Horizons Class 3

We have covered :

- Basics of Raspberry Pi
- Using Python
- How to use GPIO pins
- How to create a circuit on a breadboard
- How to control GPIO pins with Python code
- How to use LEDs, a button

GPIO review

Pin number is confusing

- Familiarise your self
- Choose your method

Pins can be:

- Input 0.5mA
- Output 16ma per pin , 50ma total
- 3V only

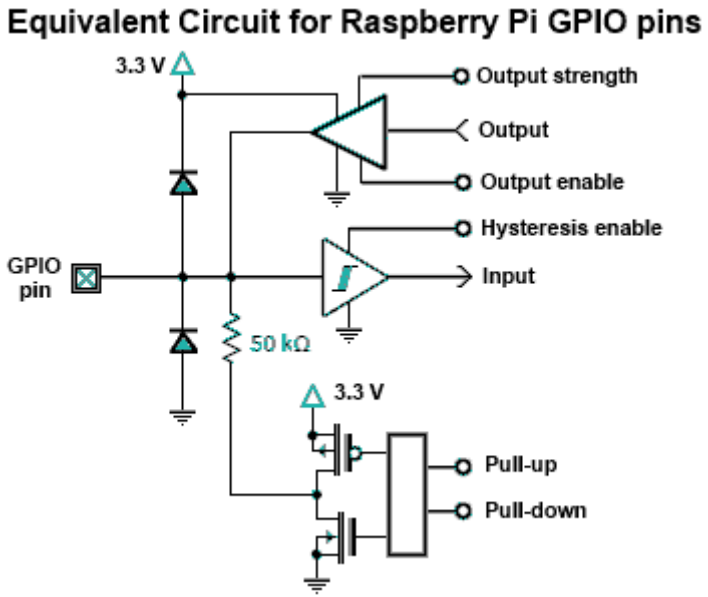
Do not connect a 5V to any GPIO pin

I/O

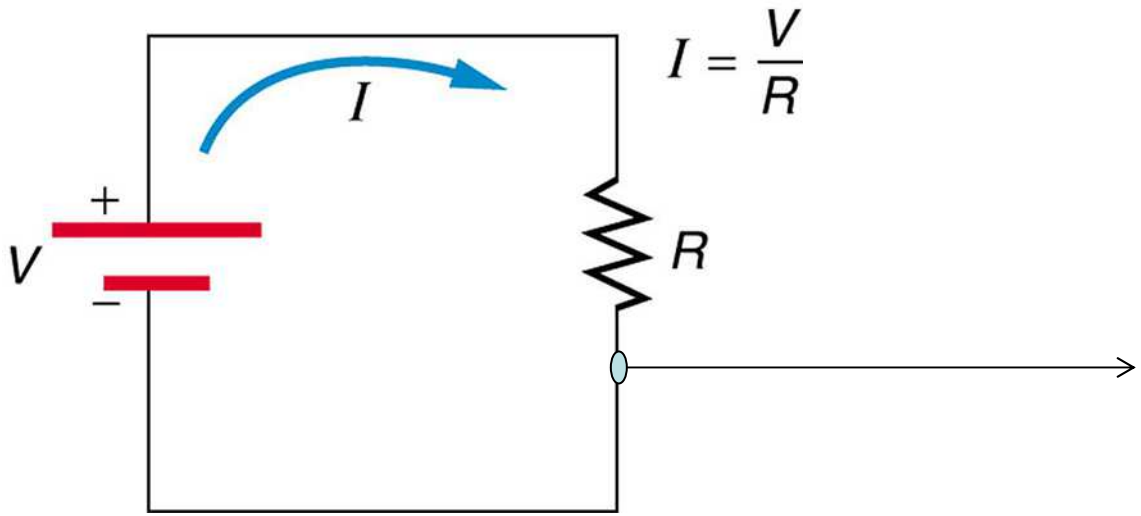
- The Raspberry Pi's GPIO pins are quite versatile, and you can modify many of their characteristics from software. You can turn on/off input pin hysteresis, limit output slew rate, and control source and sink current drive capability from 2 mA to 16 mA in 2 mA increments. These properties are set for the GPIO block as a whole, not on a pin-by-pin basis.

- Source/sink current capability does not limit the current into or out of the pin, but only specifies the maximum current for which the output signal high/low voltage specifications will be met. If misused, output pins can be damaged by excessive current irrespective of the source/sink current programmed. After a reset, the RPi comes up with the GPIO outputs set to 8 mA drive capability
- http://www.mosaic-industries.com/embedded-systems/microcontroller-projects/raspberry-pi/gpio-pin-electrical-specifications

Logic behind each i/o pin



Logical equivalent view of each pin



Or the reverse, you supply the 3 V at 0.5ma and it goes into this pin

To do anything meaningful

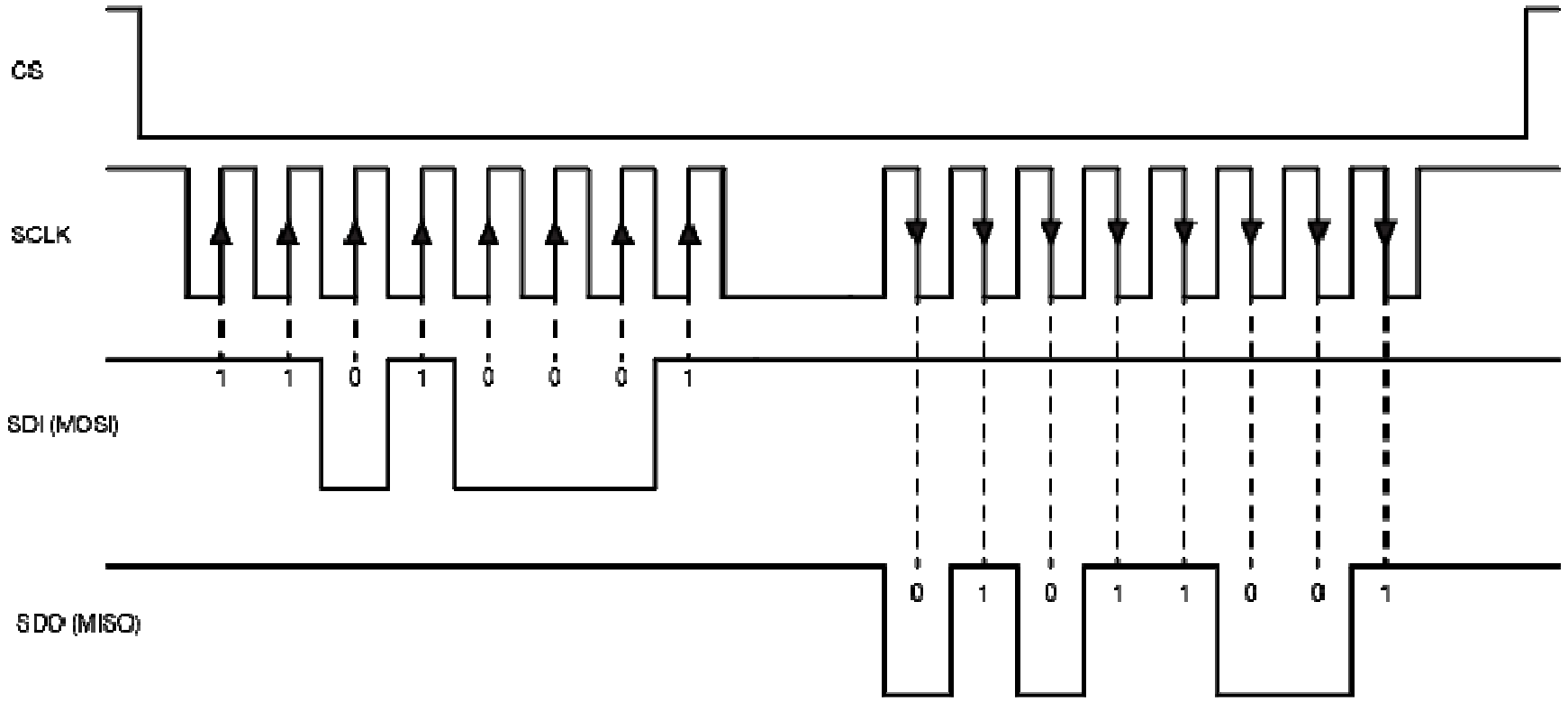
- You have to connect the pin to a driver or amplifier
- Motor take 150 ma to several amps
- Led type device or display is the only thing you can connect directly and easily to the pi as output.
- So that means you can have an external power supply that can feed your motor and the pi only operates the switch that enables the motor

Input /output

- Digital outputs: turn lights, motors, or other devices on or off
- Digital inputs: read an on or off state from a button, switch, or other sensor
- Communication with chips or modules using low-level protocols: SPI, I^2C, or serial UART
- Connections are made using GPIO ("General Purpose Input/Output") pins. Unlike USB, etc., these interfaces are not "plug and play" and require care to avoid miswiring. The Raspberry PI GPIOs use 3.3V logic levels, and can be damaged if connected directly to 5V levels (as found in many older digital systems) without level-conversion circuitry.

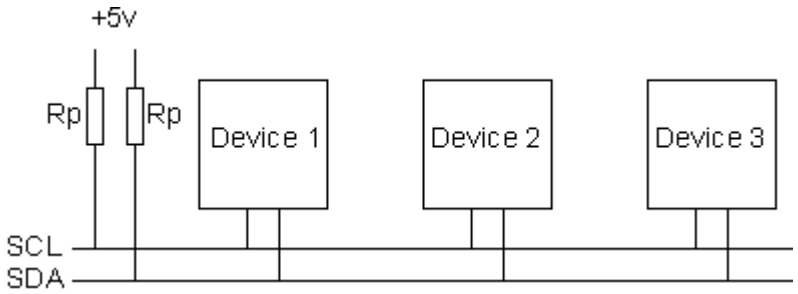
- Note that **no analogue** input or output is available. However, add-on boards such as the Rpi Gertboard provide this capability.

Digital data



Serial /Parallel/I2c

- You can have one line transferring data
- You can use 4 in parallel
- I2C Here there is a chain with each listened having an address eg LED string



Lots of sensors

- Acceleration
- Humidity
- Magentism
- Infra red
- Sound
- Uv
- Heart rate
- Colour
- Oxygen

``

You can find a ready made sensor to take your project forward

Lots of example code also available

Example of a device

Oxygen and pulse meter

- It can display data
- It can also send it

How was this first made?

Analogue circuits

- Most sensors are analogue
- Thermister; LDR; Variable resistor
- Lot of sensors use change of resistance as fundmental property
- So have to convert to digital or improvise

References

- https://www.raspberrypi.org/learning/physical-computing-with-python/analogue/
- http://www.mosaic-industries.com/embedded-systems/microcontroller-projects/raspberry-pi/gpio-pin-electrical-specifications
- http://elinux.org/RPi_Low-level_peripherals