# LAB – 1 REPORT

## Section 1 – Blink

**Objective:**  
Turn the built-in LED on and off at 1-second intervals.

Source Code:

A screenshot of a computer program

AI-generated content may be incorrect.

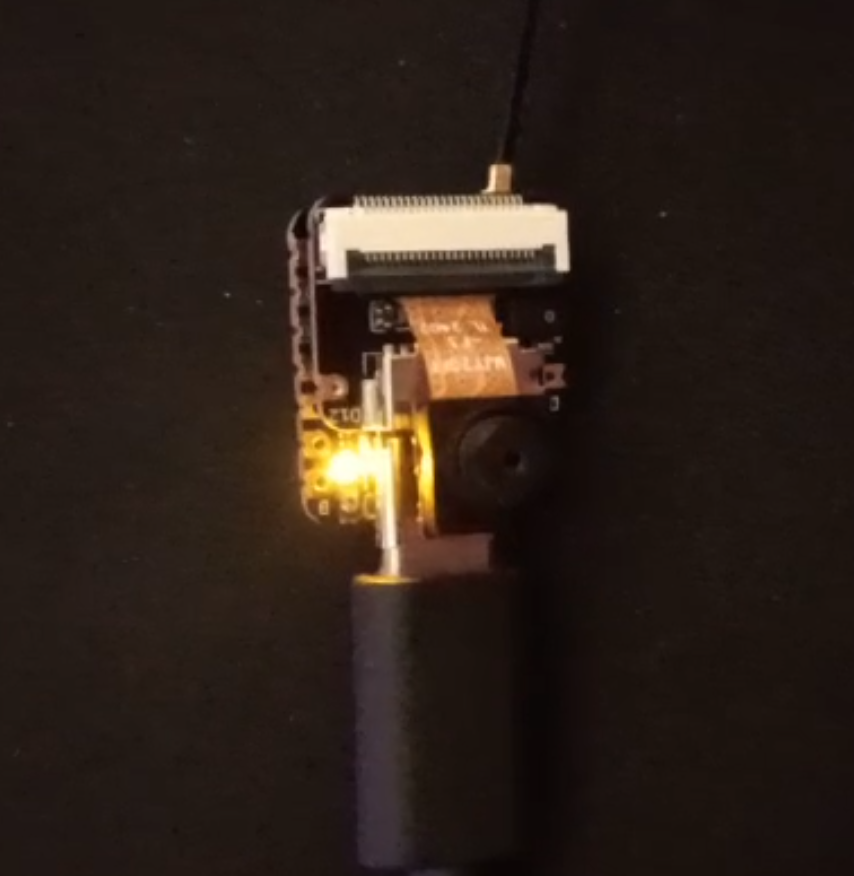
**Used Functions:**

* **pinMode(pin, mode)**
  + Library: Arduino core (part of the ESP32 Arduino framework)
  + Purpose: Sets the specified pin to behave as an input or output.
  + Used for: Configuring the built-in LED pin as an output.
* **digitalWrite(pin, value)**
  + Library: Arduino core
  + Purpose: Sets the digital pin to either HIGH or LOW.
  + Used for: Turning the LED on and off.
* **delay(ms)**
  + Library: Arduino core
  + Purpose: Pauses the program for the specified number of milliseconds.
  + Used for: Keeping the LED on or off for 1 second intervals.

**Functionality:**  
This code blinks the onboard LED at a 1 Hz frequency.

**Additional Evidence:**

* A video showing the LED blinking is included in the report folder.
* Screenshot showing LED in ON and OFF states:

A close up of a device

AI-generated content may be incorrect.

## Section 2 – Signal Reading

**Objective:**  
Read analog signal using ESP32 and visualize it over time.

**Base Signal Reference:**  
Oscilloscope image of the reference signal:

A screen with a graph on it

AI-generated content may be incorrect.

Source Code:

A computer screen shot of a program code

AI-generated content may be incorrect.

**Used Functions:**

* Serial.begin(baudRate)
  + **Library:** Arduino core (Serial communication)
  + **Purpose:** Initializes serial communication at the specified baud rate.
  + **Used for:** Starting serial communication to send data to the computer.
* analogRead(pin)
  + Library: Arduino core
  + Purpose: Reads the value from the specified analog pin (returns a value between 0 and 4095 on ESP32 by default).
  + Used for: Reading the analog voltage signal from pin A0.
* Serial.println(value)
  + Library: Arduino core (Serial communication)
  + Purpose: Prints the value to the Serial Monitor, followed by a new line.
  + Used for: Sending sampled analog values to the computer for visualization.
* micros()
  + Library: Arduino core
  + Purpose: Returns the number of microseconds since the program started running.
  + Used for: Implementing precise timing between samples (sampling interval control).

**Functionality:**  
Reads analog values from pin A0 every 1 ms and prints to serial for plotting and analysis.

**Values vs Time Graph:**

* Screenshot of Serial Plotter showing waveform:

A screenshot of a graph

AI-generated content may be incorrect.

## Section 3 – Digital Microphone

**Objective:**Capture audio from an onboard PDM microphone using I2S and print sample values.A computer screen shot of a program

AI-generated content may be incorrect.

**Used Library:**

* **ESP\_I2S.h**
  + Purpose: Provides an easy interface for using the I2S peripheral on ESP32 to handle digital audio data.
  + Installed via: Arduino Library Manager or included with the ESP32 board package.

Key Classes and Functions:

* **I2SClass I2S**
  + From: ESP\_I2S library
  + Purpose: Creates an instance of the I2S interface to communicate with I2S devices (e.g., digital microphones).
* **I2S.setPinsPdmRx(clockPin, dataPin)**
  + From: ESP\_I2S library
  + Purpose: Configures the I2S input pins for receiving PDM (Pulse Density Modulation) audio from a digital microphone.
  + Used for: Assigning GPIO 42 (clock) and 41 (data) for microphone input.
* **I2S.begin(I2S\_MODE\_PDM\_RX, 16000, I2S\_DATA\_BIT\_WIDTH\_16BIT, I2S\_SLOT\_MODE\_MONO)**
  + From: ESP\_I2S library
  + Purpose: Initializes the I2S interface in PDM receive mode with a 16 kHz sampling rate, 16-bit resolution, and mono audio format.
  + Used for: Setting up the I2S interface to match the digital microphone's output format.
* **I2S.read()**
  + From: ESP\_I2S library
  + Purpose: Reads a single audio sample from the I2S interface.
  + Used for: Capturing real-time audio data from the digital microphone.

**Sample Values:**

00:36:14.830 -> 1326

00:36:15.006 -> 1324

00:36:15.233 -> 1311

00:36:15.406 -> 1321

00:36:15.635 -> 1331

00:36:15.806 -> 1339

00:36:16.022 -> 1332

00:36:16.206 -> 1322

00:36:16.407 -> 1283

00:36:16.626 -> 1388

00:36:16.806 -> 1308

00:36:17.029 -> 1367

00:36:17.206 -> 1300

00:36:17.434 -> 1335

00:36:17.606 -> 1386

00:36:17.836 -> 1269

**Serial Graph Screenshot:**

* Screenshot of live graph showing audio signal over time:

A screenshot of a computer

AI-generated content may be incorrect.

I have also included a screen recording of the experiment in the folder.

## Section 4 – Camera (Bonus)

**Library and Code Source:**

The code used for the ESP32 camera module was taken from the example sketches provided in the ESP32 Camera library within the Arduino IDE. This example sets up a web server that streams video from the camera and allows snapshots via a browser interface.

* Library used: ESP32 board package (includes camera examples)
* Example used: File > Examples > ESP32 > Camera > CameraWebServer
* Functionality: Initializes the onboard camera, connects to Wi-Fi, and hosts a local web server that streams live camera feed.

**Note:**Due to firewall issues on the computer, the web server was accessed from a phone connected to the same network. A screenshot of the camera interface and a captured image are included in the report.

A screenshot of a video game

AI-generated content may be incorrect.A screen shot of a computer

AI-generated content may be incorrect.