

<https://www.halvorsen.blog>



Raspberry Pi with MATLAB

Hans-Petter Halvorsen

Contents

- Raspberry Pi
- MATLAB
- MATLAB Support Package for Raspberry Pi
- GPIO
- Examples
 - LED
 - PWM
 - Push Button
 - Camera

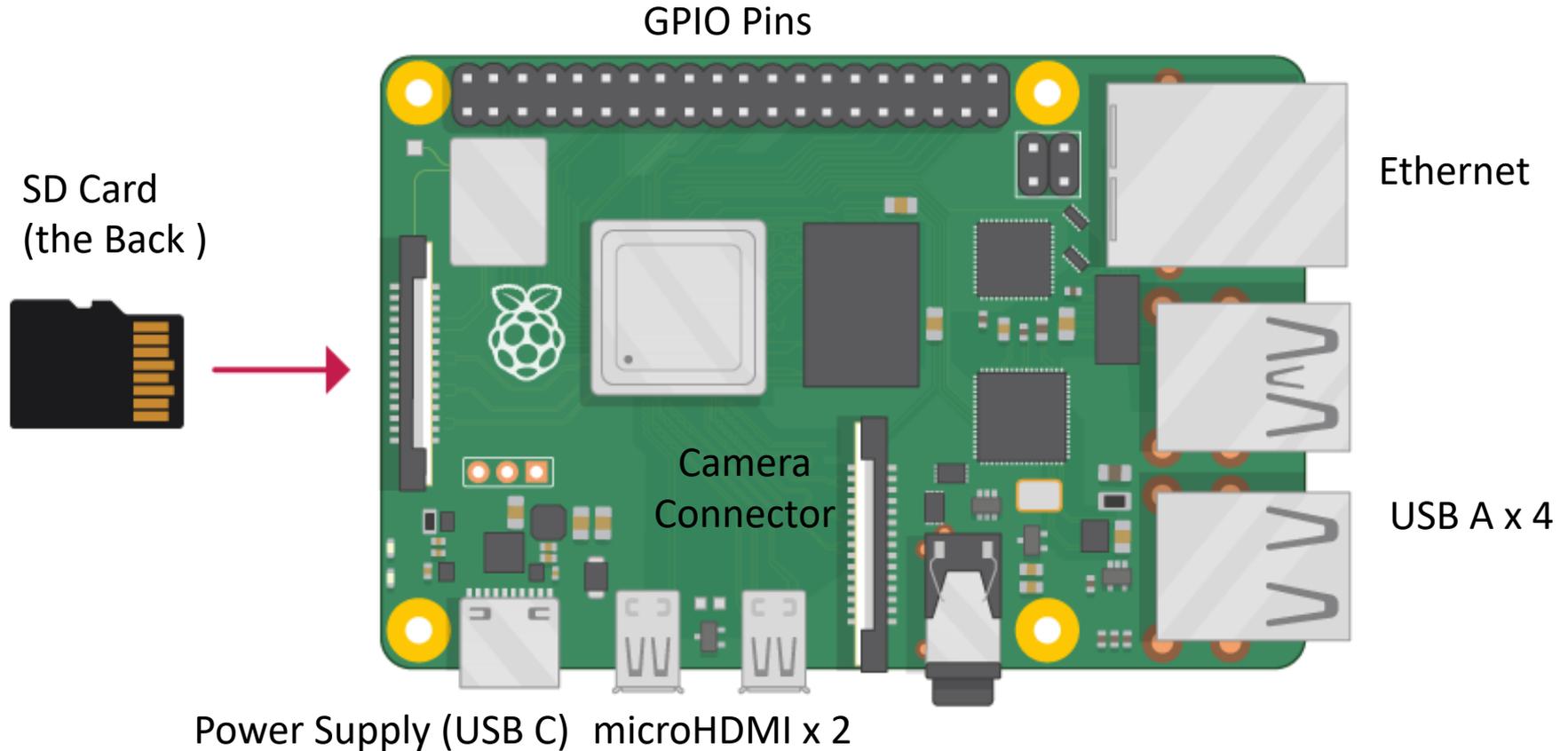


Raspberry Pi

Hans-Petter Halvorsen

[Table of Contents](#)

Raspberry Pi



Raspberry Pi

- The Raspberry Pi is a small computer that can do lots of things
- It has a small footprint (about 9x6cm) and it is cheap (\$35+)
- You plug it into a monitor and attach a keyboard and mouse
- It has so-called GPIO pins (General Purpose Input/Output) for connection sensors and other electronic components like LEDs, etc.
- Raspberry Pi is as well suited for prototyping, datalogging and different electronics projects, a media center, etc.
- It can be used to learn programming, it and other technical skills, etc.
- RP has limited power (CPU, RAM, etc.) so it cannot normally replace a desktop computer or laptop for ordinary use



MATLAB

Hans-Petter Halvorsen

[Table of Contents](#)

MATLAB

- MATLAB is a tool for technical computing, computation and visualization in an integrated environment.
- MATLAB is an abbreviation for MATrix LABoratory
- It is well suited for matrix manipulation and problem solving related to Linear Algebra, Modelling, Simulation and Control applications, etc.
- MATLAB is popular in Universities, Teaching and Research and Development (R&D)
- www.mathworks.com

MATLAB

The screenshot displays the MATLAB R2020b environment. The main window is the Editor, showing a script named `mass_spring_damper_script.m`. The script contains the following code:

```
1 %Script of mass-spring-damper simulator.
2 %Hans-Petter Halvorsen. 20.11.2009
3
4 %Modell Parameters
5 x_init=4; %[m]. Initial position.
6 dxdt_init=0; %[m/s]. Initial Speed.
7 m=20; %[kg]
8 c=4; %[N/(m/s)]
9 k=2; %[N/m]
10 t_step_F=50; %[s]
11 F_0=0; %[N]
12 F_1=4; %[N]
13
14 %Simulator Settings
15 t_stop=100; %[s]
16 T_s=t_stop/1000; %[s]
17 options=simset('solver', 'ode5', 'fixedstep', T_s);
18
19 %Starting simulation
20 sim('mass_spring_damper', t_stop, options);
```

The Command Window shows the execution of the script:

```
>> mass_spring_damper_script
fx >>
```

The Workspace window displays the following variables and their values:

Name	Value
c	4
dxdt_init	0
F_1	4
F_0	0
k	2
m	20
options	1x1 struct
T_s	0.1000
t_step_F	50
t_stop	100
tout	1000x1 do...
x_init	4

The interface includes a menu bar (HOME, PLOTS, APPS, EDITOR, PUBLISH, VIEW), a toolbar with icons for file operations, editing, and running, and a status bar at the bottom showing the encoding (UTF-8) and cursor position (Ln 1, Col 1).



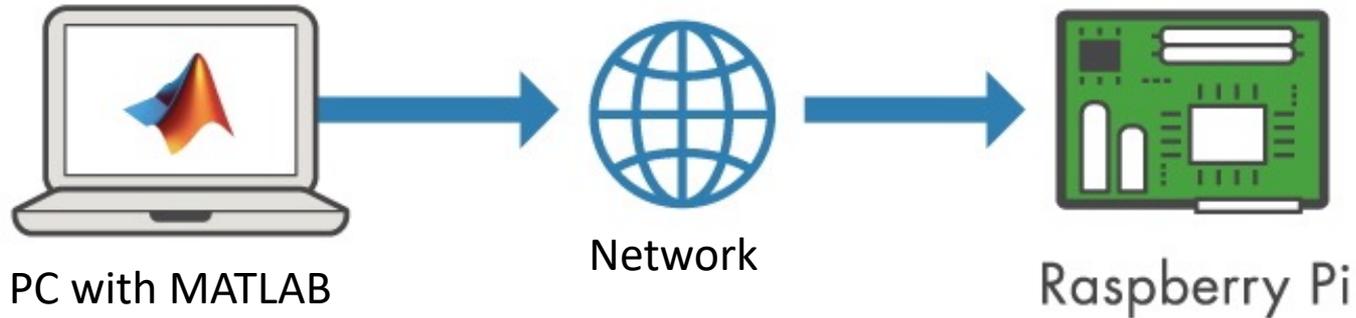
MATLAB Support Package for Raspberry Pi

Hans-Petter Halvorsen

[Table of Contents](#)

Raspberry Pi + MATLAB

MATLAB Support Package for Raspberry Pi

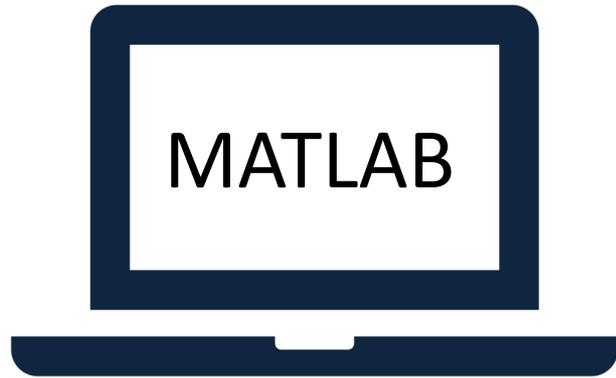


With **MATLAB Support Package for Raspberry Pi**, the Raspberry Pi is connected to a computer running MATLAB. Processing is done on the computer with MATLAB.

<https://mathworks.com/hardware-support/raspberry-pi-matlab.html>

Raspberry Pi + MATLAB

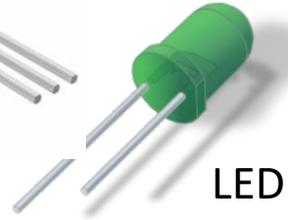
We can read data from Sensors, Cameras, control LEDs, etc. from our MATLAB environment on the Desktop Computer



PC



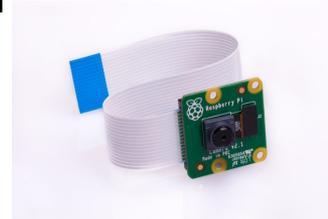
Sensors



LED



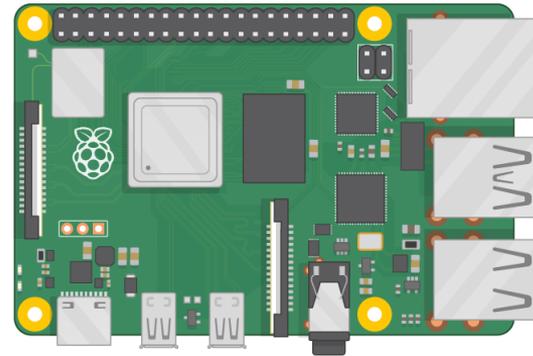
Push Button



Camera



GPIO Pins



Raspberry Pi

MATLAB Support Package for Raspberry Pi

The image shows the MATLAB R2020b interface. The top toolbar includes buttons for 'Add-Ons' and 'Help', which are circled in red. The 'Add-On Explorer' window is open, displaying search results for 'Raspberry Pi'. The search bar at the top of the Add-On Explorer is also circled in red. The first result is 'MATLAB Support Package for Raspberry Pi Hardware' by MathWorks, which is highlighted with a red box. Below it is 'Simulink Support Package for Raspberry Pi Hardware' by MathWorks, and at the bottom is 'Raspberry Pi Hardware Resource Manager' by MathWorks. The Command Window shows the prompt 'fi >> |'.

Current Folder: C:\Users\hansha\Documents\MATLAB

Command Window: fi >> |

Add-On Explorer Search: Raspberry Pi

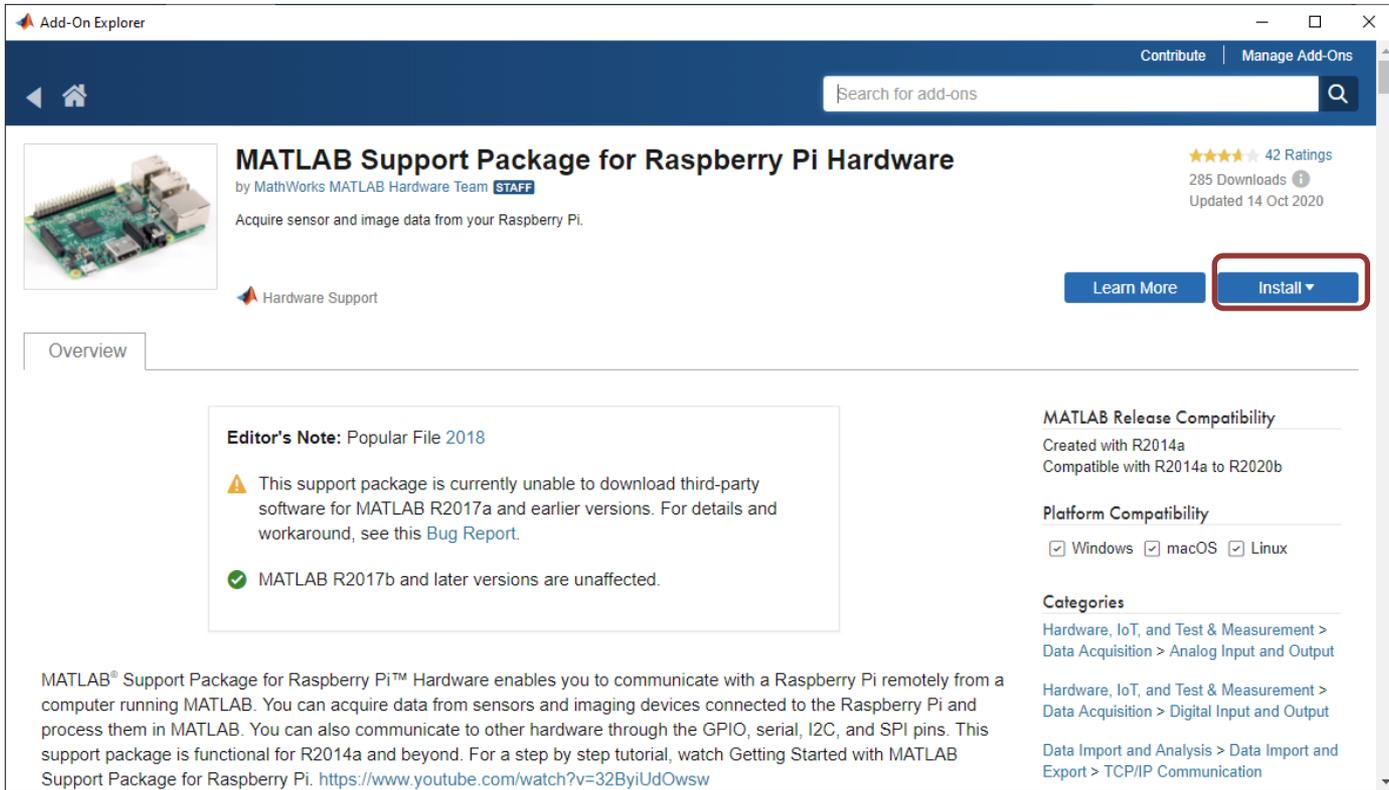
MATLAB Support Package for Raspberry Pi Hardware by MathWorks
MATLAB Hardware Team **STAFF**
285 Downloads
Updated 14 Oct 2020
Acquire sensor and image data from your Raspberry Pi.
MATLAB® Support Package for Raspberry Pi™ Hardware enables you to communicate with a Raspberry Pi remotely from a computer running MATLAB. You can acquire data from sensors and imaging devices.
Hardware Support

Simulink Support Package for Raspberry Pi Hardware by MathWorks
Simulink Team **STAFF**
185 Downloads
Updated 14 Oct 2020
Run models on Raspberry Pi.
Simulink® Support Package for Raspberry Pi™ Hardware enables you to create and run Simulink models on Raspberry Pi hardware. The support package includes a library of Simulink blocks for configuring
Hardware Support

Raspberry Pi Hardware Resource Manager version 1.0 by MathWorks
Simulink Team **STAFF**
89 Downloads
Updated 29 Jul 2019
Monitor the status of different hardware resources on the raspberry pi

Getting Started with MATLAB Support Package for Raspberry Pi: <https://youtu.be/32ByiUdOsw>

MATLAB Support Package for Raspberry Pi



Add-On Explorer

Contribute | Manage Add-Ons

Search for add-ons

MATLAB Support Package for Raspberry Pi Hardware

by MathWorks MATLAB Hardware Team **STAFF**

★★★★☆ 42 Ratings
285 Downloads ⓘ
Updated 14 Oct 2020

Acquire sensor and image data from your Raspberry Pi.

Hardware Support

Learn More

Install ▾

Overview

Editor's Note: Popular File 2018

⚠ This support package is currently unable to download third-party software for MATLAB R2017a and earlier versions. For details and workaround, see this [Bug Report](#).

✅ MATLAB R2017b and later versions are unaffected.

MATLAB Release Compatibility

Created with	R2014a
Compatible with	R2014a to R2020b

Platform Compatibility

<input checked="" type="checkbox"/> Windows	<input checked="" type="checkbox"/> macOS	<input checked="" type="checkbox"/> Linux
---	---	---

Categories

- Hardware, IoT, and Test & Measurement > Data Acquisition > Analog Input and Output
- Hardware, IoT, and Test & Measurement > Data Acquisition > Digital Input and Output
- Data Import and Analysis > Data Import and Export > TCP/IP Communication

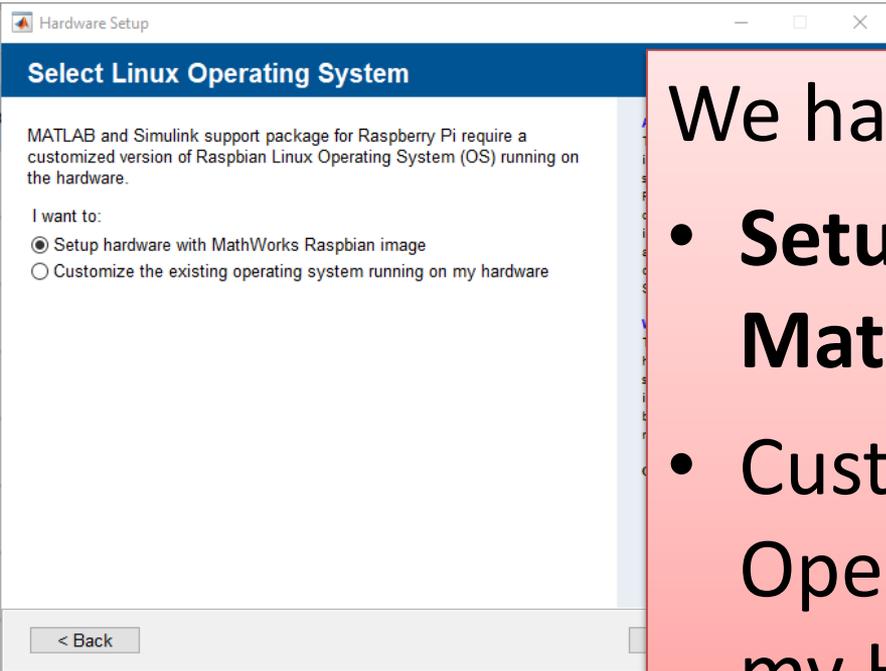
MATLAB® Support Package for Raspberry Pi™ Hardware enables you to communicate with a Raspberry Pi remotely from a computer running MATLAB. You can acquire data from sensors and imaging devices connected to the Raspberry Pi and process them in MATLAB. You can also communicate to other hardware through the GPIO, serial, I2C, and SPI pins. This support package is functional for R2014a and beyond. For a step by step tutorial, watch Getting Started with MATLAB Support Package for Raspberry Pi. <https://www.youtube.com/watch?v=32ByiUdOsw>

Getting Started with MATLAB Support Package for Raspberry Pi: <https://youtu.be/32ByiUdOsw>



Hardware Setup

Hardware Setup



We have 2 options/alternatives:

- **Setup Hardware with MathWorks Raspbian Image**
- Customize the existing Operating System running on my Hardware

Hardware Setup Alternatives

- Setup Hardware with MathWorks Raspbian Image
- Customize the existing Operating System running on my Hardware

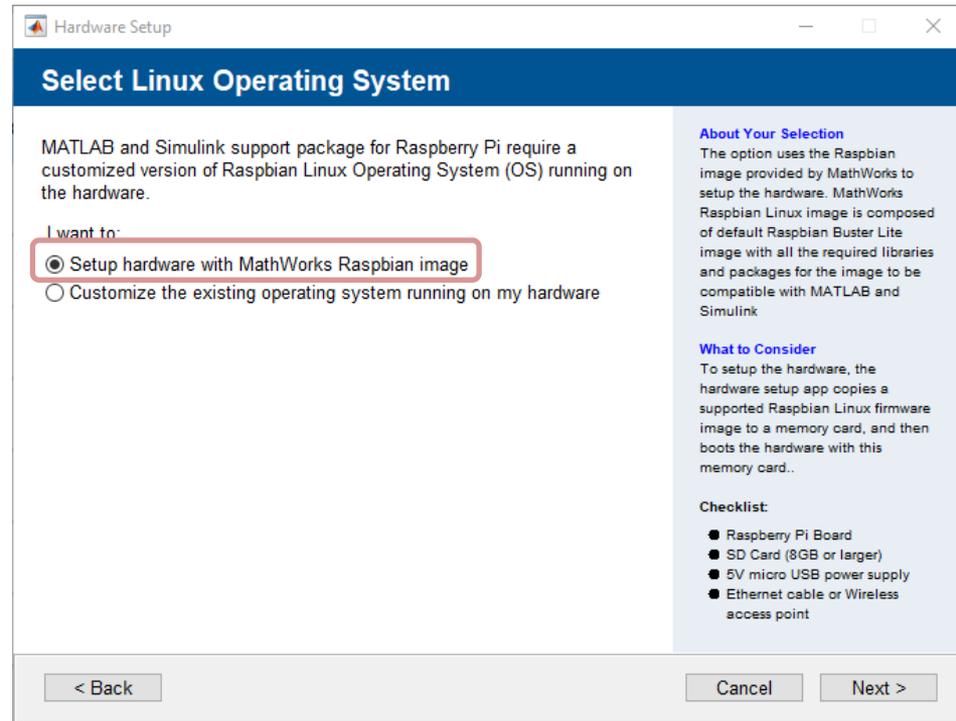
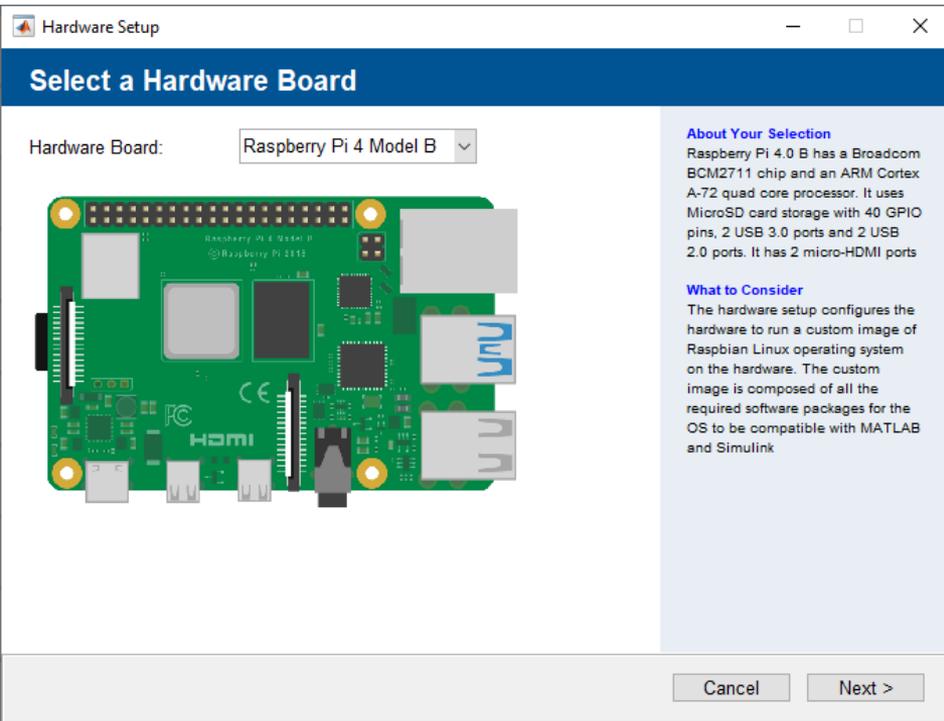


Hardware Setup

Setup Hardware with MathWorks Raspbian Image

Hardware Setup

Setup Hardware with MathWorks Raspbian Image



Hardware Setup

Select a Hardware Board

Hardware Board:

About Your Selection
Raspberry Pi 4.0 B has a Broadcom BCM2711 chip and an ARM Cortex A-72 quad core processor. It uses MicroSD card storage with 40 GPIO pins, 2 USB 3.0 ports and 2 USB 2.0 ports. It has 2 micro-HDMI ports.

What to Consider
The hardware setup configures the hardware to run a custom image of Raspbian Linux operating system on the hardware. The custom image is composed of all the required software packages for the OS to be compatible with MATLAB and Simulink.

Network Settings

Select Network configuration

- Connect to LAN or home network
- Connect to wireless network
- Connect directly to host computer
- Manually enter network settings

About Your Selection
This option in Select Network Configuration allows you to connect to a wireless network.

What to Consider
Make sure that the Raspberry Pi is in the vicinity of the wireless access point you want to connect. Then, click **Next**.

Wireless Configuration

SSID:

Security:

Password:

About Your Selection
Set the security options for the selected wireless network.

The selected IP assignment applies dynamic network settings provided by a DHCP service on the network.

What to Consider
Ensure that you have selected the correct security option for the wireless network to be connected.

IP Assignment

- Automatically get IP address
- Manually enter IP address

Select a Drive

Insert a 8 GB or larger MicroSD memory card into a memory card reader on the host computer.

Select the drive that corresponds to the memory card reader:

Drive:

What to Consider
If you do not find the memory card reader in the list of drives, insert the memory card fully. Then, click **Refresh**.

Make sure that you have properly formatted the SD card and at least 4GB is available.

Note
Navigating to next screen is allowed only when inserted SD card's file system is selected.

Slide lock switch up to unlocked position

Confirm Hardware Configuration

IP address	172.20.10.11
Host name	raspberrypi-3BdMoCwKE
User name	pi
Password	raspberrypi

Test Hardware Connection successful

What to Consider
If the Test Connection succeeds, click **Next** to finish the setup process.
If the Test Connection FAILS, click **Back** and follow the instructions again or by switching to another Network Setting.

Note
1. Your Raspberry Pi will speak its IP address through the analog audio connector when it boots.
2. You can configure your Raspberry Pi to automatically send an e-mail when IP address changes.

Test Hardware

The screenshot shows the MATLAB R2020b interface with the Command Window displaying the output of the `raspi` command. The Command Window shows the following output:

```
>> r = raspi  
  
r =  
  
raspi with properties:  
  
    DeviceAddress: '172.20.10.11'  
        Port: 18734  
    BoardName: 'Raspberry Pi 4 Model B'  
 AvailableLEDs: {'led0'}  
AvailableDigitalPins: [4, 5, 6, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27]  
AvailableSPIChannels: {'CE0', 'CE1'}  
 AvailableI2CBuses: {'i2c-1'}  
 AvailableWebcams: {}  
      I2CBusSpeed: 100000  
  
Supported peripherals
```

The Workspace window shows a variable `r` of type `1x1 raspi`.

Name	Value
r	1x1 raspi

Documentation and Examples

Documentation

CONTENTS

« Documentation Home

Category

- Installation and Setup
- Connection to Raspberry Pi Hardware
- Run on Target Hardware
- LEDs
- GPIO Pins
- Serial Port
- I2C Interface
- SPI Interface
- Camera Board
- Sense HAT
- Web Camera
- Pulse Width Modulation
- Servo
- Linux
- Display
- Audio

All Examples Functions

MATLAB Support Package for Raspberry Pi Hardware

Program sensor and image applications on Raspberry Pi

MATLAB® Support Package for Raspberry Pi™ Hardware enables you to communicate with a Raspberry Pi remotely from a computer running MATLAB or through a web browser with MATLAB Online™. You can acquire data from sensors and imaging devices connected to the Raspberry Pi and process them in MATLAB. You can also communicate with other hardware through the GPIO, serial, I2C, and SPI pins.

The support package functionality is extended if you have MATLAB Coder™. With MATLAB Coder, you can take the same MATLAB code used to interactively control the Raspberry Pi from your computer and deploy it directly to the Raspberry Pi to run as a standalone executable.

 [Release Notes](#)
 [PDF Documentation](#)

Installation and Setup

Install support for the hardware, update the firmware, and connect to the hardware

Connection to Raspberry Pi Hardware

Create a connection to Raspberry Pi hardware

Run on Target Hardware

Deploy a MATLAB function as a standalone executable on the hardware

LEDs

Use the Raspberry Pi's LED

GPIO Pins

Use the Raspberry Pi's GPIO pins

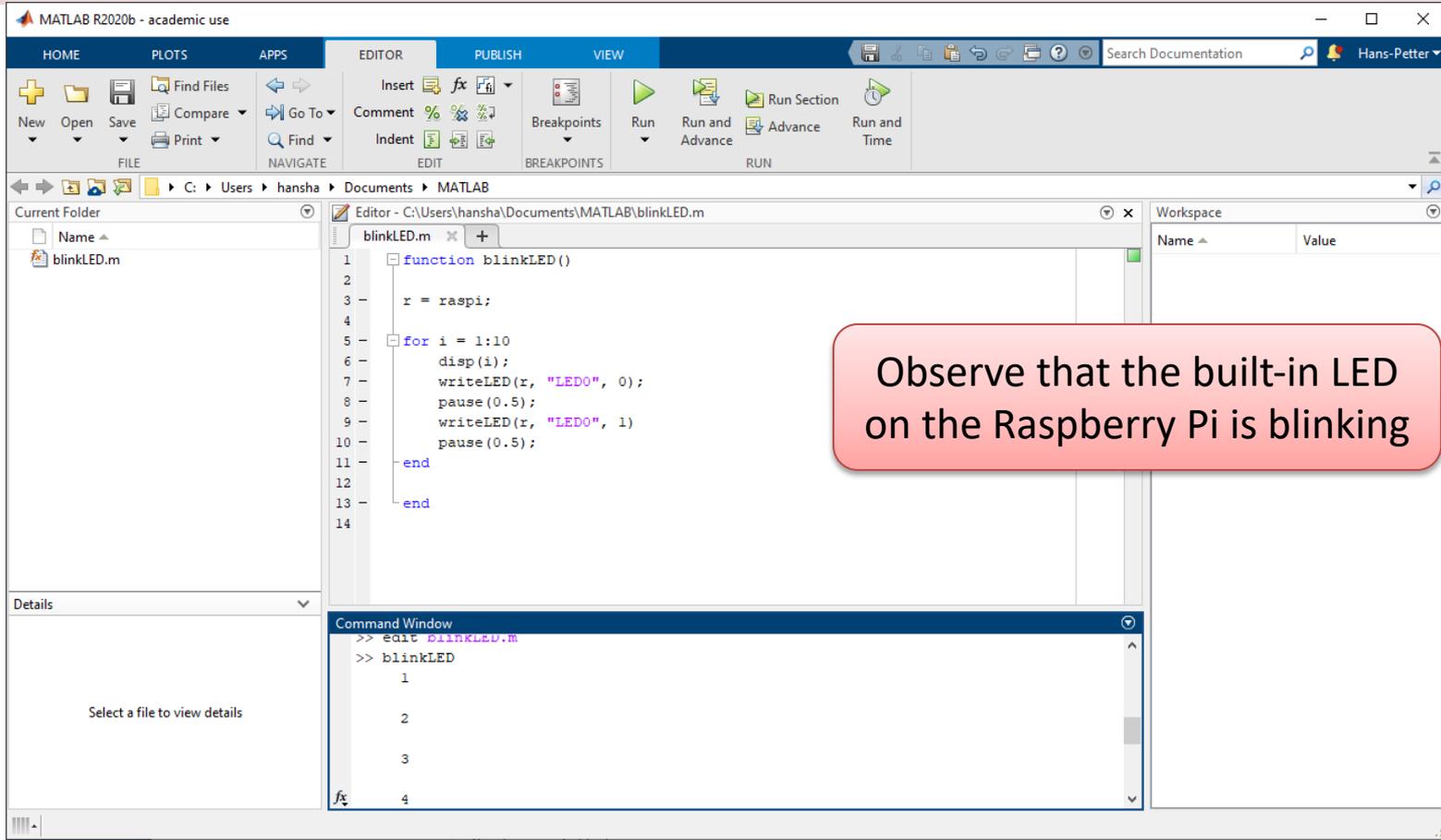
Serial Port

Use the Raspberry Pi's serial port

I2C Interface

Use the Raspberry Pi's I2C interface

Blinking LED Example



The image shows the MATLAB R2020b interface. The Editor window displays the following MATLAB code for a function named `blinkLED`:

```
1 function blinkLED()  
2  
3     r = raspi;  
4  
5     for i = 1:10  
6         disp(i);  
7         writeLED(r, "LEDO", 0);  
8         pause(0.5);  
9         writeLED(r, "LEDO", 1);  
10        pause(0.5);  
11    end  
12  
13 end  
14
```

The Command Window shows the execution output:

```
>> edit blinkLED.m  
>> blinkLED  
1  
2  
3  
4
```

A red callout box on the right side of the interface contains the text: "Observe that the built-in LED on the Raspberry Pi is blinking".

Blinking LED Example

WE use the following Function:

```
writeLED(r, "LED0", 1);
```

```
clear
clc

r = raspi;

for i = 1:10
    disp(i);
    writeLED(r, "LED0", 0);
    pause(0.5);
    writeLED(r, "LED0", 1)
    pause(0.5);
end
```



Hardware Setup

Customize the existing Operating System running on my Hardware

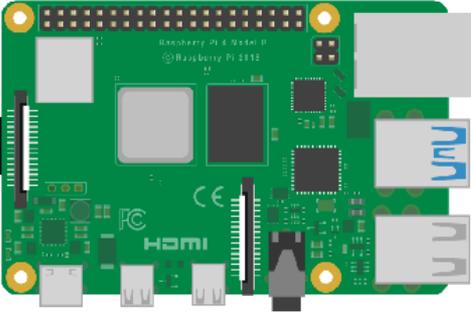
Hardware Setup

Customize the existing Operating System running on my Hardware

Hardware Setup

Select a Hardware Board

Hardware Board:



About Your Selection
Raspberry Pi 4.0 B has a Broadcom BCM2711 chip and an ARM Cortex A-72 quad core processor. It uses MicroSD card storage with 40 GPIO pins, 2 USB 3.0 ports and 2 USB 2.0 ports. It has 2 micro-HDMI ports

What to Consider
The hardware setup configures the hardware to run a custom image of Raspbian Linux operating system on the hardware. The custom image is composed of all the required software packages for the OS to be compatible with MATLAB and Simulink

Cancel Next >

Hardware Setup

Select Linux Operating System

MATLAB and Simulink support package for Raspberry Pi require a customized version of Raspbian Linux Operating System (OS) running on the hardware.

I want to:

- Setup hardware with MathWorks Raspbian image
- Customize the existing operating system running on my hardware

About Your Selection
The option uses the Raspbian image provided by MathWorks to setup the hardware. MathWorks Raspbian Linux image is composed of default Raspbian Buster Lite image with all the required libraries and packages for the image to be compatible with MATLAB and Simulink

What to Consider
To setup the hardware, the hardware setup app copies a supported Raspbian Linux firmware image to a memory card, and then boots the hardware with this memory card..

Checklist:

- Raspberry Pi Board
- SD Card (8GB or larger)
- 5V micro USB power supply
- Ethernet cable or Wireless access point

< Back Cancel Next >

Hardware Setup

Hardware Setup

Enter Login Credentials

The Operating System (OS) customization process requires a network connection to the hardware. Specify the login details of the hardware.

Device address:

Device username:

Device password:

Test connection

Ping hardware test status
SSH connection test status
SUDO user privilege verification status
Internet access verification status

< Back

Hardware Setup

Enter Login Credentials

The Operating System (OS) customization process requires a network connection to the hardware. Specify the login details of the hardware.

Device address:

Device username:

Device password:

Test connection

✓ Pinging hardware successful
✓ Establishing SSH connection successful
✓ Verifying SUDO user privilege successful
✓ Verifying Internet access successful

< Back

Cancel Next >

What to Consider

Before connecting to the hardware, ensure that the hardware:

- Is powered ON
- Is running a valid Raspbian operating system image and is bootable
- Can be connected using a Secure Shell (SSH) terminal.
- The username provided has SUDO privilege and passwordless SUDO is enabled.
- Has Internet access

If the **Test connection** succeeds click **Next** to proceed, else verify the login credentials and re-test.

Hardware Setup

Hardware Setup

Review Required Packages and Libraries

The listed software packages and libraries will be validated and installed on your hardware.

Packages

- libstd1.2-dev
- alsa-utils
- espeak
- i2c-tools
- libi2c-dev
- ssmtp
- ntpdate
- git-core
- v4l-utils
- cmake
- sense-hat
- sox
- libsox-fmt-all
- libsox-dev

Libraries

- userland
- wiringpi
- pigpio

What to Consider
Installation of software packages and libraries on the Raspbian Pi OS requires Internet access. Ensure that the hardware has internet access.

< Back Cancel Next >

Hardware Setup

Install Packages and Libraries

Install the required packages and libraries on the Raspbian Operating System (OS) running on the hardware.

What to Consider
This task will modify the OS by installing new software packages and libraries that are missing. Estimated time to complete this task is approximately 45 minutes. **We recommend that you perform a backup of the operating system running on your hardware before proceeding with the installation.**

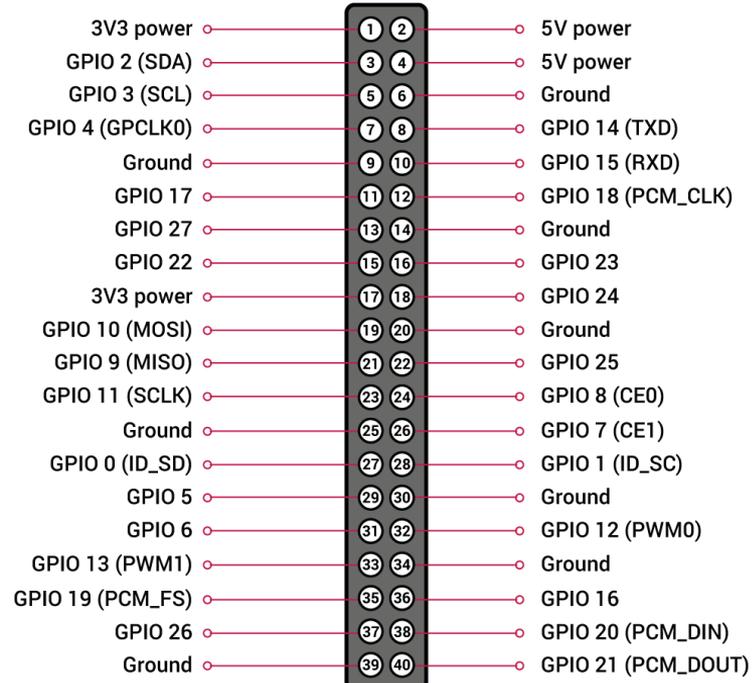
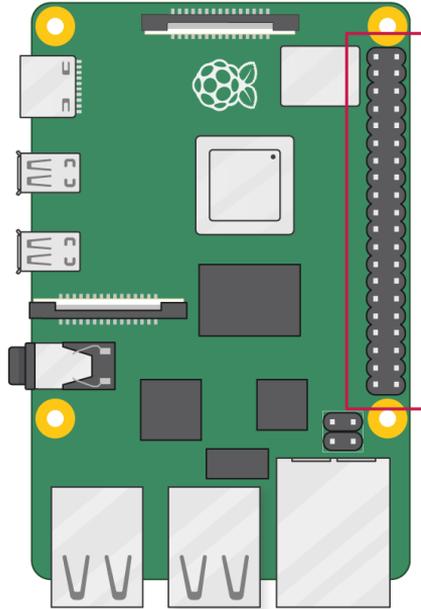
Progress bar: [] **Install**

< Back Cancel Next >

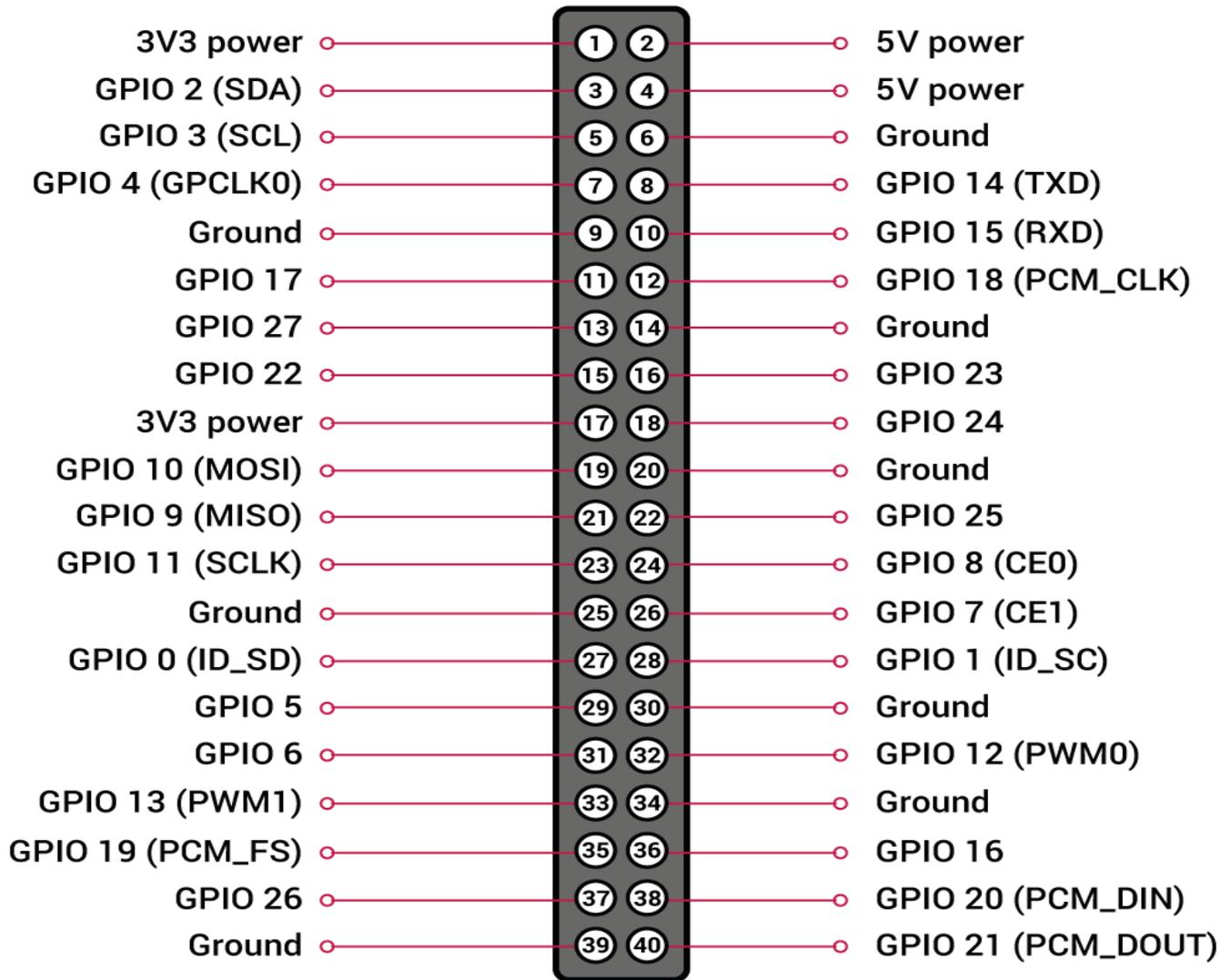


GPIO

GPIO



GPIO



GPIO Features

The GPIO pins are Digital Pins which are either True (+3.3V) or False (0V). These can be used to turn on/off LEDs, etc.

The Digital Pins can be either Output or Input.

In addition, some of the pins also offer some other Features:

- PWM (Pulse Width Modulation)

Digital Buses (for reading data from Sensors, etc.):

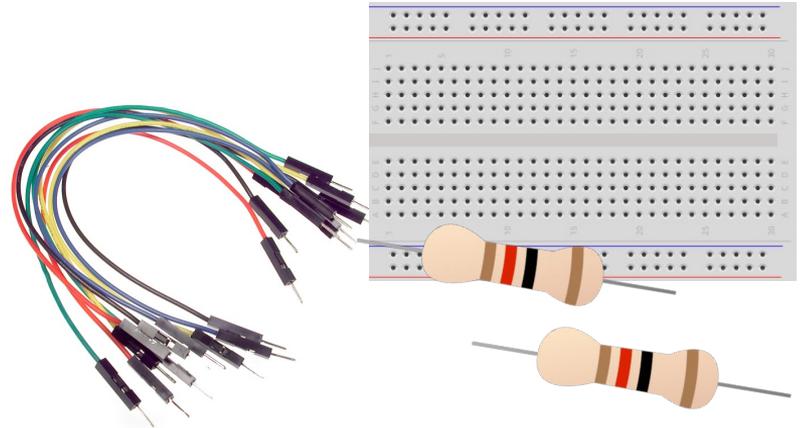
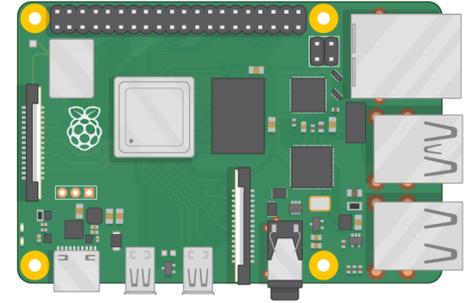
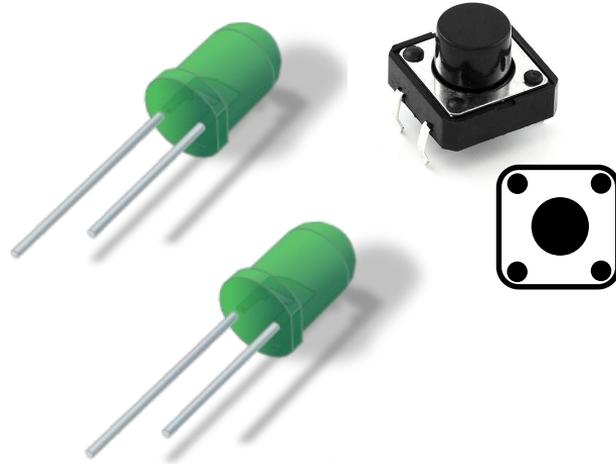
- SPI
- I2C



GPIO Code Examples

Necessary Equipment

- Raspberry Pi
- Breadboard
- LEDs
- Push Buttons
- Resistors
- Wires (Jumper Wires)

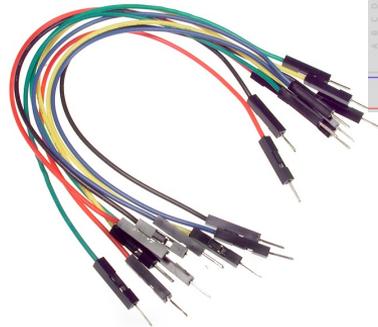
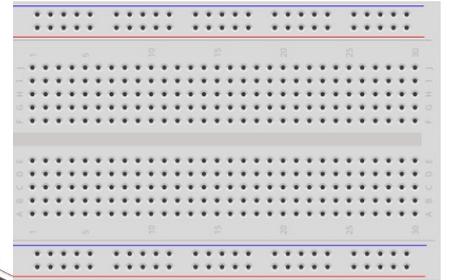
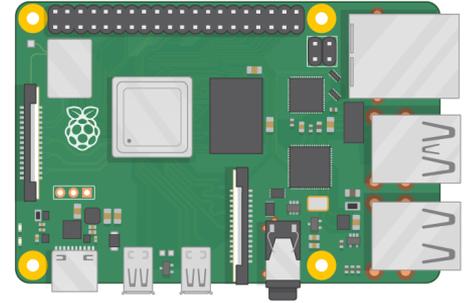




LED Example

Necessary Equipment

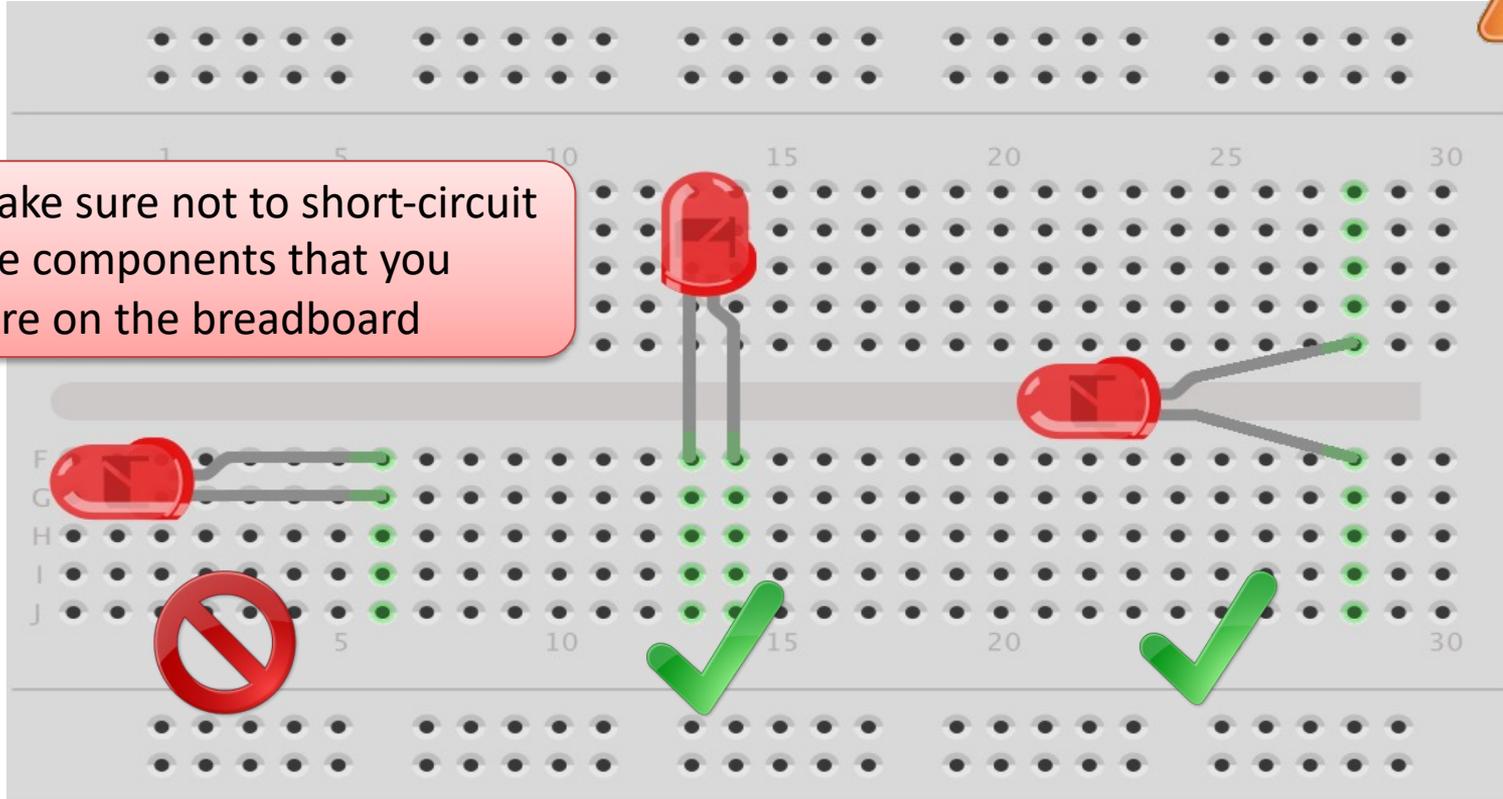
- Raspberry Pi
- Breadboard
- LED
- Resistor (270Ω)
- Wires (Jumper Wires)



Breadboard Wiring

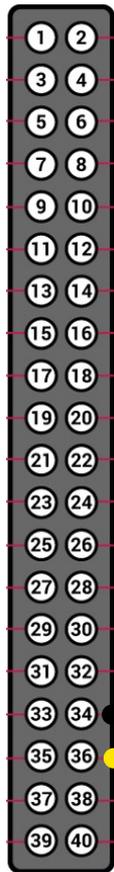


Make sure not to short-circuit the components that you wire on the breadboard



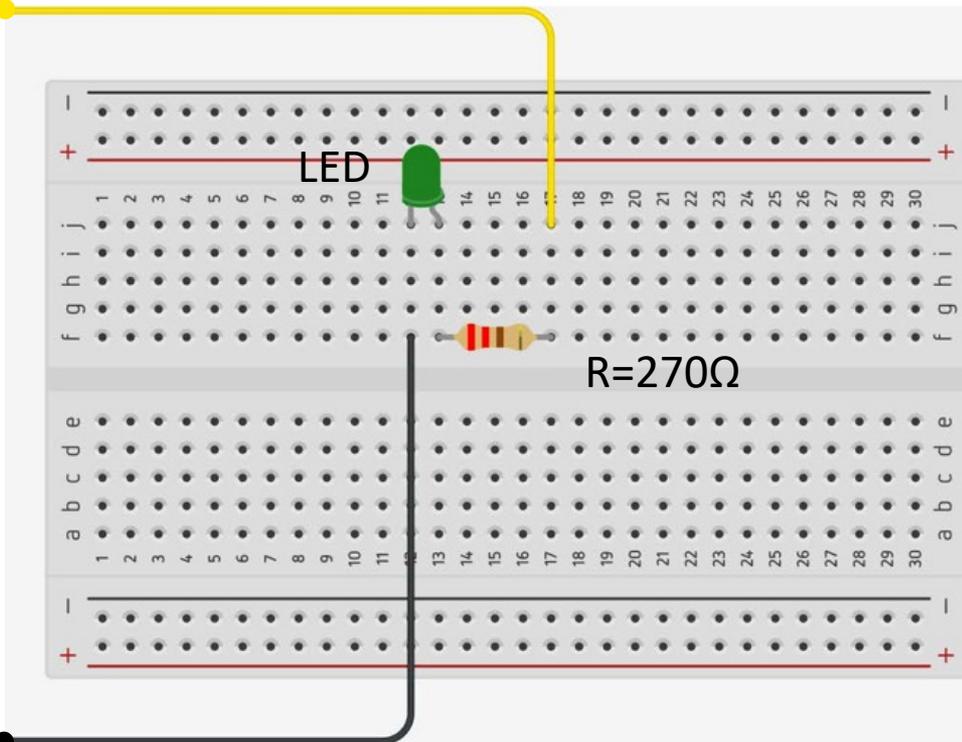
Wiring

Raspberry Pi GPIO Pins



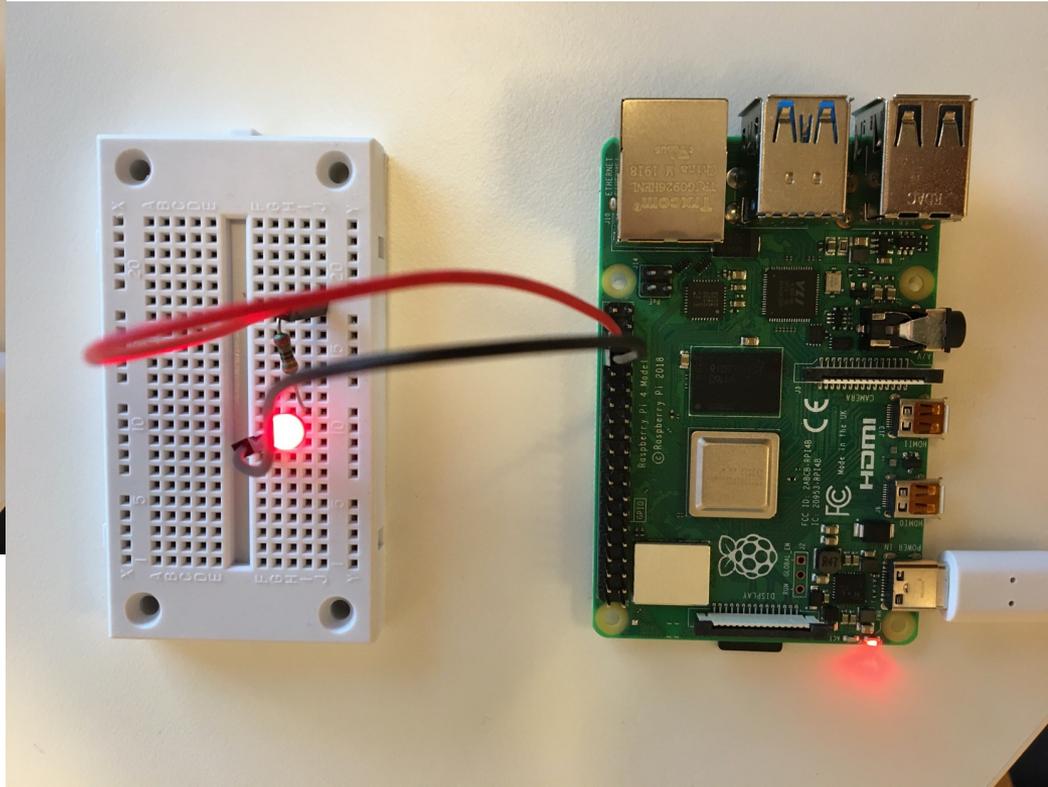
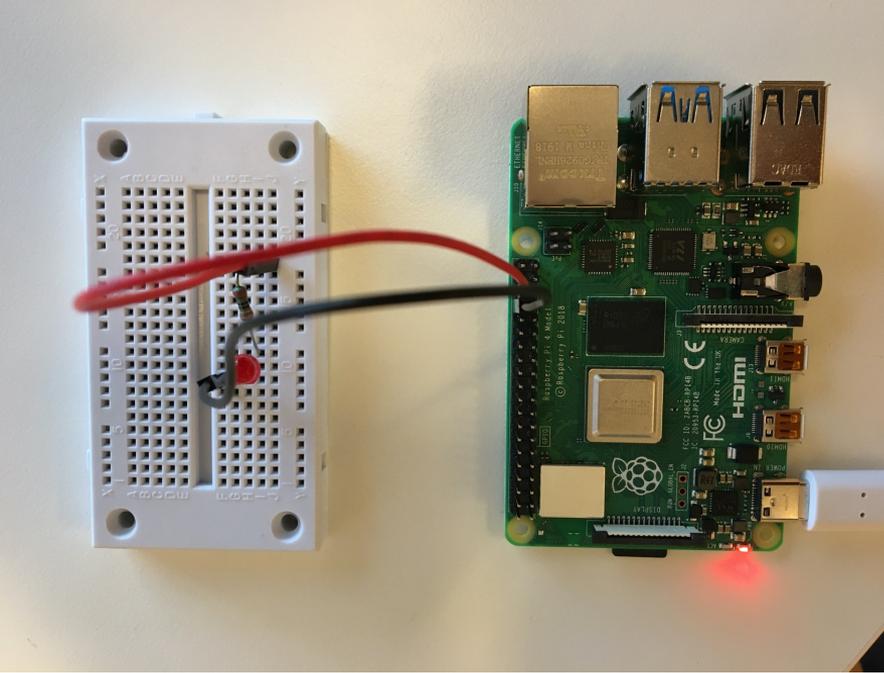
GND (Pin 34)

GPIO16 (Pin 36)



Breadboard

Blinking LED GPIO Example



Turn LED ON Example



```
clear rpi
rpi = raspi();
gpiopin = 16;
ledvalue = 1;
writeDigitalPin(rpi, gpiopin, ledvalue);
```

Blinking LED GPIO Example

```
clear rpi
rpi = raspi();
gpiopin = 16;
```

```
for i = 1:10
```

```
    ledvalue = 1;
```

```
    writeDigitalPin(rpi, gpiopin, ledvalue);
```

```
    pause(0.5);
```

```
    ledvalue = 0;
```

```
    writeDigitalPin(rpi, gpiopin, ledvalue);
```

```
    pause(0.5);
```

```
end
```



Blinking LED GPIO Example

```
clear rpi
rpi = raspi();
gpiopin = 16;

for i = 1:10
    ledvalue = 1;
    writeDigitalPin(rpi, gpiopin, ledvalue);
    pause(0.5);
    ledvalue = 0;
    writeDigitalPin(rpi, gpiopin, ledvalue);
    pause(0.5);
end
```



PWM

Pulse Width Modulation

Hans-Petter Halvorsen

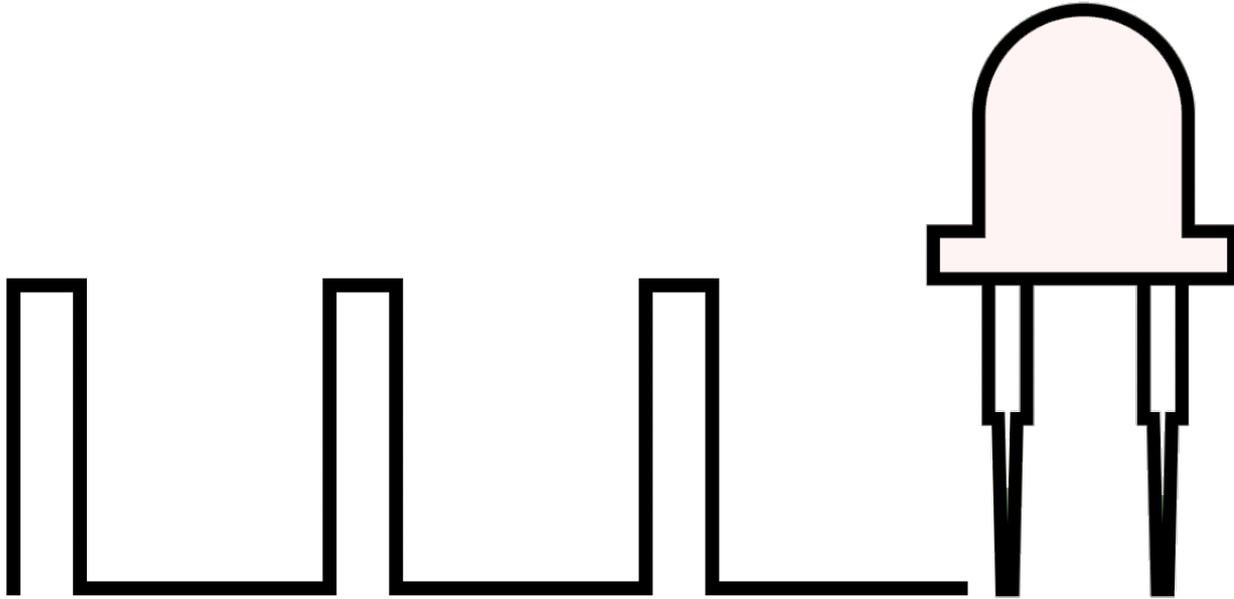
[Table of Contents](#)

Controlling LED Brightness using PWM

- We've seen how to turn an LED on and off, but how do we control its brightness levels?
- An LED's brightness is determined by controlling the amount of current flowing through it, but that requires a lot more hardware components.
- A simple trick we can do is to flash the LED faster than the eye can see!
- By controlling the amount of time the LED is on versus off, we can change its perceived brightness.
- This is known as *Pulse Width Modulation* (PWM).

Controlling LED Brightness using PWM

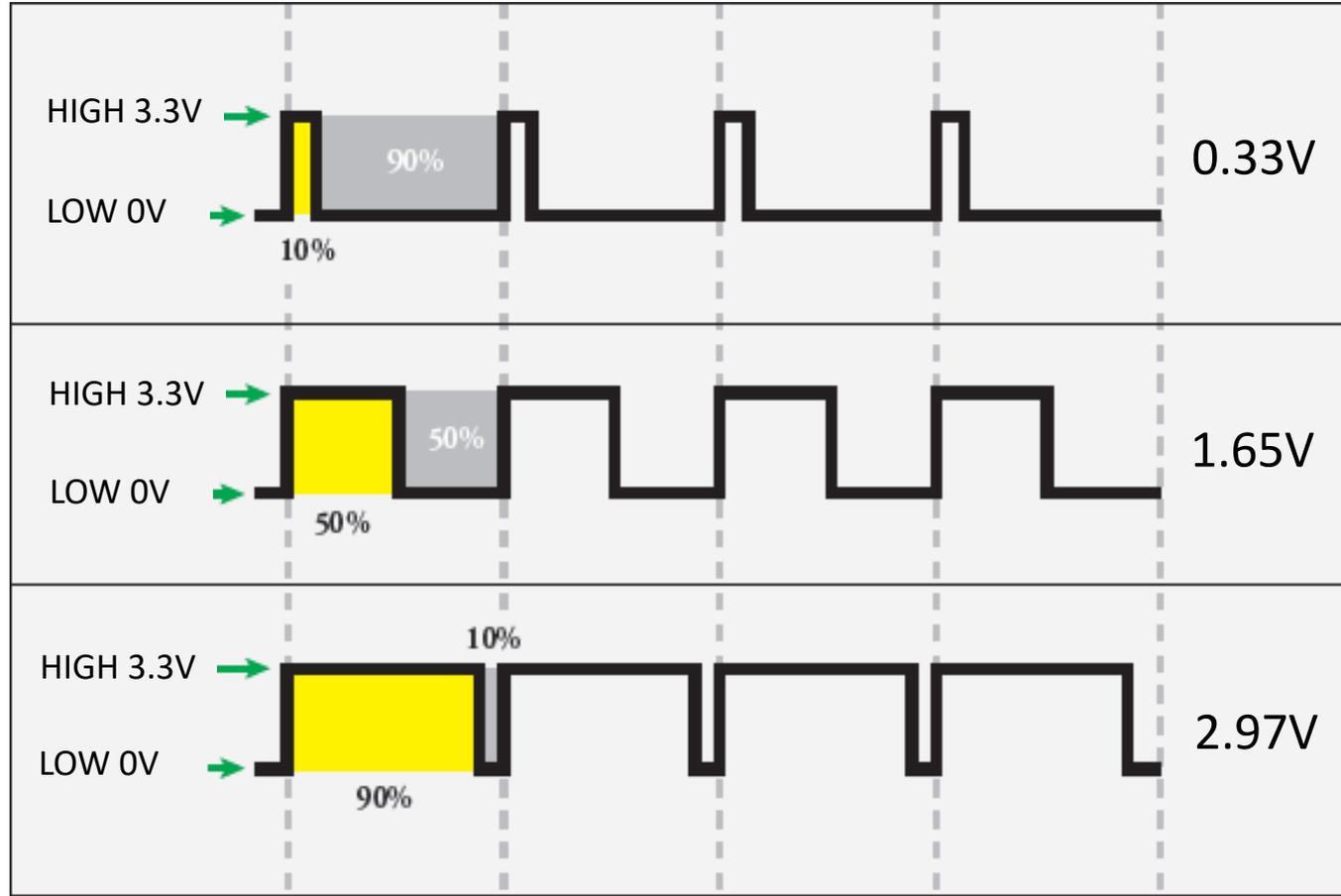
Below we see how we can use PWM to control the brightness of a LED



PWM as “Analog Out”

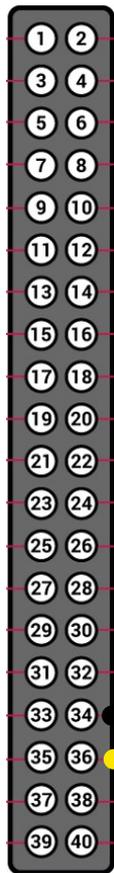
The Raspberry Pi has no real Analog Out pins, but we can use a PWM pin. PWM can be used to control brightness of a LED, control the speed of a Fan, control a DC Motor, etc.

0 – 3.3V



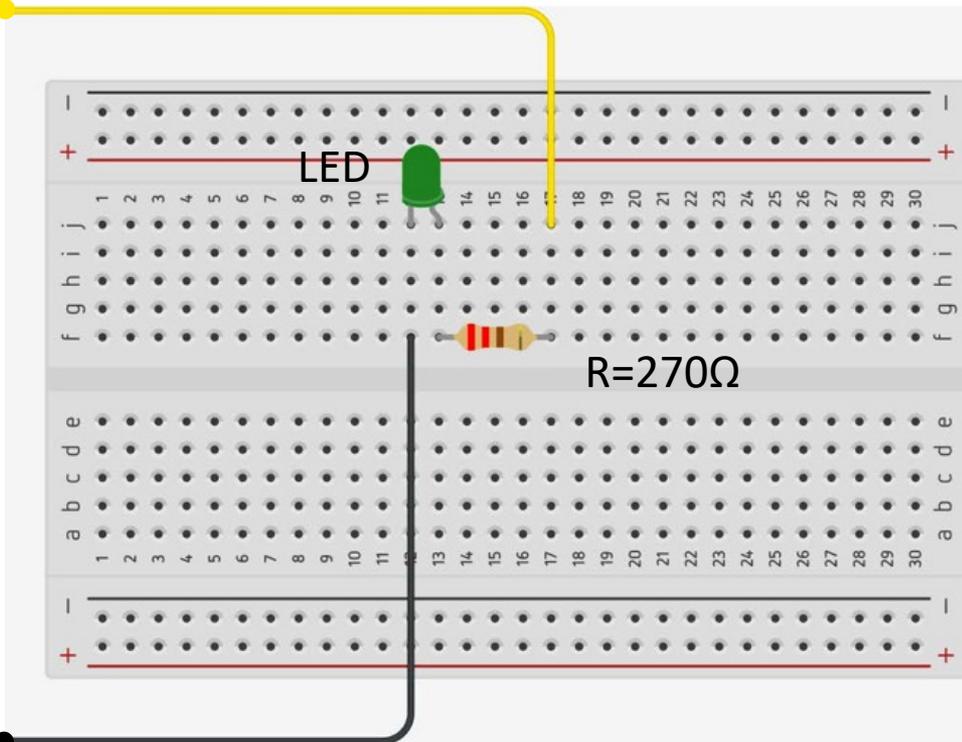
Wiring

Raspberry Pi GPIO Pins



GND (Pin 34)

GPIO16 (Pin 36)



R=270Ω

Breadboard

writePWMSVoltage

```
clear mypi  
mypi = raspi()
```

```
gpiopin = 16;
```

```
configurePin(mypi, gpiopin, 'PWM');  
writePWMSFrequency(mypi, gpiopin, 2000);
```

```
volt = 1.65
```

```
writePWMSVoltage(mypi, gpiopin, volt);
```

volt between 0V (0%) and 3.3V (100%)
This means 1.65V = 50% brightness of the LED

writePWMDutyCycle

```
clear mypi  
mypi = raspi()
```

```
gpiopin = 16;
```

```
configurePin(mypi, gpiopin, 'PWM');  
writePWMFrequency(mypi, gpiopin, 2000);
```

```
dutycycle = 0.5  
writePWMDutyCycle(mypi, gpiopin, dutycycle);
```

dutycycle between 0 (0%) and 1 (100%)
This means 0.5 = 50% brightness of the LED

PWM in MATLAB - Summary

```
clear mypi
mypi = raspi()

gpiopin = 16;

configurePin(mypi, gpiopin, 'PWM');
writePWMFrequency(mypi, gpiopin, 2000);

volt = 1.65
writePWMPVoltage(mypi, gpiopin, volt);
```

volt between 0V (0%) and 3.3V (100%)
This means 1.65V = 50% brightness of
the LED

dutycycle between 0 (0%) and 1 (100%)
This means 0.5 = 50% brightness of the LED

```
clear mypi
mypi = raspi()

gpiopin = 16;

configurePin(mypi, gpiopin, 'PWM');
writePWMFrequency(mypi, gpiopin, 2000);

dutycycle = 0.5
writePWMDutyCycle(mypi, gpiopin, dutycycle);
```

PWM with MATLAB

```
clear rpi
mypi = raspi()

gpiopin = 16;

configurePin(mypi, gpiopin, 'PWM');
writePWMFrequency(mypi, gpiopin, 2000);

for dutyCycle = 0:0.1:1
    writePWMDutyCycle(mypi, gpiopin, dutyCycle);
    pause(0.5);
end

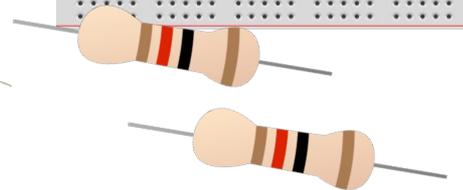
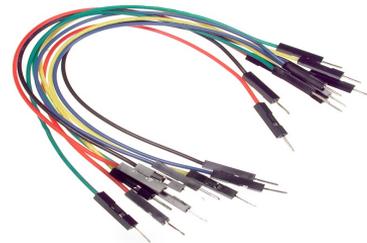
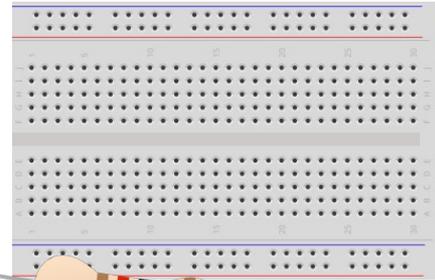
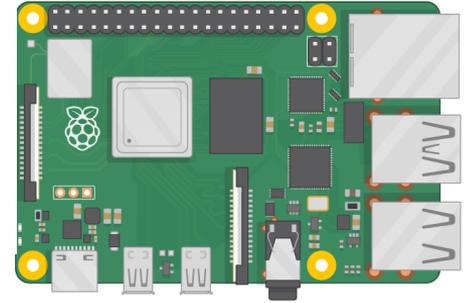
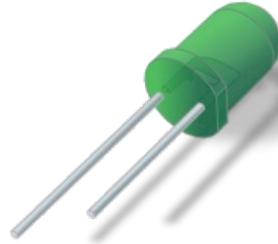
dutyCycle = 0;
writePWMDutyCycle(mypi, gpiopin, dutyCycle);
```



Push Button

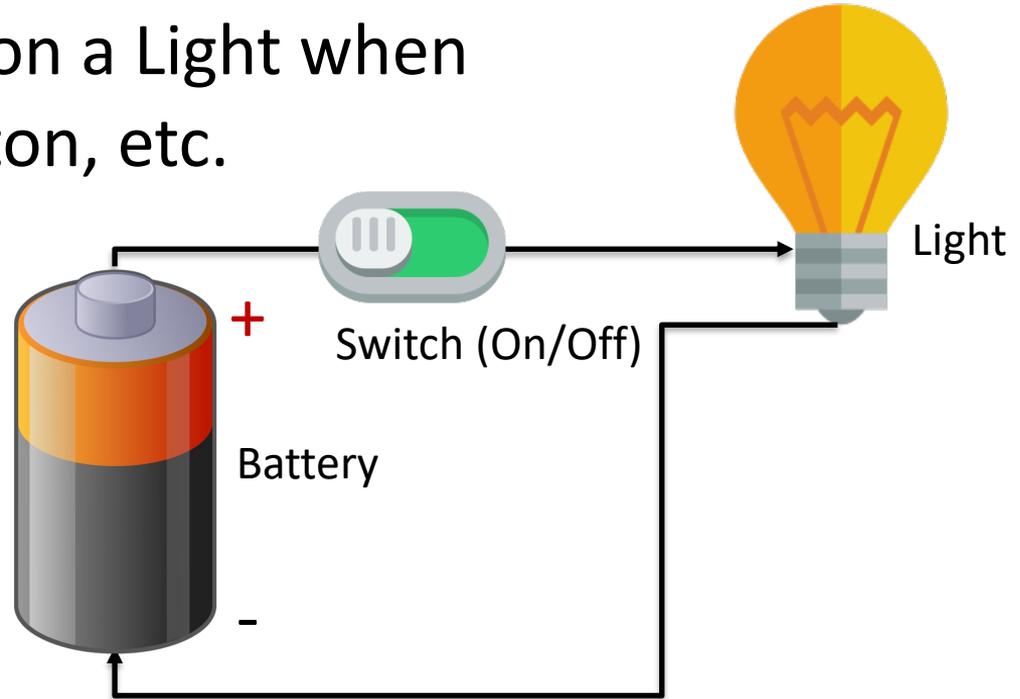
Necessary Equipment

- Raspberry Pi
- Breadboard
- Push Button
- LED
- Resistors, $R = 270\Omega$, $R = 10k\Omega$
- Wires (Jumper Wires)



Push Button/Switch

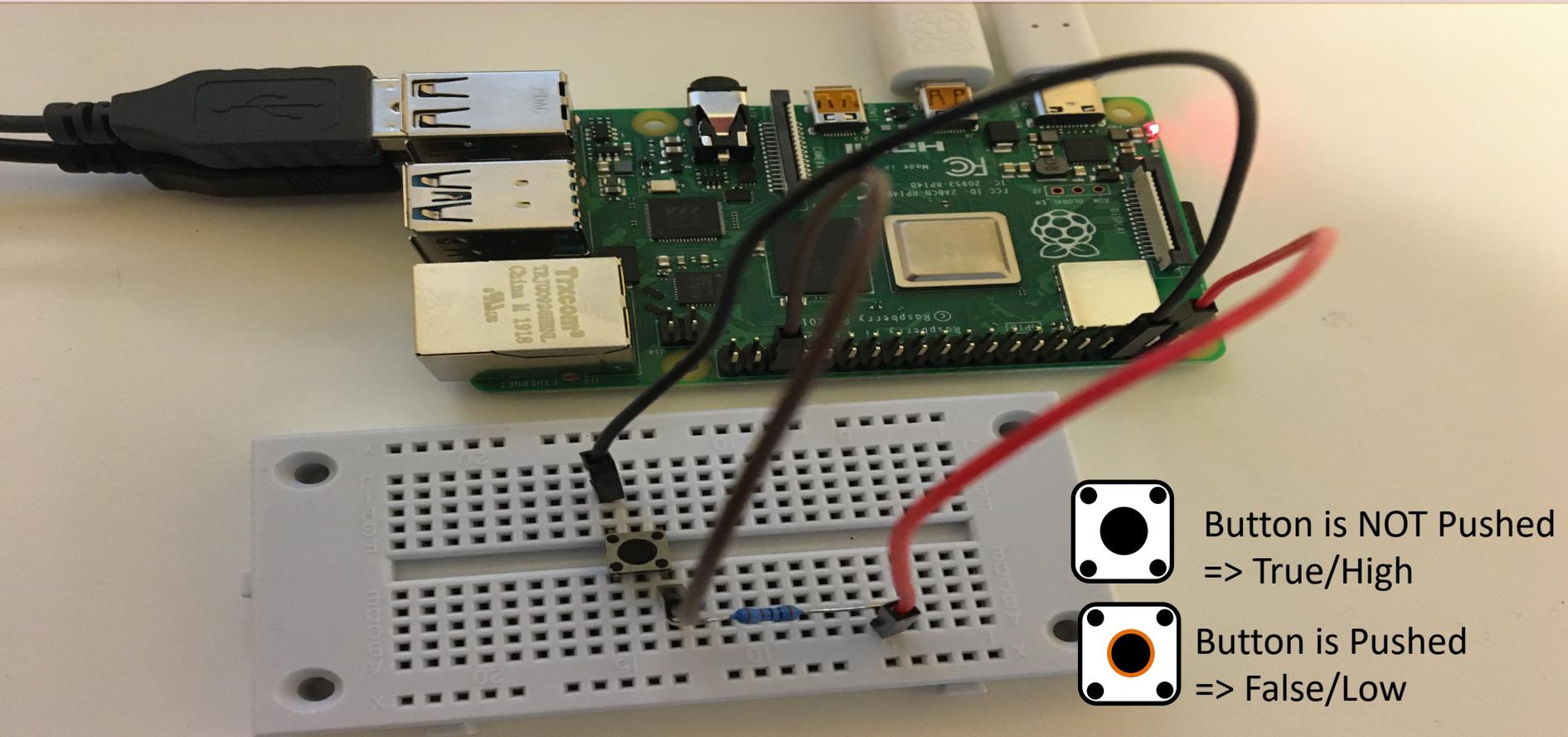
- Pushbuttons or switches connect two points in a circuit when you press them.
- You can use it to turn on a Light when holding down the button, etc.



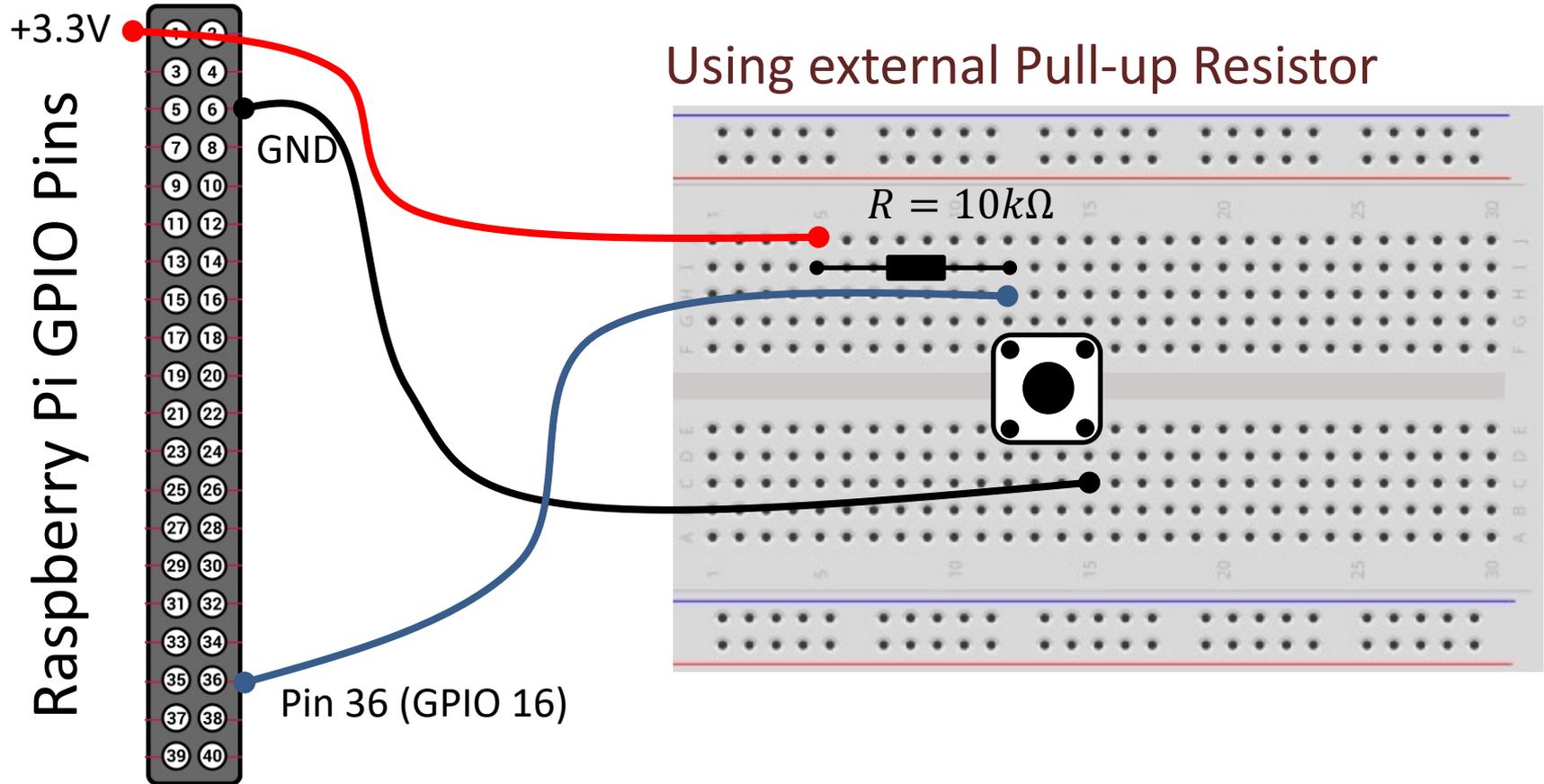


Push Button (Pull-up Resistor)

Push Button (Pull-up Resistor)

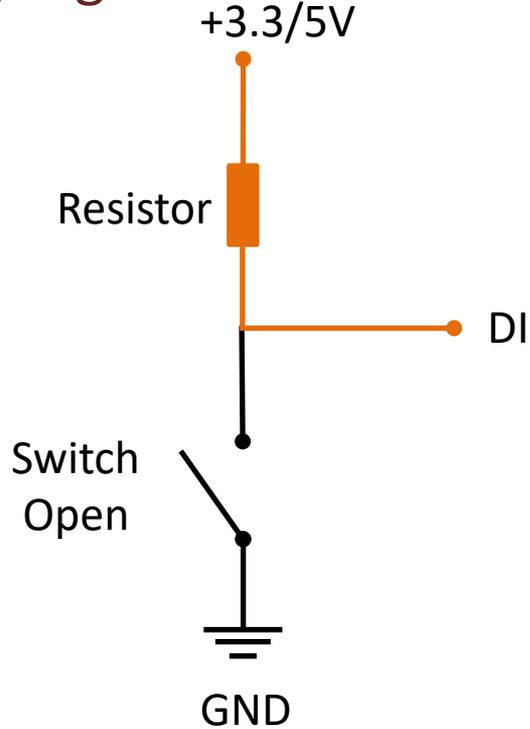


Button Setup (Pull-up Resistor)



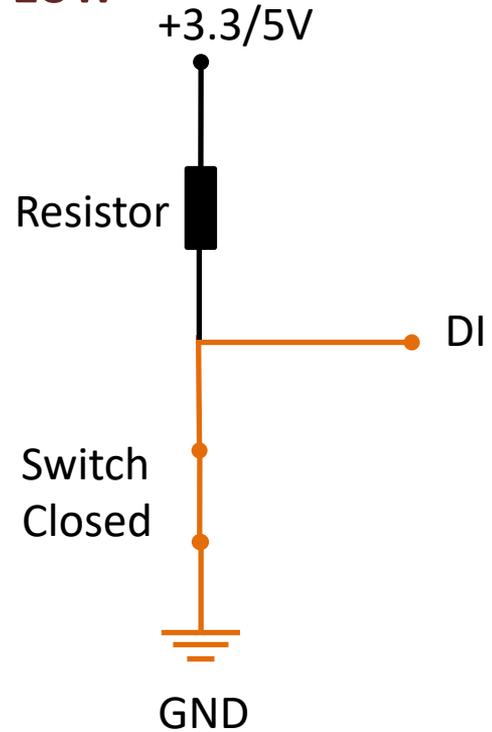
Pull-up Resistor

True/High



We Push the Button

False/Low



Pull-down/Pull-up Resistor

Why do we need a pull-up or pull-down resistor in the circuit?

- If you disconnect the digital I/O pin from everything, it will behave in an irregular way.
- This is because the input is "floating" - that is, it will randomly return either HIGH or LOW.
- That's why you need a pull-up or pull-down resistor in the circuit.

```
clear r
clc

r = raspi();
gpiopin_read = 16;

for i = 1:10
    status = readDigitalPin(r, gpiopin_read);

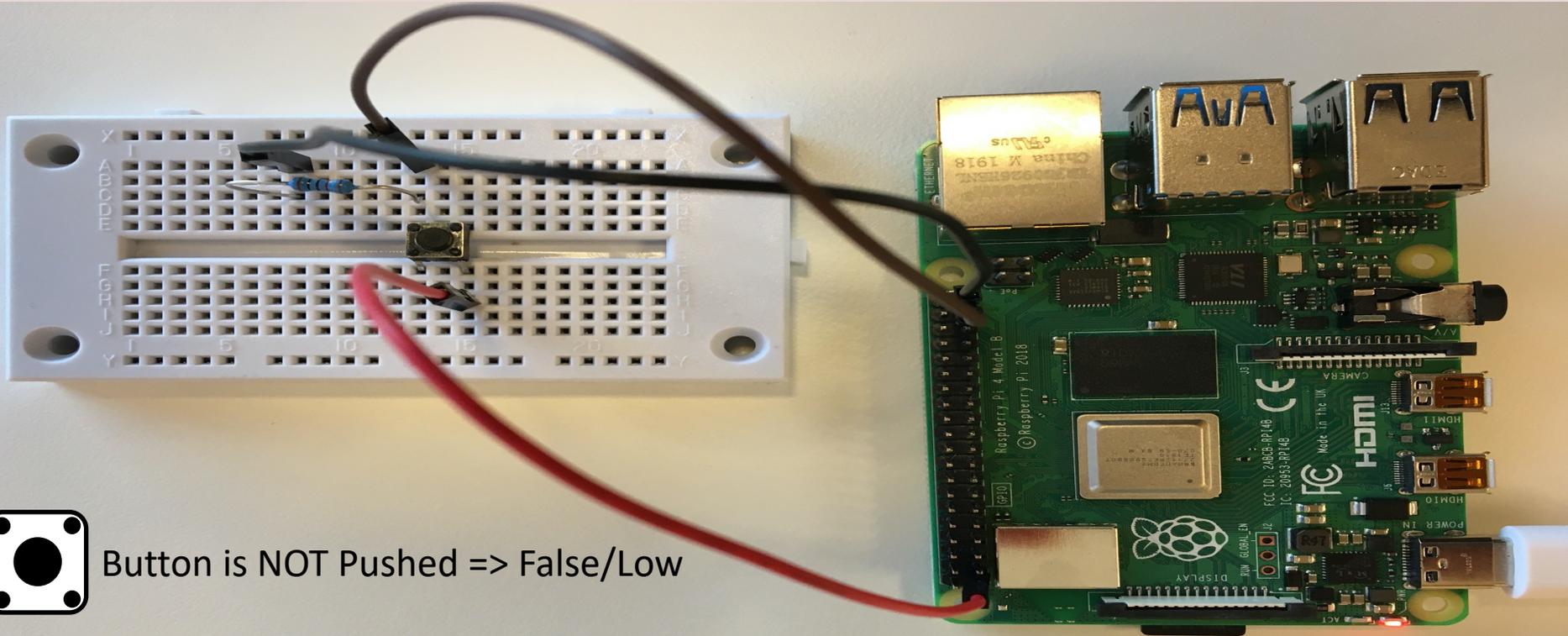
    if (status == 0)
        disp("Button Pushed")
    else
        disp("Please Push the Button")
    end

    pause(1);
end
```



Push Button (Pull-down Resistor)

Push Button (Pull-down Resistor)

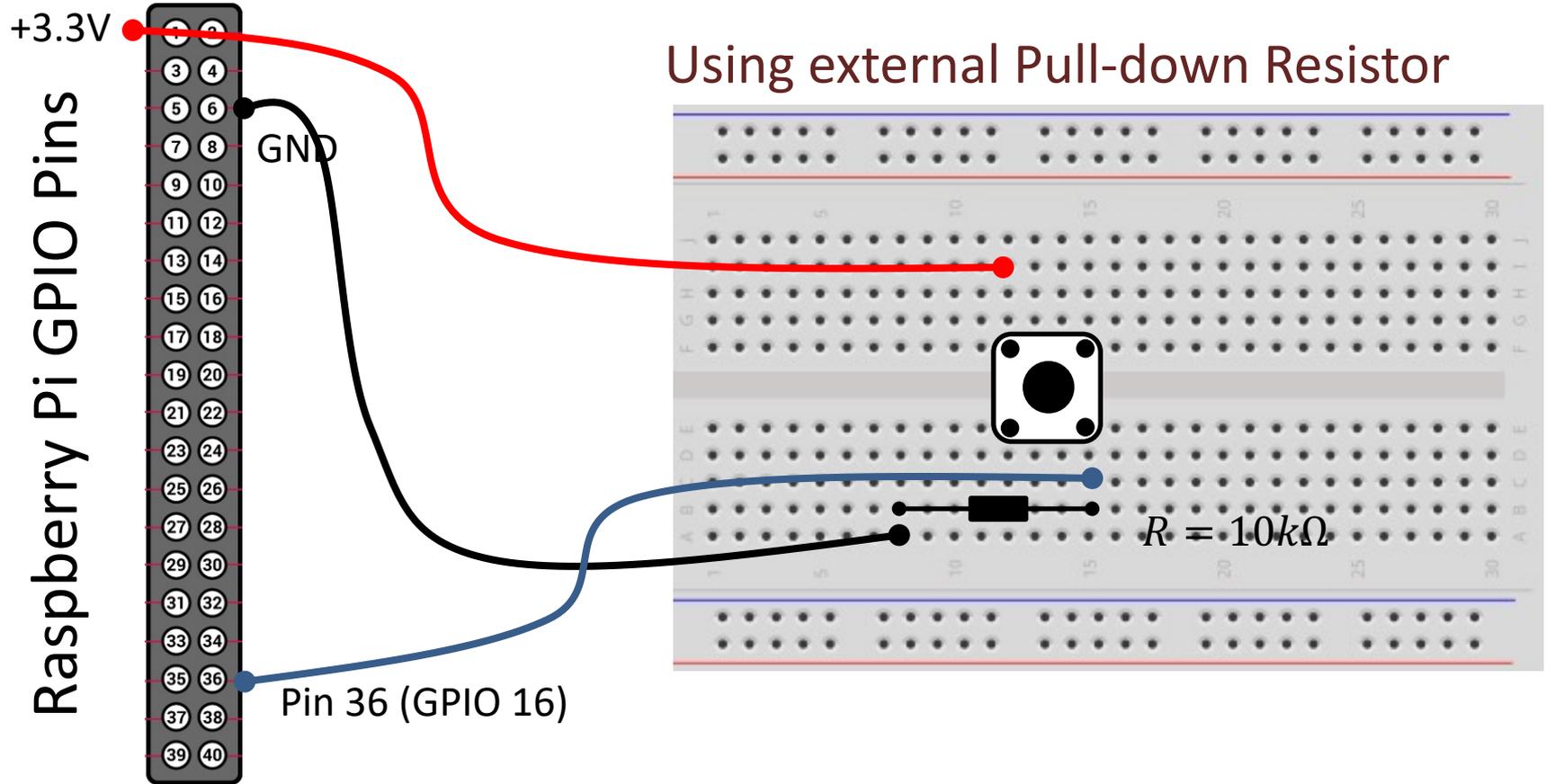


Button is NOT Pushed => False/Low



Button is Pushed => True/High

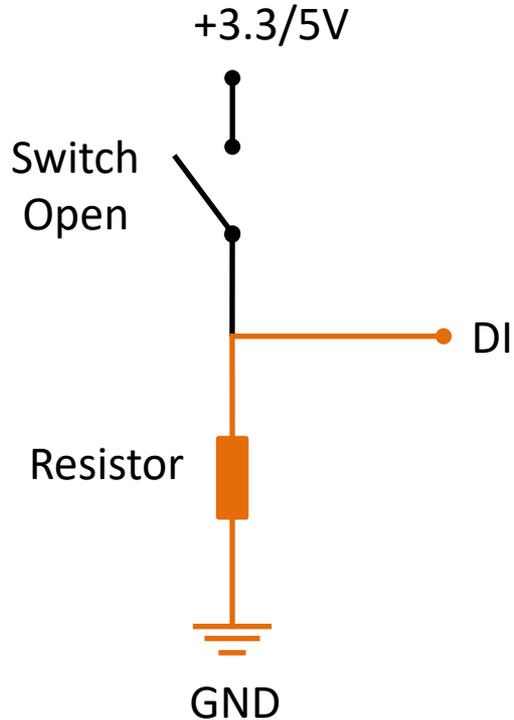
Button Setup (Pull-down Resistor)



Pull-down Resistor

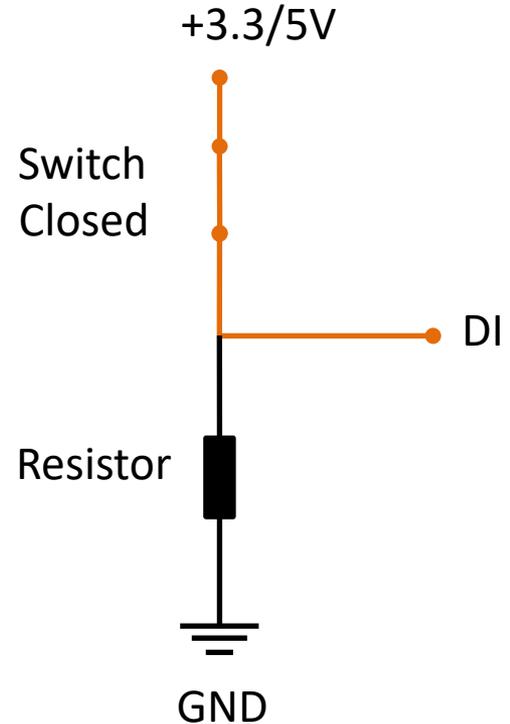
We could also have wired according to a “Pull-down” Resistor

False/Low



True/High

→
We Push the Button



```
clear r
clc

r = raspi();
gpiopin_read = 16;

for i = 1:10
    status = readDigitalPin(r, gpiopin_read);

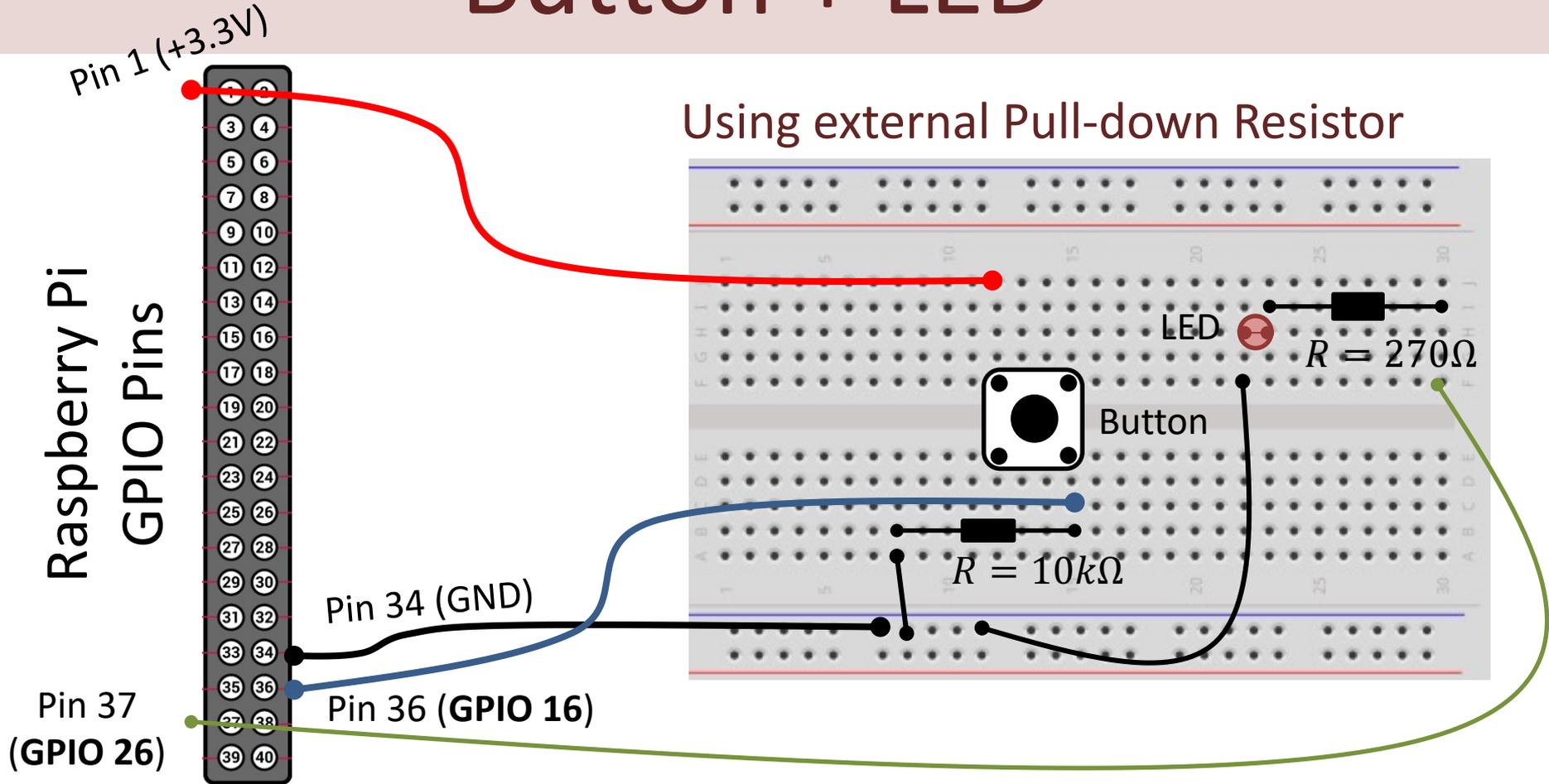
    if (status == 1)
        disp("Button Pushed")
    else
        disp("Please Push the Button")
    end

    pause(1);
end
```



Push Button + LED

Button + LED



```
clear r
clc
gpiopin_button = 16;
gpiopin_led = 26;

r = raspi();
disp("Raspberry Pi Ready")

for i = 1:10

    status = readDigitalPin(r, gpiopin_button);

    if (status == 1)
        ledvalue = 1;
        writeDigitalPin(r, gpiopin_led, ledvalue);
        disp("LED On")
    else
        ledvalue = 0;
        writeDigitalPin(r, gpiopin_led, ledvalue);
        disp("LED Off")
    end

    pause(1);

end
clear r
disp("Program is Finished")
```



Camera

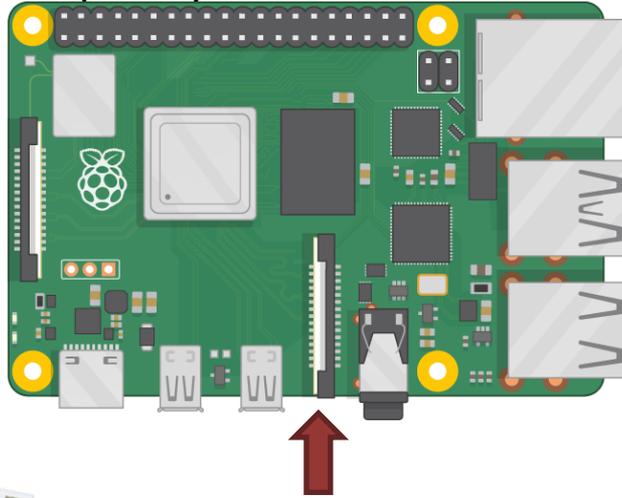
Raspberry Pi + Camera

We can either use a specialized Raspberry Pi Camera or an ordinary/standard USB Camera

Raspberry Pi Camera Module v2



Raspberry Pi



Camera Connector

CSI (Camera Serial Interface)

USB Connector



Standard USB Web Camera

Web Camera Example

The image shows the MATLAB R2020b interface with a script named `webcamex.m` open in the Editor. The script contains the following code:

```
1 mypi = raspi
2 mycam = webcam(mypi)
3
4 img = snapshot(mycam)
5 imagesc(img)
6 drawnow
7
8 clear
```

Below the script, a portion of the MATLAB Command Window is visible, showing a grid of numerical data:

34	66	144	164	153	166	193	196	195
34	78	113	164	157	156	168	198	197
95	107	104	128	156	155	148	185	198
101	92	76	101	137	156	148	162	191
64	95	76	100	112	151	148	150	171
54	63	87	105	100	128	149	147	145
95	97	103	95	82	98	140	151	145
107	105	84	99	48	82	104	147	146
99	97	67	93	70	91	97	122	145
65	95	75	67	90	101	82	80	132
93	92	104	104	97	64	38	37	93
112	100	93	80	62	48	44	46	64
112	111	109	106	103	103	102	105	111
111	109	111	113	114	117	119	120	117
105	107	107	109	111	116	118	119	116

Overlaid on the MATLAB interface is a red rounded rectangle containing the text:

Connect an USB Web Camera to the USB port on the Raspberry Pi

In the bottom right corner, a window titled "Figure 1" displays a live video feed from a web camera. The video shows a man with glasses and a beard, wearing a dark jacket, sitting at a desk in a workshop or office. He is pointing his right index finger towards the camera. The background includes shelves with various items and a window with blinds.

Web Camera Example

```
clear
clc

myipi = raspi
mycam = webcam(myipi)

img = snapshot(mycam)
imagesc(img)
drawnow
```

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: <https://www.halvorsen.blog>

