# Embedded Software – B31DG Assignment 1

Rowan Daff – H00349503 - rad2000@hw.ac.uk

| Date | Author Name | Version Number | Notes |
|---|---|---|---|
| 1/02/2025 | Rowan Daff | 1.0 | Initial Draft |

## Contents



## Introduction

This assignment involves developing a software application for the ESP32 module, designed to generate a periodic digital signal controlled by two push buttons. The implementation is carried out in two environments: Arduino IDE and ESP-IDF within Visual Studio Code. The Arduino version served as the initial implementation, which was later reimplemented using ESP-IDF to achieve functionality in both environments. The generated signal is uniquely derived from the author's surname, "DAFF." This report outlines the calculations derived from the surname and presents the corresponding output data. It also includes the source code, hardware setup, and oscilloscope images and videos.

# Hardware Configuration

The hardware was designed to meet the requirements specified in the assignment PDF. It includes two LEDs, each connected in series with a 330-ohm resistor. These LEDs are wired to separate GPIO pins, with GPIO26 and GPIO27 selected for this purpose. GPIO16 and GPIO17 are used for the push buttons, which are connected to the ground via 1100-ohm resistors. For signal measurement, the oscilloscope replaces the LEDs. The circuit layout is illustrated in figure 1 and the real-world setup can be seen in figure 2.

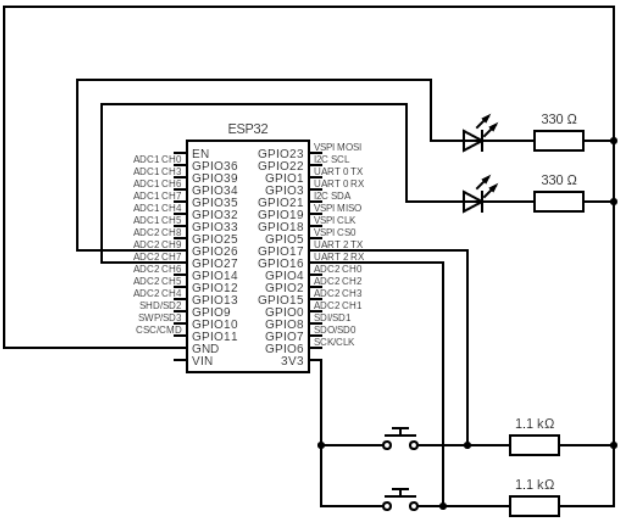


Figure 1: Circuit diagram showing the setup of the ESP32

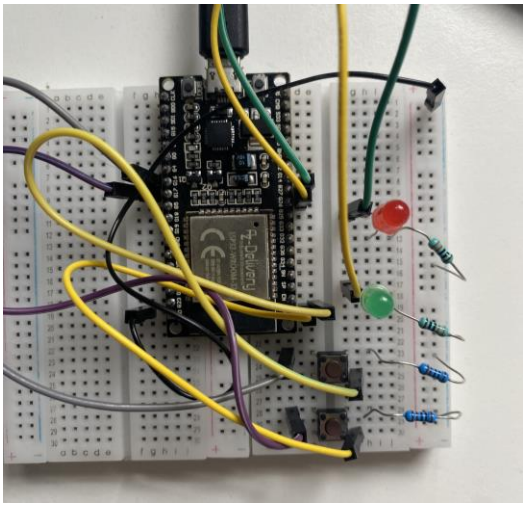

Figure 2: Hardware setup showing ESP32 connected to 2 push switches 4 resistors and 2 LEDS on a breadboard.

# Application Calculations

As previously mentioned, the assignment is tailored to each individual's surname. According to the mapping chart provided in the assignment outline, the name "DAFF" translates to 4166. The following calculations demonstrate how each letter corresponds to specific waveform attributes.

Surname = DAFF

D = 4, A = 1, F = 6, F = 6

Inital Pulse Width: $a = D \cdot 100us = 4 \cdot 100us = 400us$

Spaces Between Pulses: $b = A \cdot 100us = 1 \cdot 100us = 100us$

Number of Pulses (Default form): $c = F + 4 = 6 + 4 = 10$

Space Between Blocks: $d = F + 500us = 6 + 500us = 3000us$

$$T_{ON(n)} = a + \big((n-1)x\,50us\big),\ where\ 2 \leq n$$

This formula above shows that each pulses width will incrementally increase by 50us every pulse starting with the initial width of 400us.

$$Option\ Number = (F\ \%\ 4) + 1 = (6\ \%\ 4) + 1 = 2 + 1 = 3$$

The above formula shows the fifth (actually fourth as my name only has four letters) letter of my surname F which calculates giving us mode 3.

Mode 3: Insert an extra 3 pulses into each data waveform cycle (i.e. c+3 pulses in a data waveform cycle) until the Output Select push button is pressed again.

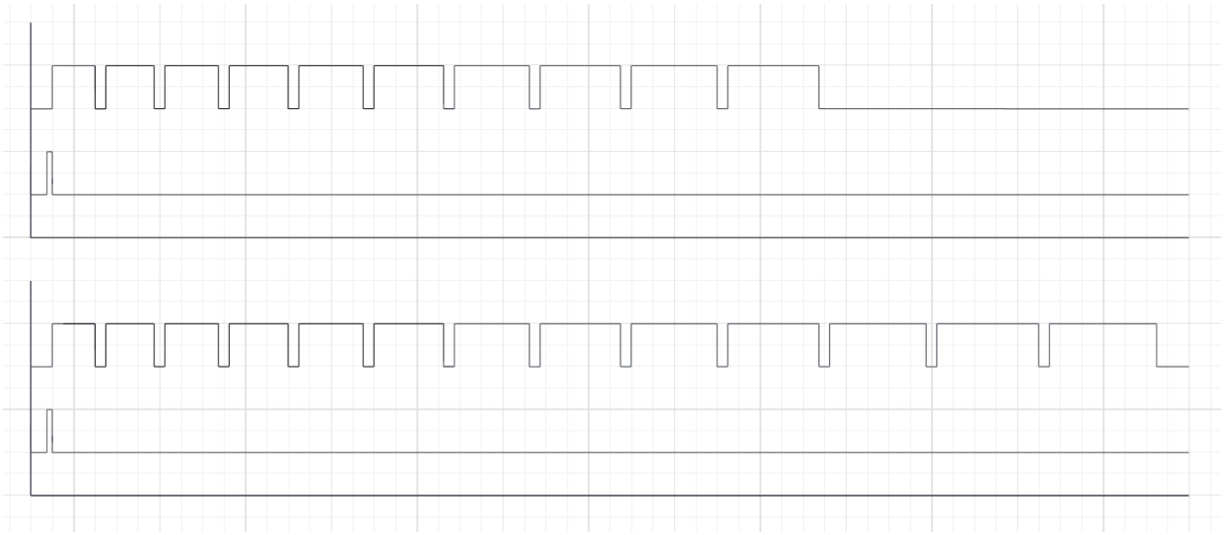

Figure 3: Oscilloscope desired outcome, showing the Default signal with 10 pulses above and the Alternative signal with 13 pulses below, each with trigger signals below their main waveform.

## Process flow chart

Figure 4 illustrates the code's process flow. It begins by initialising the GPIOs and reading Switch 1 to determine whether waveform generation should be enabled. If disabled, Signals A and B are turned off, and the system halts. If enabled, switch 2 is read to check for a mode change. The system then generates Signal A, using a pulse counter to incrementally increase the pulse width by 50µs if additional pulses remain. Each cycle involves setting Signal A HIGH, waiting for the designated pulse width, and then setting it LOW. This process repeats until the required number of pulses is generated.

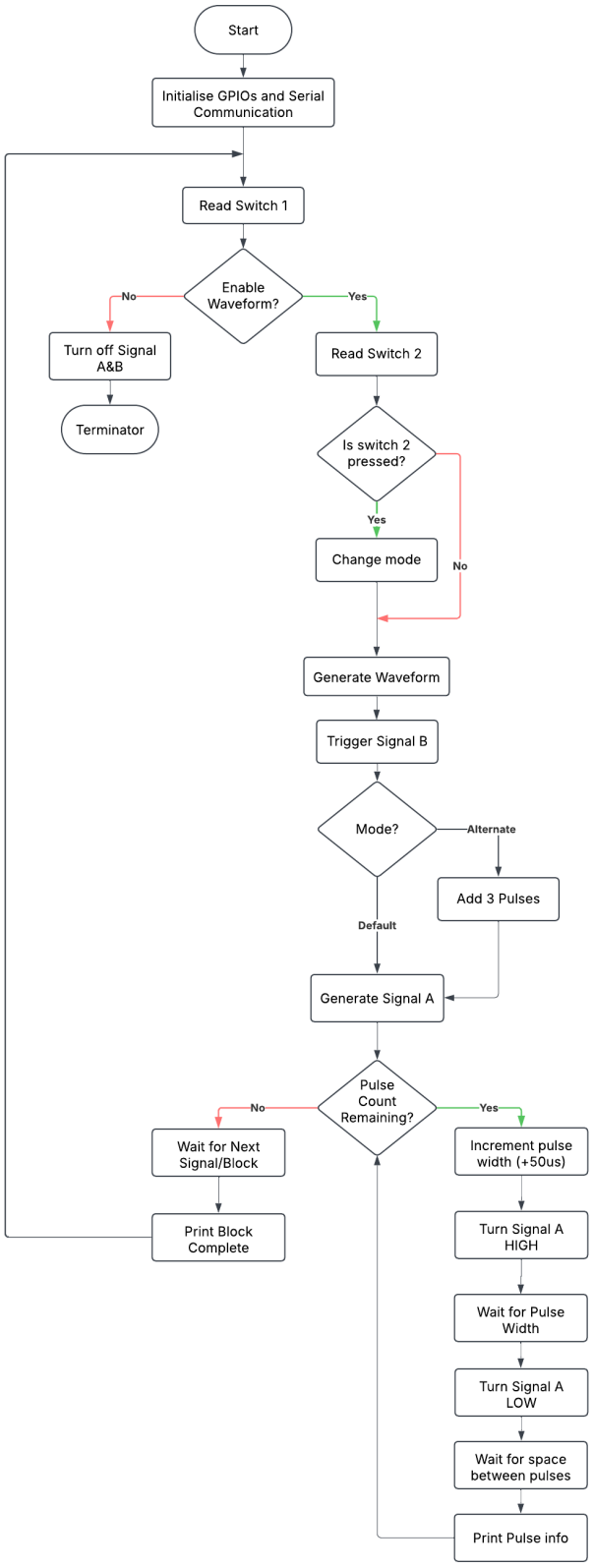


Figure 4: Flowchart of the program showing application control

# Output Signal

As shown in Figures 5 and 6, the program runs successfully, producing the expected results. When compared to Figure 3, the output matches the desired outcome, displaying clearly incrementing pulse widths and the correct number of pulses for each mode.

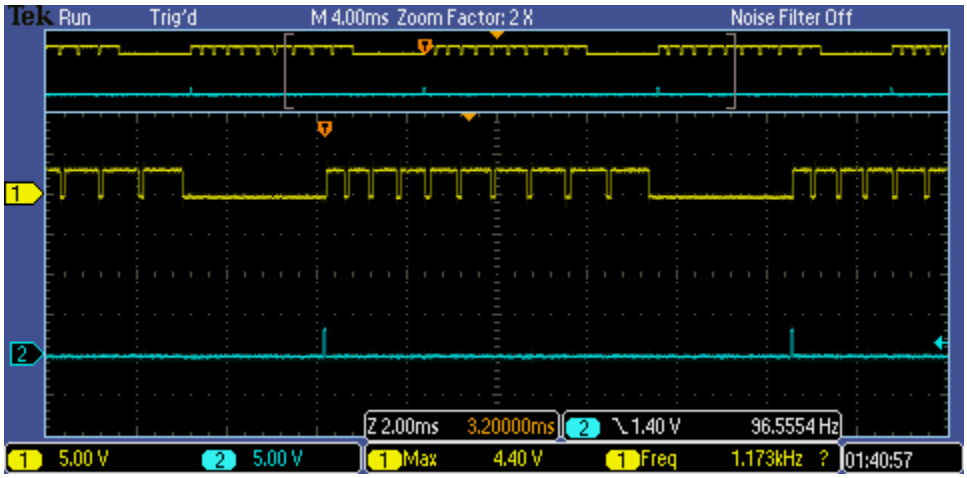


Figure 5: Oscilloscope screenshot showing the default waveform with 10 pulses (Yellow) and the trigger signal (Blue).

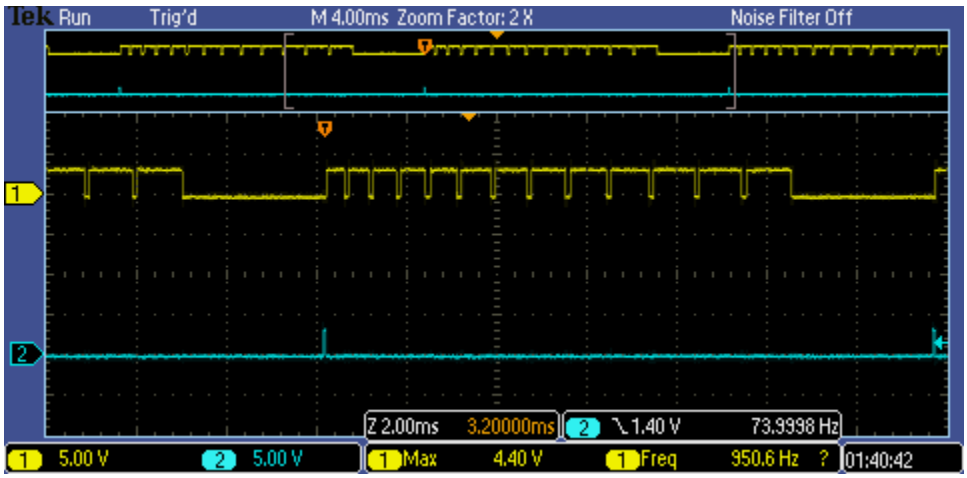


Figure 6: Oscilloscope screenshot showing the Alternative Signal with 13 pulses (Yellow) and the trigger signal (Blue).

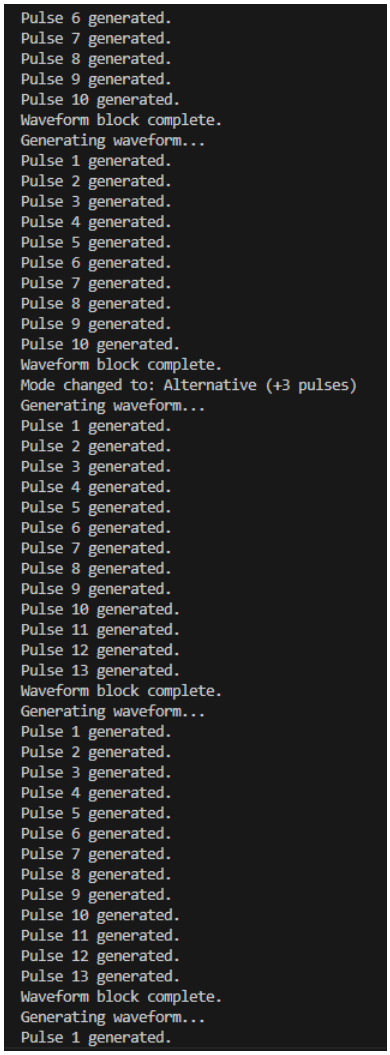


Figure 7: Console output showing print messages for each start, pulse, end, and mode change of each block within the signal.

# Conclusion

This assignment successfully devloped a software application for the ESP32 to generate a periodic signal controlled by push buttons, this was completed in both Arduino IDE and ESP-IDF, Ensuring compatibility across both environments. The parameters of the waveform were derived from the authors surname, which in this case was DAFF, with calculations determining the pulses widths, spacing, frequency, and modes. This assignment provided a chance to use the ESP32 devices which was enjoyable. Correctly installing and structuring ESP-IDF was difficult and took more time than was intended but was rewarding in the end. In conclusion the successful execution of the software as well as the desired outcome was achieved in both environments.