# LAB 2 - REPORT 2

## Introduction

## PART 1 – Analog-to-Digital Conversion

**Objective**

The goal of this task was to configure the ESP32S3 XIAO Sense board to sample analog input signals using its built-in ADC and to analyze the sampled data using Python. The analysis included time-domain waveform visualization and frequency-domain spectrum plots (amplitude and phase).

### Hardware Setup

* **ADC input pin**: A0 (D0)
* **Signal source**: Function generator
* **Oscilloscope**: Used for reference waveform

**Original Signal:**

A screen with a graph on it

AI-generated content may be incorrect.

**Signal settings:**

|  |  |
| --- | --- |
| **Parameter** | **Value** |
| Frequency | 100 Hz |
| Period | 10 ms |
| Peak-to-peak voltage | 500 mV |
| Duty Cycle | 50% |
| Rise time | 2.78 ms |
| Fall time | 2.81 ms |

### ESP32 ADC Configuration:

A screenshot of a computer program

AI-generated content may be incorrect.

**Key Configuration Details:**

* Resolution: 12-bit (0–4095 range)
* Attenuation: ADC\_11db (allows full 0–3.3 V input range)
* Sampling rate: 1 kHz (using delayMicroseconds(1000))

Sampled Signal on Arduino Serial Plotter

A screenshot of a graph

AI-generated content may be incorrect.

The waveform observed on the Serial plot matched the 100 Hz sinusoidal input, validating correct sampling.

### Data Transfer and PC-Side Analysis (Python)

Data was retrieved from the ESP32’s Serial output using Python. The script processed the data to generate the waveform and compute the FFT for amplitude and phase spectrum plots.

**Python source code:**

A screenshot of a computer program

AI-generated content may be incorrect.

**Comparison with Oscilloscope**

Waveform and frequency spectrum plots generated from ESP32 ADC samples were compared with the reference waveforms captured on a digital oscilloscope. The 100 Hz fundamental frequency was clearly visible in both cases, confirming the correctness of the ADC sampling.

[Unfortunately I lost the images of the graphs but python code that generates it is included]

### Conclusions

The ESP32S3 XIAO Sense successfully digitized a 100 Hz analog signal using a 12-bit ADC with 1 kHz sampling rate. The Serial data transmission was reliable, and the waveform and FFT spectra matched those obtained from the oscilloscope. Minor discrepancies in signal shape and noise levels were attributed to sampling resolution and timing accuracy. Overall, this demonstrated that the ESP32 ADC is suitable for low-frequency signal analysis tasks.

## PART 2 – Signal Synthesis using DDS and Digital-to-Analog Conversion

This part of the lab focused on generating an analog waveform using Direct Digital Synthesis (DDS) and Digital-to-Analog Conversion (DAC) on the ESP32S3 XIAO Sense board. The synthesized signal was output via PWM and analyzed in both time and frequency domains.

**Configuration and Implementation:**

- Signal Output Method: PWM

- PWM Resolution: 8-bit

- PWM Frequency: 20 kHz

**DDS Configuration:**

**- LUT (Look-Up Table)**: 256-point sine wave

**- Phase Accumulator**: Initialized to 0

**- Phase Increment**: 1000

**- Desired Output Frequency**: 1000 Hz

**- DDS Clock (Update Rate)**: 10,000 Hz

The sine wave was synthesized by iterating through the LUT using the DDS method, where the phase accumulator determined the current position in the LUT. The calculated sample was then output through the PWM peripheral configured for analog signal generation.

The waveform displayed on the oscilloscope confirmed the expected 1 kHz sine wave, demonstrating successful signal synthesis using the DDS algorithm.

A screen with a graph on it

AI-generated content may be incorrect.

As in Part 1, the synthesized signal was sampled and transferred to a PC. A Python script was used to visualize the waveform and compute the frequency spectrum.

### Conclusions

The DDS-based signal synthesis on the ESP32S3 XIAO Sense, using PWM as a DAC, successfully generated a 1 kHz sine wave with a 256-sample LUT and a 10 kHz update rate. The waveform captured via oscilloscope and analyzed via Python confirmed the frequency and shape of the generated signal. Minor quantization and PWM-related artifacts were expected due to 8-bit resolution and PWM nature. Nonetheless, the approach proved effective for low-frequency waveform generation tasks.