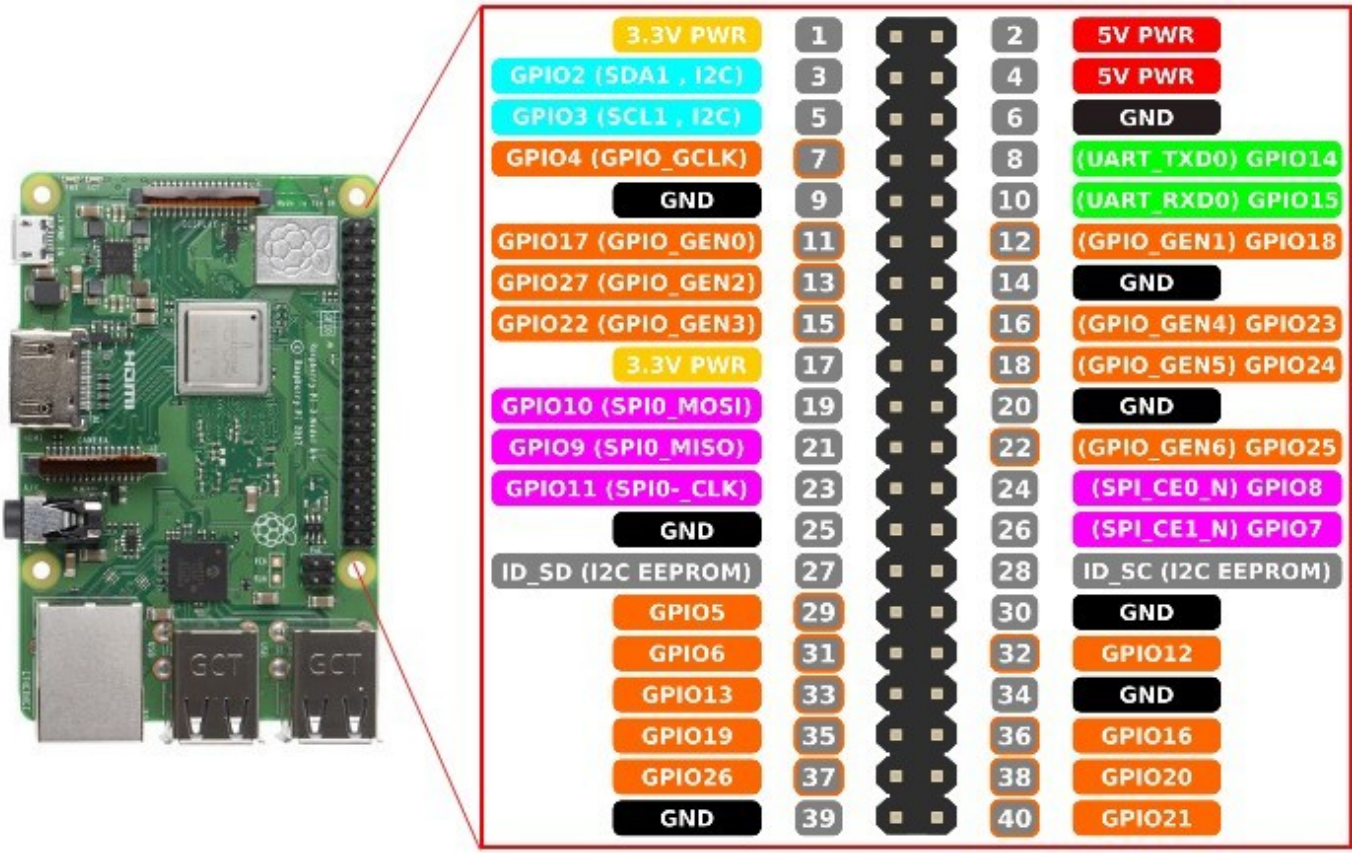
RASPBERRY GPIO

Una delle caratteristiche che rendono la Raspberry molto versatile ed utile per tante applicazioni, è la possibilità di gestire delle GPIO (**G**eneral **P**urpose **I**nput ed **O**utput) cioè dei terminali che possono essere utilizzati come ingressi o uscite.



Sul connettore della Raspberry troviamo tutti i terminali che consentono alla scheda di collegarsi direttamente con dispositivi esterni. Molti di questi hanno una duplice funzione, come ad esempio i verdi (GPIO e comunicazione RS232) i fucsia (GPIO e comunicazione SPI) e gli azzurri (GPIO e comunicazione I2C). I restanti terminali indicati, con il colore arancione, hanno l'unica funzione di GPIO cioè possono essere INPUT o OUTPUT a seconda dell'impostazione che verrà assegnata dal software.

Perciò oltre che dialogare con i 3 protocolli di comunicazione indicati sopra, potremo gestire direttamente l'accensione o lo spegnimento un dispositivo di uscita, ovviamente digitale cioè che può assumere solo due stati 0 e 1, OFF ed ON (come ad esempio un led) o potremo leggere lo stato di un segnale digitale, cioè un ingresso che può valere anch'esso 0 o 1, OFF o ON, come ad esempio un pulsante.

Anche se può sembrare una cosa semplice (ed in effetti lo è) non è questa una possibiità che altri SBC (Single Board Computer) non hanno, come sicuramente non hanno i normali PC.

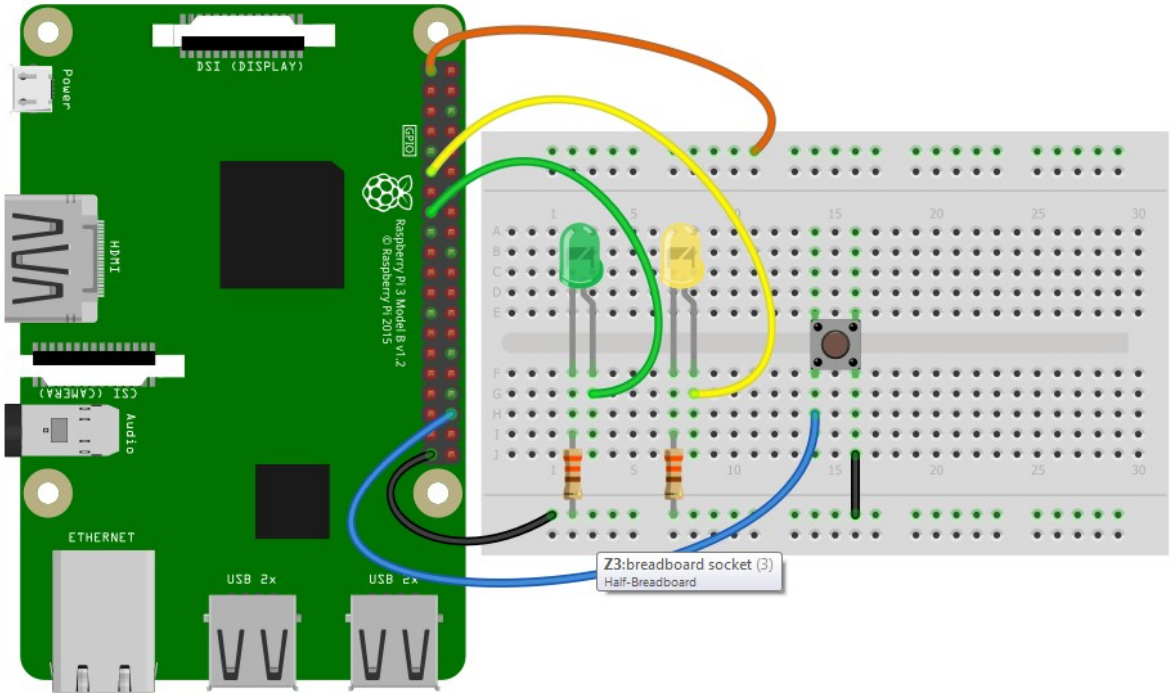
La possibilità di interfacciarsi con altri dispositivi o di gestire direttamente dispositivi di ingresso ed uscita, rendono la scheda Raspberry estremamente versatile, e la rendono utilizzabile oltre che come un normale PC, come un PLC (Programmable Logic Controller cioè quei dispositivi programmabili utilizzati nell'ambito dell'automazione industriale). E' importante ricordare che tutte le GPIO funzionano a 3,3 Volt, e per evitare la rottura di tutta la scheda bisogna adattare il livello di tensione ed evitare di collegarle direttamente a tensioni più elevate.

Dobbiamo inoltre ricordare che:

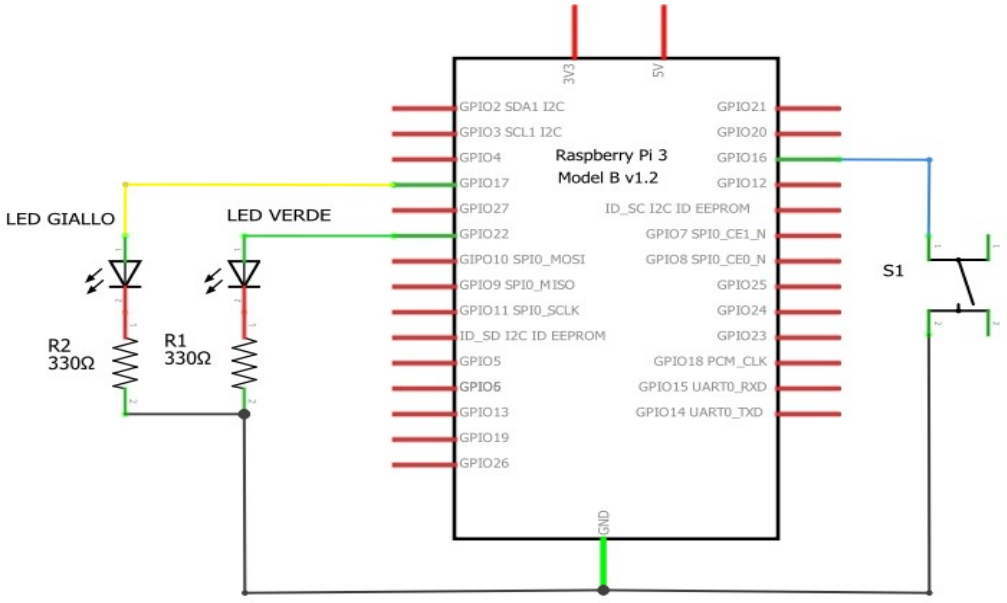
- I pin 5V, possono fornire la corrente dell'alimentatore meno quella consumata dalla scheda.
- I pin di alimentazione a 3,3V possono erogare al massimo nel loro complesso 50mA.
- I pin di Massa sono collegati tutti insieme.

E' consuetudine, quando si fanno i primi passi in un ambiente di programmazione, realizzare il classico esempio per scrivere a video "Hello World!". Nel nostro caso (o come avviene con le schede a microcontrollore come Arduino) il primo esempio è il classico LED lampeggiante. Noi realizzeremo invece uno schema leggermente più complesso con due LED (2 output) ed un pulsante (1 input).

Il primo passo sarà quello di realizzare il seguente schema di montaggio, utilizzando una breadboard, e dei componenti, 2 LED colorati (**L**ight **E**mitter **D**iode, cioè diodo emettitore di luce) un pulsante e 2 resistenze da 330Ohm.



Seguendo lo schema di montaggio della figura sopra, si realizzerà di fatto il seguente collegamento:



Vedremo ora come gestire le GPIO utilizzando il linguaggio Python ed il linguaggio C, dato che nel S.O. Raspbian, risulta presente il classico gcc, compilatore per programmi scritti in linguaggio C, ed un interprete Python, in grado di eseguire programmi scritti con questo potente linguaggio.

UTILIZZO DELLE GPIO CON LINGUAGGIO PYTHON – LIBRERIA Rpi.GPIO

https://sourceforge.net/p/raspberry-gpio-python/wiki/Home/

Ci sono diverse librerie che ci consentono di utilizzare le GPIO, con Python noi proveremo in due modi differenti. Il primo metodo utilizza la llibreria **Rpi.GPIO.**

Per installare questa libreria dobbiamo effettuare i seguenti passi da terminale:

- Installare il gestore dei pacchetti Python con il comando ***sudo apt-get install python-pip***
- Installare gli strumenti di sviluppo di Python 3 con il comando ***sudo apt-get install python3-dev***
- Installare il pacchetto ***sudo pip install distribute***
- Installare il pacchetto ***sudo pip install RPi.GPIO***

Successivamente aprire l'editor da terminale con il comando ***sudo nano GPIO.py,*** e scrivere il seguente programma. Esso farà lampeggiare uno dei due led a seconda dello stato del pulsante (premuto o rilasciato) per avviarlo occorre digitare il comando ***python GPIO.py***.

Il funzionamento delle funzioni di questa libreria è molto simile a quello delle librerie di Arduino. In pratica con la funzione ***GPIO.setup***, si definisce se il GPIO è input o output, nel primo caso (input) si può attivare la resistenza di ***pull_up*** (resistenza da 10k connessa tra 3,3V e l'ingresso) o la resistenza di ***pull_down*** (resistenza da 10k connessa tra GND e l'ingresso). Se un terminale viene configurato come uscita, potrà invece successivamente portato a livello alto o basso con la funzione ***GPIO.output***, dove il livello può essere indicato con il valore numerico 1 o 0 o con le diciture ***GPIO.HIGH*** o ***GPIO.LOW***.

```
import RPi.GPIO as GPIO
from time import sleep

LedGiallo=17
LedVerde=22
Pulsante=16

#Modalita' di indicazione del led (numerazione GPIO)
GPIO.setmode(GPIO.BCM)

#Seleziono gli ingressi e le uscite
GPIO.setup(LedGiallo, GPIO.OUT)  #uscita digitale
GPIO.setup(LedVerde, GPIO.OUT)   #uscita digitale
GPIO.setup(Pulsante, GPIO.IN, pull_up_down = GPIO.PUD_UP)    #ingresso con resistenza di pull_up
#GPIO.setup(Pulsante, GPIO.IN, pull_up_down = GPIO.PUD_DOWN)#ingresso con resistenza di pull_down

while True:
    if GPIO.input(Pulsante):            #test del pulsante
        GPIO.output(LedGiallo,GPIO.HIGH)#mette l'uscita a livello alto
        #GPIO.output(LedGiallo,1)       #mette l'uscita a livello alto
    else:
        GPIO.output(LedVerde,GPIO.HIGH) #mette l'uscita a livello alto
        #GPIO.output(LedVerde,1)        #mette l'uscita a livello alto
    sleep(1)
    GPIO.output(LedGiallo,GPIO.LOW)     #mette l'uscita a livello basso
    GPIO.output(LedVerde,GPIO.LOW)      #mette l'uscita a livello basso
    #GPIO.output(LedGiallo,0)           #mette l'uscita a livello basso
    #GPIO.output(LedVerde,0)            #mette l'uscita a livello basso
    sleep(1)
```

UTILIZZO DELLE GPIO CON LINGUAGGIO PYTHON – LIBRERIA gpiozero

https://www.raspberrypi.org/documentation/usage/gpio/python/README.md

Un altra libreria che si può utilizzare è la **gpiozero**. Questa libreria è di norma già presente nella distro Raspbian, perciò può essere utilizzata già alla prima accensione della scheda. Qualora non fosse disponibile, la libreria è comunque installabile con il comando: ***sudo apt install python3-gpiozero.***

Lo stesso identico programma scritto precedentemente con la libreria ***RPi.GPIO***, diventa molto più semplice con la libreria ***gpiozero*** e cioè:

```
from gpiozero import LED, Button
from time import sleep

LedGiallo = LED(17)
LedVerde = LED(22)
Pulsante = Button(16)

while True:
    if Pulsante.is_pressed:
        LedGiallo.on()
    else:
        LedVerde.on()
    sleep(1)
    LedGiallo.off()
    LedVerde.off()
    sleep(1)
```

Anche in questo caso possiamo editare un programma da terminale con l'editor nano, ***sudo nano GPIO.py***, e lanciarlo successivamente con ***python GPIO.py.***

Questa libreria offre inoltre altre funzioni alternative, come ad esempio ***Pulsante.wait_for_press()*** e ***Pulsante.wait_for_release()***, o come ***Pulsante.when_pressed*** e ***Pulsante.when_released.***

Di seguito due esempi, tratti dal link ufficiale scritto sopra, tutta la documentazione ufficiale della libreria la si può trovare al seguente link: **https://gpiozero.readthedocs.io/en/stable**

In entrambi gli esempi il led si accende a seconda dello stato del pulsante.

```
from gpiozero import LED, Button

LedGiallo = LED(17)
Pulsante = Button(16)

while True:
    Pulsante.wait_for_press()
    LedGiallo.on()
    Pulsante.wait_for_release()
    LedGiallo.off()
```

```
from gpiozero import LED, Button

LedGiallo = LED(17)
Pulsante = Button(16)

while True:
    Pulsante.when_pressed = LedGiallo.on
    Pulsante.when_released = LedGiallo.off
```

N.B.

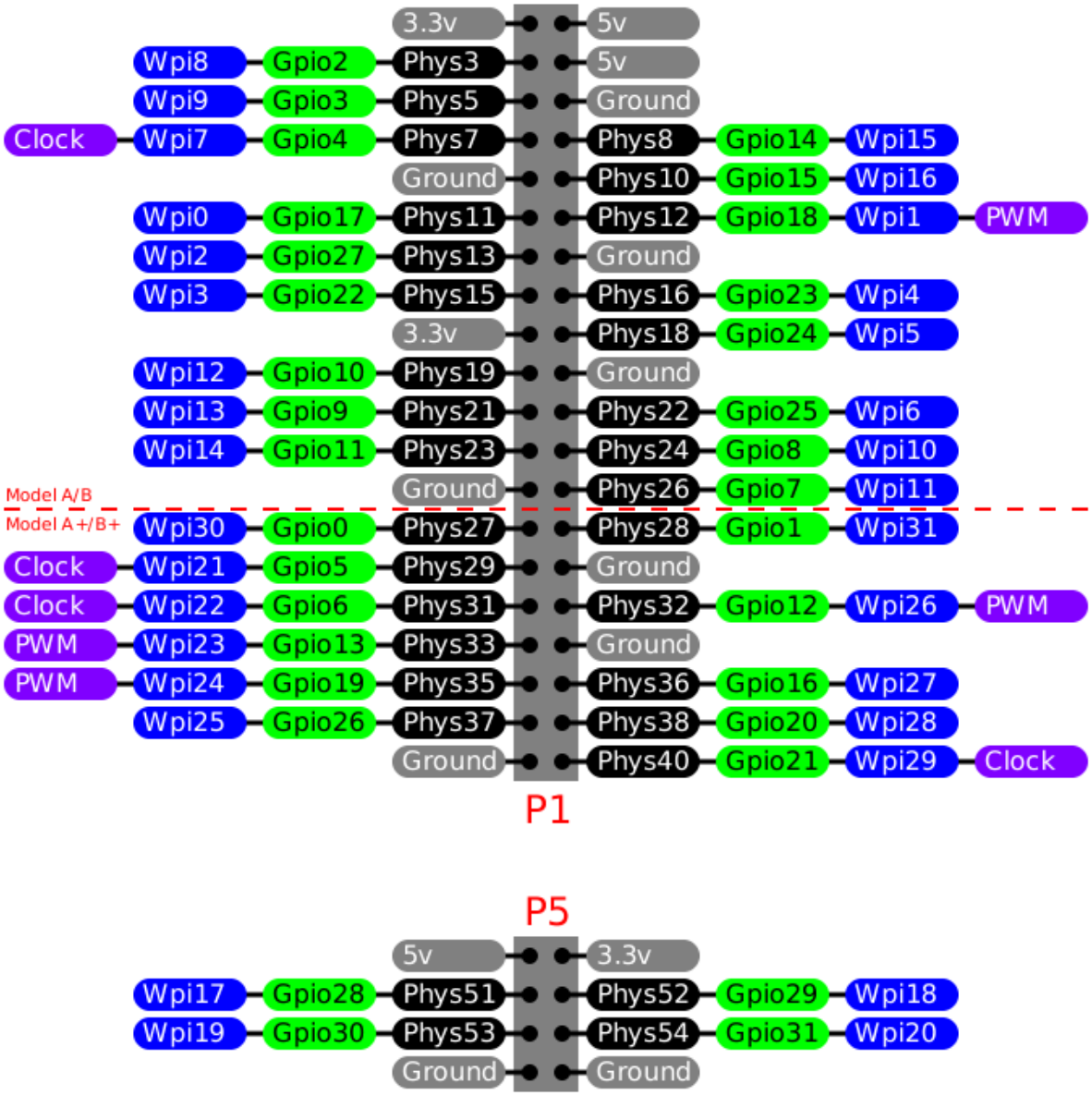
In tutti i casi sopra descritti bisogna fare attenzione al fatto che il pulsante è collegato verso GND, pertanto lo stato di pulsante premuto corrisponde ad un livello logico basso sull'ingresso. Ciò significa che quando scriviamo, ***if Pulsante.is_pressed,*** oppure ***Pulsante.wait_for_press()*** o anche ***Pulsante.when_pressed***, o ancora come nell'esempio precedente ***if GPIO.input(Pulsante),*** stiamo identificando **la condizione di pulsante non premuto.**

UTILIZZO DELLE GPIO CON LINGUAGGIO C – Libreria wiringpi

http://wiringpi.com/reference/ www.wiringpi.com

La libreria **wiringpi** è già presente nel sistema operativo Raspbian, ma qualora non lo fosse può essere installata seguendo attentamente le istruzioni al seguente link: **http://wiringpi.com/download-and-install/** .

Quando si utilizza la **wiringpi** bisogna fare attenzione al nome dei GPIO, nella figura presente vediamo in nero la numerazione fisica del connettore, in verde il numero del GPIO ed in blu il corrispondente numero della libreria wiringpi. Perciò se si vuol utilizzare, come nel nostro caso il GPIO17 ed il GPIO 22, dove sono collegati i due LED, bisogna fare riferimento al numero 0 e 3 (Wpi0 e Wpi3).



l programma di esempio che utilizza la libreria wiringpi in linguaggio C è il seguente:

```c
#include <wiringPi.h>

#define LedGiallo 0 //GPIO 17
#define LedVerde 3  //GPIO 22
#define Pulsante 27 //GPIO 16

int main (void)
 {
   //inizializzazione e definizione ingressi ed uscite
   wiringPiSetup () ;
   pinMode(LedGiallo, OUTPUT);
   pinMode(LedVerde, OUTPUT);
   pinMode(Pulsante, INPUT);
   //pullUpDnControl([pin], [PUD_OFF, PUD_DOWN, PUD_UP])
   pullUpDnControl(Pulsante,PUD_UP);  //attivo le resistenze di pull_up

   while(1)
    {
     if(digitalRead(Pulsante) == HIGH) digitalWrite(LedGiallo, HIGH);
     else                              digitalWrite(LedVerde, HIGH);
     delay(1000);
     digitalWrite(LedGiallo, LOW);
     digitalWrite(LedVerde, LOW);
     delay(1000);
    }
 return 0 ;
}
```

Possiamo scriverlo da terminale con il solito editor ***sudo nano gpio.c***, da compilare poi con il seguente comando, ***sudo gcc gpio.c -o gpio.out -lwringPi.***

Possiamo lanciare il programma da terminale con il comando ***./gpio.out.***

Anche in questo caso si avrà il lampeggio di uno dei due led a seconda dello stato del pulsante, che è collegato verso GND e pertanto necessità dell'attivazione delle resistenze di pull-up.

L'istruzione ***pullUpDnControl*** permette di attivare resistenze di pull-up con l'opzione PUD_UP, di pull-down con l'opzion PUD_DOWN, o di disattivarle entrambe con l'opzione PUD_OFF.

Un altro link dove trovare informazioni sulla wiringpi per linguaggio C è l seguente:
https://learn.sparkfun.com/tutorials/raspberry-gpio/c-wiringpi-api

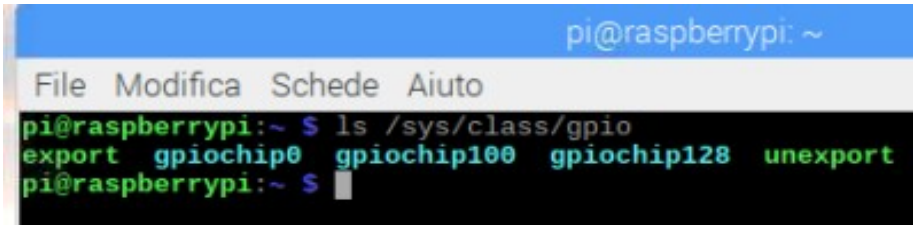
La libreria wiringpi esiste anche per Python, tutte le informazioni al seguente link:
https://raspi.tv/2013/how-to-use-wiringpi2-for-python-on-the-raspberry-pi-in-raspbian

UTILIZZO DELLE GPIO UTILIZZANDO I FILE DI SISTEMA

Fino ad ora abbiamo sempre utilizzato delle librerie per gestire le GPIO, ma possiamo anche fare a meno delle librerie ed utilizzare i file di sistema di RASPBIAN per attivare e disattivare uscite o leggere ingressi.
Questo perché nel sistema operativo Linux, tutti i dispositivi sono visti come dei file, e le GPIO non fanno eccezione in quanto fanno parte anch'esse del filsystem del sistema.

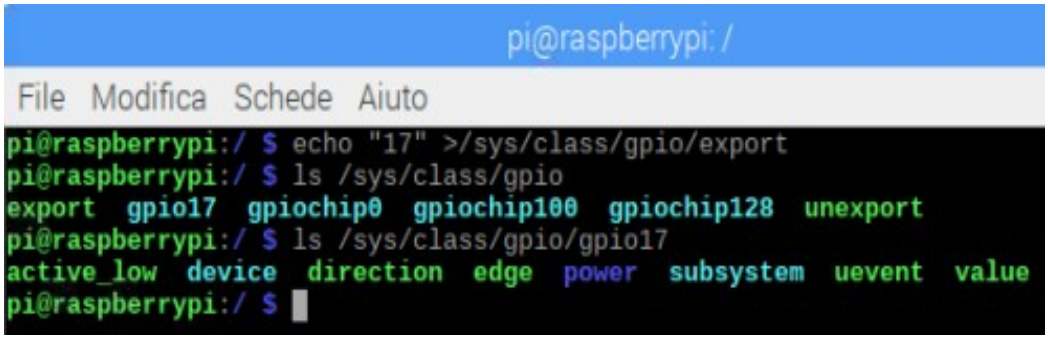
I file che gestiscono le GPIO si trovano nel percorso ***/sys/class/gpio.***

Da terminale scriviamo ***ls /sys/class/gpio:***



Notiamo la presenza di 3 cartelle (in azzurro) e di due files (in verde) concentriamoci su questi ultimi due files, che di fatto sono due programmi a cui passeremo dei parametri.
Con il primo file ***export***, possiamo creare i files che ci permettono di lavorare in ingresso o in uscita su una delle tante GPIO. Questo programma crea di fatto una struttura che parte da una directory che porterà lo stesso nome della GPIO scelta.

Ad esempio scrivendo,
echo "17" >/sys/class/gpio/export passiamo al programma ***export*** il parametro 17, ed in questo modo verrà creata la struttura per gestire la GPIO 17.



Come possiamo vedere dall'immagine sopra dopo il comando, è stata creata la cartella ***gpio17*** nel percorso ***/sys/class/gpio/gpio17***. Dentro la cartella c'è l'insieme d file e cartelle che permetteranno di impostare la direzione (input o output) di attivare a GPIO o di leggerne lo stato.
Cominciamo con l'impostazione della direzione, se vogliamo impostare la GPIO17 come uscita dobbiamo passare al programma ***direction*** (presente nella cartella appena creata) il parametro **out** con il seguente comando da terminale, ***echo "out" >/sys/class/gpio/gpio17/direction***.
Successivamente potremo accendere e spegnere la GPIO 17 passando il parametro **0** o **1** al programma ***value*** presente sempre nella stessa directory:

echo "1" >/sys/class/gpio/gpio17/value accende l'uscita GPIO17
echo "0" >/sys/class/gpio/gpio17/value spegne l'uscita GPIO17

Se invece vogliamo impostare la GPIO 16 come ingresso dovremo creare la struttura ed impostare la direzione come input con i seguenti due comandi:

echo "16" >/sys/class/gpio/export ***echo "out" >/sys/class/gpio/gpio17/direction***.
Per leggere lo stato dell'ingresso possiamo digitare ***cat /sys/class/gpio/gpio16/value.***
Al termine dell'utilizzo delle GPIO per cancellare le due strutture create bisogna utilizzare il comando ***unexport***,
echo "17" >/sys/class/gpio/unexport ***echo "16" >/sys/class/gpio/unexport***

Riepilogando in maniera grafica le operazioni da fare sono le seguenti:

1. ***Creare la struttura della GPIO*** **numero della GPIO** → **/sys/class/gpio/export**
2. ***Impostare la direzione della porta*** **parametro in o out** → **/sys/class/gpio/gpioxx/direction**
3. ***Scrittura del valore*** **0 oppure 1** → **/sys/class/gpio/gpioxx/value**
4. ***Lettura del valore*** **0 oppure 1** ← **/sys/class/gpio/gpioxx/value**
5. ***Cancellare la struttura della GPIO*** **numero della GPIO** → **/sys/class/gpio/unexport**

Possiamo pertanto utilizzare questo metodo per lavorare sulle GPIO utilizzando un qualsiasi linguaggio di programmazione, come ad esempio il linguaggio C.

Di seguito un esempio scritto in linguaggio C per eseguire la stessa funzione prevista nei programmi precedenti e cioè il lampeggio dei due led a seconda dello stato del pulsante.

```c
#include<stdio.h>
#include<stdlib.h>

#define Ritardo 10000000

int main() {
 FILE *handle;  //dichiaro un puntatore ad un file
 long cont;
 int lettura,cicli=100;

 /*------Crea porta GPIO17------*/
 handle=fopen("/sys/class/gpio/export","w"); //apro il file in scrittura
 fprintf(handle,"17");  //utilizzo il puntatore handle per passare il numero della GPIO
 fclose(handle);        //chiudo il file
 /*------Crea porta GPIO22------*/
 handle=fopen("/sys/class/gpio/export","w");
 fprintf(handle,"22");
 fclose(handle);
  /*------Crea porta GPIO16------*/
 handle=fopen("/sys/class/gpio/export","w");
 fprintf(handle,"16");
 fclose(handle);

 /*-----GPIO17 in uscita------*/
 handle=fopen("/sys/class/gpio/gpio17/direction","w");
 fprintf(handle,"out");
 fclose(handle);
 /*-----GPIO22 in uscita------*/
 handle=fopen("/sys/class/gpio/gpio22/direction","w");
 fprintf(handle,"out");
 fclose(handle);
 /*-----GPIO16 in ingresso------*/
 handle=fopen("/sys/class/gpio/gpio16/direction","w");
 fprintf(handle,"in");
 fclose(handle);
```

```
while(cicli)
 {
      //leggo lo stato del pulsante
      handle=fopen("/sys/class/gpio/gpio16/value","r"); //apro il file in lettura
      fscanf(handle,"%d",&ettura);    //acquisisco il valore
      fclose(handle);                 //chiudo il file
      if(lettura==1)
       {
          /*------Accende diodo Led giallo-----*/
          handle=fopen("/sys/class/gpio/gpio17/value","w");
          fprintf(handle,"1");
          fclose(handle);
       }//if
      else
       {
          /*------Accende diodo Led verde-----*/
          handle=fopen("/sys/class/gpio/gpio22/value","w");
          fprintf(handle,"1");
          fclose(handle);
       }//if

      /*----------Ritardo--------*/
      for(cont=1;cont<=Ritardo;cont++);

      /*------------Spegne diodo Led giallo------*/
      handle=fopen("/sys/class/gpio/gpio17/value","w");
      fprintf(handle,"0");
      fclose(handle);
      /*------------Spegne diodo Led verde------*/
      handle=fopen("/sys/class/gpio/gpio22/value","w");
      fprintf(handle,"0");
      fclose(handle);

      /*----------Ritardo--------*/
      for(cont=1;cont<=Ritardo;cont++);

      cicli--;
 }//while

/*----Rilascia porta GPIO17------*/
handle=fopen("/sys/class/gpio/unexport","w");
fprintf(handle,"17");
fclose(handle);
/*----Rilascia porta GPIO22------*/
handle=fopen("/sys/class/gpio/unexport","w");
fprintf(handle,"22");
fclose(handle);
/*----Rilascia porta GPIO16------*/
handle=fopen("/sys/class/gpio/unexport","r");
fprintf(handle,"16");
fclose(handle);
return 0;
}//main
```

Ovviamente il programma in linguaggio C (o in altri linguaggi, può utilizzare metodi diversi da quello utilizzato sopra che utilizza un puntatore ai file (handle) che vengono aperti o chiusi , ed a cui vengono passati dei parametri. Esistono comunque altre librerie per utilizzare le GPIO, con un po' di volonta in rete possiamo trovare di tutto e realizzando lo stesso schema, possiamo sbizzarrirci a verificare il funzionamento di tutte le librerie, come è stato fatto per redigere questo breve tutorial.