

Using Raspberry Pi to control LED

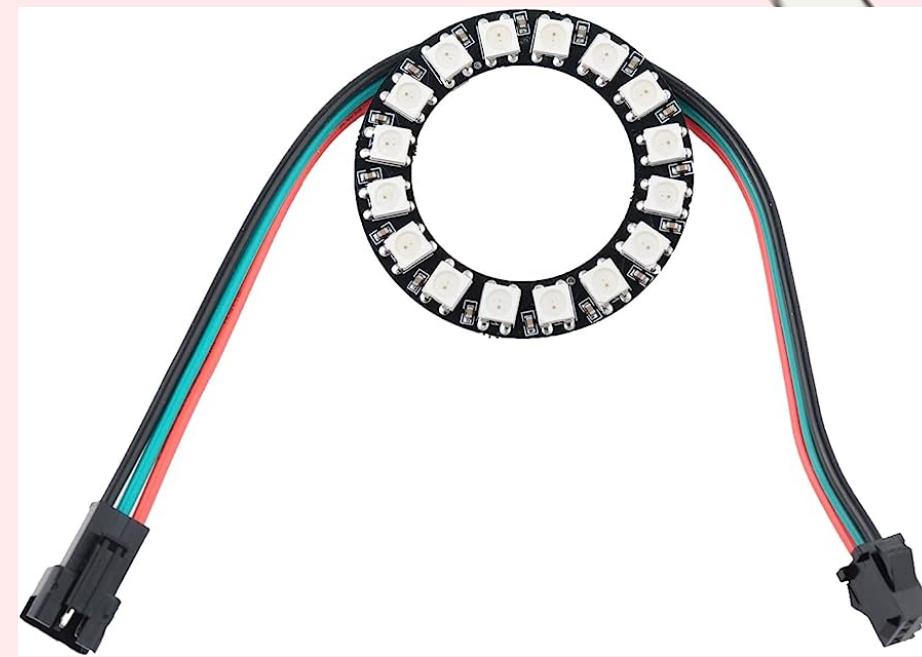
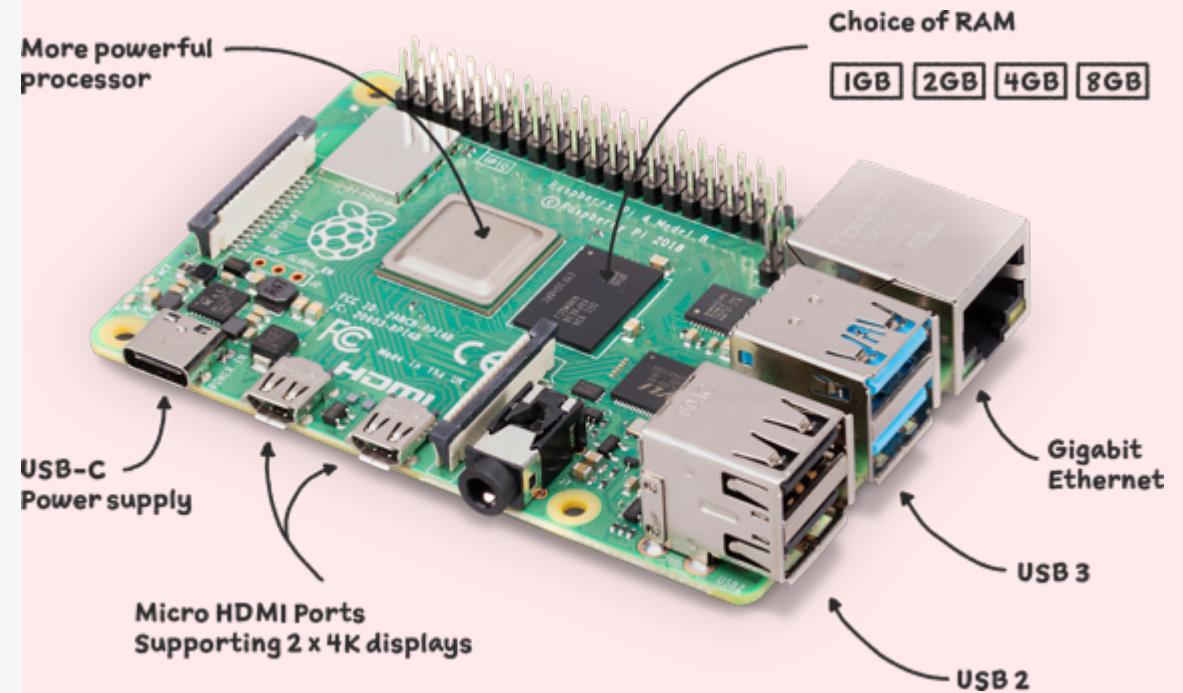
Danyan Liao

Project Abstract:

- Controlling a 45 RGB LED with a Raspberry Pi by using the C language knowledge learned.
- Project expected outcome: To control the even lights on and off or either odd lights on and off using a tree node data structure.

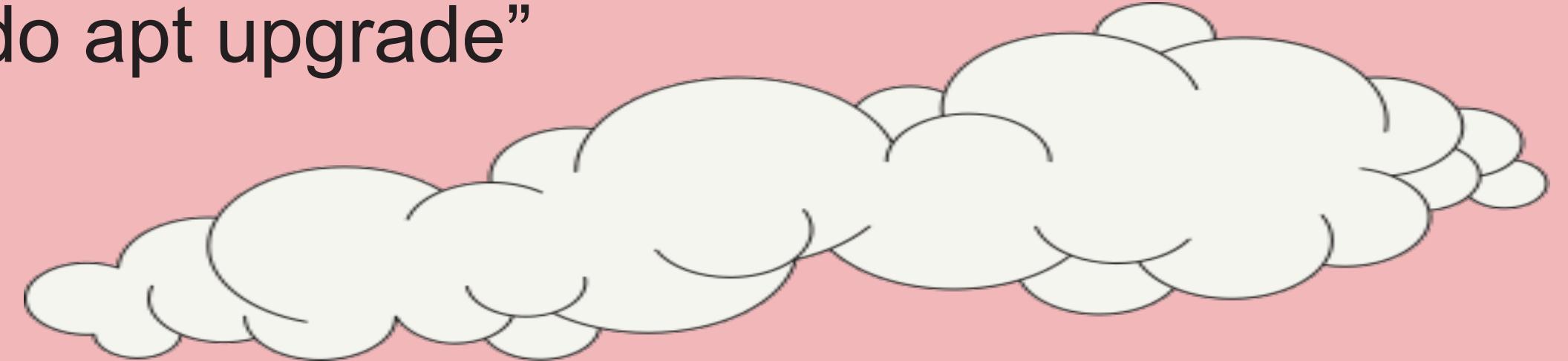
Required parts.

- Raspberry Pi : model 4
- 45 RGB LED Ring (WS2812)

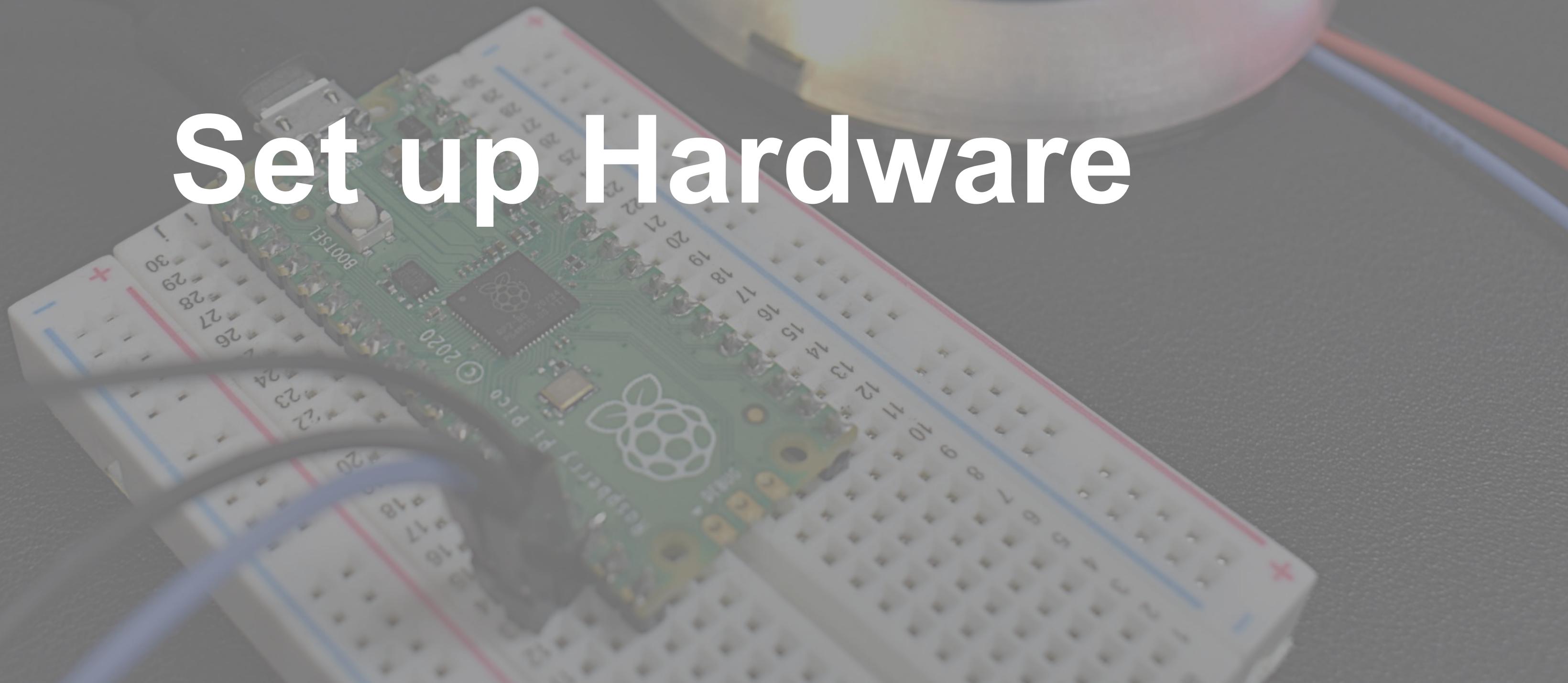




Steps to update Raspberry Pi:

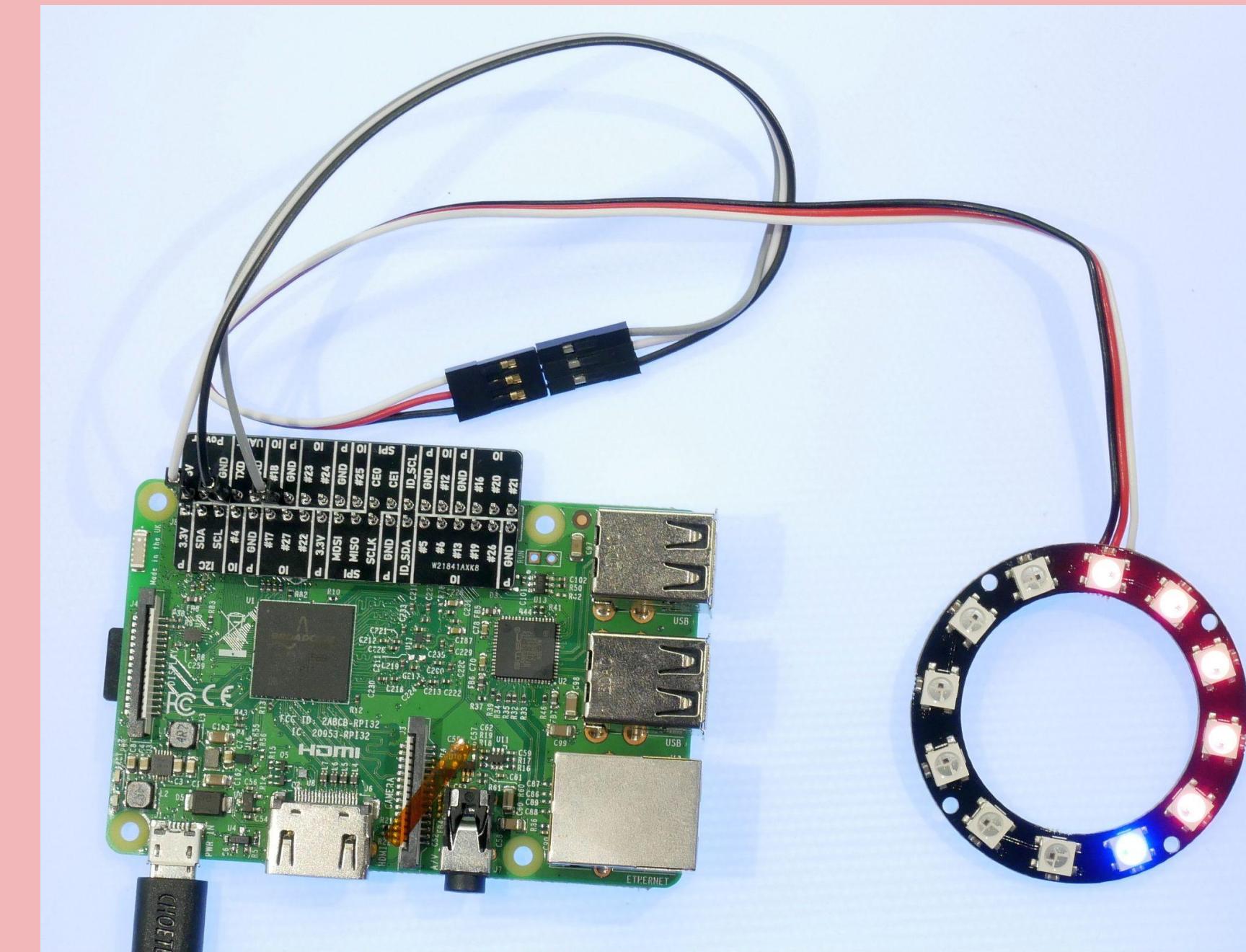
- Connect the raspberry pi to a computer or access to terminal
 - Open a terminal
 - Update the package list
 - with the command “ sudo apt update”
 - Upgrade the installed packages
 - with the command “ sudo apt upgrade”
- 

Set up Hardware

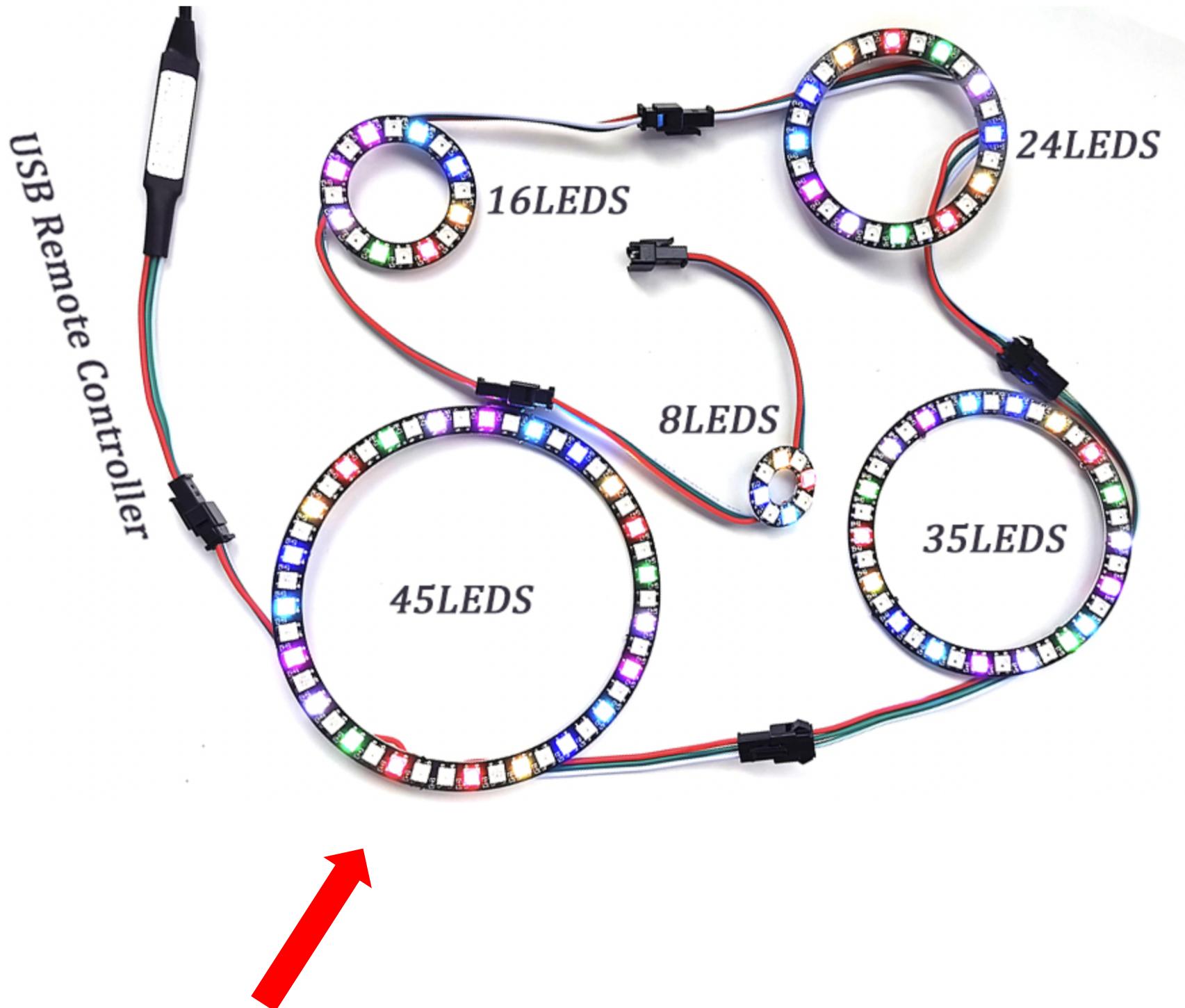


Three Steps to connect LED Ring to Raspberry Pi

- (1)Connect the **5V** pin of the WS2812 ring to the **5V** pin on the Raspberry Pi
- (2)Connect the **GND** pin of the WS2812 to any **GND** pin on the Raspberry Pi
- (3)Connect the **DIN** pin of the WS2812 to a **GPIO pin** on the Raspberry Pi



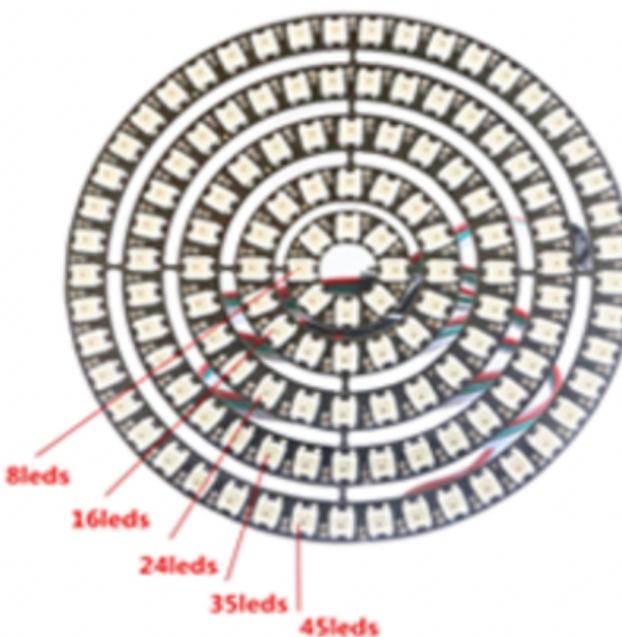
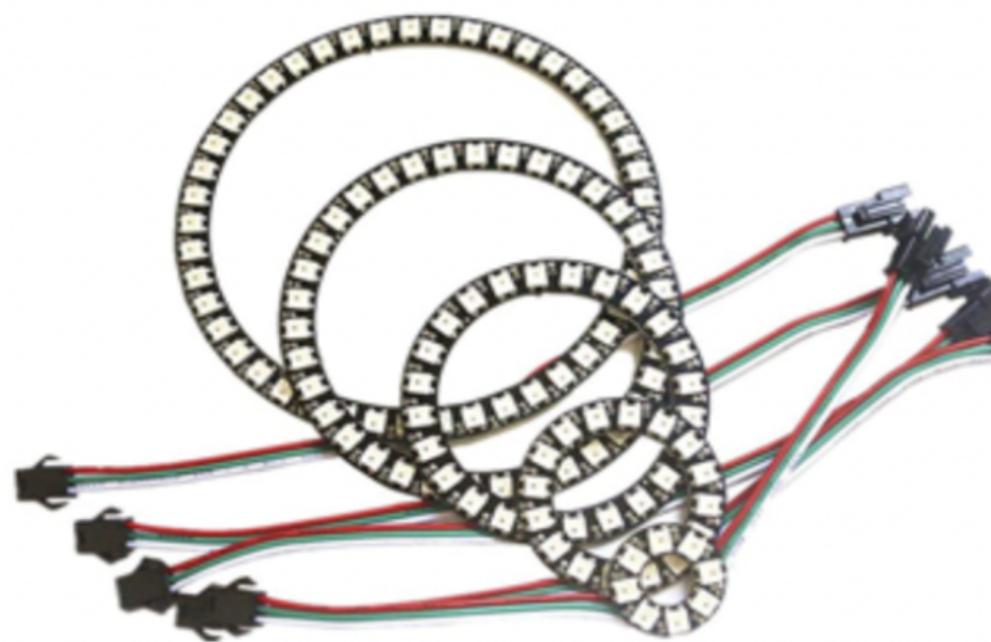
45 RGB LED Ring - Descriptions



We're using this 45 LEDS

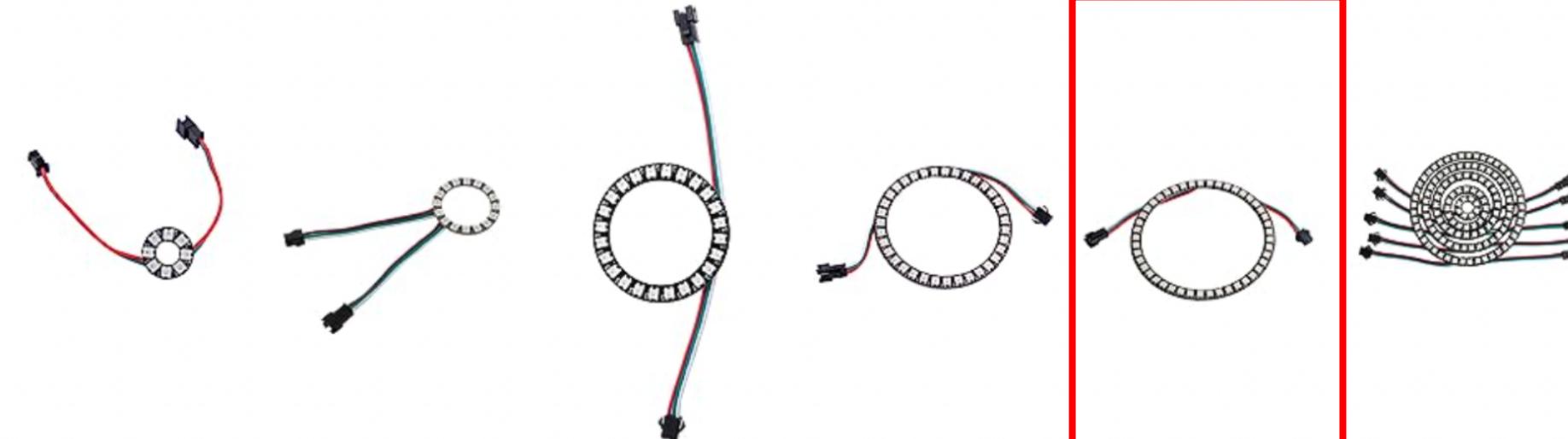
Brand	DIYmall
Manufacturer	DIYmall
Item Weight	0.705 ounces
Product Dimensions	3.35 x 1.89 x 0.59 inches
Assembled Height	1.5 centimeters
Assembled Length	8.5 centimeters
Assembled Width	4.8 centimeters
Color	Black
Shape	Round
Special Features	Dimmable
Batteries Included?	No
Batteries Required?	No

45 RGB LED Ring - Descriptions



LED Numbers	Out Diameter	Inner Diameter	PCB Width	Maximum power
8LED	27mm	12mm	9mm	≤2W/PCS
16LED	48mm	31mm	9mm	≤5W/PCS
24LED	72mm	54mm	9mm	≤6WPCS
35LED	96mm	78mm	9mm	≤9WPCS
45LED	120mm	102mm	9mm	≤11WPCS

45 RGB LED Ring - Descriptions



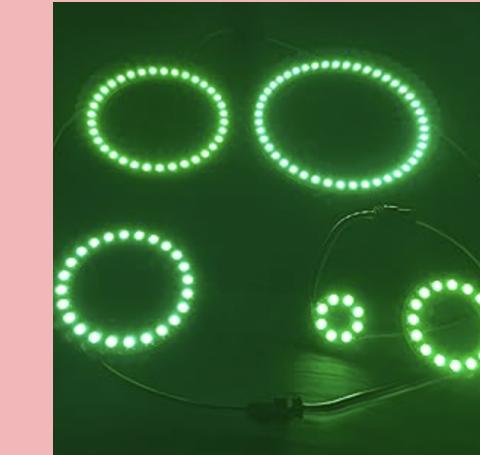
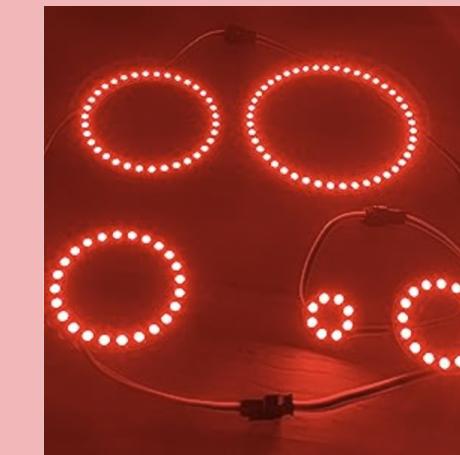
	8 RGB LED	16 RGB LED	24 RGB LED	35 RGB LED	45 RGB LED	128 RGB LED
Voltage	DC5V	DC5V	DC5V	DC5V	DC5V	DC5V
LED Chip	WS2812B	WS2812B	WS2812B	WS2812B	WS2812B	WS2812B
Each LED Current(mA)	18	18	18	18	18	18
Red LED Wavelength(nm)/Luminous(mcd)	620-625/700-1000	620-625/700-1000	620-625/700-1000	620-625/700-1000	620-625/700-1000	620-625/700-1000
Green LED Wavelength(nm)/Luminous(mcd)	522.5-525/1500-2200	522.5-525/1500-2200	522.5-525/1500-2200	522.5-525/1500-2200	522.5-525/1500-2200	522.5-525/1500-2200
Blue LED Wavelength(nm)/Luminous(mcd)	467.5-470/700-1000	467.5-470/700-1000	467.5-470/700-1000	467.5-470/700-1000	467.5-470/700-1000	467.5-470/700-1000
Emit Colors Numbers	16777216	16777216	16777216	16777216	16777216	16777216

Availability: 45-LED rings are commercially available and affordable in the desired size, color, or configuration.

Compatibility: A 45-LED ring is compatible with the software or libraries being used in this project.

Power Consumption: 45 LEDs is a suitable number that doesn't draw too much power from the device.

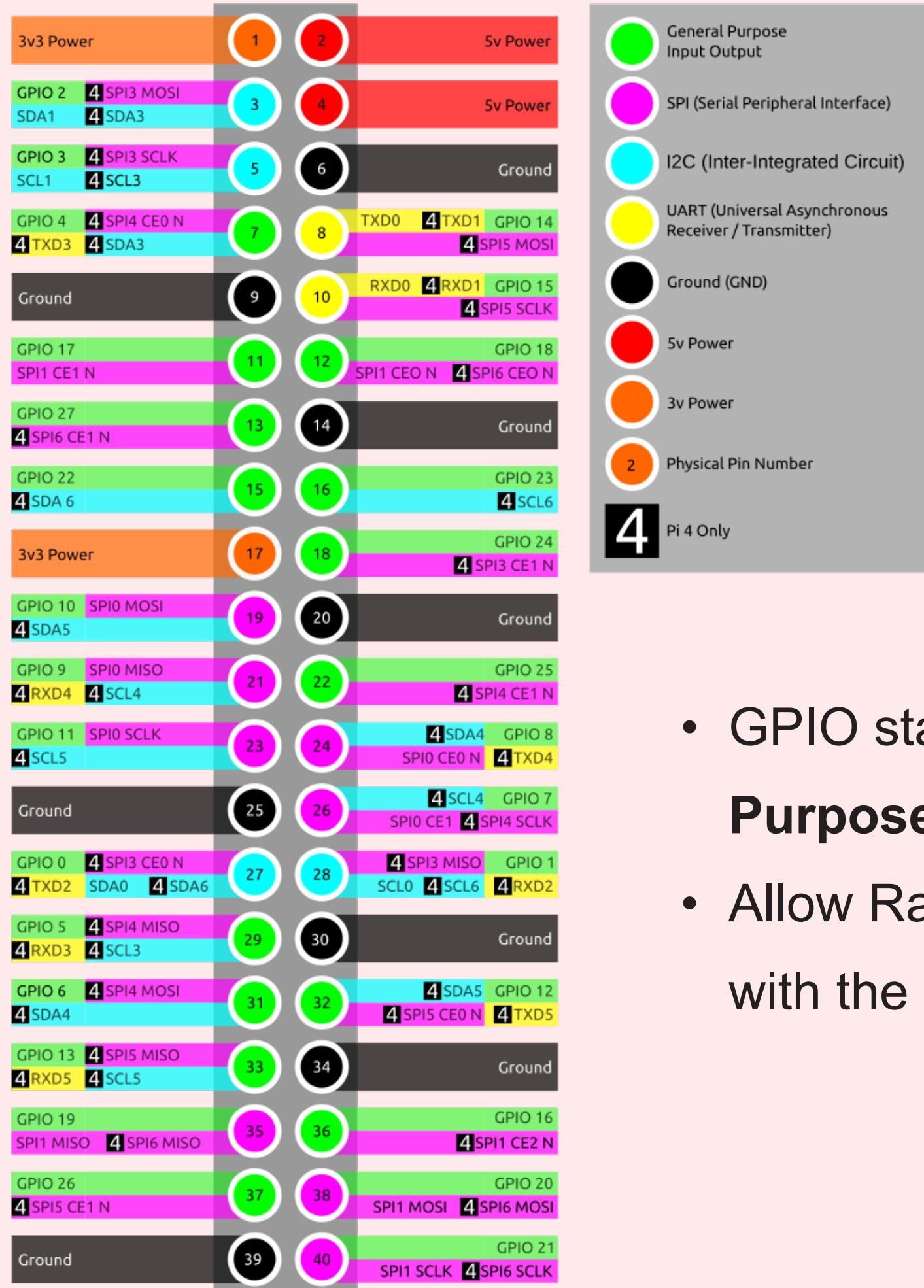
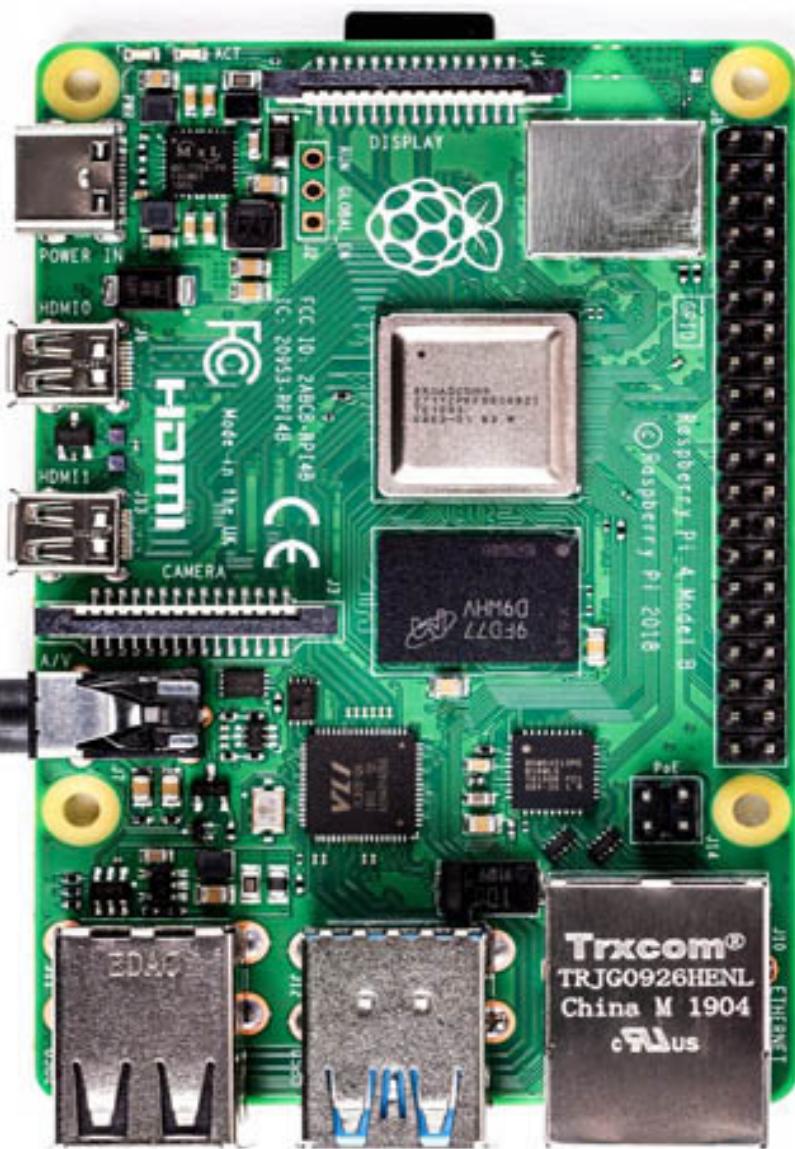
45 RGB LED Ring - Full Color



Full color refers to the ability to reproduce the entire range of colors that can be perceived by the human eye.

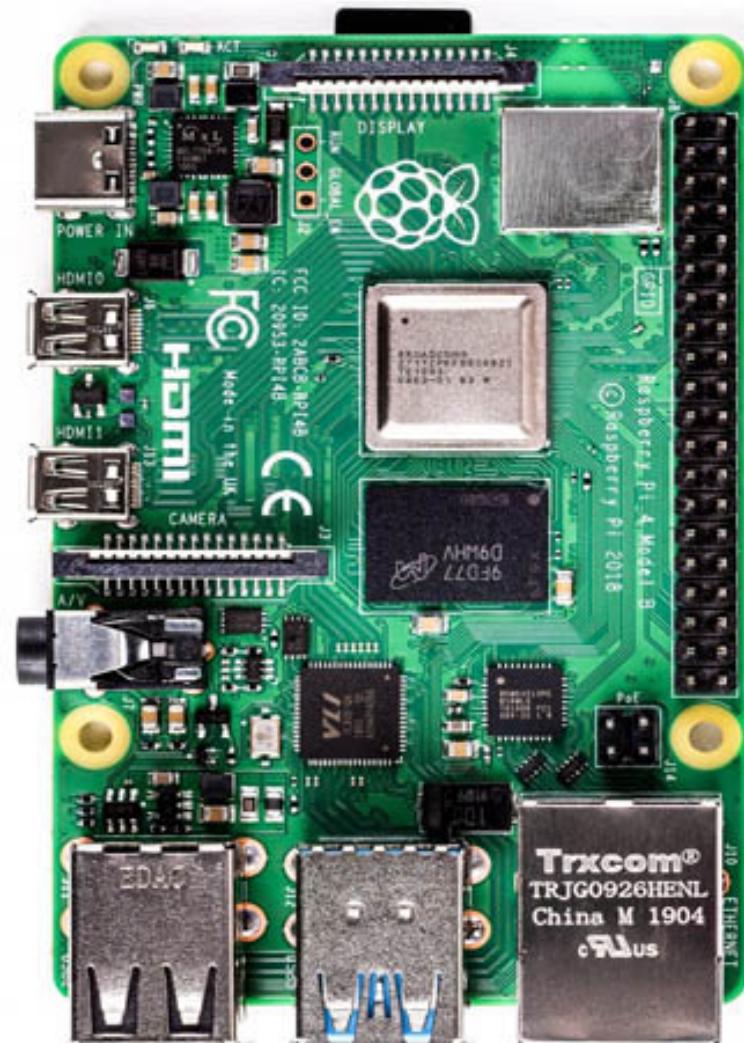
In digital devices like LED rings, full color is typically achieved through the combination of **Red, Green, and Blue (RGB) light**, allowing for the display of thousands or even millions of distinct colors.

What is a GPIO Pin?



- GPIO stands for **General Purpose Input/Output**
- Allow Raspberry Pi to interact with the outside world

What is a GND Pin?

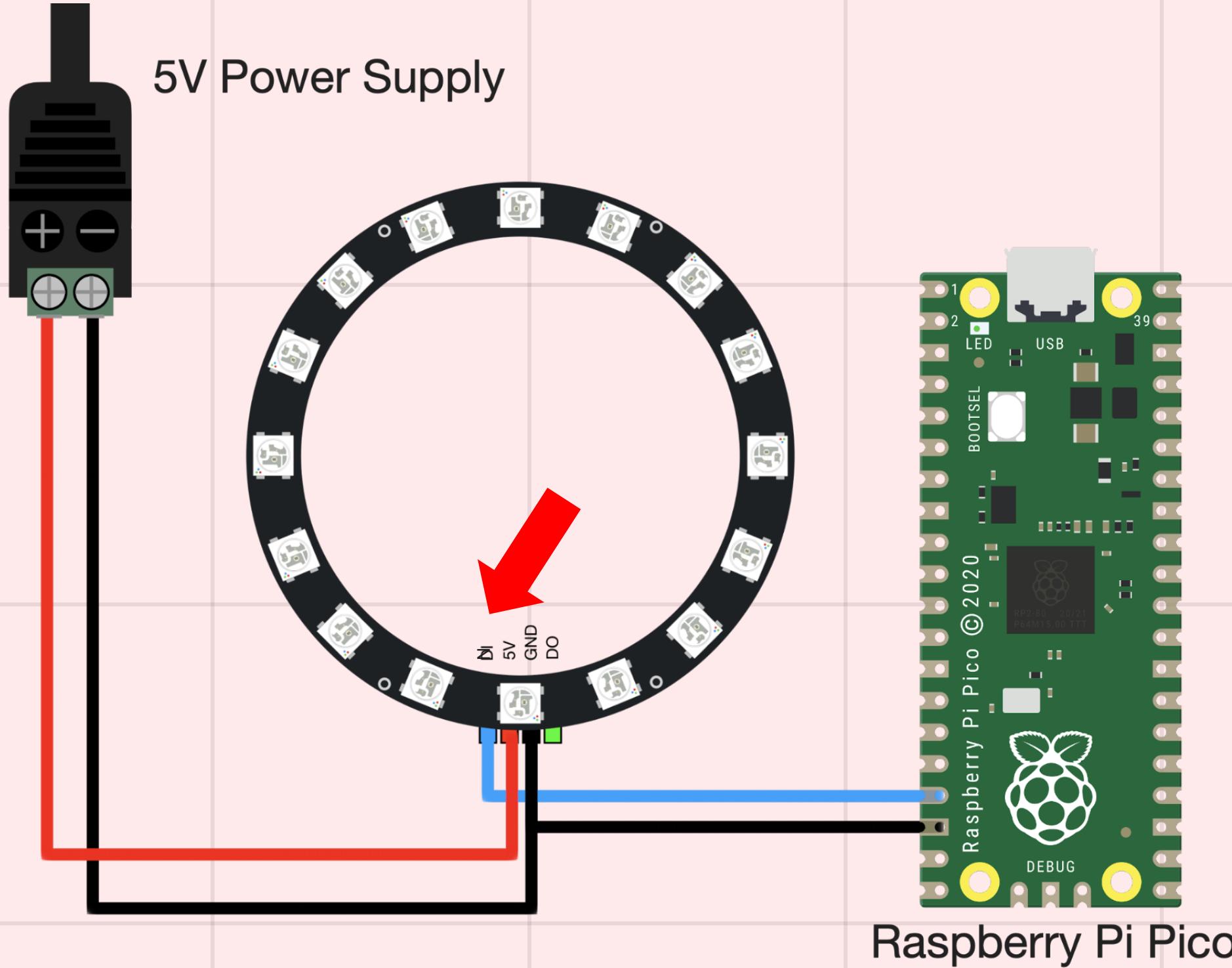


Physical Pins					
Function	BCM	pin#	pin#	BCM	Function
3.3 Volts		1	2		5 Volts
GPIO/SDA1 (I2C)	2	3	4		5 Volts
GPIO/SCL1 (I2C)	3	5	6		GND
GPIO/GCLK	4	7	8	14	TX UART/GPIO
GND		9	10	15	RX UART/GPIO
GPIO	17	11	12	18	GPIO
GPIO	27	13	14		GND
GPIO	22	15	16	23	GPIO
3.3 Volts		17	18	24	GPIO
MOSI (SPI)	10	19	20		GND
MISO(SPI)	9	21	22	25	GPIO
SCLK(SPI)	11	23	24	8	CEO_N (SPI)
GND		25	26	7	CE1_N (SPI)
RESERVED		27	28		RESERVED
GPIO	5	29	30		GND
GPIO	6	31	32	12	GPIO
GPIO	13	33	34		GND
GPIO	19	35	36	16	GPIO
GPIO	26	37	38	20	GPIO
GND		39	40	21	GPIO

Raspberry Pi GPIO pin layout

- GND stands for **Ground**
- A common return path for electric current
- Prevent electrical damage
- Give a common reference point for all components in the circuit

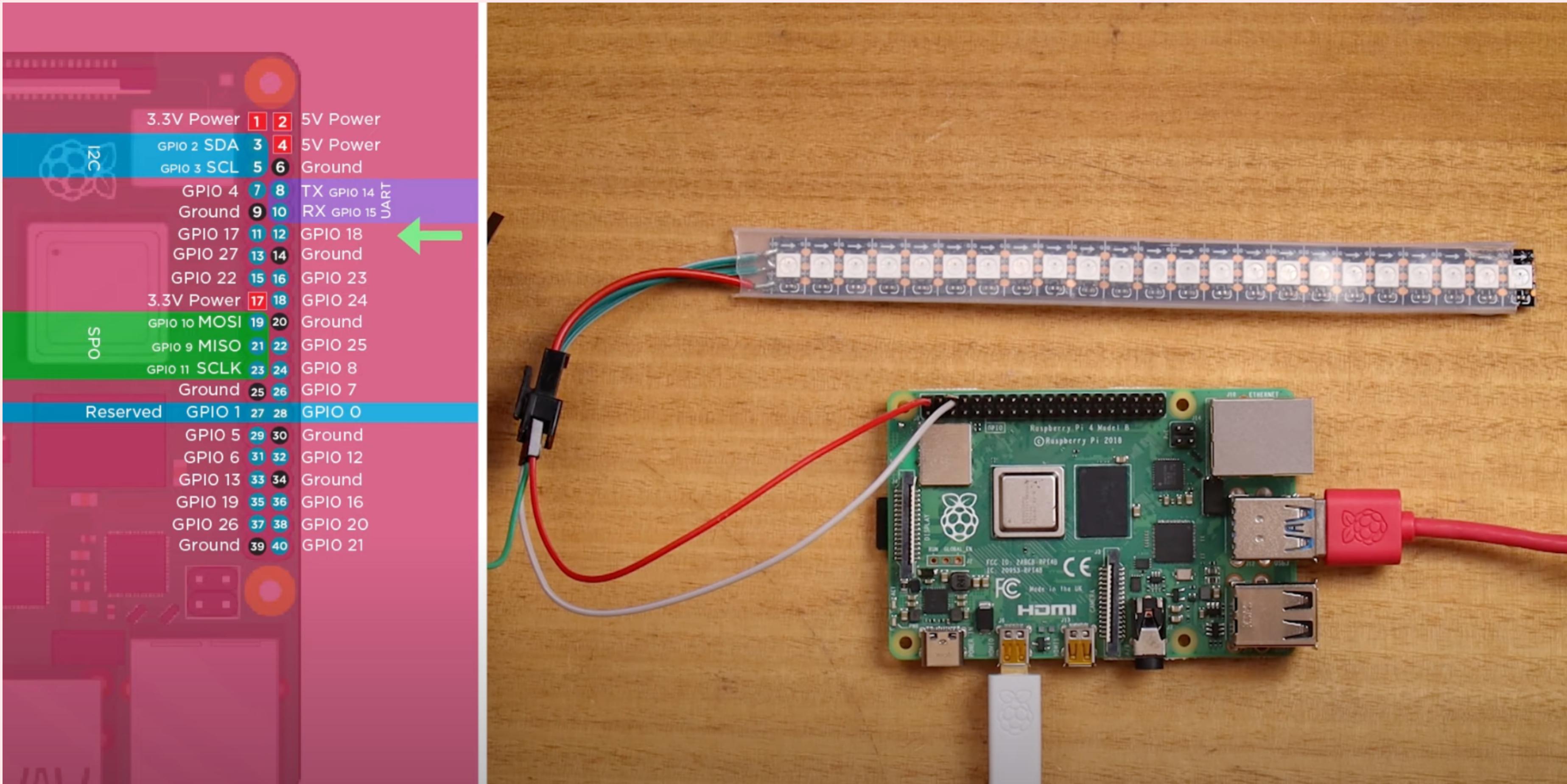
What is a DIN Pin (on the LED Ring)?



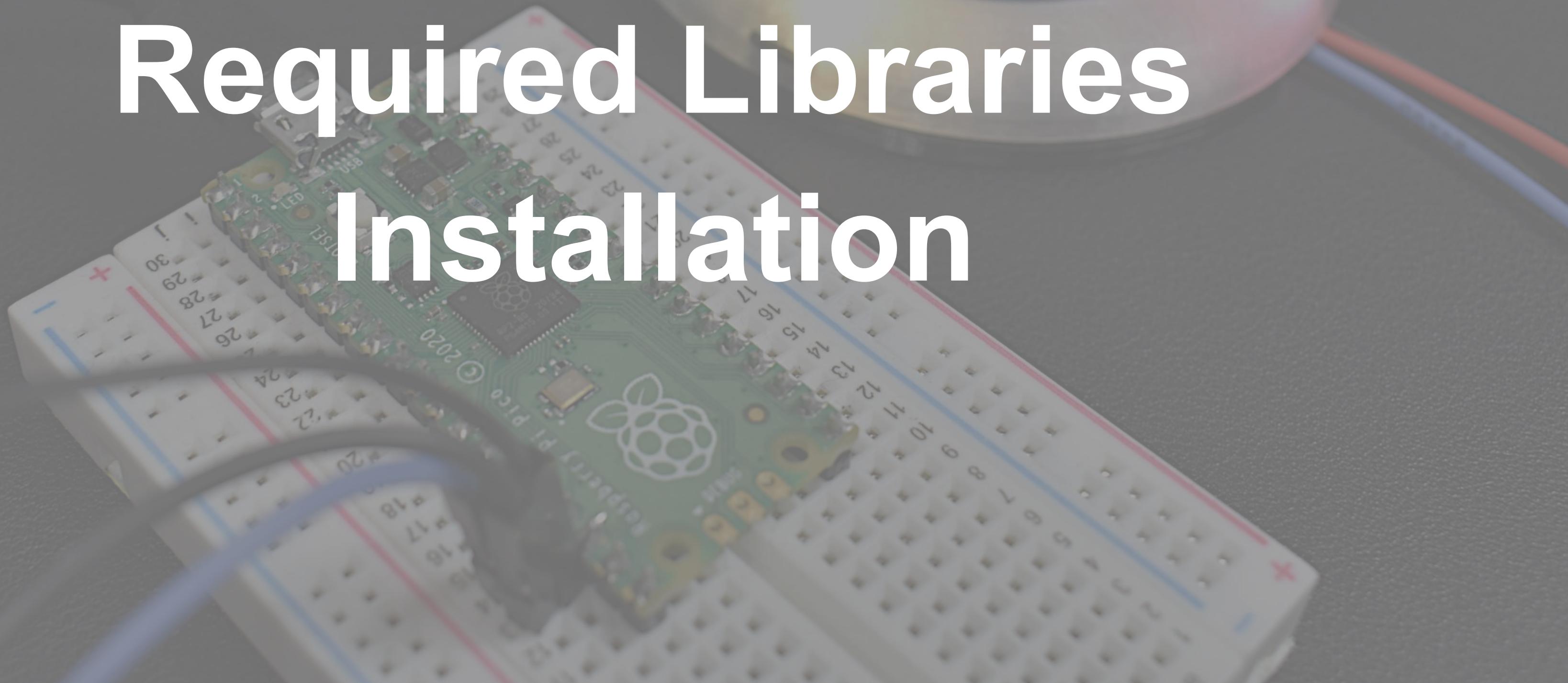
- DIN stands for **Data IN**
- DIN pin is where it **receives data**
- Color and brightness information for each LED

Hardware Set up Instruction

<https://www.youtube.com/watch?v=aNlaj1r7NKo>



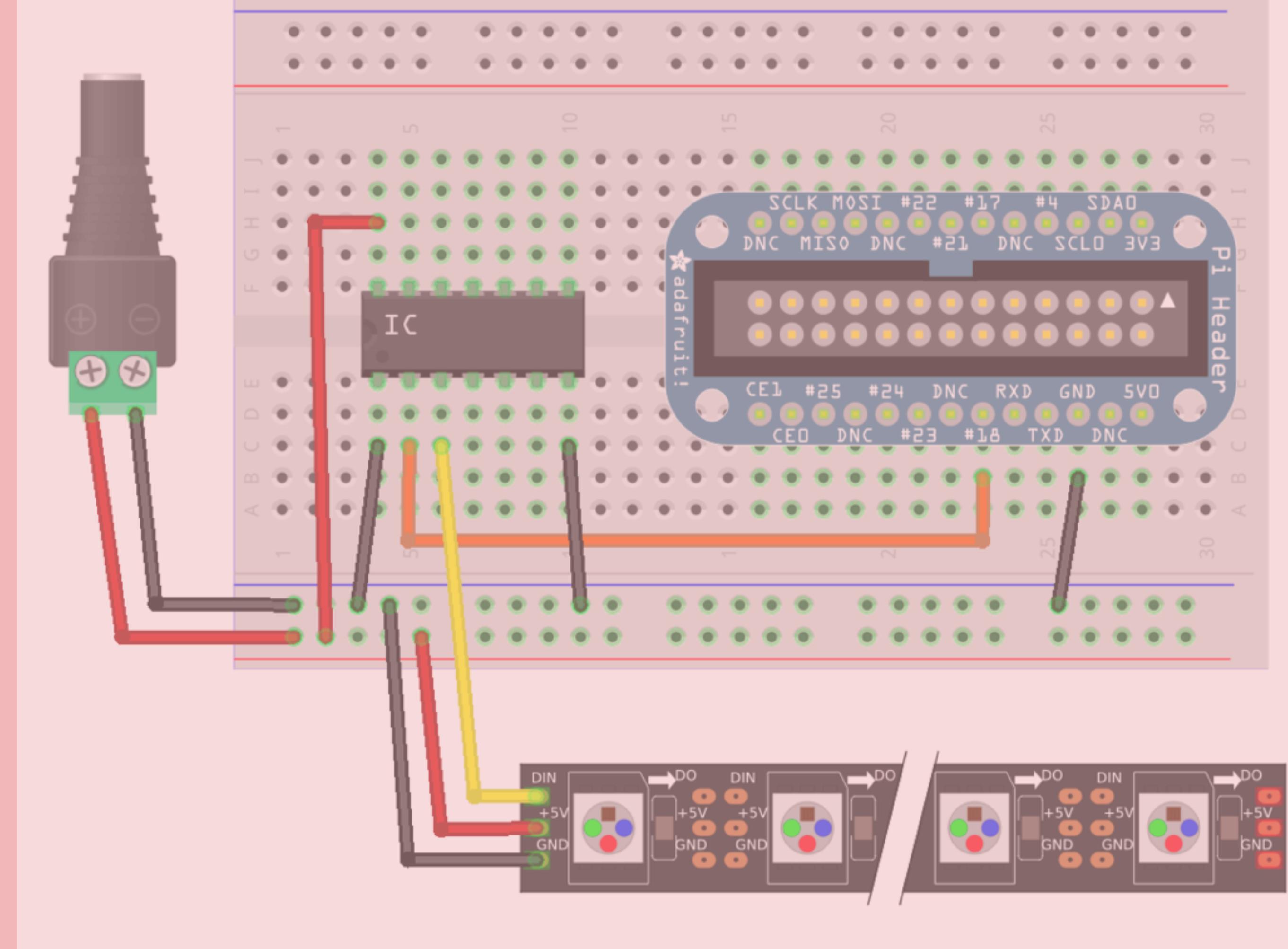
Required Libraries Installation



Main Library

rpi_ws281x

- git clone https://github.com/jgarff/rpi_ws281x.git
- cd rpi_ws281x
- scons
- sudo scons install



Data Structures and Functions in Main Library

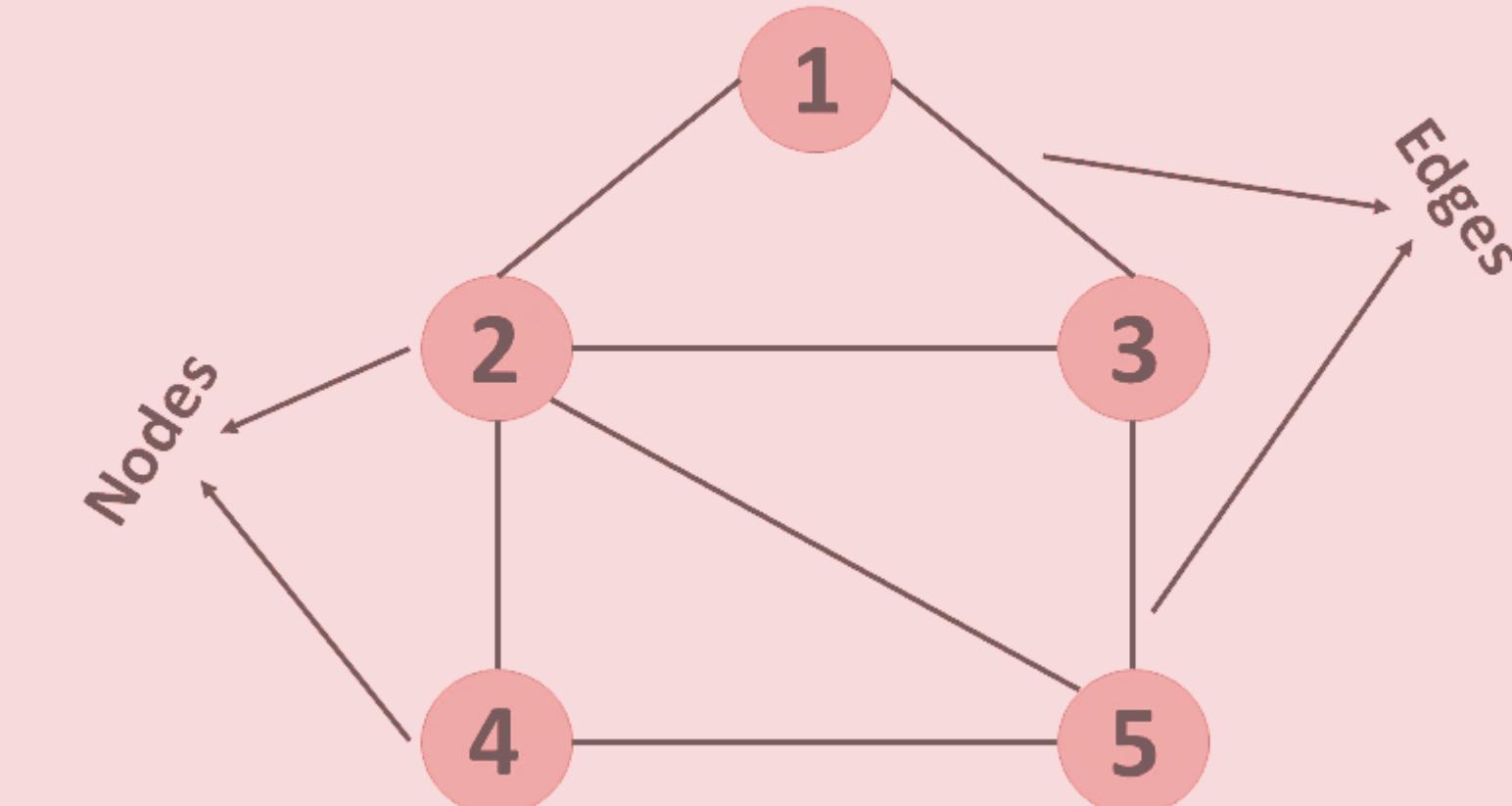
rpi_ws281x

Data Structures

- initializing the parameters of the LED strip.
- setting the parameters of each channel of the LED strip.

Functions

- ws2811_init()
- ws2811_render()
- ws2811_fini()



Other Libraries

WiringPi

- ‘wringPilSR()’ function
 - example in C: int wiringPilSR(int pin, int edgeType, void (*function)(void))

‘pin’: The GPIO pin number to set up the ISR for.

‘edgeType’: The type of edge triggering the interrupt.

‘function’: A pointer to the function that will be called when the interrupt is triggered.

- Install dependencies in terminal:

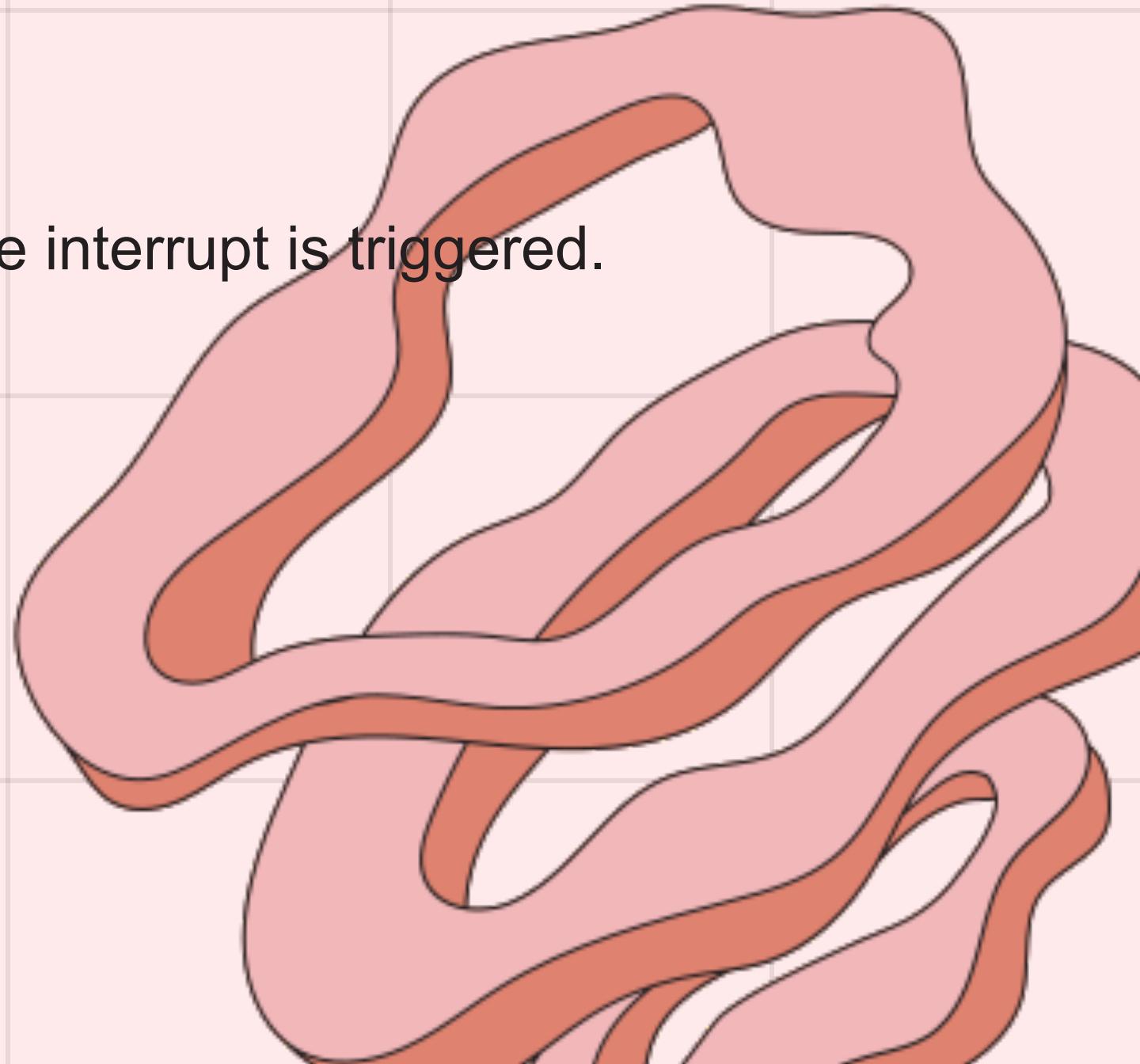
sudo apt-get update

sudo apt-get install git-core

- git clone <https://github.com/WiringPi/WiringPi.git>

- cd WiringPi

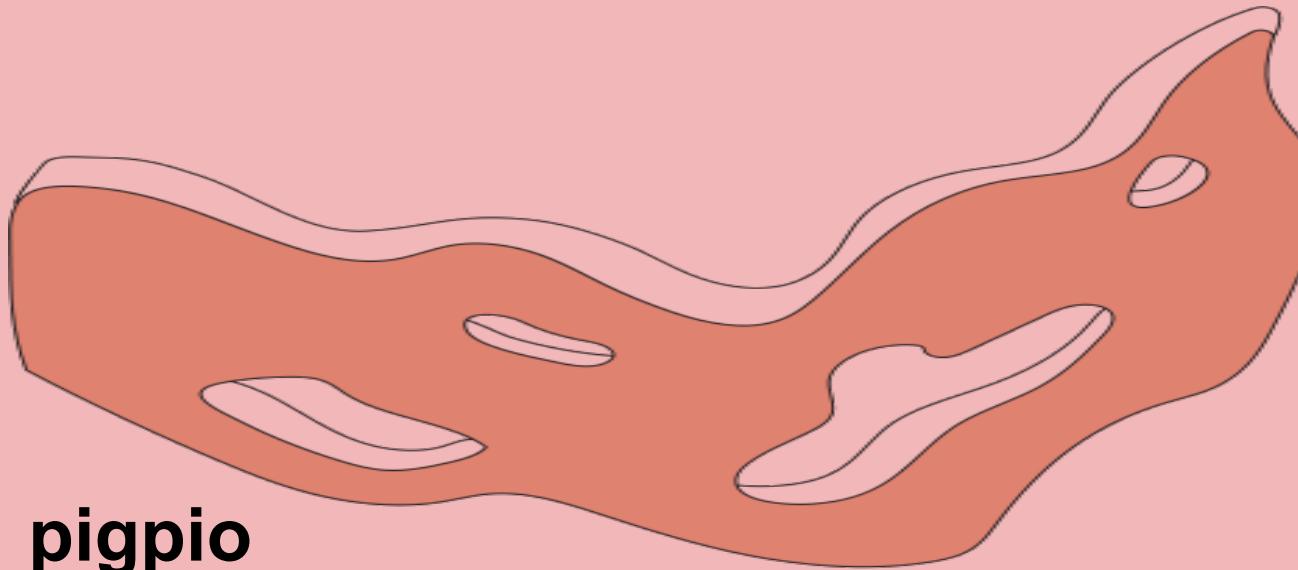
- ./build



Main reasons for using WiringPi

- **Simplicity and Abstraction:** abstracts the low-level hardware operations, making it easier for developers to control the GPIO pins without having to deal with the intricacies of direct hardware manipulation.
- **Consistency:** provides a consistent API across different models of Raspberry Pi, allowing developers to write code that works across multiple hardware revisions.
- **Community Support:** gained popularity and has an active community, which means that you can find plenty of resources, tutorials, and examples to help you get started and troubleshoot any issues.
- **Flexibility:** supports various programming languages, including C, C++, and Python, making it accessible to a wide range of developers with different language preferences.
- **Compatibility:** provides functions for working with I2C, SPI, and serial communication, in addition to GPIO, making it suitable for projects that require multiple types of communication protocols.
- **Speed and Performance:** it is optimized for performance, allowing developers to achieve efficient GPIO operations even in time-critical applications.

Other Libraries

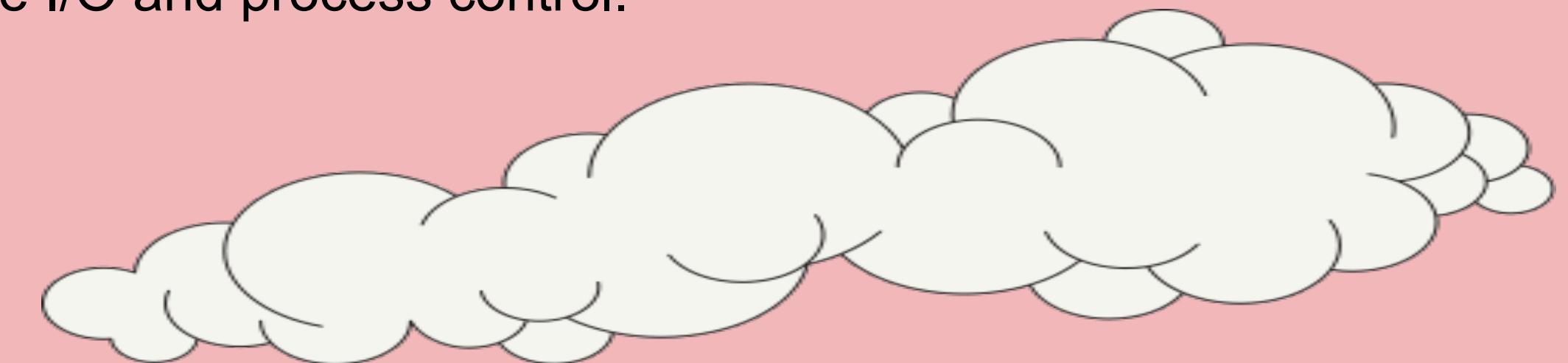


pigpio

- It is an alternative to WiringPi and allows access to Raspberry Pi GPIO pins from user space.
- Offers similar functionality as WiringPi and is chosen by some users due to its advanced features and capabilities.

unistd.h

- unistd.h is a C standard library header file.
- Provides access to POSIX operating system APIs, enabling interaction with functions like read, write, and sleep for tasks such as file I/O and process control.





**Write C Program to
control odd lights on or
even lights on**

Firstly, we need to import the libraries:

```
#include <ws2811.h>  
#include <unistd.h>
```

```
typedef struct TreeNode
TreeNode* createNode(int led_index)
void traverse Tree
void freeTree(TreeNode root)
```

