analog-sensor-based TRNGs.
- Experimental results and limitations.
- Potential for distributed entropy networks.

5.1 Future Work

- Hardware whitening circuits and amplification.
- FPGA or ASIC implementations.
- Scaling to larger networks and entropy pooling.

References

[1] C. E. Shannon, *A Mathematical Theory of Communication*, Bell System Technical Journal, vol. 27, pp. 379–423, 623–656, 1948.

[2] R. M. Gray, *Entropy and Information Theory*, Springer, 1990.

[3] Cerberus Security, *Hardware Random Number Generators*, 2020.
`https://cerberus-laboratories.com/blog/random_number_generators/`

[4] Statistics How To, *Shannon Entropy Definition*,
`https://www.statisticshowto.com/shannon-entropy/`

[5] G. Gundersen, *Random Noise and the Central Limit Theorem*, Blog post, 01 February 2019.
`https://gregorygundersen.com/blog/2019/02/01/clt/`