

# B31DG : Embedded Software

## *Assignment 1.*

---



## Summary

<b>1</b>	<b>Hardware</b>	<b>2</b>
1.1	Schematics . . . . .	2
1.2	Photos . . . . .	3
<b>2</b>	<b>Software</b>	<b>4</b>
2.1	Delay parameters . . . . .	4
2.2	Code commentary . . . . .	4
2.3	Nassi Shneiderman diagram . . . . .	6
2.4	Results . . . . .	7

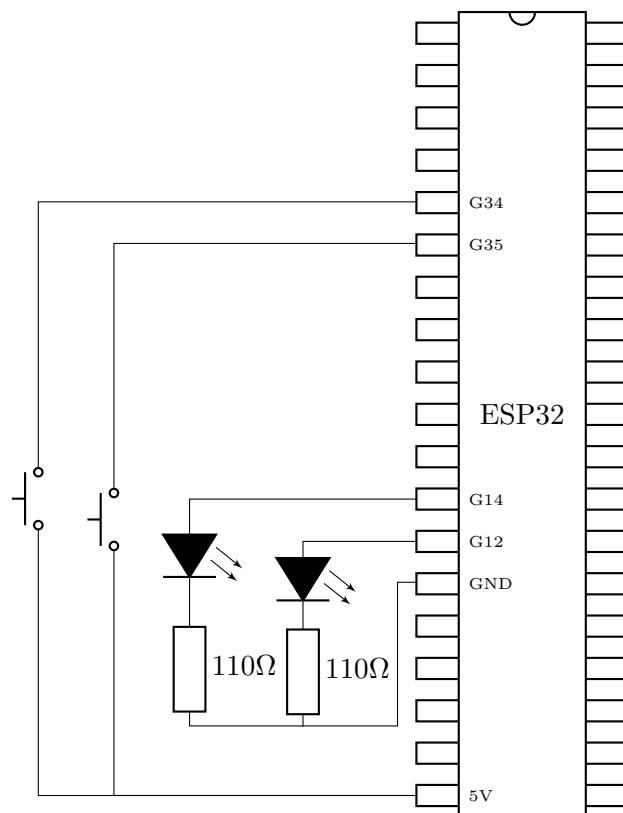
# 1 Hardware

## 1.1 Schematics

We are working on an ESP32

### Material used

- ESP 32
- 2 Resistor of  $230\Omega$
- 2 Green LED
- 2 switchs



## 1.2 Photos

Photos of the electronic assembly

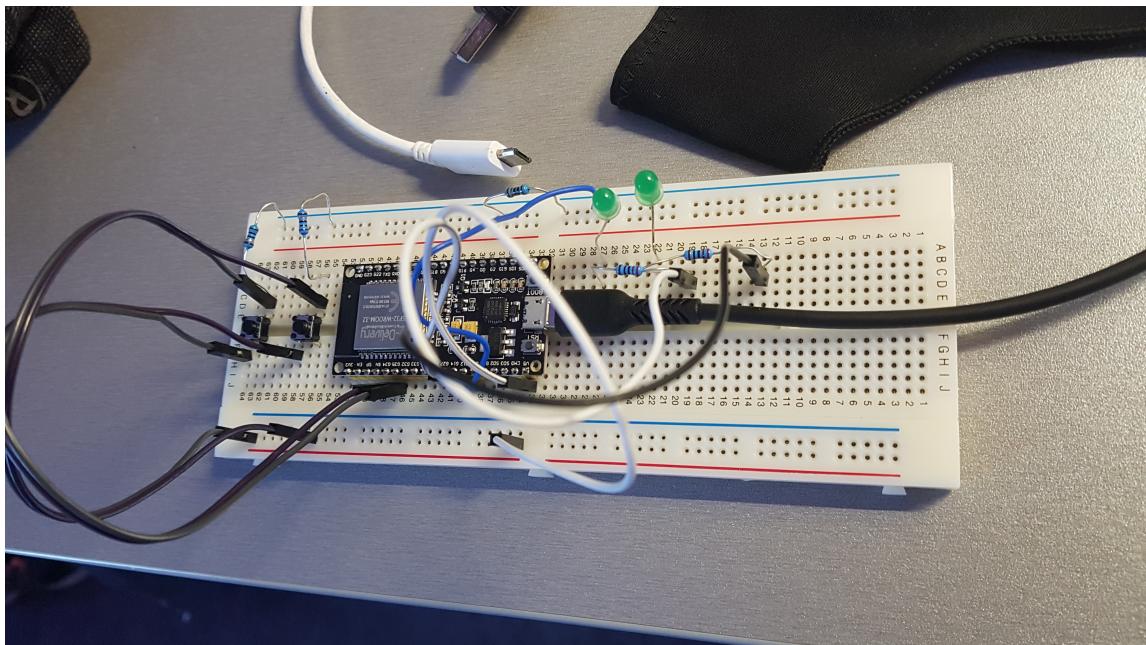


FIGURE 1 – Signal A off

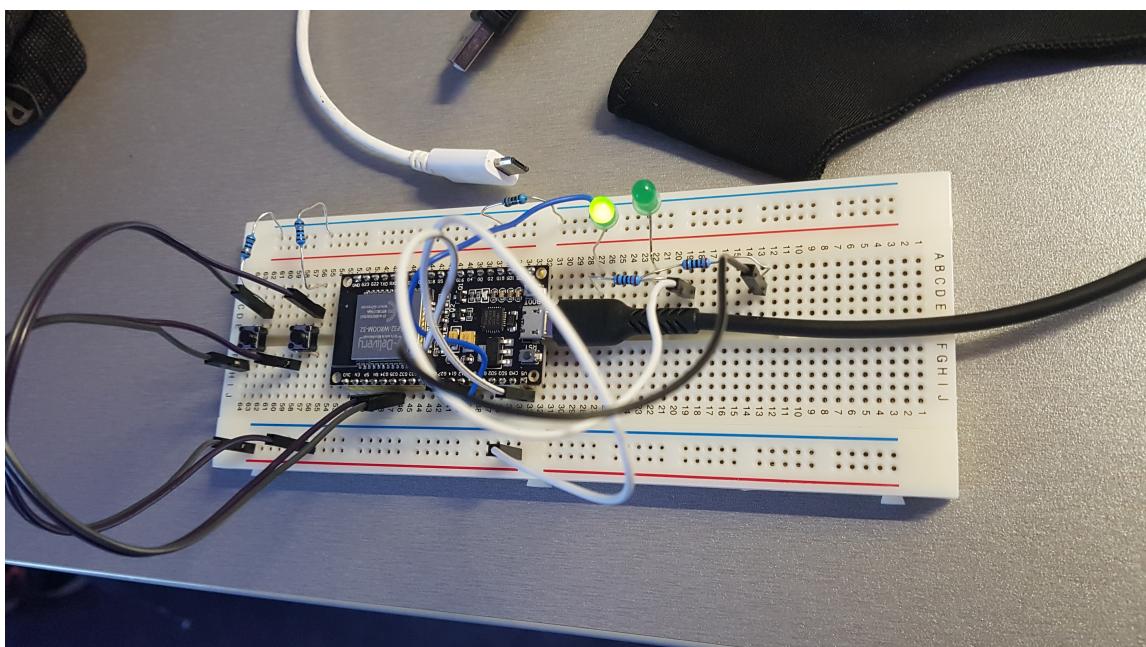


FIGURE 2 – Signal A on

## 2 Software

### 2.1 Delay parameters

In this section, We are going to calculate the delays of the signal using my name "emonot":

$$a = e \times 100 = 500\mu S$$

$$b = m \times 100 = 1300\mu S$$

$$c = o + 4 = 16 pulses$$

$$d = n \times 500 = 6500\mu S$$

$$B = 50ms$$

$$Mode = rem(\frac{o}{4}) + 1 = 1$$

If we pressed the second button, we apply the mode one which remove 3 pulses from c. So :

$$c = 13$$

### 2.2 Code commentary

This section describe the general structure of the code. This structure is divided into 4 sections.

#### sections

- Pin assignment
- Constant delay for the signal
- Pin property
- Algorithm loop

Section for the pin assignment :

```
1 // defining constant for the INPUT and OUTPUT of the ESP32
2 #define SIGNAL_A 14
3 #define SIGNAL_B 12
4 #define INPUT_1 35
5 #define INPUT_2 34
```

Some example of the constant declaration

```
1 int input_2.Btn = 0;
2 int const B = 50*1000;
3 int const b = 1300*1000;
```

Section for assigning pin property

```
1 void setup() {
2     // initialize digital pins as inputs and outputs
3     pinMode(SIGNAL_A, OUTPUT);
4     pinMode(SIGNAL_B, OUTPUT);
5     pinMode(INPUT_1, INPUT);
6     pinMode(INPUT_2, INPUT);
7 }
```

This is an example of a section of the algorithm that read the states of the buttons and verify a condition. If the condition is met, it executes a certain number of pulses with the delay provided by the calculus in chapter 2.1.

```

1  input_1_Btn = digitalRead(INPUT_1);
2  input_2_Btn = digitalRead(INPUT_2);
3
4  //4 conditions to verify the states of the buttons as there are pressed or not
5  // if both button are not pressed , we run in normal mode
6
7  if(input_1_Btn == LOW and input_2_Btn == LOW)
8  {
9      //we apply the bliking of the LEDS as requested in the assignement
10     digitalWrite(SIGNAL_B, HIGH);
11     delayMicroseconds(B);
12     digitalWrite(SIGNAL_B, LOW);
13
14     //we have 16 blink with a period increment between each blink
15     for(i=0;i<16;i++){
16         digitalWrite(SIGNAL_A, HIGH);
17         delayMicroseconds(a);
18         digitalWrite(SIGNAL_A, LOW);
19         a += 50000;
20         delayMicroseconds(b);
21     }
22     delayMicroseconds(d);
23     digitalWrite(SIGNAL_B, LOW);
24     a = 500*1000;
25 }
26 }
```

The whole code is published on git, the link can be find in the README file

### 2.3 Nassi Shneiderman diagram

This diagram follows the code structure

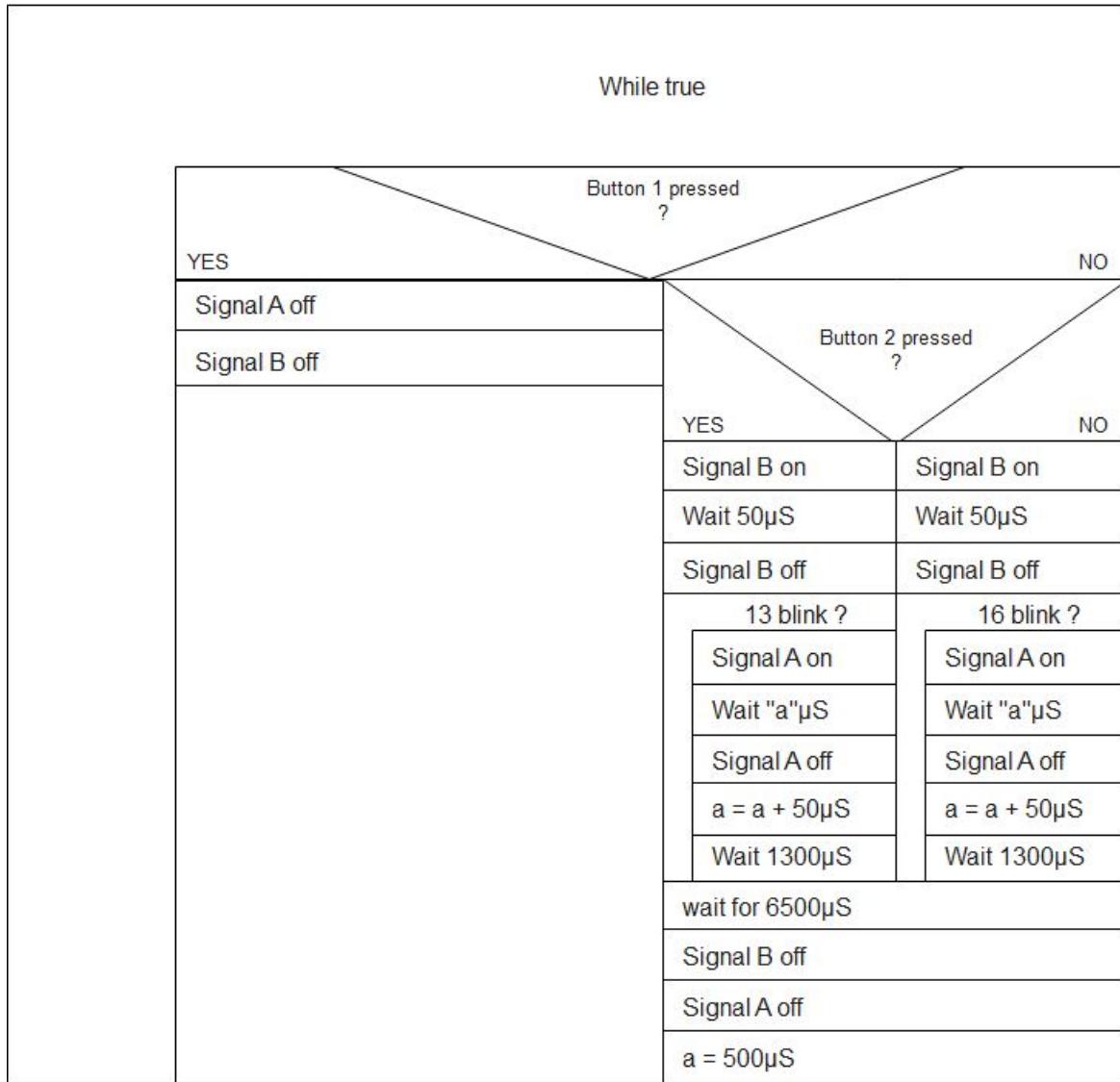


FIGURE 3 – Nassi Shneiderman diagram

## 2.4 Results

The code provided on git contains 2 files because we have 2 ways to verify our program. First we can increase the delay of the constant to see if the algorithm is working properly. If we run the normal code with delays in microseconds, then we verify some information on the oscilloscope.

In this case we can see that for each new 16 pulse of signal A small signal B is detected.



FIGURE 4 – The line in yellow is SIGNAL A, The line in blue is SIGNAL B

On this second image we can see that after that the pulse's delay of the A signal is bigger than the next ones which mean that the delay "a" is correctly increasing.



FIGURE 5 – The line in yellow is SIGNAL A, The line in blue is SIGNAL B

## 2.5 Entire source code

```

1  /*
2 First assignement embedded software
3
4 Name : EMONOT—DE CAROLIS Evrard
5 ID : H00385163
6
7 */
8
9 // defining constant for the INPUT and OUTPUT of the ESP32
10 #define SIGNAL_A 14
11 #define SIGNAL_B 12
12 #define INPUT_1 35
13 #define INPUT_2 34
14
15 // defining the period calculated with my name
16 int input_1.Btn = 0;
17 int input_2.Btn = 0;
18 int const B = 50;
19 int const b = 1300;
20 int const d = 6500;
21 int a = 500;
22 int i = 0;
23
24 // the setup function runs once we boot the board
25 void setup() {
26     // initialize digital pins as inputs and outputs
27     pinMode(SIGNAL_A, OUTPUT);
28     pinMode(SIGNAL_B, OUTPUT);
29     pinMode(INPUT_1, INPUT);
30     pinMode(INPUT_2, INPUT);
31 }
32
33 // the loop function acts like a while true , that runs while the ESP32 is power
34 void loop() {
35
36     //turning the LED off
37     digitalWrite(SIGNAL_B, LOW);
38     digitalWrite(SIGNAL_A, LOW);
39
40     //read the buttons
41     input_1.Btn = digitalRead(INPUT_1);
42     input_2.Btn = digitalRead(INPUT_2);
43
44     //4 conditions to verify the states of the buttons as there are pressed or not
45     // if both button are not pressed , we run in normal mode
46
47     if(input_1.Btn == LOW and input_2.Btn == LOW)
48     {
49         //we apply the bliking of the LEDS as requested in the assignement
50         digitalWrite(SIGNAL_B, HIGH);
51         delayMicroseconds(B);
52         digitalWrite(SIGNAL_B, LOW);
53
54         //we have 16 blink with a period increment between each blink
55         for(i=0;i<16;i++){
56             digitalWrite(SIGNAL_A, HIGH);
57             delayMicroseconds(a);
58             digitalWrite(SIGNAL_A, LOW);
59             a += 50;
60             delayMicroseconds(b);
61         }
62     }
63 }
```

```

62     delayMicroseconds(d);
63     digitalWrite(SIGNAL_B, LOW);
64     a = 500;
65 }
66
67 // if only the second button is pressed , we run in the second mode
68 else if(input_2.Btn == HIGH and input_1.Btn == LOW)
{
    //we apply the bliking of the LEDS as requested in the assignment
    digitalWrite(SIGNAL_B, HIGH);
    delayMicroseconds(B);
    digitalWrite(SIGNAL_B, LOW);

    //we have 13 blinks with "a" period increment between each blink
    for(i=0;i<13;i++){
        digitalWrite(SIGNAL_A, HIGH);
        delayMicroseconds(a);
        digitalWrite(SIGNAL_A, LOW);
        a += 50;
        delayMicroseconds(b);
    }
    delayMicroseconds(d);
    digitalWrite(SIGNAL_A, LOW);
    digitalWrite(SIGNAL_B, LOW);S
    a = 500; //reset a to the normal period
}
88
89 //on all the other cases , we just turns the LEDS off
90
91 else if(input_2.Btn == HIGH and input_1.Btn == HIGH)
{
    digitalWrite(SIGNAL_B, LOW); // turn the LED on (HIGH is the voltage level)
    digitalWrite(SIGNAL_A, LOW);
    delay(5);
}
97
98 else if(input_2.Btn == LOW and input_1.Btn == HIGH)
{
    digitalWrite(SIGNAL_B, LOW); // turn the LED on (HIGH is the voltage level)
    digitalWrite(SIGNAL_A, LOW);
    delay(1000);
}
104
105 else
{
    digitalWrite(SIGNAL_B, LOW); // turn the LED off by making the voltage LOW
    digitalWrite(SIGNAL_A, LOW);
    delay(1000);
}
111
112 }
113 //END

```