Module 1 – Part B
Module 2 – Part A
Assignment
IOT internship

Team Members: -

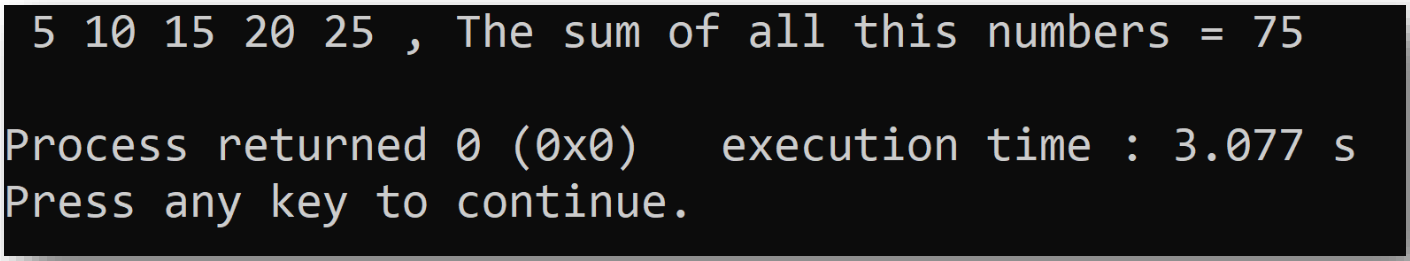
TL: Mahmoud Essam Fathy: 20221460231
Ziad Ashraf Hafez Gaber: 20221374025
Ziad Ashraf Ibrahim Taher: 20221369225
Muhammad Ashraf: 20221372763

Assignment 2 – Module 1 – Part B – P1

Pseudo Code

```
1-declare constant called num_elements with value of 5
2-Creating function called array_sum_function to take input array and retrieving
summation
  function will contain
  1.1-varaible called sum_result
  1.2-for loop to trace all array values
    1.1.1-loop will print each value in array to make sure that all values are choose
succesfully
    1.1.2-storing summation of all this values into sum_result
  1.3-returning the value of sum
3-in main function declaring array with size of "num_elements" and put the needed
values
4-declaring sum variable and putting it equal to the function passed the array into
5-printing the values of the array and printing sum value
```

Output



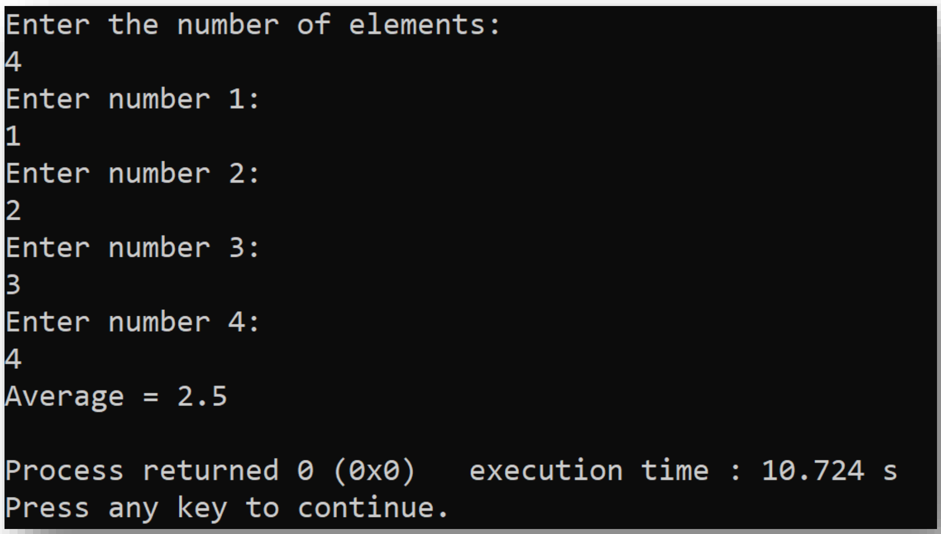
```
 5 10 15 20 25 , The sum of all this numbers = 75

Process returned 0 (0x0)   execution time : 3.077 s
Press any key to continue.
```

Module 1 Assignment
Part A – Problem 2

Pseudo Code

```
1. Average function take array as parameter and size of array and calculate average and return it
2. In main function make a for loop to make end user write numbers and store them in array
3. Then declaring variable will store the value of this avarage after calling the function
4. Printing the results
```

Output



```
Enter the number of elements:
4
Enter number 1:
1
Enter number 2:
2
Enter number 3:
3
Enter number 4:
4
Average = 2.5

Process returned 0 (0x0)   execution time : 10.724 s
Press any key to continue.
```

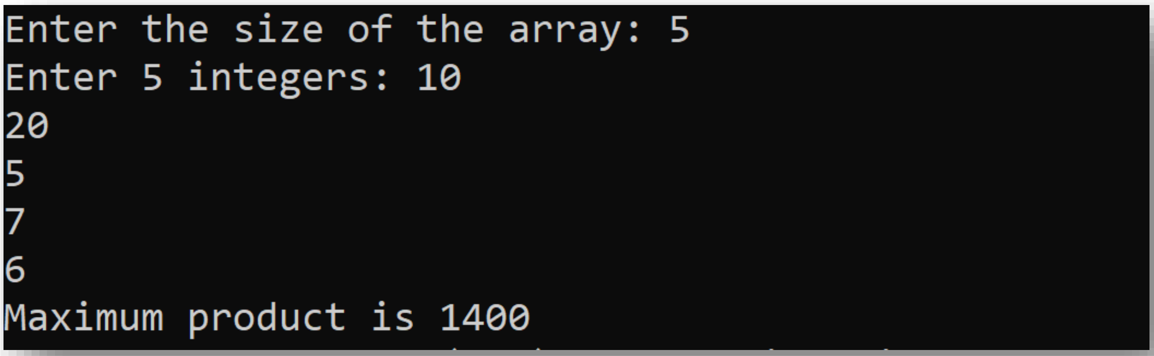
Mahmoud Essam, Ziad Ashraf, Ziad Ashraf , Muhammed ashraf
Supervised by: Eng. Muhammed Hatem

Assignment 2 – Module 1 – Part B – P3

Pseudo Code

```
Initialize function max_of_Product with parameters arr[] and n
2. If n is less than 3, return -1
END IF
Initialize max_product as INT_MIN
Loop through the array with index i from 0 up to n-2(for loop with index "i" iterates through each element in the array
from the beginning up to the third-to-last element.)
    a. Loop through the array with index j from i+1 up to n-1(for loop with index "j" iterates through each element in the
array from "i+1" up to the second-to-last element.)
        i. Loop through the array with index k from j+1 up to n(for loop with index "k" iterates through each element in the
array from "j+1" up to the last element.)
End for
            1. Find the product of arr[i], arr[j], and arr[k]
            2. If the product is greater than max_product, update max_product with the product
Return max_product
END FUNCTION
Declare an array arr with values {10, 3, 5, 6, 20} and size n as the size of the array divided by the size of the first element
8. Call the max_of_Product function with parameters arr[] and n and store the result in max
9. If max is equal to -1, print "No Triplet Exists"
10. Else, print "Maximum product is " concatenated with max.
```

Output

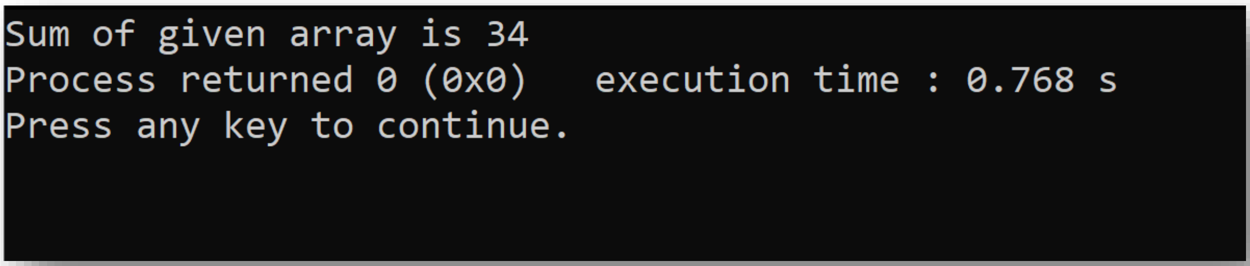

```
Enter the size of the array: 5
Enter 5 integers: 10
20
5
7
6
Maximum product is 1400
```

Module 1 Assignment
Part A – Problem 5

Pseudo Code

```
INITIALIZE iostream library
FUNCTION sum(int arr[],int n)
DECLARE integer sum=0
FOR (int i=0, i<n,i++)
COMPUTE sum=arr[i]+sum
ENDFOR
RETURN sum
ENDFUNCTION
DECLARE integer array arr={12,3,4,15}
DECLARE integer n=4
DISPLAY "Sum of given array is "+ sum(arr, n)
RETURN SUCCESS
```

Output



```
Sum of given array is 34
Process returned 0 (0x0)   execution time : 0.768 s
Press any key to continue.
```

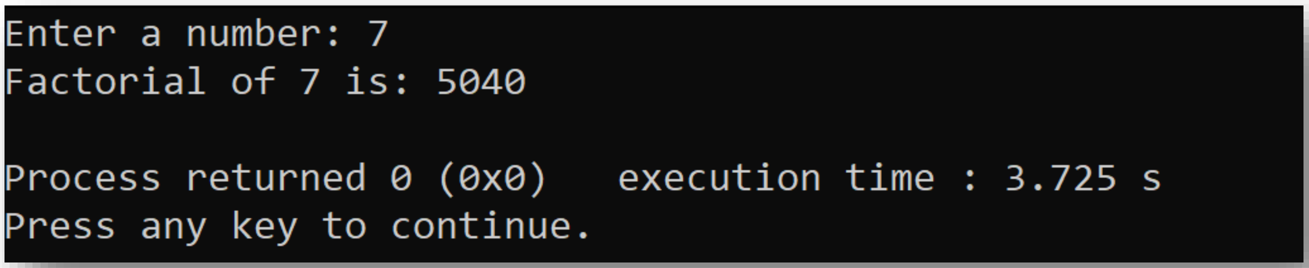
Mahmoud Essam, Ziad Ashraf, Ziad Ashraf , Muhammed ashraf
Supervised by: Eng. Muhammed Hatem

Assignment 2 – Module 1 – Part B – P6

Pseudo Code

```
1. Define a function called factorial that takes an integer n as input:
   a. Check if n < 0:
      i. If yes, return -1 as factorial is not defined for negative numbers.
   b. Check if n is 0:
      i. If yes, return 1 as factorial of 0 is 1.
   c. Otherwise, calculate the factorial of n using recursion and return the result.

2. In the main function:
   a. Prompt the user to enter an integer n.
   b. Read the value of n from the user.
   c. Call the factorial function with n as input.
   d. If the value returned by the factorial function is -1, print a message indicating that
factorial is not defined for negative numbers.
   e. Otherwise, print the value of the factorial.
```

Output



```
Enter a number: 7
Factorial of 7 is: 5040

Process returned 0 (0x0)   execution time : 3.725 s
Press any key to continue.
```

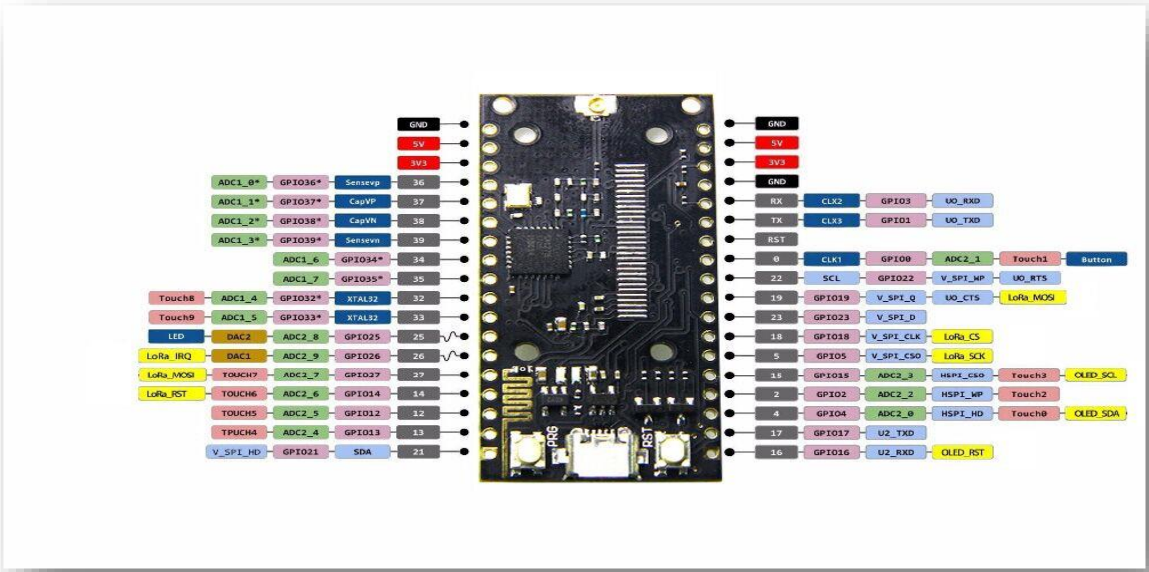
Module 2 Assignment Part B – Theoretical

< Write down a brief description of the Tensilica Xtensa LX6 Daul >
< core microcontroller >

<Answer>

Tensilica Xtensa LX6 Dual core 32-bit
Is the CPU (main microprocessor) of the microcontroller ESP32. It operates at clock frequency up to 240MHz with 2 cores allowing multitasking. It comes built in with Internal memory:

ROM: 448 Kb (For booting and core functions.)
SRAM: 520 Kb (For data and instruction.)


GND
5V
3V3
ADC1_0* GPIO36* Sensevp 36
ADC1_1* GPIO37* CapVP 37
ADC1_2* GPIO38* CapVN 38
ADC1_3* GPIO39* Sensevn 39
ADC1_6 GPIO34* 34
ADC1_7 GPIO35* 35
Touch8 ADC1_4 GPIO32* XTAL32 32
Touch9 ADC1_5 GPIO33* XTAL32 33
LED DAC2 ADC2_8 GPIO25 25
LoRa_IRQ DAC1 ADC2_9 GPIO26 26
LoRa_MOSI TOUCH7 ADC2_7 GPIO27 27
LoRa_RST TOUCH6 ADC2_6 GPIO14 14
TOUCH5 ADC2_5 GPIO12 12
TPUCH4 ADC2_4 GPIO13 13
V_SPI_HD GPIO21 SDA 21
PRG
RST
GND
5V
3V3
GND
RX CLX2 GPIO3 U0_RXD
TX CLX3 GPIO1 U0_TXD
RST
0 CLK1 GPIO0 ADC2_1 Touch1 Button
22 SCL GPIO22 V_SPI_WP U0_RTS
19 GPIO19 V_SPI_Q U0_CTS LoRa_MOSI
23 GPIO23 V_SPI_D
18 GPIO18 V_SPI_CLK LoRa_CS
5 GPIO5 V_SPI_CS0 LoRa_SCK
15 GPIO15 ADC2_3 HSPI_CS0 Touch3 OLED_SCL
2 GPIO2 ADC2_2 HSPI_WP Touch2
4 GPIO4 ADC2_0 HSPI_HD Touch0 OLED_SDA
17 GPIO17 U2_TXD
16 GPIO16 U2_RXD OLED_RST

Module 2 Assignment Part B – Theoretical

< Different comparisons micro-controllers to micro-processors >

<Answer>

There are various similarities between the microprocessor and microcontroller as they perform relatively the same tasks. There exist differences between microprocessor and microcontroller which are elaborated in the table provided below on various factors

Microprocessor	Microcontroller
Heart of the system.	Heart of the embedded system.
Externally connected with input-output components.	input-output components are embedded.
The circuit may be large depending upon usage.	The circuit is very small.
Not cost-effective.	Cost-effective.
The total consumption of power is high.	Total consumption of power is less.
Power saving mode is not generally available.	Power saving mode is generally offered.
Used in PC.	Used in MP3 players, washing machines, etc.
Memories like RAM and ROM are absent.	Carries RAM, ROM, etc.
Runs at a very high speed.	Runs at a relatively lower speed.
It is complex and costly.	Simple and cheap.
Example: DEC Alpha 21164, IBM RS6000, etc	Example: Intel 8031/8051, PIC1x, etc.

Is ESP32 a Microprocessor?

The answer is yes, the ESP32 is a microprocessor. ESP32 microprocessor is a chip-based system that offers a full range of functions, such as a microcontroller, as well as built-in capabilities for Wi-Fi and Bluetooth connectivity. The ESP32 can be programmed using the Arduino IDE or other development environments, making it easy to use for hobbyists and professionals alike.

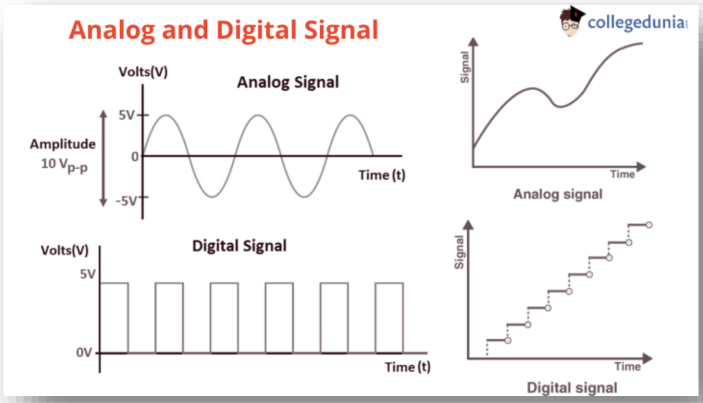
The ESP32 is a type of microcontroller that is cost-effective along with low energy requirements. It has built-in Wi-Fi and can use both Bluetooth modes. It uses either a dual-core or single-core Tensilica Xtensa LX6 microprocessor or a single-core RISC-V microprocessor.

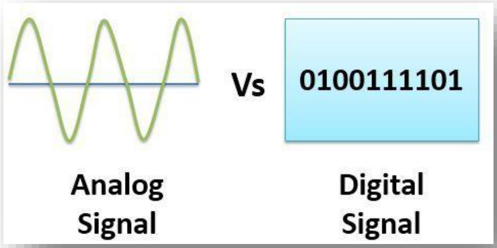
Module 2 Assignment Part B – Theoretical

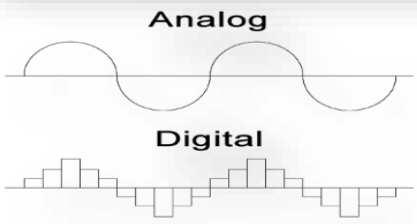
< Compare digital signals to analog signal with at least 5 different Comparisons >

<Answer>

Analog signals	Digital signals
Analog signals are difficult to get analyzed at first.	Digital signals are easy to analyze.
Analog signals are more accurate than digital signals.	Digital signals are less accurate.
Analog signals take time to be stored. It has infinite memory.	Digital signals can be easily stored.
To record an analog signal, the technique used, preserves the original signals.	In recording digital signal, the sample signals are taken and preserved.
There is a continuous representation of signals in analog signals.	There is a discontinuous representation of signals in digital signals.
Analog signals produce too much noise.	Digital signals do not produce noise.
Examples of analog signals are Human voice, Thermometer, Analog phones etc.	Examples of digital signals are Computers, Digital Phones, Digital pens, etc.







Mahmoud Essam, Ziad Ashraf, Ziad Ashraf , Muhammed ashraf
Supervised by: Eng. Muhammed Hatem

Module 2 Assignment Part B – ESP32 Problem 1 (Bar Led)

Description of connections

This code controls a set of LEDs using a potentiometer to adjust how many LEDs are turned on. The number of LEDs is set to 4 and their pin numbers are defined in an array. The potentiometer pin is set to pin 34.

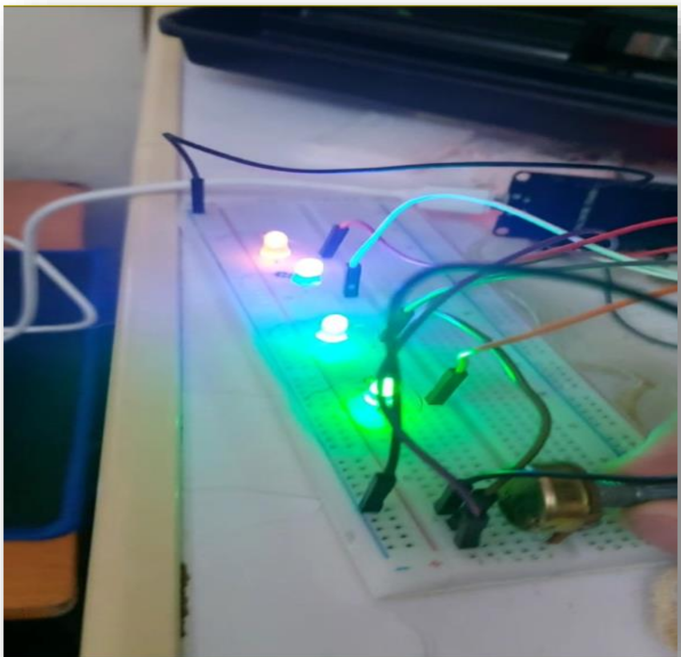
The setup() function sets the LED pins as outputs.

The loop() function reads the potentiometer value and waits for user input from the serial monitor. If a value greater than 0 is entered, the number of LEDs to be turned on is set to that value. The potentiometer value is then mapped to a range of 0 to the selected number of LEDs. The number of LEDs to be turned on is printed to the serial monitor.

The for loop iterates through each LED and turns on the appropriate number, based on the mapped value of the potentiometer reading.

Finally, a delay is added to reduce flicker.

Everything Provided in the folder "Videos – Circuits – Schematic – Real-life – and TinkerCAD action"



Mahmoud Essam, Ziad Ashraf, Ziad Ashraf , Muhammed ashraf
Supervised by: Eng. Muhammed Hatem

Module 2 Assignment Part B – ESP32 Button Press

Description of connections

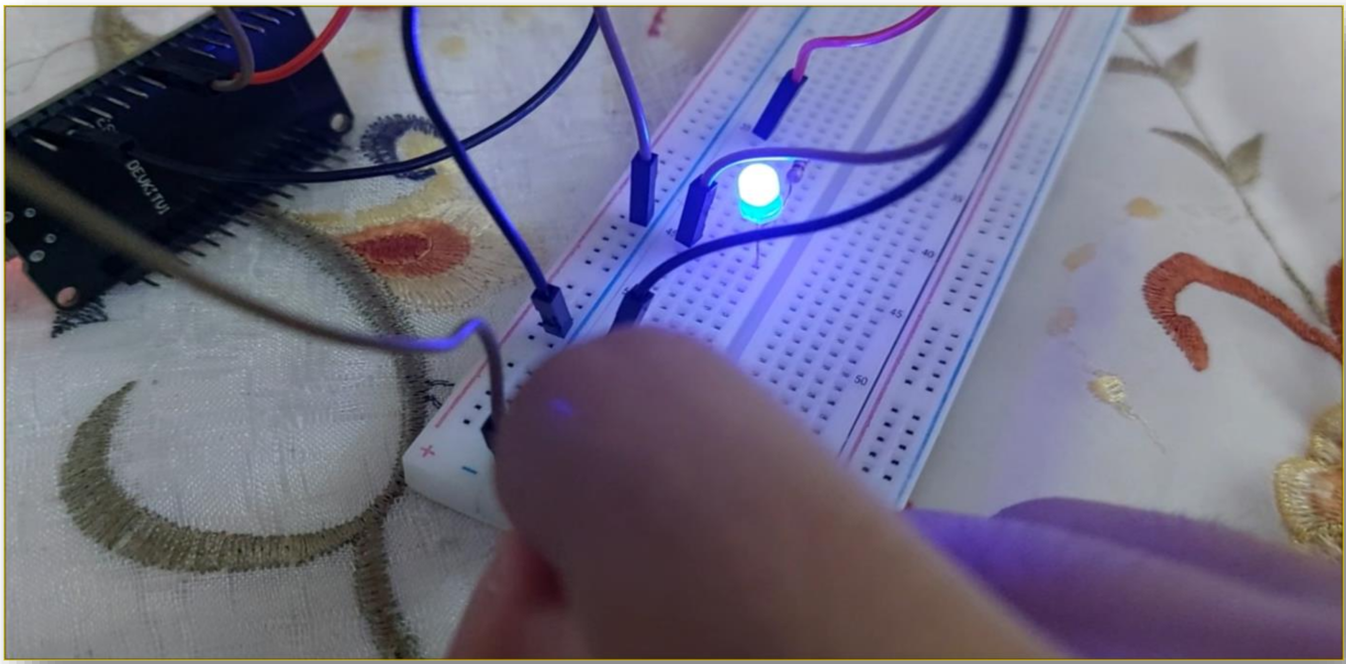
The LED is connected to pin **18** of the ESP32 board, and the push button switch is connected to pin **5.**

The setup() function initializes the **serial monitor**, sets the LED pin as an **output**, and configures the push button pin as an **input with a pull-up resistor enabled**.

In the loop() function, the code reads the value of the push button pin using the **digitalRead()** function. If the button is not pressed, **the LED is turned off** using **digitalWrite()** with a value of **LOW**. If the button is pressed, the LED is turned on using **digitalWrite()** with a value of **HIGH**.

The code continuously checks the state of the push button and updates the LED accordingly.

Everything Provided in the folder "Videos – Circuits – Schematic – Real-life – and TinkerCAD action"



Mahmoud Essam, Ziad Ashraf, Ziad Ashraf , Muhammed ashraf
Supervised by: Eng. Muhammed Hatem

Module 2 Assignment Part B – ESP32 Problem 1 (LDR Buzzr)

Description of connections

This code is designed to monitor the light level using a light-dependent resistor (LDR) connected to GPIO pin 34 on the microcontroller board. If the light level falls below a certain threshold (set by the threshold variable), an alarm is triggered using a buzzer connected to GPIO pin 13. An LED connected to GPIO pin 12 is also used to provide visual feedback.

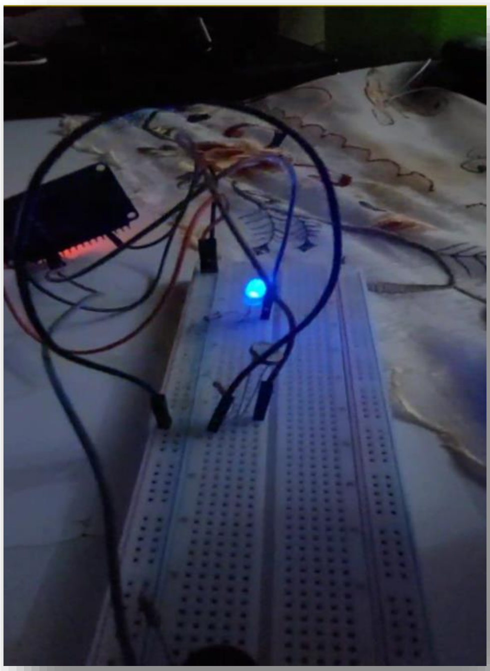
Here is a simplified description of the connections:

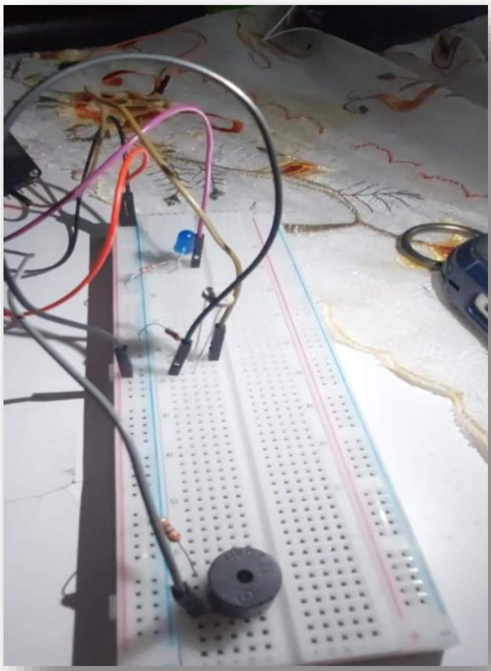
- Connect the LDR to GPIO pin 34 on the microcontroller board.
- Connect the positive lead of the buzzer to GPIO pin 13 on the microcontroller board, and the negative lead to ground.
- Connect the positive lead of the LED to GPIO pin 12 on the microcontroller board, and the negative lead to ground.

Note that you may need to adjust the threshold value to suit your specific application, depending on the light levels you are working with.

You can adjust this value by changing the threshold variable in the code.

Everything Provided in the folder "Videos – Circuits – Schematic – Real-life – and TinkerCAD action"





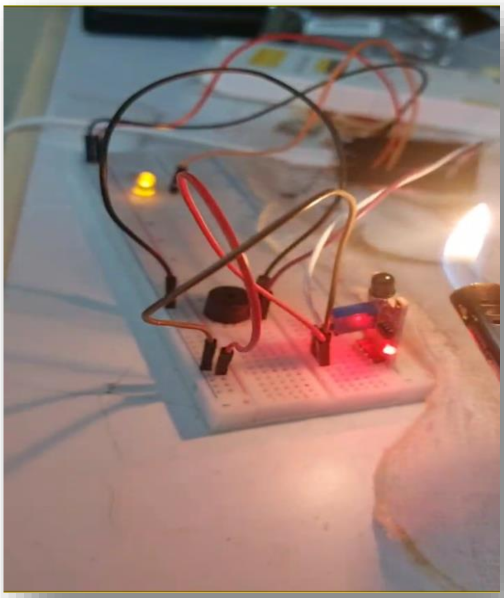
Mahmoud Essam, Ziad Ashraf, Ziad Ashraf , Muhammed ashraf
Supervised by: Eng. Muhammed Hatem

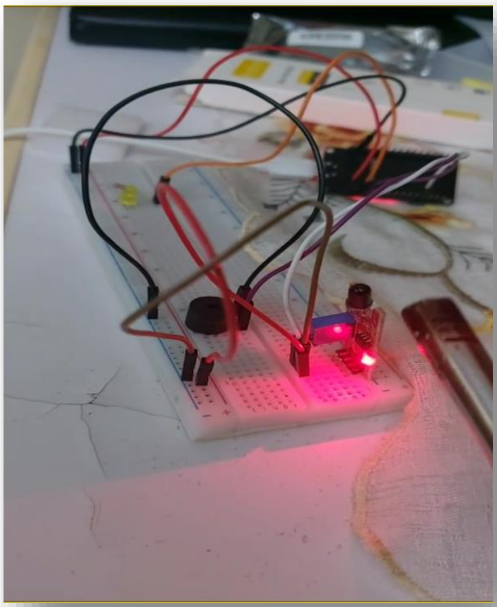
Module 2 Assignment Part B – ESP32 Flame Sensor "Bonus"

Description of connections

This code is designed to monitor the temperature using a **temperature** sensor connected to GPIO pin **34** on the microcontroller board. If the temperature falls below 0 degrees Celsius, an alarm is triggered using a **buzzer** connected to GPIO pin **12** and an **LED** connected to GPIO pin **13** is turned on.
Here is a description of the connections:
Connect the **temperature sensor** to GPIO pin **34** on the microcontroller board.
Connect the positive lead of the **buzzer** to GPIO pin **12** on the microcontroller board, and the negative lead to ground.
Connect the positive lead of the **LED** to GPIO pin **13** on the microcontroller board, and the negative lead to ground.
Note that the code assumes that a temperature below 0 degrees Celsius indicates a fire, which may not be accurate or appropriate for all situations. **You may need to adjust the code to suit your specific application and temperature range.**
Also, note that the map() function is converting the raw analog value **(which ranges from 0 to 4095)** to a voltage value **(which ranges from 0 to 3.3V)**, assuming that the maximum voltage input is 3.3V. If your temperature sensor uses a different voltage range, you may need to adjust the map() function accordingly.

Everything Provided in the folder "Videos – Circuits – Schematic – Real-life – and TinkerCAD action"





Mahmoud Essam, Ziad Ashraf, Ziad Ashraf , Muhammed ashraf
Supervised by: Eng. Muhammed Hatem