



PRÁCTICA 1

Programación de la Raspberry Pi2

OBJETIVOS

Aprender a

- programar la Raspberry Pi 2 desde distintas plataformas.
- programar los pines GPIO de la Raspberry Pi para realizar funciones de entrada/salida.



MATERIAL NECESARIO

Hardware

- Raspberry Pi 2
- Protoboard, LEDs, pulsador...

Software

- PuTTY y NetBeans 8.0.2
- Librerías WiringPi

Manuales (en Moodle)

- RaspberriPi para RSA
- API WiringPi : <http://wiringpi.com/reference/>

1. Puesta en marcha

Para usar los pines del puerto J8 de la Raspberry Pi 2 vamos a usar la librería Wiring Pi¹ preparada para C/C++. Para instalar esta librería seguiremos los pasos del documento RaspberriPi para RSA:

- Instalar GIT
 - `sudo apt-get install git-core`
- Update & Upgrade Raspberry Pi

¹ Wiring Pi. <http://wiringpi.com/>

- `sudo apt-get update`
- `sudo apt-get upgrade`

- Descargar el proyecto WiringPi

Si ya ha sido descargado previamente devolverá el mensaje: *"Fatal destination path 'wiringPi' already exists and is not an empty directory"*. Si es así se puede pasar al siguiente punto.

- `git clone git://git.drogon.net/wiringPi`

- Entrar en la carpeta y actualizar el contenido

- `cd wiringPi`
- `git pull origin`

- Compilar las librerías

- `./build`

- Comprobar que la librería funciona

- `gpio -v`
- `gpio readall`

| Pi 2 | | | | | | | | | | | |
|------|-----|---------|------|---|----------|----|------|------|---------|-----|----|
| BCM | wPi | Name | Mode | V | Physical | V | Mode | Name | wPi | BCM | |
| | | 3.3v | | | 1 | 2 | | 5v | | | |
| 2 | 8 | SDA.1 | IN | 1 | 3 | 4 | | 5V | | | |
| 3 | 9 | SCL.1 | IN | 1 | 5 | 6 | | 0v | | | |
| 4 | 7 | GPIO. 7 | IN | 1 | 7 | 8 | 1 | ALT0 | TxD | 15 | 14 |
| | | 0v | | | 9 | 10 | 1 | ALT0 | RxD | 16 | 15 |
| 17 | 0 | GPIO. 0 | IN | 0 | 11 | 12 | 0 | IN | GPIO. 1 | 1 | 18 |
| 27 | 2 | GPIO. 2 | IN | 1 | 13 | 14 | | 0v | | | |
| 22 | 3 | GPIO. 3 | IN | 1 | 15 | 16 | 0 | IN | GPIO. 4 | 4 | 23 |
| | | 3.3v | | | 17 | 18 | 0 | IN | GPIO. 5 | 5 | 24 |
| 10 | 12 | MOSI | IN | 0 | 19 | 20 | | 0v | | | |
| 9 | 13 | MISO | IN | 0 | 21 | 22 | 0 | IN | GPIO. 6 | 6 | 25 |
| 11 | 14 | SCLK | IN | 0 | 23 | 24 | 1 | IN | CE0 | 10 | 8 |
| | | 0v | | | 25 | 26 | 1 | IN | CE1 | 11 | 7 |
| 0 | 30 | SDA.0 | IN | 1 | 27 | 28 | 1 | IN | SCL.0 | 31 | 1 |
| 5 | 21 | GPIO.21 | IN | 1 | 29 | 30 | | 0v | | | |
| 6 | 22 | GPIO.22 | IN | 1 | 31 | 32 | 0 | IN | GPIO.26 | 26 | 12 |
| 13 | 23 | GPIO.23 | IN | 0 | 33 | 34 | | 0v | | | |
| 19 | 24 | GPIO.24 | IN | 0 | 35 | 36 | 0 | IN | GPIO.27 | 27 | 16 |
| 26 | 25 | GPIO.25 | IN | 0 | 37 | 38 | 0 | IN | GPIO.28 | 28 | 20 |
| | | 0v | | | 39 | 40 | 0 | IN | GPIO.29 | 29 | 21 |
| BCM | wPi | Name | Mode | V | Physical | V | Mode | Name | wPi | BCM | |

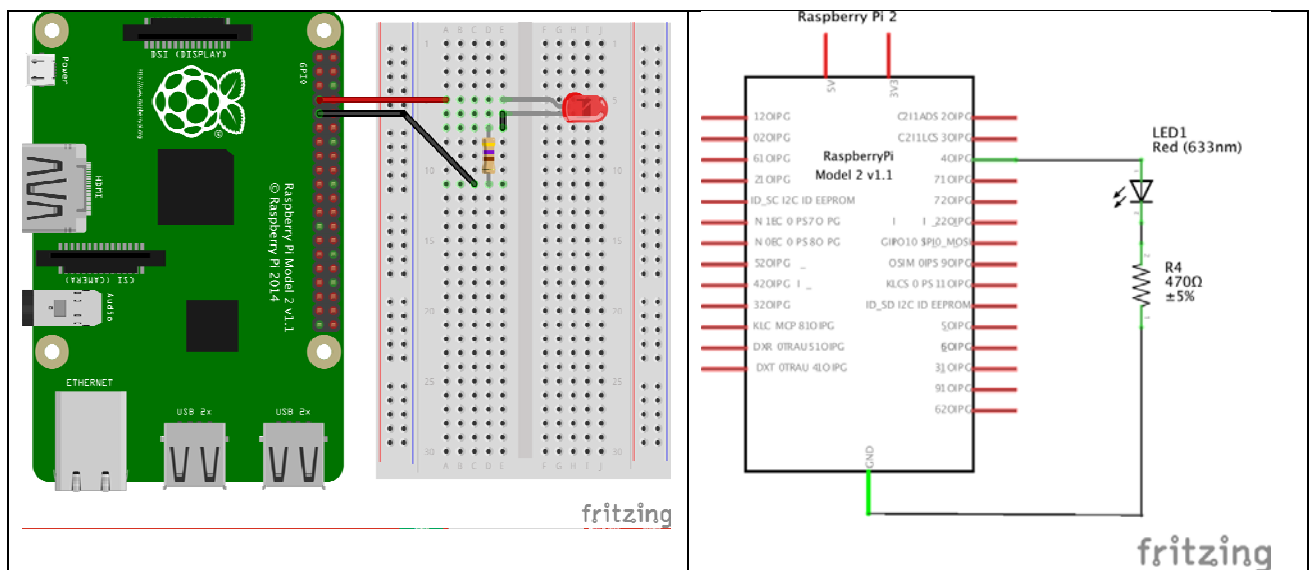
Wiring Pi lleva en funcionamiento desde las primeras versiones de Raspberry Pi. El orden y disposición del GPIO ha ido cambiando para las distintas revisiones de la placa.

2. Materiales

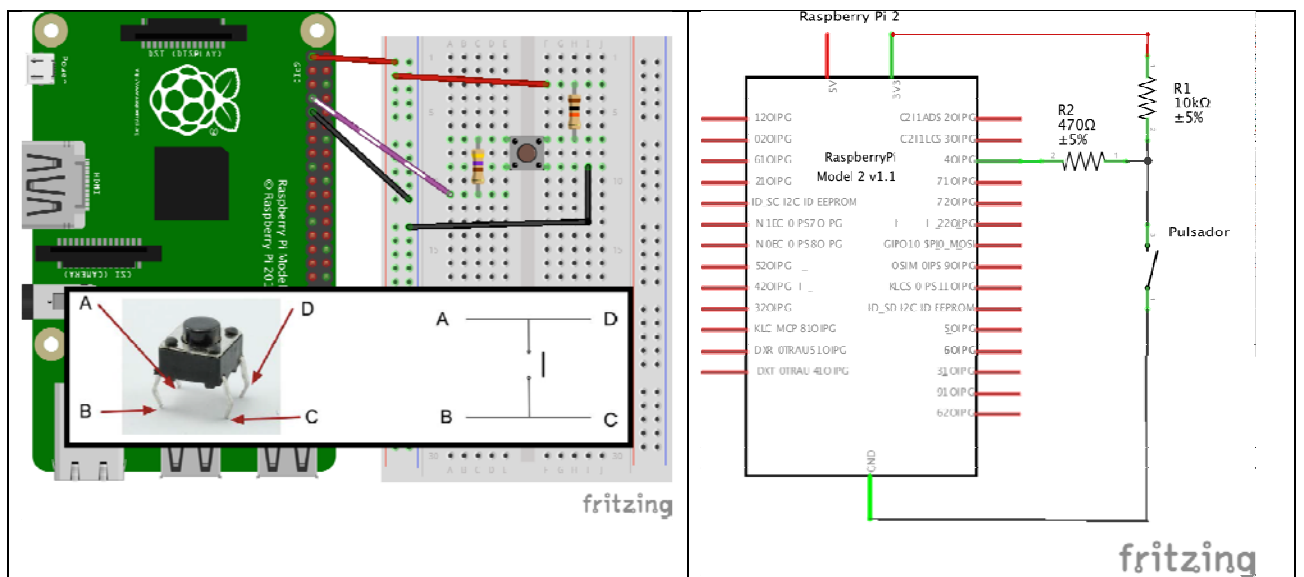
Para realizar esta práctica es necesario:

- Placa de prototipado
- Cables (rojos y negros)
- 1 LED
- 1 Pulsador
- 2 Resistencias de $470\ \Omega$
- 1 Resistencia de $10\ \text{k}\Omega$
- Mazo de cables negros
- Tira de 9 pines

Conexión del LED:



Conexión del pulsador:



3. Uso de las librerías

WiringPi está compuesta por una serie de funciones que facilitan el uso de la entrada/salida al programar en C/C++ la Raspberry Pi.

Es necesario en primer lugar incluir referencias a los archivos .h que definen las funciones que vamos a usar:

```
#include <wiringPi.h>
#include <softPwm.h> (para usar PWM)
```

En esta práctica se hace uso de las siguientes funciones:

| | |
|--|---|
| wiringPiSetup () | Configura el puerto GPIO para usar la numeración por defecto de la librería WiringPi |
| pinMode (Param1, Param2) | Selecciona si un pin funciona com entrada o salida: <ul style="list-style-type: none">• Param1: Identificador del pin• Param2: INPUT / OUTPUT |
| digitalWrite (Param1, Param2) | Establece el valor de un pin configurado de salida: <ul style="list-style-type: none">• Param1: Identificador del pin• Param2: HIGH / LOW |
| int digitalRead(Param) | Adquiere el valor de un pin configurado de entrada: <ul style="list-style-type: none">• Param: Identificador del pin• Return: 1 - High / 0 - Low |
| delay(Param); | Pausa la ejecución de un programa: <ul style="list-style-type: none">• Param: Número de milisegundos |
| softPwmCreate(Param1, Param2, Param3); | Configura un pin de salida para el uso de la modulación de anchura de pulso (PWM): <ul style="list-style-type: none">• Param1: Identificador del pin• Param2: Valor inicial del pin• Param3: Rango máximo |
| softPwmWrite(Param1, Param2) | Establece el valor del PWM de un pin de salida: <ul style="list-style-type: none">• Param1: Identificador del pin• Param2: Valor del PWM respecto al rango |

El resto de funciones de la librería se pueden encontrar consultar en:
<http://wiringpi.com/reference/>

4. Compilación de programas WiringPi

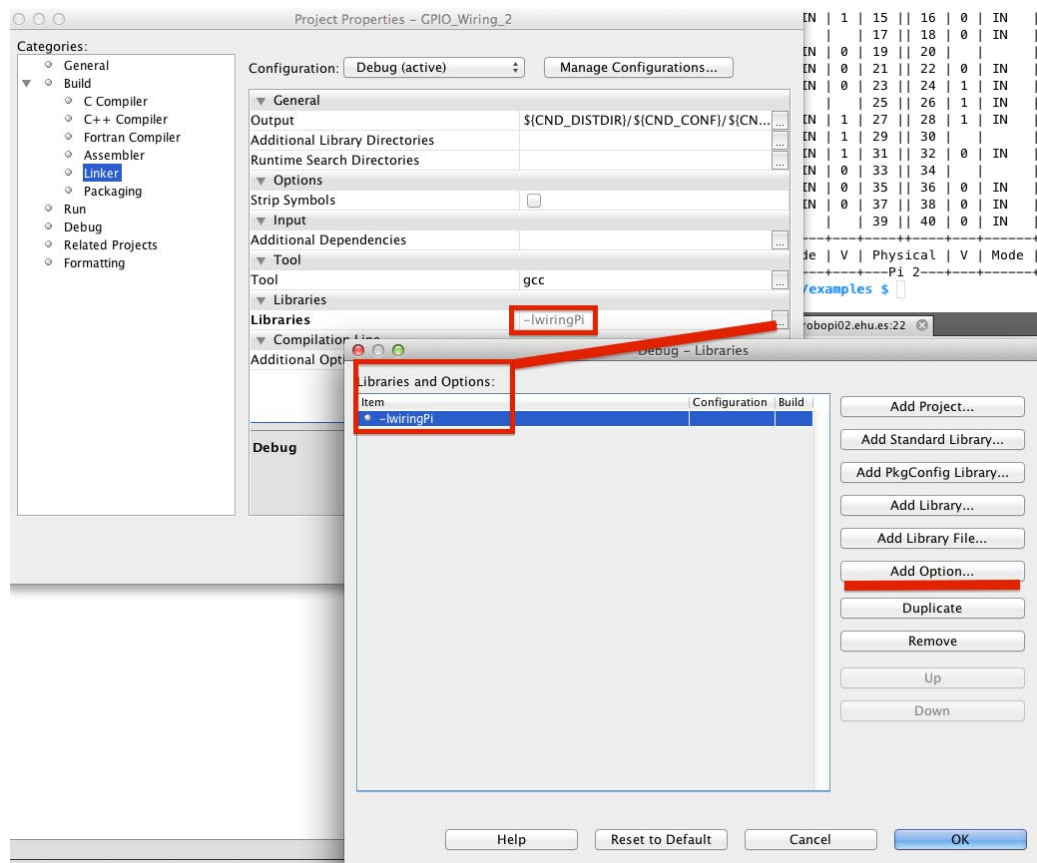
Los proyectos que usan WiringPi requieren que se haga referencia a las librerías estáticas que se instalan. Para ello hay que añadir “-lwiringPi” a las opciones de compilación:

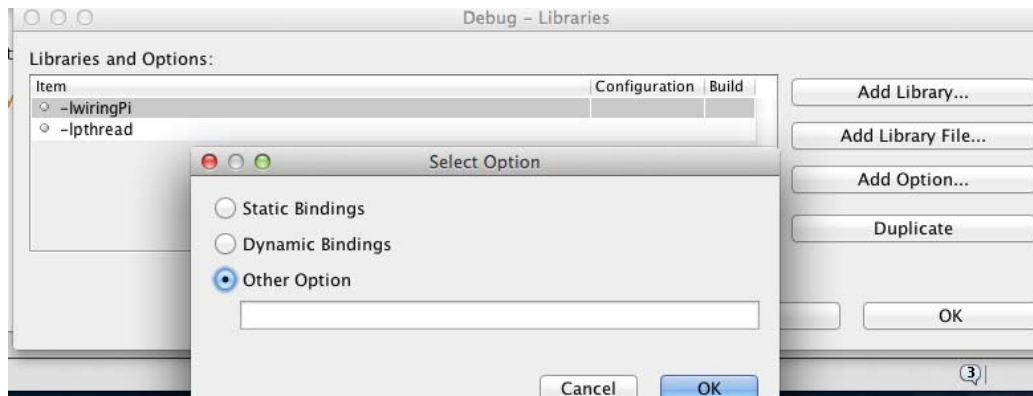
```
➤ gcc -Wall myFile.c -o programa -lwiringPi
```

Si se usa la funcionalidad de PWM también hay que incluir la opción “-lpthread”. Para ejecutar el programa hay que hacerlo como administrador:

```
➤ sudo ./programa
```

En NetBeans es necesario añadir las librerías de forma manual en las “propiedades del proyecto”, apartado “Linker”:





Escribir

-lwiringPi

en la ventana Other Option

5. Programas

1. Programación en C directamente sobre Raspberry Pi2.

Establecer una conexión SSH (mediante PuTTY o NetBeans) y **extender** el programa “Hola Mundo” escrito en C :

```
#include <stdio.h>
int main() {
    printf("Hola mundo");
    return 0;
}
```

de manera que **incluya** la función “scanf” para leer el teclado y la definición:

```
int main(int argc, char *argv[])
```

2. Programación en C desde NetBeans

Crear un proyecto NetBeans para ejecutar el mismo programa del apartado anterior.

3. Programación en C de la GPIO

Crear un proyecto NetBeans para controlar las E/S de la GPIO..

- Conectar un LED y escribir un programa que lo encienda y apague en intervalos de 1 segundo.
- Añadir un pulsador y escribir un programa que haga que se encienda el LED al pulsar el pulsador y se apague al volver a pulsarlo (tened en cuenta los rebotes del pulsador).
- Controlarla luminosidad del LED mediante PWM: Al pulsar el botón, la luminosidad del LED va aumentando de 0 al máximo y al volver a pulsarlo disminuye desde el máximo a 0.

4. Programación con threads de Wiring Pi

Escribir un programa que cree tres *threads* que escriban su propio nombre por pantalla. Para que no sigan la misma secuencia, introducir esperas diferentes en cada uno de ellos (por ejemplo, 1, 2 y 3 segundos).

6. Resultados

- a. Antes de desmontar los circuitos, es necesario hacer una demostración al profesor del funcionamiento de los cinco programas
 - a. P1: "Hola Mundo extendido"
 - b. P2: "Hola Mundo extendido" desde NetBeans
 - c. P3: LED intermitente
 - d. P4: LED controlado mediante pulsador
 - e. P5: LED con luminosidad variable controlado mediante pulsador
 - f. P&: Programa con threads
 - b. Cada grupo redactará y subirá a eGela un breve informe con el siguiente contenido:
 - Código fuente de **todos los programas comentados.**
 - Informe de la práctica redactado según la plantilla "Informe de la Práctica 1" disponible en eGela
- Comprimir todos los ficheros en un unico zip con el siguiente nombre:
- **Practica1_GrupoXY_Apellido1_Apellido2**