

# My first Raspberry Pi hands-on session

---

Paolo Burgio  
paolo.burgio@unimore.it



**UNIMORE**  
UNIVERSITÀ DEGLI STUDI DI  
MODENA E REGGIO EMILIA

High Performance  
Real Time **Lab**

“

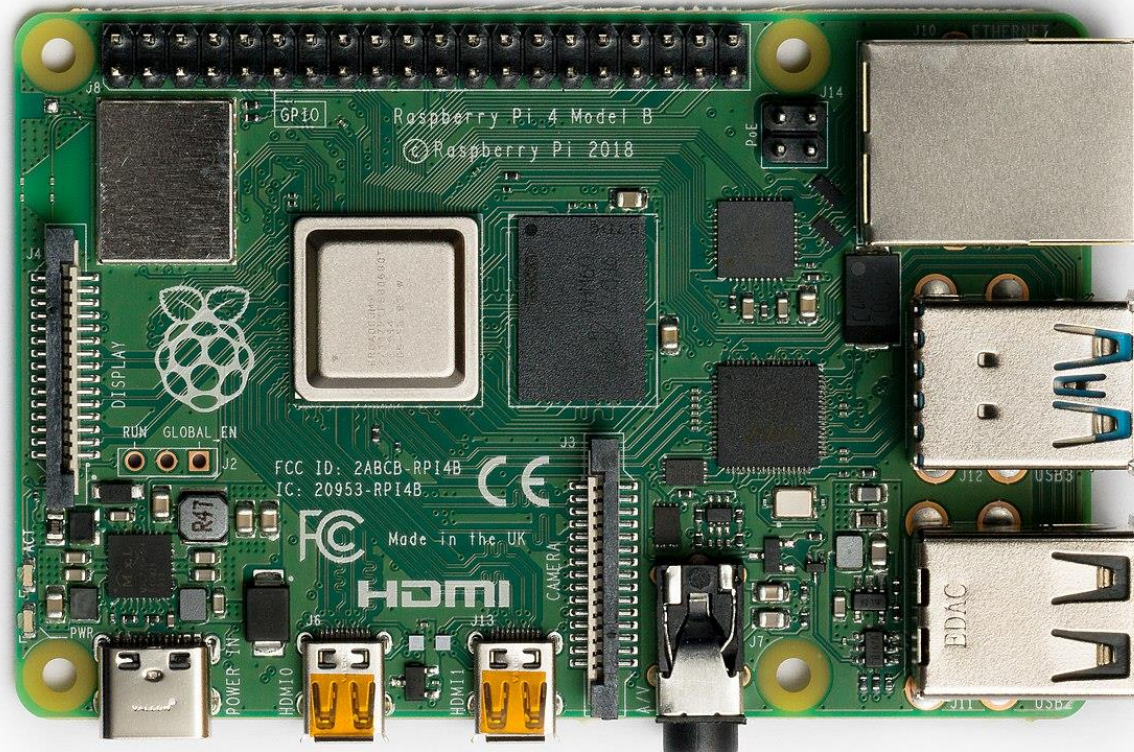
Programming is a skill  
best acquired by practice  
and example rather than  
from books.

ALAN TURING



# Our guy (Pi4)

---



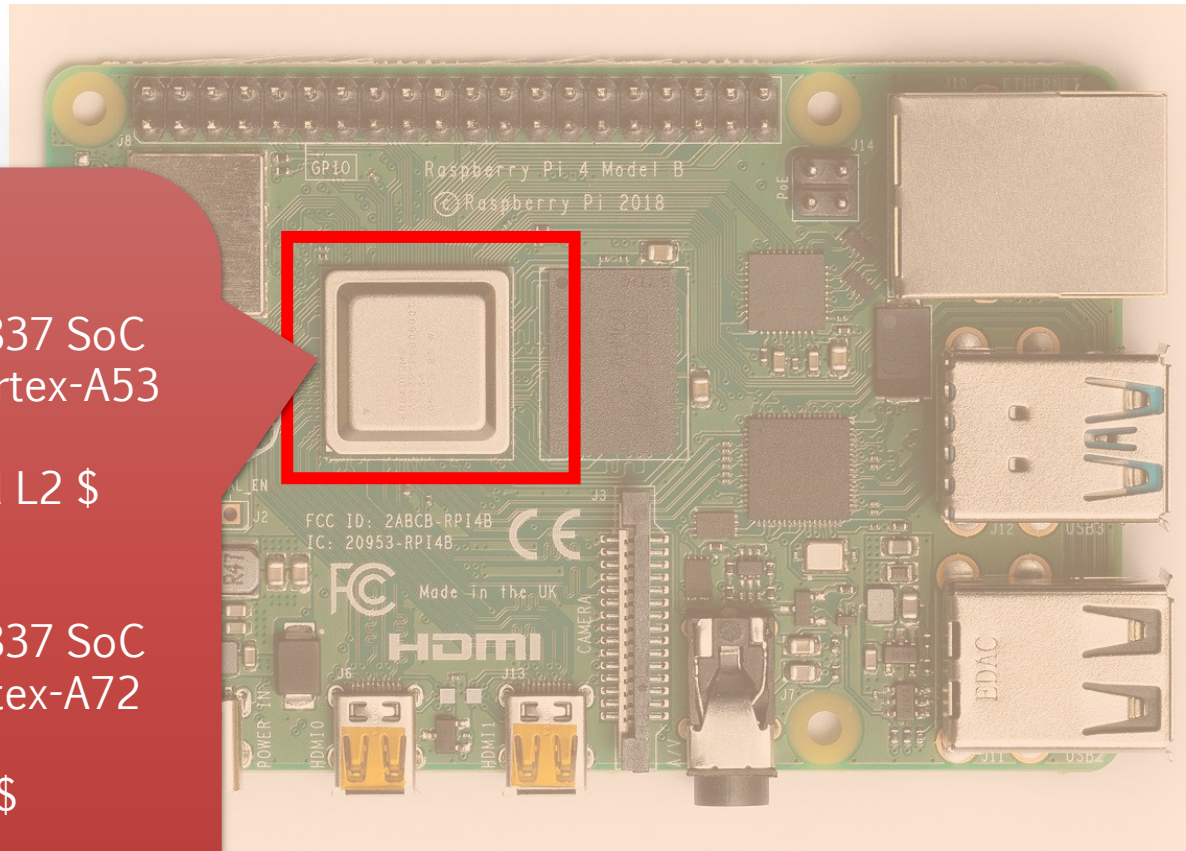




## Broadcom BCM2837 SoC

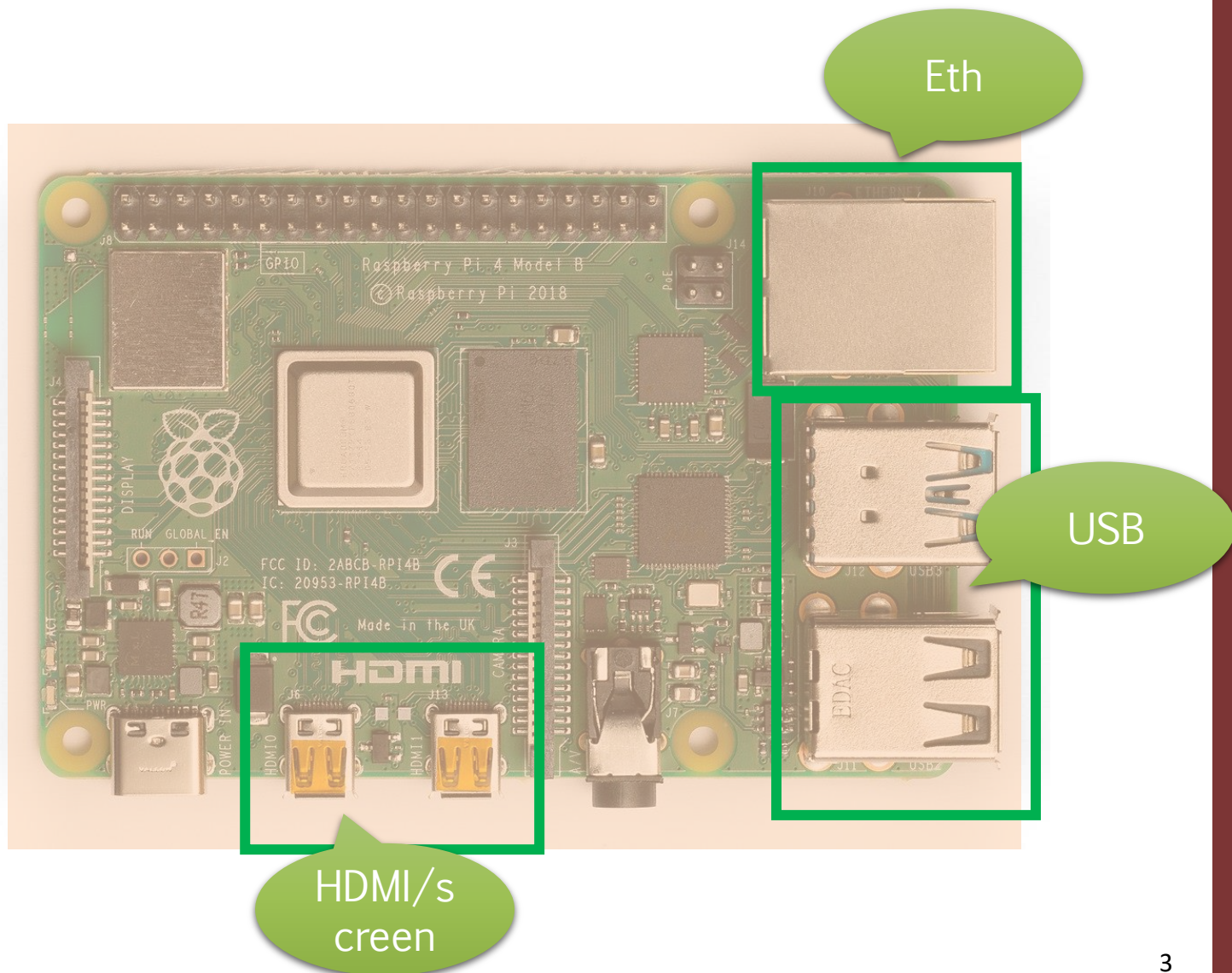
- 64-bit ARM Cortex-A53
- @1.2 GHz
- 512 KiB shared L2 \$

- Broadcom BCM2837 SoC
  - 64-bit ARM Cortex-A72
  - @1.5 GHz
  - 1MiB shared L2 \$





# Our guy (Pi4)

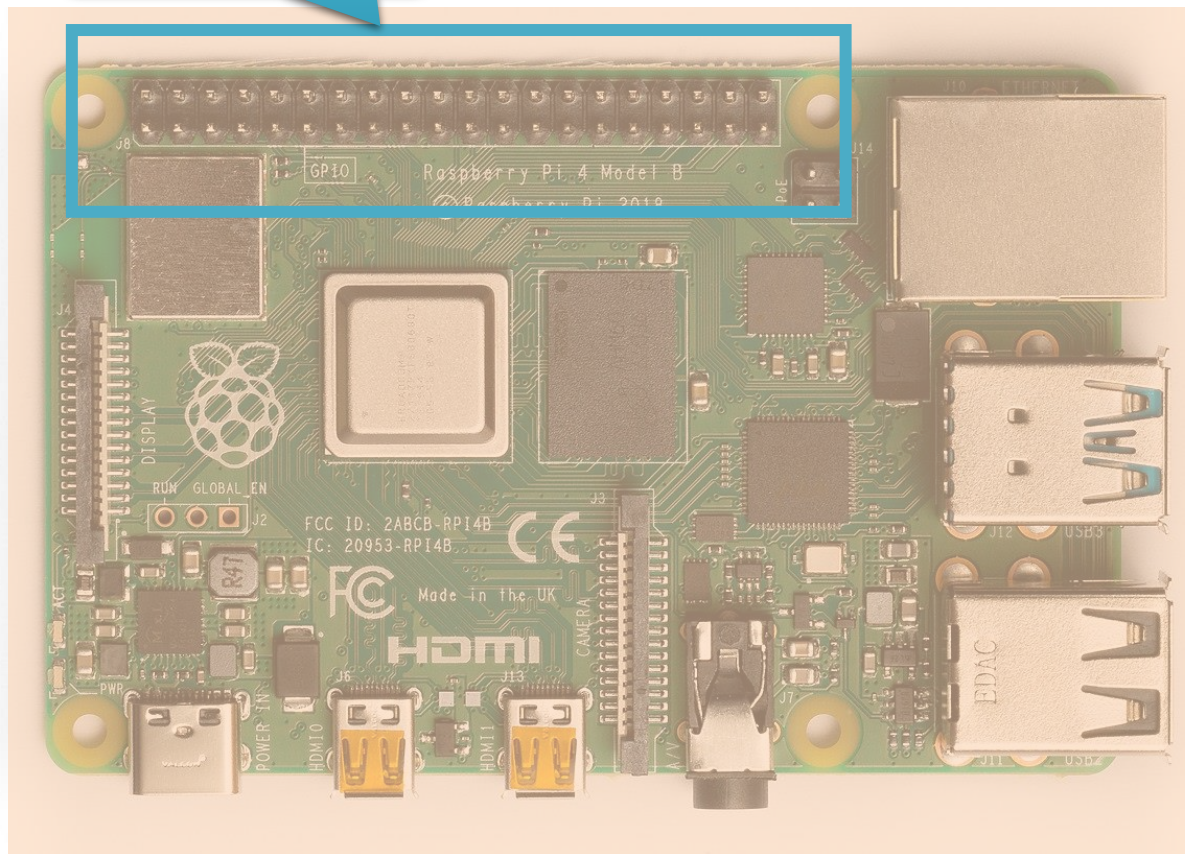






# Our guy (Pi4)

General Purpose  
I/O ports (GPIO)

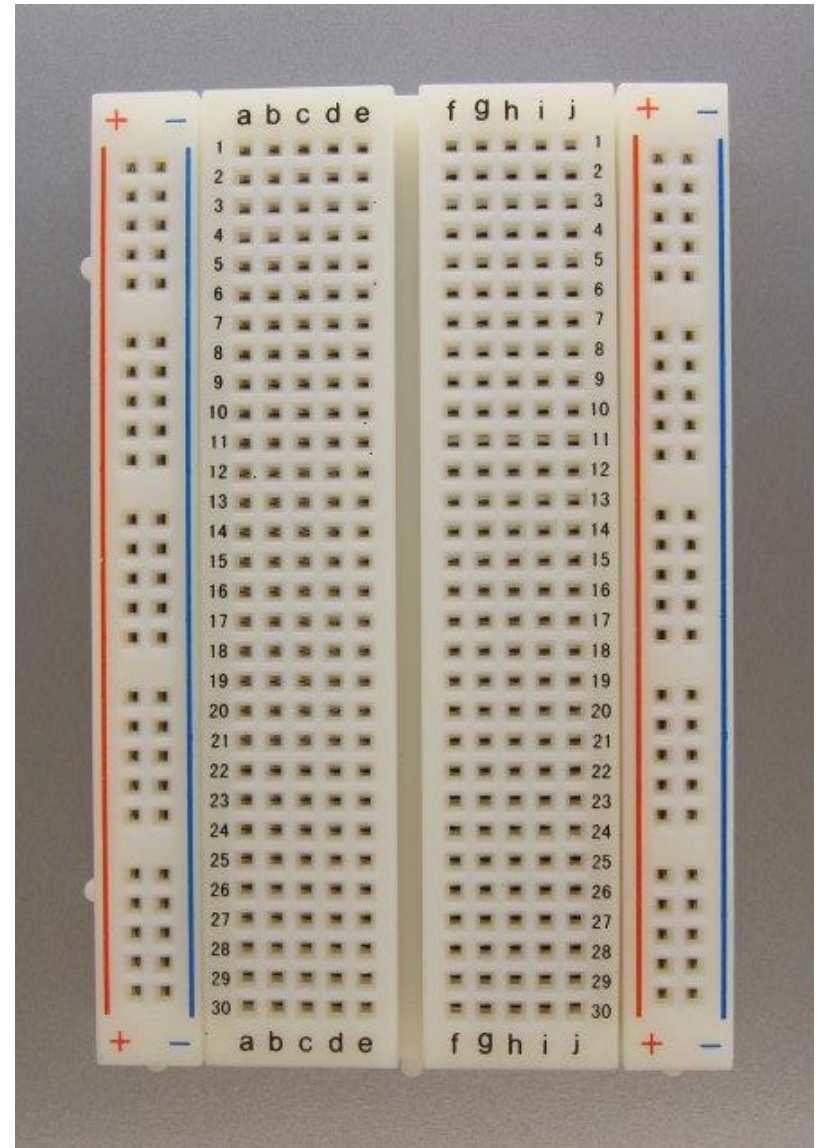




# Breadboard

Provides electrical connectivity

- › Vertical vs. horizontal rails
- › (Typically, power vs other)
- › Can use jumper wires

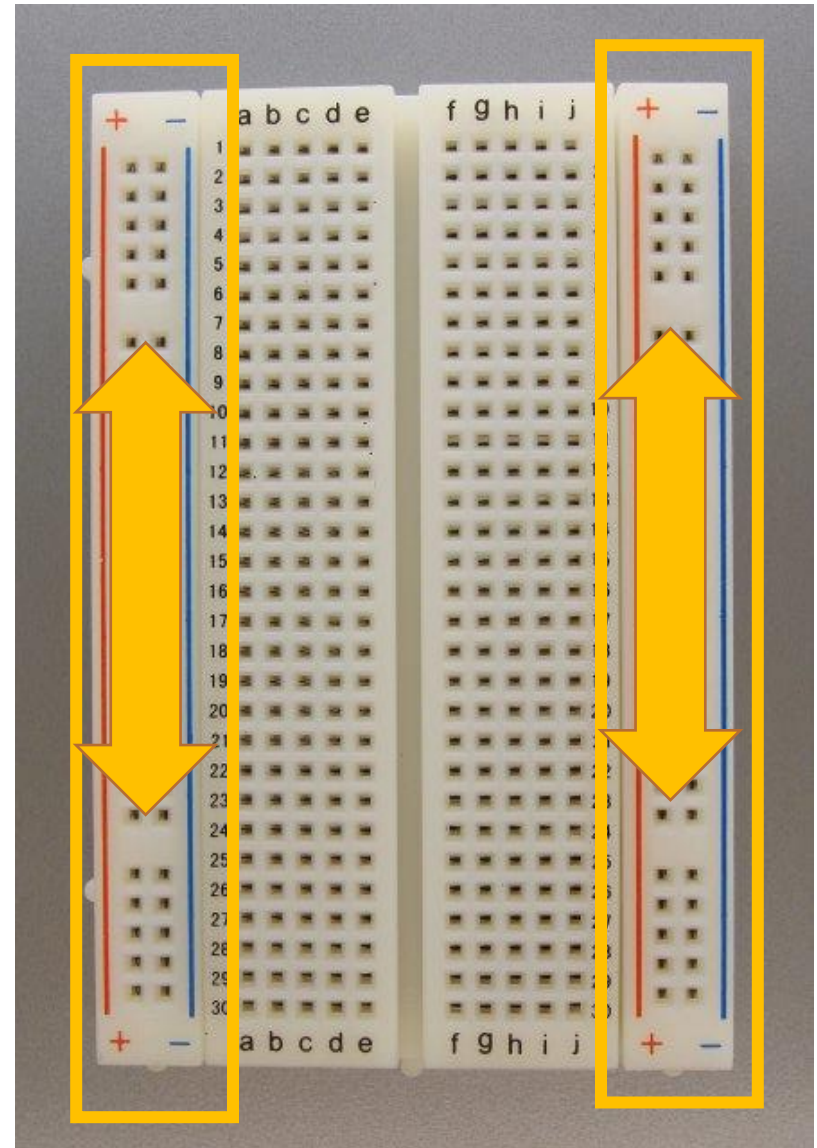




# Breadboard

Provides electrical connectivity

- › Vertical vs. horizontal rails
- › (Typically, power vs other)
- › Can use jumper wires



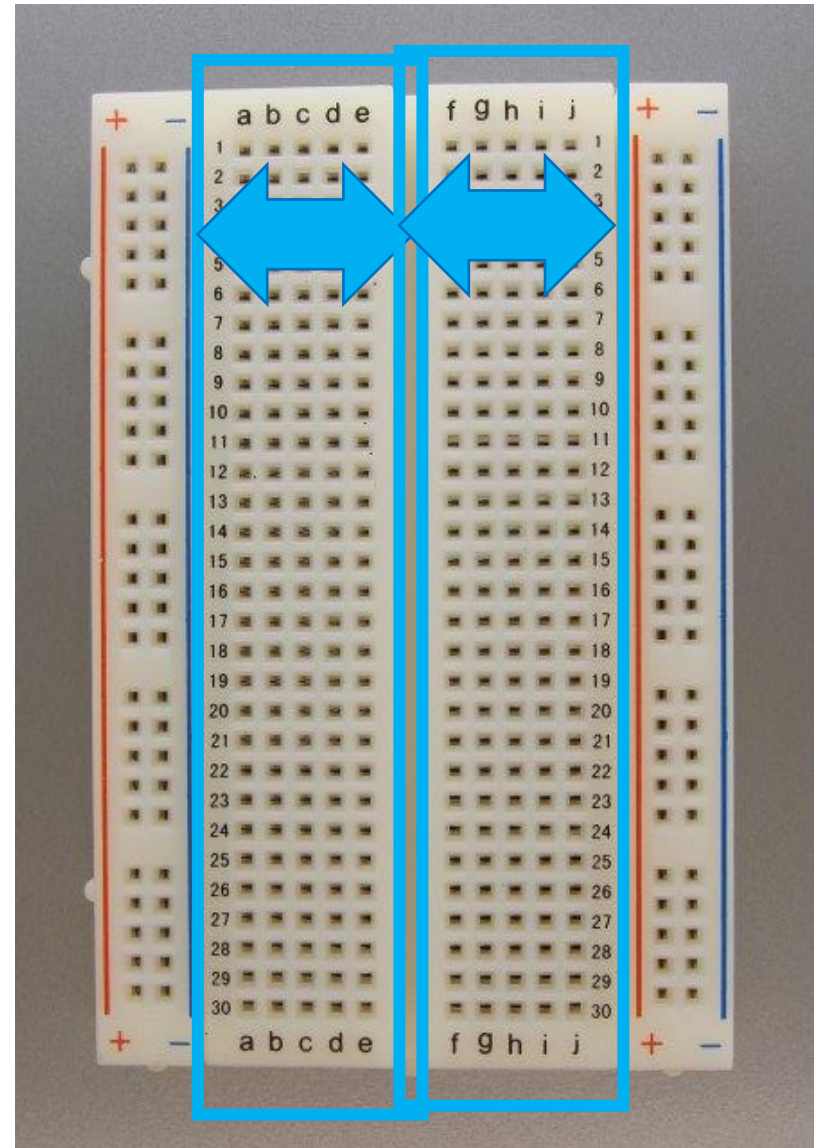




# Breadboard

Provides electrical connectivity

- › Vertical vs. horizontal rails
- › (Typically, power vs other)
- › Can use jumper wires





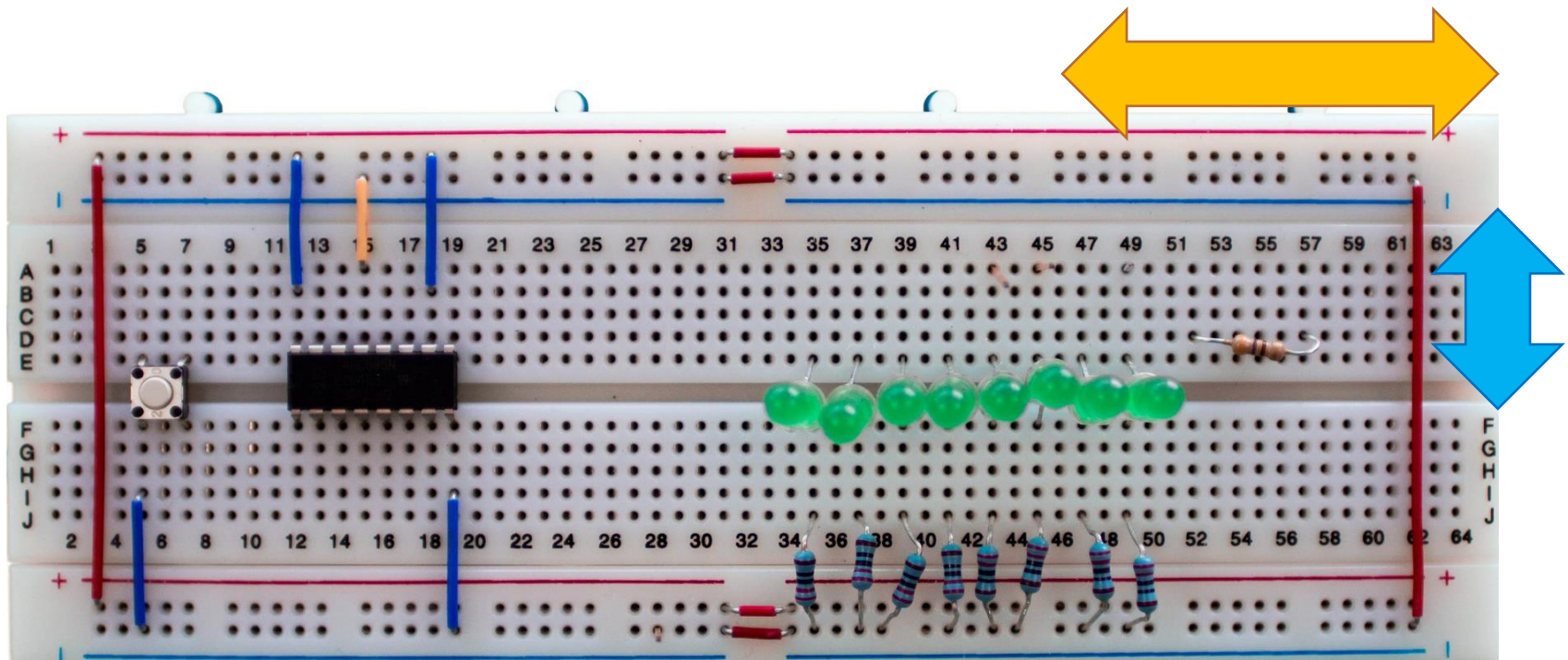
# Breadboard

The two sides of the + and - rails are wired together

- › Typically, used for power/GND

Brought to the internal rails with jumper wires

- › Where core/chip and other stuff reside





# Finally...LEDs

---

## Light Emitting Diodes

- › You feed with electrons; they light up
- › They have a side!!!!
- › They need a resistance to lower the charge



Wrong wiring => you burn them...





# Finally...LEDs

## Light Emitting Diodes

- › You feed with electrons; they light up
- › They have a side!!!!
- › They need a resistance to lower the charge

Wrong wiring => you burn them...



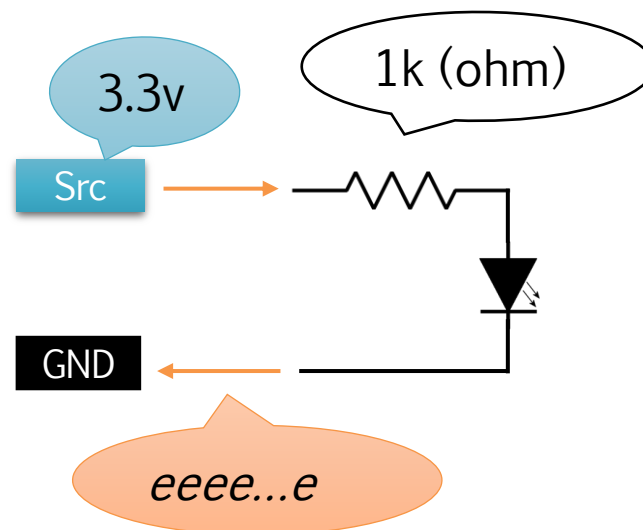


# Finally...LEDs

## Light Emitting Diodes

- › You feed with electrons; they light up
- › They have a side!!!!
- › They need a resistance to lower the charge

Wrong wiring => you burn them...



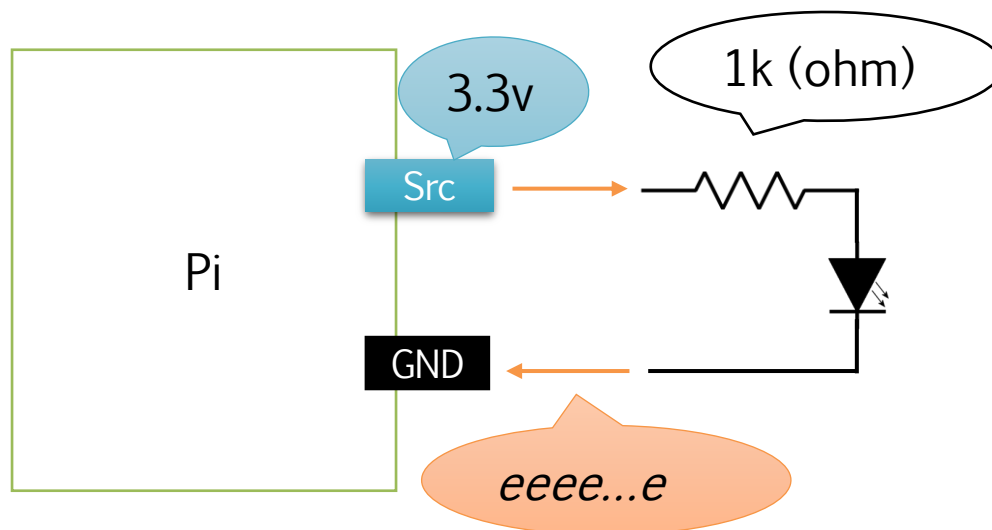


# Finally...LEDs

## Light Emitting Diodes

- › You feed with electrons; they light up
- › They have a side!!!!
- › They need a resistance to lower the charge

Wrong wiring => you burn them...







# General Purpose I/O Ports

Our interface towards the external world

- › <https://pinout.xyz/pinout/#>
- › BCM vs. Standard Wiring



3v3 Power	1		2	5v Power
GPIO 2 (I2C1 SDA)	3		4	5v Power
GPIO 3 (I2C1 SCL)	5		6	Ground
GPIO 4 (GPCLK0)	7		8	GPIO 14 (UART TX)
Ground	9		10	GPIO 15 (UART RX)
GPIO 17	11		12	GPIO 18 (PCM CLK)
GPIO 27	13		14	Ground
GPIO 22	15		16	GPIO 23
3v3 Power	17		18	GPIO 24
GPIO 10 (SPI0 MOSI)	19		20	Ground
GPIO 9 (SPI0 MISO)	21		22	GPIO 25
GPIO 11 (SPI0 SCLK)	23		24	GPIO 8 (SPI0 CE0)
Ground	25		26	GPIO 7 (SPI0 CE1)
GPIO 0 (EEPROM SDA)	27		28	GPIO 1 (EEPROM SCL)
GPIO 5	29		30	Ground
GPIO 6	31		32	GPIO 12 (PWM0)
GPIO 13 (PWM1)	33		34	Ground
GPIO 19 (PCM FS)	35		36	GPIO 16
GPIO 26	37		38	GPIO 20 (PCM DIN)
Ground	39		40	GPIO 21 (PCM DOUT)



# Software

---

## Operative system

- › Debian-based GNU/Linux Distro called **Raspberry Pi OS**
  - Aka *Raspbian*
- › Also Ubuntu and Win10 IoT are supported
- › (and many more...)

## (A number of) dev environments

- › **Standard GCC toolchain**
- › Arduino IDE (micro-kernel)
- › Google's TensorFlow for AI ;)
- › ....



# WiringPi

- › Library to interact with I/O
- › Uses “progressive” wiring



Raspberry Pi GPIO Header

BCM	WiringPi	Name	Physical	Name	WiringPi	BCM
		3.3v	1	2 5v		
2	8	SDA.1	3	4 5V		
3	9	SCL.1	5	6 0v		
4	7	1-Wire	7	8 TxD	15	14
		0v	9	10 RxD	16	15
17	0	GPIO. 0	11	12 GPIO. 1	1	18
27	2	GPIO. 2	13	14 0v		
22	3	GPIO. 3	15	16 GPIO. 4	4	23
		3.3v	17	18 GPIO. 5	5	24
10	12	MOSI	19	20 0v		
9	13	MISO	21	22 GPIO. 6	6	25
11	14	SCLK	23	24 CE0	10	8
		0v	25	26 CE1	11	7
0	30	SDA.0	27	28 SCL.0	31	1
5	21	GPIO.21	29	30 0v		
6	22	GPIO.22	31	32 GPIO.26	26	12
13	23	GPIO.23	33	34 0v		
19	24	GPIO.24	35	36 GPIO.27	27	16
26	25	GPIO.25	37	38 GPIO.28	28	20
		0v	39	40 GPIO.29	29	21
BCM	WiringPi	Name	Physical	Name	WiringPi	BCM





# WiringPi API

---

Include library header

```
#include <wiringPi.h>
```

(In desktop environments doesn't exist, so you shall use macro to remove this code, e.g., NO\_PI)

Init library, and **every** GPio port

```
wiringPiSetup(); // Init lib  
pinMode(0, OUTPUT); // GPio 0 is output port
```

Write to port

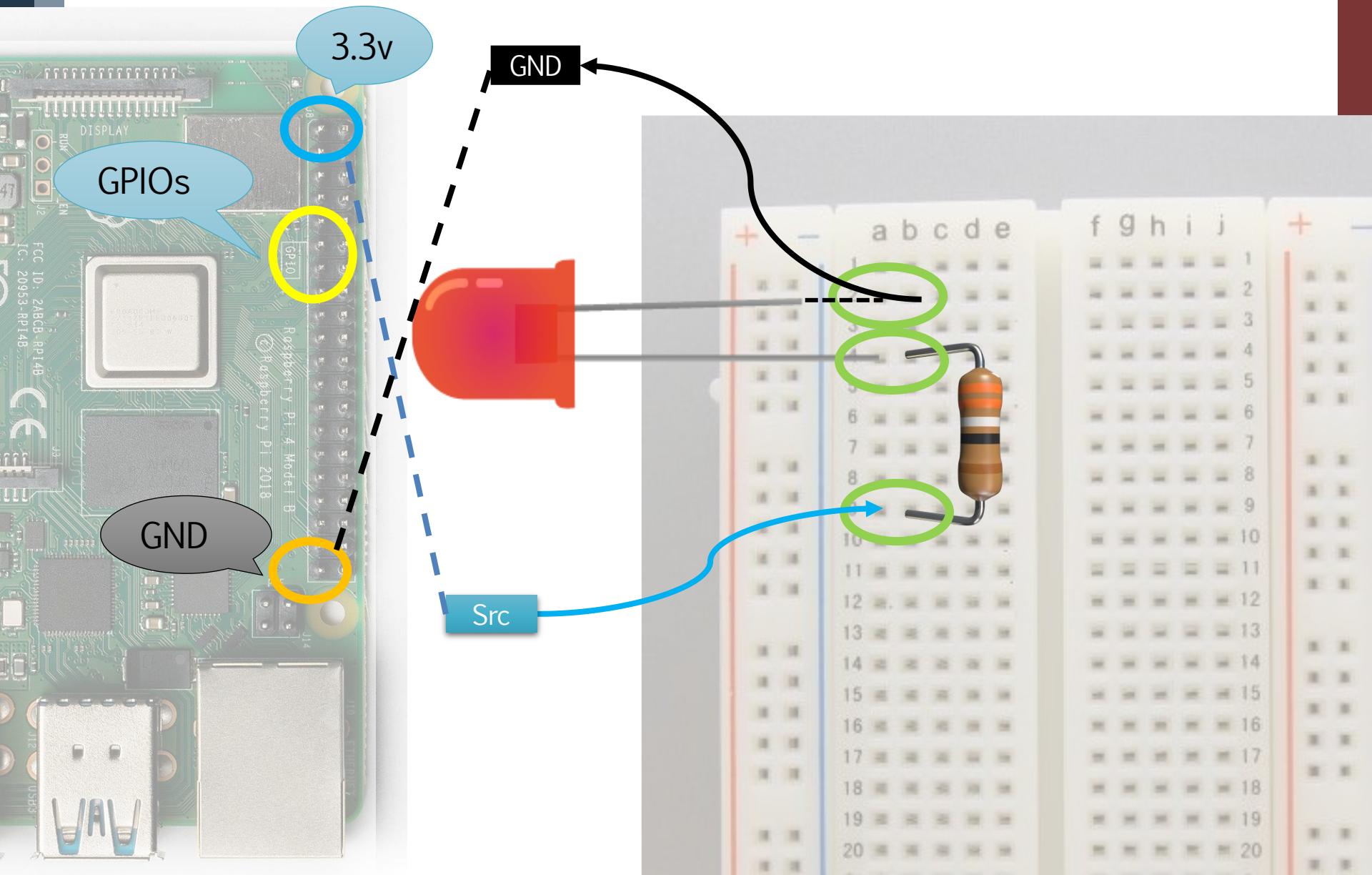
```
digitalWrite(0, true); // Set port 0
```

Link library

```
$ gcc ..... -l wiringPi
```



# E/E system





# Exercise

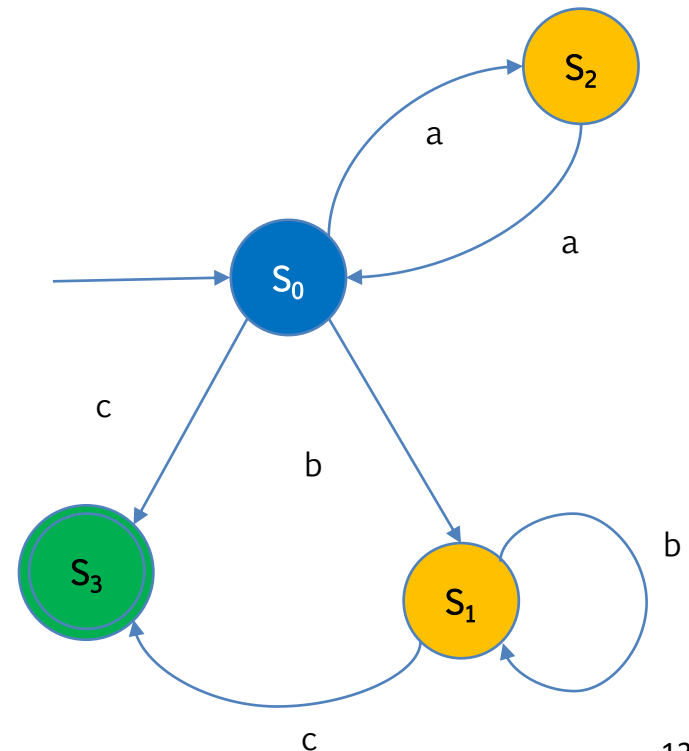
Let's  
code!

- › Implement the Moore machine of the FSM that understands whether a words is from L

*“Identify even sequences of a (even empty),  
followed by one, or more, or no, b, ended by c”*

- › ..and turns on the corresponding led color

- **Blue** => GPIO 0
- **Red (error state)** => GPIO 1
- **Yellow** => GPIO 2
- **Green** => GPIO 3







# How to run the examples

---

Let's  
code!

- › Find them in Code/ folder from the course website

For C++: compile

```
› $ gcc code.cpp [-DNO_PI] -o code -Wall -l stdc++ -l wiringPi
```

Run (Unix/Linux)

```
$ ./code
```

Run (Win/Cygwin)

```
$ ./code.exe
```



# References

---



## Course website

- › [http://hipert.unimore.it/people/paolob/pub/Industrial Informatics/index.html](http://hipert.unimore.it/people/paolob/pub/Industrial_Informatics/index.html)

## My contacts

- › [paolo.burgio@unimore.it](mailto:paolo.burgio@unimore.it)
- › <http://hipert.mat.unimore.it/people/paolob/>

## Resources

- › <https://www.digikey.com/en/maker/blogs/2019/how-to-use-gpio-on-the-raspberry-pi-with-c>
- › <http://maxembedded.com/2014/07/using-raspberry-pi-gpio-using-python/>
- › <https://www.digikey.com/en/resources/conversion-calculators/conversion-calculator-resistor-color-code>
- › [https://pinout.xyz/pinout/pin11\\_gpio17#](https://pinout.xyz/pinout/pin11_gpio17#)
- › <http://wiringpi.com/>
- › A "small blog "
  - <http://www.google.com>