RASPBERRY PI GPIOS

CS95003 - Applied Robotics Lab

Ing. Gerardo Carmona

Differences between a computer and RPi

- While the Raspberry Pi is, in essence, a very inexpensive Linux computer, there are a few things that distinguish it from laptop and desktop machines that we usually use for writing email, browsing the web, or word processing.
- One of the main differences is that the Raspberry Pi can be directly used in electronics projects because it has **general purpose input and output pins** right on the board.

GPIO -General Purpose Input Output

Raspberry Pi Model B

Rev 1 P1 GPIO Header

	Pin No.		
3.3V	1	2	5V
GPIO0	3	4	5V
GPIO1	5	6	GND
GPIO4	7	8	GPIO14
GND	9	10	GPIO15
GPIO17	11	12	GPIO18
GPIO21	13	14	GND
GPIO22	15	16	GPIO23
3.3V	17	18	GPIO24
GPIO10	19	20	GND
GPIO9	21	22	GPIO25
GPIO11	23	24	GPIO8
GND	25	26	GPIO7

Raspberry Pi Model A/B

Rev 2 P1 GPIO Header

	Pin No.		
3.3V	1	2	5V
GPIO2	3	4	5V
GPIO3	5	6	GND
GPIO4	7	8	GPIO14
GND	9	10	GPIO15
GPIO17	11	12	GPIO18
GPIO27	13	14	GND
GPIO22	15	16	GPIO23
3.3V	17	18	GPIO24
GPIO10	19	20	GND
GPIO9	21	22	GPIO25
GPIO11	23	24	GPIO8
GND	25	26	GPIO7

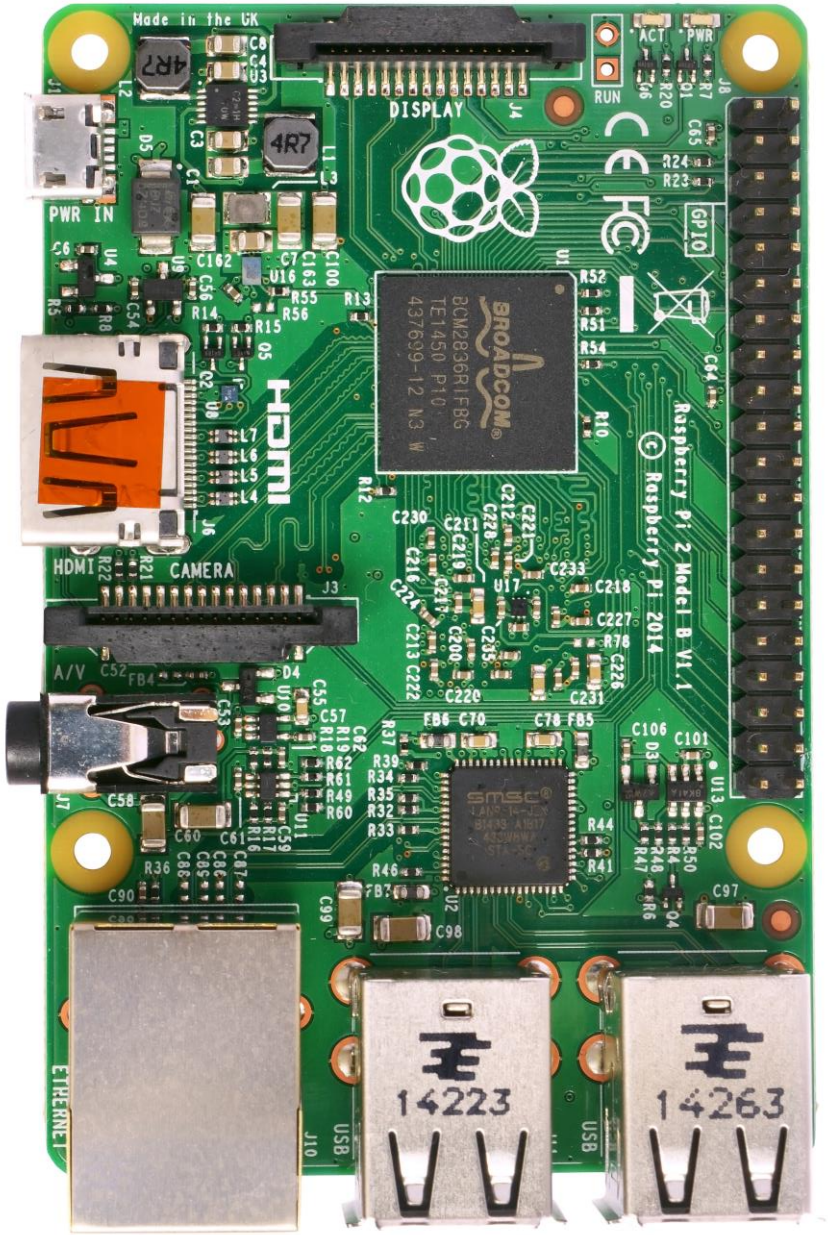
Raspberry Pi Model A+/B+/Pi 2

B+ J8 GPIO Header

	Pin No.		
3.3V	1	2	5V
GPIO2	3	4	5V
GPIO3	5	6	GND
GPIO4	7	8	GPIO14
GND	9	10	GPIO15
GPIO17	11	12	GPIO18
GPIO27	13	14	GND
GPIO22	15	16	GPIO23
3.3V	17	18	GPIO24
GPIO10	19	20	GND
GPIO9	21	22	GPIO25
GPIO11	23	24	GPIO8
GND	25	26	GPIO7
DNC	27	28	DNC
GPIO5	29	30	GND
GPIO6	31	32	GPIO12
GPIO13	33	34	GND
GPIO19	35	36	GPIO16
GPIO26	37	38	GPIO20
GND	39	40	GPIO21

Key

Power +	UART
GND	SPI
I²C	GPIO



Raspberry Pi Modelo 2B Pinout

	Pin No.		
3.3V	1	2	5V
GPIO2	3	4	5V
GPIO3	5	6	GND
GPIO4	7	8	GPIO14
GND	9	10	GPIO15
GPIO17	11	12	GPIO18
GPIO27	13	14	GND
GPIO22	15	16	GPIO23
3.3V	17	18	GPIO24
GPIO10	19	20	GND
GPIO9	21	22	GPIO25
GPIO11	23	24	GPIO8
GND	25	26	GPIO7
DNC	27	28	DNC
GPIO5	29	30	GND
GPIO6	31	32	GPIO12
GPIO13	33	34	GND
GPIO19	35	36	GPIO16
GPIO26	37	38	GPIO20
GND	39	40	GPIO21

Key

Power +	UART
GND	SPI
I²C	GPIO

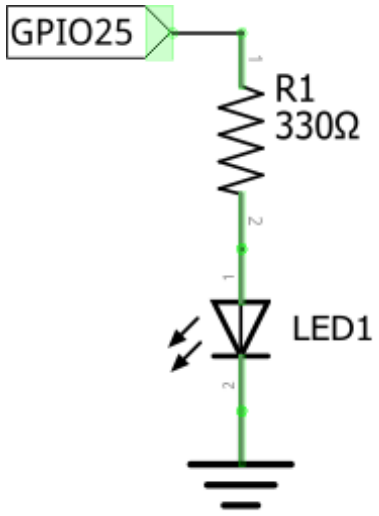
Differences having an OS

- Several inputs: You have a mouse, keyboard, Ethernet connection, monitor, SD card without need to connect additional electronics
- Filesystem: Being able to read and write data in the Linux file system will make many projects much easier.
- Linux tools: packaged in the Raspberry Pi's Linux distribution is a set of core command-line utilities, which let you work with files, control processes, and automate many different tasks.
- Languages: There are many programming languages out there and embedded Linux systems like the Raspberry Pi give you the flexibility to choose whichever language you're most comfortable with

FIRST PROGRAM

First Program

- We will turn on and off and LED
- Connect and LED to GPIO25 using a 330 ohm resistor



	Pin No.		
3.3V	1	2	5V
GPIO2	3	4	5V
GPIO3	5	6	GND
GPIO4	7	8	GPIO14
GND	9	10	GPIO15
GPIO17	11	12	GPIO18
GPIO27	13	14	GND
GPIO22	15	16	GPIO23
3.3V	17	18	GPIO24
GPIO10	19	20	GND
GPIO9	21	22	GPIO25
GPIO11	23	24	GPIO8
GND	25	26	GPIO7
DNC	27	28	DNC
GPIO5	29	30	GND
GPIO6	31	32	GPIO12
GPIO13	33	34	GND
GPIO19	35	36	GPIO16
GPIO26	37	38	GPIO20
GND	39	40	GPIO21

Key	
Power +	UART
GND	SPI
I²C	GPIO

Digital Output: Lighting Up an LED

- You can then use the Linux command line to turn the LED on and off.
- Steps:
  1) In graphic mode, double click on the LXTerminal
  2) In order to access the input and output pins from the command line, you'll need to run the commands as root, the superuseraccount on the Raspberry Pi. To start running commands as root, type sudo su at the command line and press enter:

```
pi@raspberrypi ~ $ sudo su
root@raspberrypi:/home/pi#
```

The root account has administrative access to all the functions and files on the system and there is very little protecting you from damaging the operating system if you type a command that can harm it.

Digital Output: Lighting Up an LED

3) Before you can use the command line to turn the LED on pin 25 on and off, you need to export the pin to the userspace(in other words, make the pin available for use outside of the confines of the Linux kernel), this way:

```
root@raspberrypi:/home/pi# echo 25 > /sys/class/gpio/export
```

The echo command writes the number of the pin you want to use (25) to the export file, which is located in the folder /sys/class/gpio. When you write pin numbers to this special file, it creates a new directory in /sys/class/gpio that has the control files for the pin. In this case, it created a new directory called /sys/class/gpio/gpio25.

Digital Output: Lighting Up an LED

4) Change to that directory with the cd command and list the contents of it with ls:

```
root@raspberrypi:/home/pi# cd /sys/class/gpio/gpio25
root@raspberrypi:/sys/class/gpio/gpio25# ls
active_low direction edge power subsystem uevent value
```

The echo command writes the number of the pin you want to use (25) to the export file, which is located in the folder /sys/class/gpio. When you write pin numbers to this special file, it creates a new directory in /sys/class/gpio that has the control files for the pin. In this case, it created a new directory called /sys/class/gpio/gpio25.

Digital Output: Lighting Up an LED

5) The directionfile is how you'll set this pin to be an input (like a button) or an output (like an LED). Since you have an LED connected to pin 25 and you want to control it, you're going to set this pin as an output:

```
root@raspberrypi:/sys/class/gpio/gpio25# echo out > direction
```

6) To turn the LED on, you'll use the echo command again to write the number 1 to the value file:

```
root@raspberrypi:/sys/class/gpio/gpio25# echo 1 > value
```

← LED On

7) After pressing enter, the LED will turn on! Turning it off is as simple as using echo to write a zero to the value file:

```
root@raspberrypi:/sys/class/gpio/gpio25# echo 0 > value
```

← LED Off

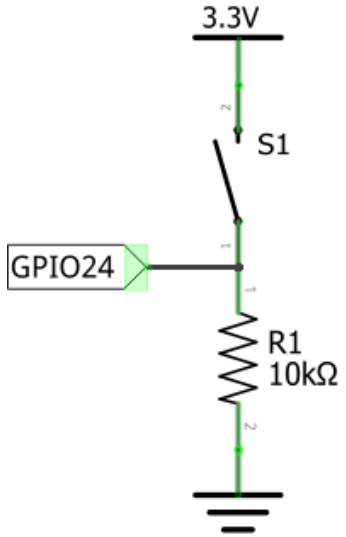
Virtual Files

- The files that you're working with aren't actually files on the Raspberry Pi's SD card, but rather are a part of Linux's virtual file system, which is a system that makes it easier to access low-level functions of the board in a simpler way.
- For example, you could turn the LED on and off by writing to a particular section of the Raspberry Pi's memory, but doing so would require more coding and more caution.

SECOND PROGRAM

Digital Inputs

- We will read a digital input and display its status
  - "0" for GND and "1" for 3.3v.
- Connect the following diagram



	Pin No.		
3.3V	1	2	5V
GPIO2	3	4	5V
GPIO3	5	6	GND
GPIO4	7	8	GPIO14
GND	9	10	GPIO15
GPIO17	11	12	GPIO18
GPIO27	13	14	GND
GPIO22	15	16	GPIO23
3.3V	17	18	GPIO24
GPIO10	19	20	GND
GPIO9	21	22	GPIO25
GPIO11	23	24	GPIO8
GND	25	26	GPIO7
DNC	27	28	DNC
GPIO5	29	30	GND
GPIO6	31	32	GPIO12
GPIO13	33	34	GND
GPIO19	35	36	GPIO16
GPIO26	37	38	GPIO20
GND	39	40	GPIO21

Key	
Power +	UART
GND	SPI
I²C	GPIO

Digital Input

- Almost same instructions. Remember to run commands as root.

```
root@raspberrypi:/home/pi# echo 24 > /sys/class/gpio/export
root@raspberrypi:/home/pi# cd /sys/class/gpio/gpio24
root@raspberrypi:/sys/class/gpio/gpio24# echo in > direction
root@raspberrypi:/sys/class/gpio/gpio24# cat value
0
```

(1)
(2)
(3)
(4)
(5)

(1) Export the pin input to userspace.
(2) Change directory.
(3) Set the direction of the pin to input.
(4) Read the value of the of the pin using cat command.
(5) Print the result of the pin, zero when you aren't not pressing the button.

ABOUT GPIOS

GPIOs

- All GPIOs are 3.3 V tolerant
- They can provide maximum 16 mA, but not exceeding 50 mA from all at the same time
- 3.3 V pin can deliver 50 mA maximum
- 5 V pin can deliver your power supply – 700 mA for the raspberry*

GPIOS WITH PYTHON

Python GPIOs

- We can use Python to control the GPIOs.
- Open Python by typing on the Linux console:

```
sudo python
```

- First make sure this library its already installed on the Raspberry Pi. In console type:

```
>>> import RPi.GPIO as GPIO
```

- If you don’t get an error, you’re all set.
- Close the console

How to install Python library

- Type the following command on the Linux console

```
$ wget http://pypi.python.org/packages/source/R/RPi.GPIO/RPi.GPIO-0.1.0.tar.gz
$ tar zxf RPi.GPIO-0.1.0.tar.gz
$ cd RPi.GPIO-0.1.0
$ sudo python setup.py install
```

- *The instructions here refer to an early version of RPi.GPIO. Please search the web for the latest version and replace the version numbers in the instructions below. On newer Raspbian distributions library is included.*

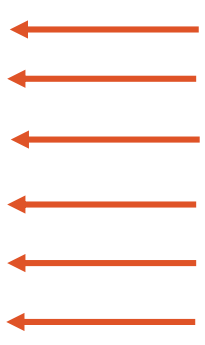
Python's IDLE

- IDLE: Integrated Development Enviroment
- Locate on the desktop "IDLE" icon and double click on it
- Or you can find Python under:
  - **start menu >> programming >> IDLE**
- Now you can write your own scripts

Installing and Testing GPIO in Python

- On the Python console type:

```
>>> import RPi.GPIO as GPIO
>>> GPIO.setmode(GPIO.BCM)
>>> GPIO.setup(25, GPIO.OUT)
>>> GPIO.output(25, GPIO.HIGH)
>>> GPIO.output(25, GPIO.LOW)
>>> exit()
```



Import GPIO library
Use BCM convention for the names of the GPIOs
Pin 25 as output
Turn on pin 25 (send 3.3v)
Turn off pin 25 (connect to ground)
Close python interpreter

- BCM is for Broadcom **BCM** 2835 chip, the chip that is containned in the Raspberry Pi. When we set mode as BCM we are telling to the library that I want to use the real pin names of the BCM chip. There are other configurations that we will not use in this class (such as board).

Blinking an LED

- We will use a Python Script.
- Create a new file, name it blink.py
- Write the following code:

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setup(25, GPIO.OUT)
while True:
    GPIO.output(25, GPIO.HIGH)
    time.sleep(1)
    GPIO.output(25, GPIO.LOW)
    time.sleep(1)
```

Execute Blinking an LED

- Open LXTerminal and go to the directory that contains your “py” file
  - Remember that you can use cd, cd .., and ls commands
- Type:

```
pi@raspberrypi ~ $ sudo python blink.py
```

- Your LED should now be blinking!
- Hit Ctrl+C to stop the script and return to the command line

Read a Button

- Here's the code:

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setup(24, GPIO.IN)
count = 0
while True:
    inputValue = GPIO.input(24)
    if (inputValue == True):
        count = count + 1
        print("Button pressed " + str(count) + " times.")
        time.sleep(.3)          #Wait for the user
    time.sleep(.01)             #So the CPU is not at 100%
```

YOUR TURN

Project

- Connect 3 LEDs and 2 buttons.

B2	B1	LED3	LED2	LED1
0	0	0	0	0
0	1	0	0	1
1	0	0	1	0
1	1	1	0	0