

**Press the Buttons  
& LED pattern**

**Getting started**

A circuit board

Description automatically generated Before turning on the Raspberry Pi wire up  
the board using the details below.

|  |  |
| --- | --- |
| **LED&KEY** | **Raspberry Pi** |
| VCC | 3.3V (Pin 1) |
| GND | GND (Pin 6) |
| STB | GPIO 22 (Pin15) |
| CLK | GPIO 21 (Pin 40) |
| DIO | GPIO 17 (Pin11) |

 Time to connect the   
power to boot up the   
Raspberry Pi.

**Code to read the buttons**

 Let’s start coding:

First we’re going to read the buttons and learn a bit about binary.

 Run Python 3 and open a new file [File]-[New].  
Save the file and call it **pressbuttons.py**

You could use any name but it makes it easier for the workshop if everyone is the same

Enter the code below.

|  |
| --- |
| #!/usr/bin/env python3  # Output the value for the buttons.  # TM1638  import TM1638  # These are the pins the display is connected to. Adjust accordingly.  # In addition to these you need to connect to 3.3V and ground.  DIO = 17  CLK = 21  STB = 22  display = TM1638.TM1638(DIO, CLK, STB)  display.enable(1)  display.set\_led(0, True)  count = 0  active = True  while active == True:  ····keys = display.get\_buttons()  ····display.set\_text(str(keys))  ····print(str(keys))  ····if keys == 128:  ········active = False |

Run your program and press the buttons.

NOTE: If you press the button on the right the program will exit (where is this in the code)

If all is working well then different numbers will appear on the 7 segment display.

Press each button one by one. Can you see a pattern in the numbers.

Try pressing more than one button at a time. Do they still make sense.

The numbers are decimal version of binary (except kind of backwards)

|  |  |
| --- | --- |
| **Binary** | **Decimal** |
| 00000001 | 1 |
| 00000010 | 2 |
| 00000100 | 4 |
| 00001000 | 8 |
| 00010000 | 16 |
| 00100000 | 32 |
| 01000000 | 64 |
| 10000000 | 128 |

Binary numbers are where each digit can only be zero (0) or one (1). This is how computers store data/information. For computers each 0/1 is called a Bit. Eight Bits is called a Byte. Computer store their data in Byte sized pieces. So, this board having 8 LEDs, 8 7-Segment display and 8 buttons is no coincident, It is the most that can be handled with Bytes. If there were 9 then another set of Bytes would be needed.

For this board the buttons are mapped on the opposite way to the bits.  
The button on the left is the bit on the right.

**Light up some LEDs**

 A fun pattern for LEDs is to make them light up one by one from left to right and then back again:

 In Python 3 open a new file [File]-[New].  
Save the file and call it **ledpattern.py**

Enter the code below.

|  |
| --- |
| #!/usr/bin/env python3  # LEDs pattern  # TM1638  import TM1638  import time  # These are the pins the display is connected to. Adjust accordingly.  # In addition to these you need to connect to 5V and ground.  DIO = 17  CLK = 21  STB = 22  pause = 0.01  display = TM1638.TM1638(DIO, CLK, STB)  display.enable(1)  display.set\_led(0, True)  for x in range(5):  ····for i in range(7):  ········display.set\_led(i,True)  ········time.sleep(pause)  ········display.set\_led(i,False)  ········time.sleep(pause)  ····for i in range (7,0,-1):  ········display.set\_led(i,True)  ········time.sleep(pause)  ········display.set\_led(i,False)  ········time.sleep(pause) |

Run the code and enjoy the pattern.