

Uso de la Raspberry Pi 2 en Robótica

Raspberry Pi 2

- es un mini-ordenador basado en el microcontrolador de Broadcom BMC 2836 a 900Mhz con 1GB de RAM.
 - [<https://www.raspberrypi.org/products/raspberry-pi-2-model-b/>]
- admite diferentes sistemas operativos.
 - usaremos “Raspberry Pi OS”, (antes llamado Raspbian, versión de la distribución Linux Debian, adaptada al hardware de la Raspberry Pi).
 - Raspberry Pi OS : [<https://www.raspberrypi.org/documentation/raspbian/>]



Configuración de la Pi para el laboratorio de Robótica

- Usaremos Raspberry Pi 2 para controlar el iRobot create.
 - Las placas Raspberry Pi 2 están preconfiguradas
 - Conviene verificarlo y actualizar el sistema operativo (Si hay que reconfigurarlas -> ANEXO 1)
- Conectar a la placa Raspberry Pi del grupo:
 - un teclado y un ratón USB
 - un adaptador de HDMI-VGA a la pantalla disponible.
 - un adaptador USB-WIFI
 - una fuente de alimentación (a Raspberry PI 2 usa alimentación de 5V a 2A aunque internamente trabaja con 3.3V).
- Cada placa Raspberry Pi del laboratorio tiene pegada una etiqueta en la que aparece el nombre que la identifica con su nombre en la red: “robopiXY”.
- Login del usuario por defecto (por seguridad, podría no existir):
 - *login:* pi
 - *Password:* robopi
- Login de cada grupo
 - *login:* grupoXY
 - *Password:* grupoXY (dónde XY es el número de grupo, XY: {01,02,...10})
- Login del supersusuario
 - *Login:* root
 - *password :* toor



Verificación, mantenimiento y actualización

1. Iniciar sesión:

- robopiXY login: **grupoXY**
- Password: **grupoXY**

grupoXY@robopyXY ~\$

2. Comprobar la asignación de dirección de red:

```
grupoXY@robopyXY ~$ ifconfig wlan0
```

```
wlan0
Link encap:Ethernet  HWaddr 74:da:38:2e:2a:29
inet addr:192.168.1.1XY
    Bcast:192.168.1.255  Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST  MTU:1500
    Metric:1
RX packets:6 errors:0 dropped:12 overruns:0
  frame:0
TX packets:8 errors:0 dropped:0 overruns:0
  carrier:0
collisions:0 txqueuelen:1000
RX bytes:1028 (1.0 KiB)  TX bytes:1070 (1.0
  KiB)
```

3. Actualizar el sistema operativo

- ~\$ sudo apt update
- ~\$ sudo apt upgrade

4. Cómo apagar y reiniciar (halt and reboot)

• Reiniciar:

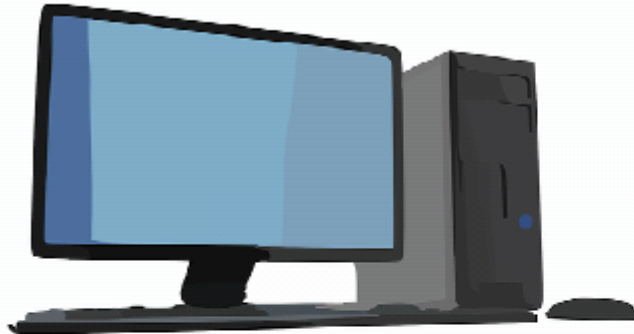
- ~\$ sudo shutdown -r now
- o
- ~\$ sudo reboot

• Apagar:

- ~\$ sudo shutdown -h now
- o
- ~\$ sudo halt

Modo de trabajo

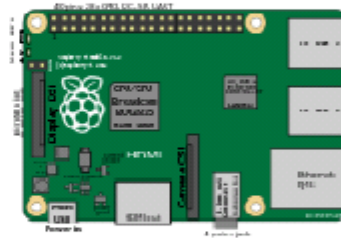
Puesto de trabajo



- Sistema Operativo: Windows 10
- Entorno de desarrollo: NetBeans
- Lenguaje: C/C++
- Conexión WiFi a red "RaspberryPiLab"

Raspberry Pi 2

- Sistema Operativo: Raspberry Pi OS
- Entorno de desarrollo: compiladores gcc/g++
- Conexión WiFi a red "RaspberryPiLab"



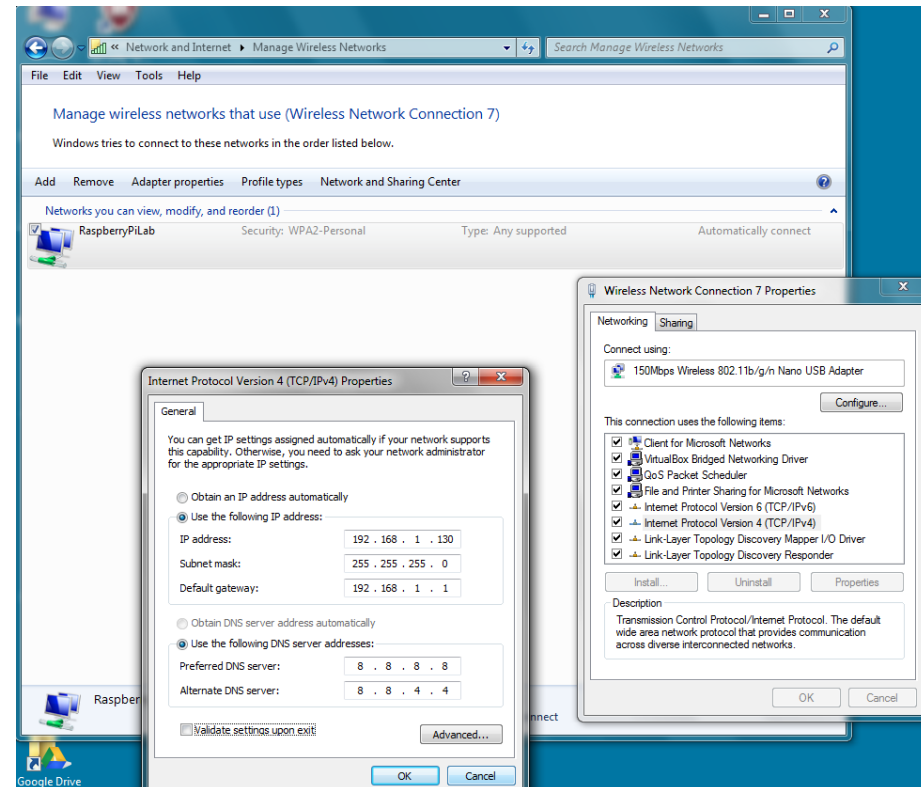
Router "RaspberryPiLab"
Pass: 'robopi'

Conexión de la Raspberry Pi 2 con PC/Windows a través de WiFi

- Usaremos la red wifi “RaspberryPiLab” con dos adaptadores WiFi USB.
- Configurar la dirección IP de los PC en un rango que permita trabajar a las Raspberry PI 2 sin conflictos: [131–140].

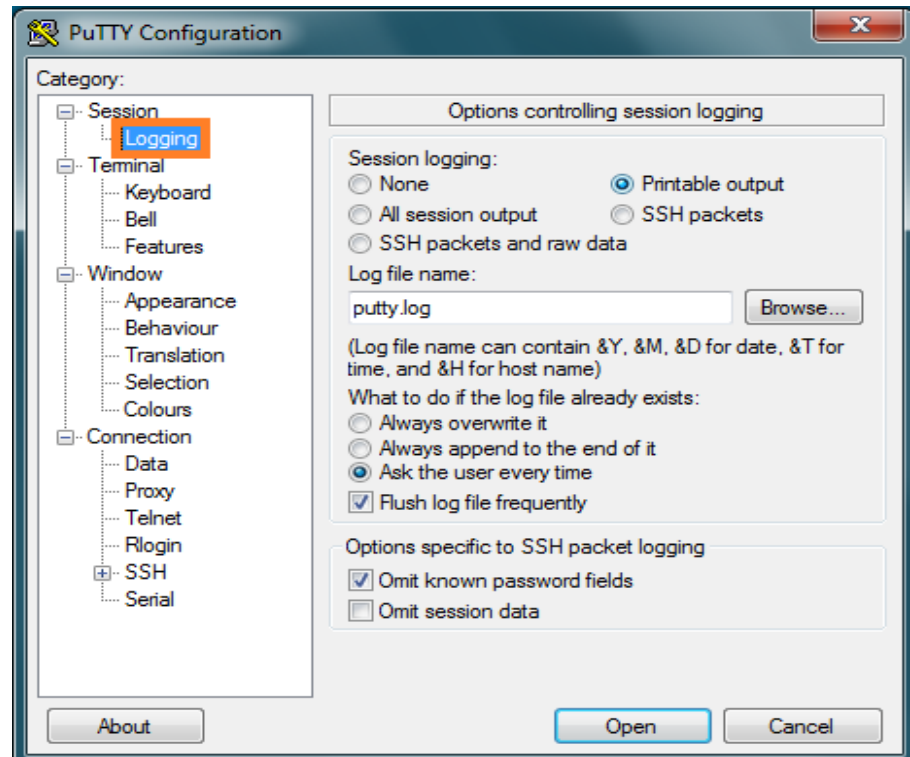
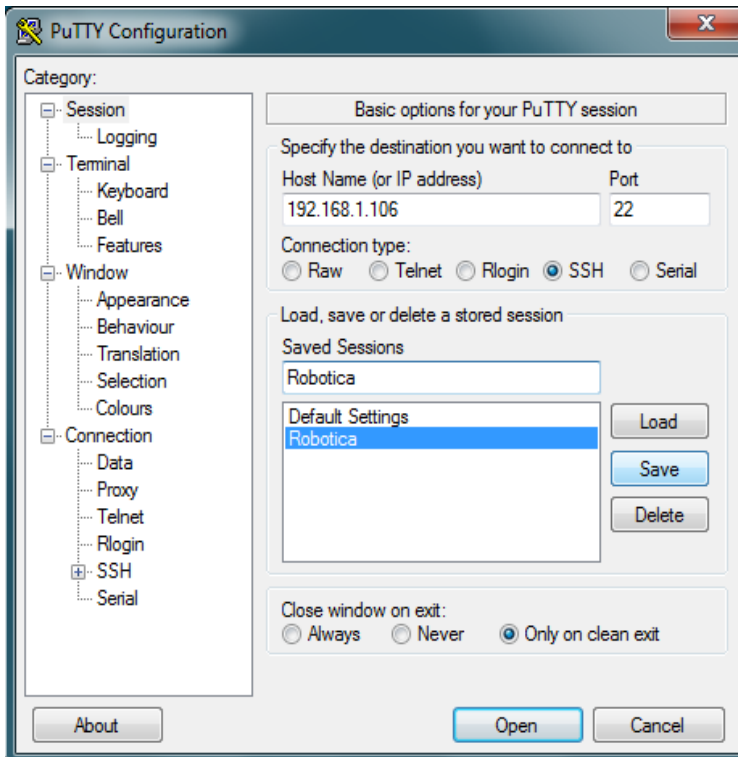
Asignación de números de IP a cada grupo

Gr	Raspberry Pi	PC
01	192.168.1.101	192.168.1.131
02	192.168.1.102	192.168.1.132
03	192.168.1.103	192.168.1.133
04	192.168.1.104	192.168.1.134
05	192.168.1.105	192.168.1.135
06	192.168.1.106	192.168.1.136
07	192.168.1.107	192.168.1.137
08	192.168.1.108	192.168.1.138
09	192.168.1.109	192.168.1.139
10	192.168.1.110	192.168.1.140



Conexión remota SSH

- Usamos un cliente ssh para conectarnos de manera remota: **PuTTY**
 - conexión remota protocolo SSH (Secure Shell): dirección IP de la Raspberry PI 2; puerto 22.
 - permite almacenar un Log con los comandos y respuestas de la sesión remota si se configura en: Session -> Logging
- Se puede descargar gratuitamente desde:
<http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>



Programación en C en la Raspberry Pi

- Se pueden escribir programas en C/C++ y compilarlos sin necesidad de equipos exteriores.
- Raspbian incluye editores, compiladores gcc y g++, linkeditor, etc.

Ejercicio 1: “Hola mundo” en C

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

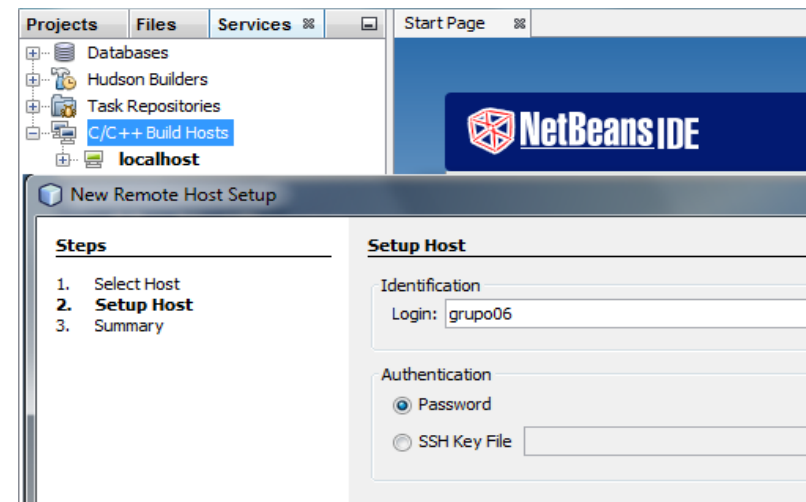
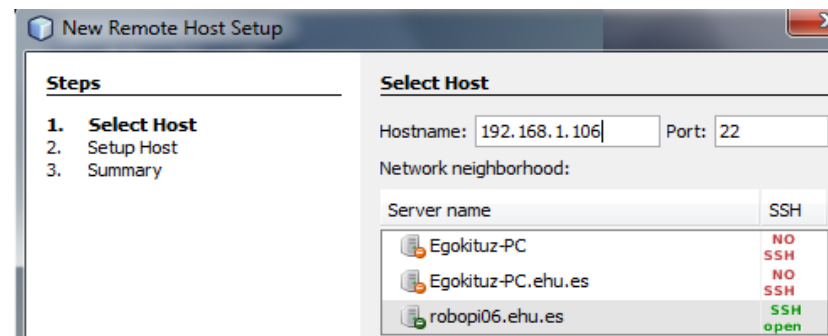
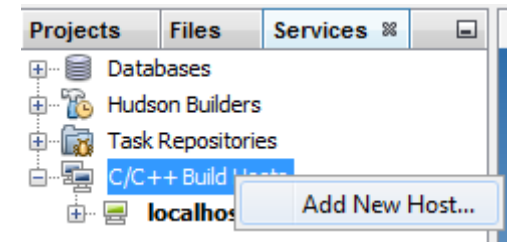
- Para ello, pueden ser útiles los siguientes comandos:

Comando Raspbian	Función
<code>touch programa.c</code>	Crea un archivo
<code>nano programa.c</code>	Edita el archivo (Ctrl+O guardar, Ctrl+X Salir)
<code>gcc -Wall programa.c -o programa</code>	Compila en C el archivo programa.c
<code>g++ -Wall programa.c -o programa</code>	Compila en C++ el archivo programa.c
<code>./programa</code>	Ejecuta el programa

Programación Raspberry Pi 2 usando el entorno NetBeans

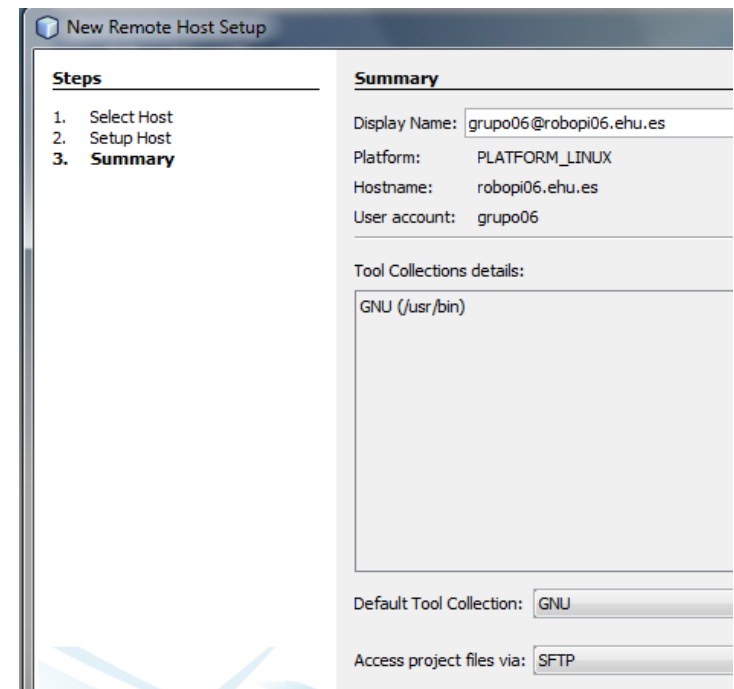
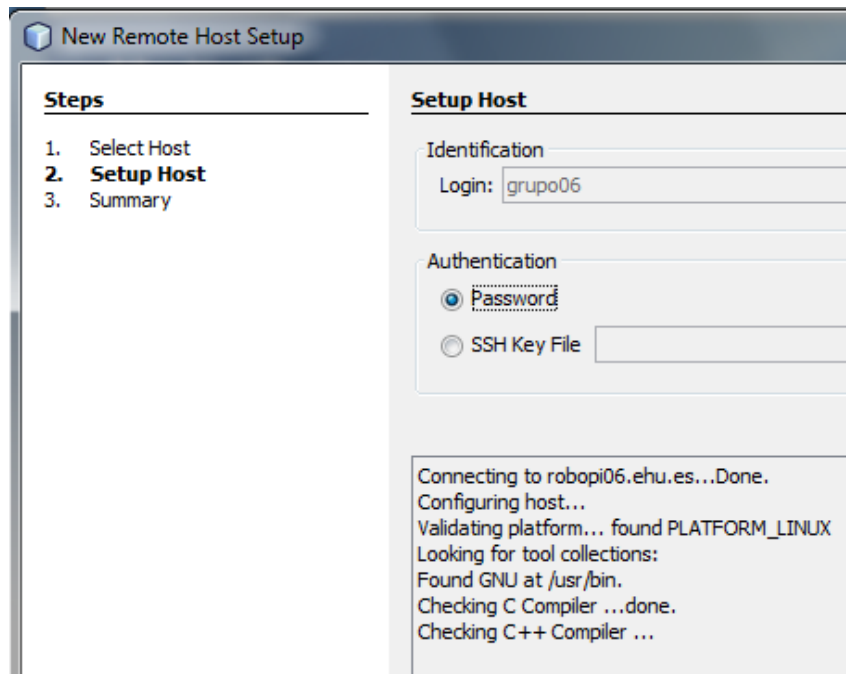
Para desarrollar programas utilizaremos NetBeans en Windows 10

- Descargar NetBeans 8.0.2:
<https://netbeans.org/community/releases/80/>
- Configurar NetBeans para C/C++ (Tutorial):
<https://netbeans.org/community/releases/80/cpp-setup-instructions.html>
- Desarrollo remoto en NetBeans (Tutorial)
<https://netbeans.org/kb/docs/cnd/remotedev-tutorial.html>
- Fijar un host remoto (la Raspberry Pi) para compilar los programas.
 - Hostname: dirección IP de la Raspberry Pi
 - login: el usuario que tenemos asignado
 - Password: el del grupo
 - aparece una advertencia de seguridad que hay que aceptar
- NetBeans detecta y configura el entorno de programación automáticamente para los compiladores que se encuentran instalados en la Raspberry PI
- Dado que los compiladores y el código que se genera están en la Raspberry Pi, para compilar es necesario que haya una conexión activa



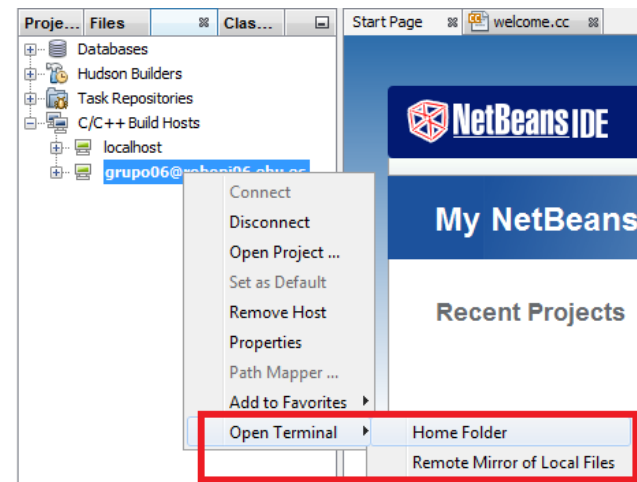
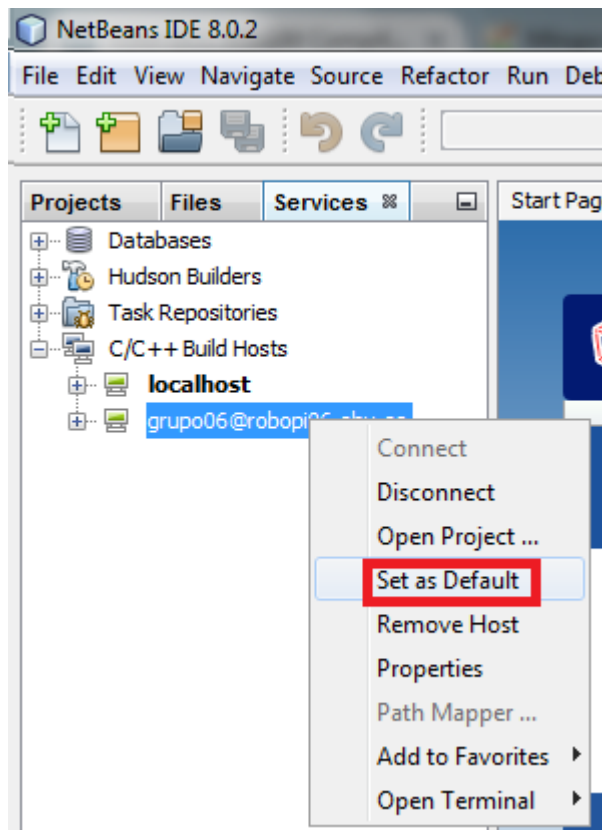
Programación Raspberry Pi 2 usando el entorno NetBeans

- Hay que asegurarse de que use el protocolo SFTP para copiar archivos (puede haber problemas a la hora de crear un nuevo proyecto)



Programación Raspberry Pi 2 usando el entorno NetBeans

- Es necesario configurar el nuevo "Host por defecto" para que los proyectos compilen directamente sobre la Raspberry Pi
- Es posible abrir un cliente SSH remoto usando las opciones "Open Terminal" de NetBeans.
- Equivale a la conexión que habíamos hecho con PUTTY



Programación Raspberry Pi 2 usando el entorno NetBeans

- Hay que asegurarse que el “Host” es el adecuado antes de compilar

The image displays three screenshots from the NetBeans IDE environment:

- Top Left:** The "New Sample Project - Welcome" dialog box. The "Project Name and Location" tab is active. The "Build Host" dropdown menu is highlighted with a red rectangle, showing "grupo06@robopi06.ehu.es" selected.
- Top Right:** A context menu for the "welcome.cc" file. The "Set Build Host" option is highlighted, and a sub-menu is open showing the selected host "grupo06@robopi06.ehu.es" and a "Manage Hosts..." option.
- Bottom:** The main IDE window showing the source code of "welcome.cc" and the output console. The output console displays the build process, including the command "g++ -c -g -MD -MF 'build/Debug/GNU-Linux-x86/welcome.o.d' -o build/Debug/GNU-Linux-x86/welcome.o welcome.cc" and the final message "BUILD SUCCESSFUL (total time: 2s)".

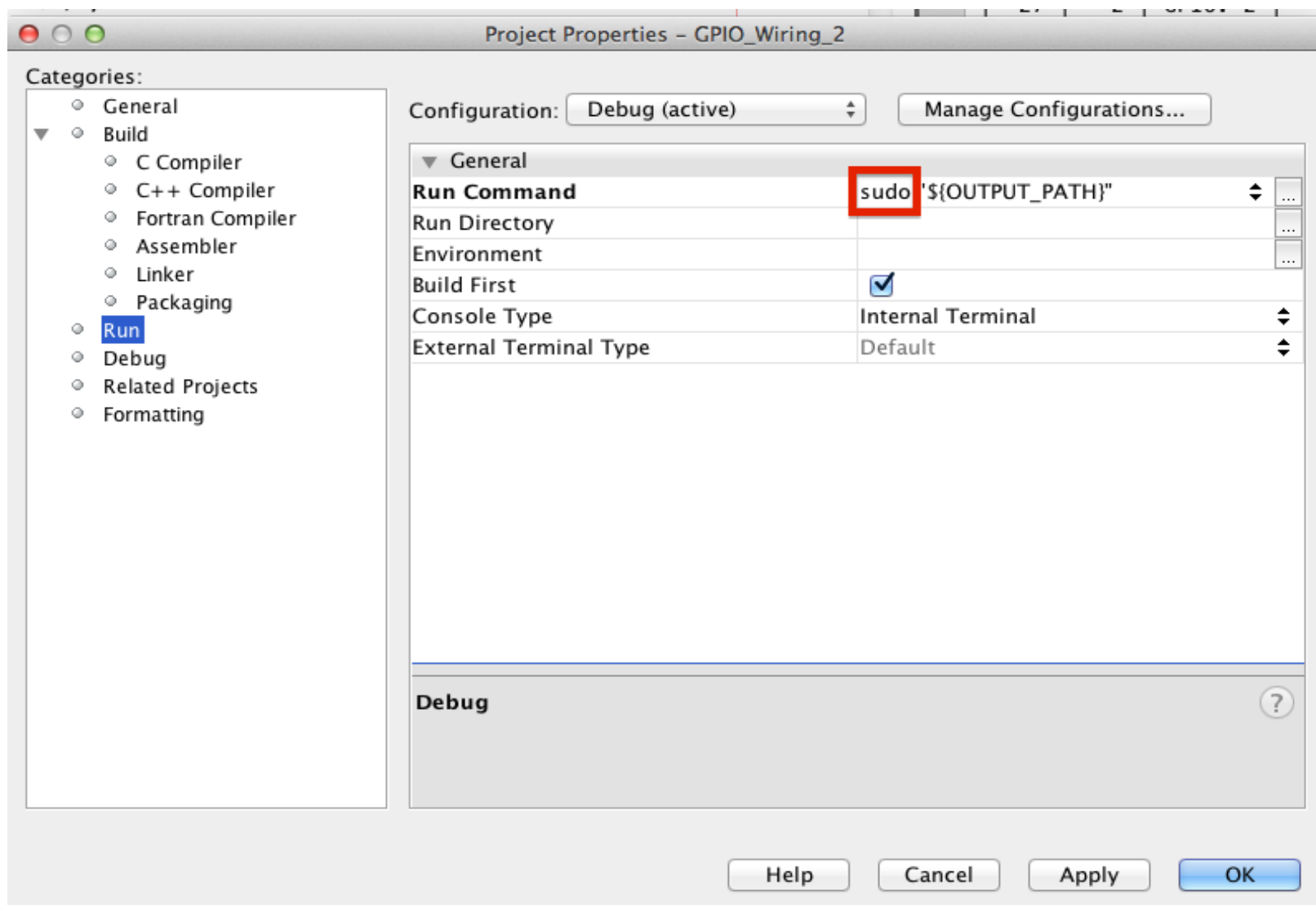
- Si el compilador finaliza sin errores, aparecerá un mensaje en color verde

Ejecución en la Raspberry Pi

Para lanzar un programa como root desde NetBeans, hay que configurar la opción **Run** en las propiedades del proyecto

- NetBeans deja el código ejecutable resultante de la compilación y linkedición en un directorio de la Raspberry Pi. Normalmente deberíamos ir a ese directorio y ejecutar con la orden
 - **~\$./programa**
- Para poder ejecutar el programa usando el “Run Project” de NetBeans, lo que resulta mucho más cómodo que tener que ir al directorio que contiene el ejecutable:
 - En una sesión de Raspberry Pi hacer lo siguiente:
 1. Pasar a ser root:
 - **~\$ sudo su**
 2. Poner password al root:
 - **~\$ passwd root <R> toor**
 - En la ventana de NetBeans:
 - Services, C/C++ Built Host borrar el antiguo host y crear uno nuevo:
 - Add new host:
 - » Host name: **dirección IP**
 - » Setup Host -> Name: **root**, Password: **toor**
- Otra alternativa es establecer el host por defecto con el usuario root de nuestra Raspberry Pi
- Ahora se puede compilar y ejecutar directamente con el triángulo verde “Run Project”

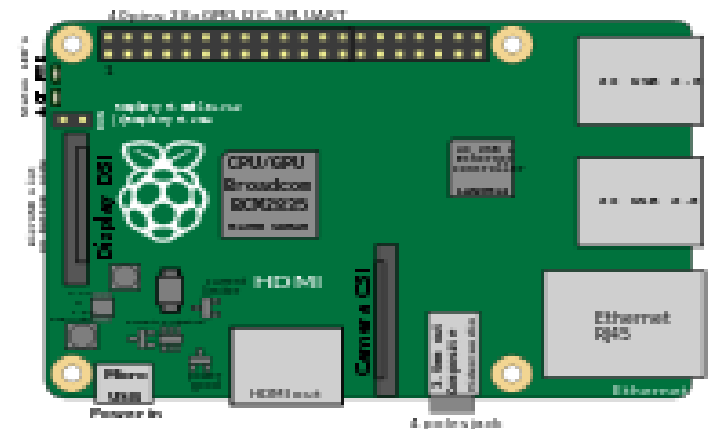
Ejecución en la Raspberry Pi



Prueba NetBeans-Raspberry Pi

- Para cargar nuevos proyectos
 - a. Crear un nuevo proyecto modificando uno de los ejemplos:
> File: New Project:
Samples: Ejemplo_X
 - b. Si partís de una plantilla usada:
>File: "import project"
- Probad que los programas de ejemplo incluidos en NetBeans se ejecutan correctamente.
 - Por ejemplo el "Project welcome"
> File: New Project: Samples: Welcome

De otra manera puede haber problemas para que encuentre los ficheros



Periféricos BCM2836

- Asignaciones del conector de 40 pines de la Raspberri Pi.
- Es posible que el software que utilicemos para controlar las estradas salidas, asigne nombres diferentes a esos mismos pines.



Raspberry Pi2 GPIO Header

Pin#	NAME		NAME	Pin#
01	3.3v DC Power	Red	DC Power 5v	02
03	GPIO02 (SDA1 , I ² C)	Blue	DC Power 5v	04
05	GPIO03 (SCL1 , I ² C)	Blue	Ground	06
07	GPIO04 (GPIO_GCLK)	Green	(TXD0) GPIO14	08
09	Ground	Black	(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)	Green	(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)	Green	Ground	14
15	GPIO22 (GPIO_GEN3)	Green	(GPIO_GEN4) GPIO23	16
17	3.3v DC Power	Red	(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)	Purple	Ground	20
21	GPIO09 (SPI_MISO)	Purple	(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)	Purple	(SPI_CE0_N) GPIO08	24
25	Ground	Black	(SPI_CE1_N) GPIO07	26
27	ID_SD (I ² C ID EEPROM)	Yellow	(I ² C ID EEPROM) ID_SC	28
29	GPIO05	Green	Ground	30
31	GPIO06	Green	GPIO12	32
33	GPIO13	Green	Ground	34
35	GPIO19	Green	GPIO16	36
37	GPIO26	Green	GPIO20	38
39	Ground	Black	GPIO21	40

Rev. 1
26/01/2014

<http://www.element14.com>

Precauciones importantes:

- No introducir más de 3.3V en un pin GPIO
- No consumir más de 3mA por los pines de salida (asegurarse de protegerlos usando resistencias)
- No tocar el puerto GPIO con destornilladores o objetos metálicos
- No alimentar la Raspberry Pi con más de 5V
- No consumir más de 50mA de los pines a 3.3V
- No consumir más de 250mA de los pines a 5V

Acceso al puerto GPIO desde la Raspbian

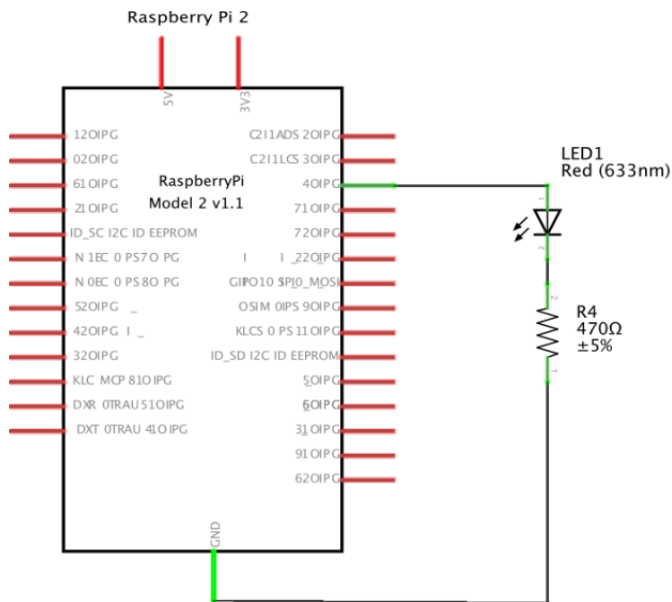
Para programar la función de los pines de entrada salida hay que manipular los ficheros que se encuentran en la carpeta `/sys/class/gpio` lo que **requiere permisos de root**.

- Para saber si tiene permisos de acceso al grupo GPIO ejecutar:
`~$ groups grupoXY`
- Si responde
`grupoXY: grupoXY audio video gpio`
es que tiene permiso para usar GPIO.
- Si no fuera así, hay que añadir el usuario grupoXY al grupo GPIO:
`~$ sudo usermod -aG gpio grupoXY`
Ahora, repetir la verificación con el comando groups.
- Para activar un pin en concreto hay que crear un acceso a él usando “export”
`~$ echo 4 > /sys/class/gpio/export`
- Hay que configurarlo para que se comporte como pin de entrada o salida (in/out)
`~$ echo out > /sys/class/gpio/gpio4/direction`
- En caso de que esté configurado como salida se puede escribir 1 o 0 para asignarle un voltaje “High” o “Low”.
`~$ echo 1 > /sys/class/gpio/gpio4/value`
- Tanto si es una salida como si es una entrada se puede leer el valor del voltaje en el pin con el comando “cat”
`~$ cat /sys/class/gpio/gpio4/value`

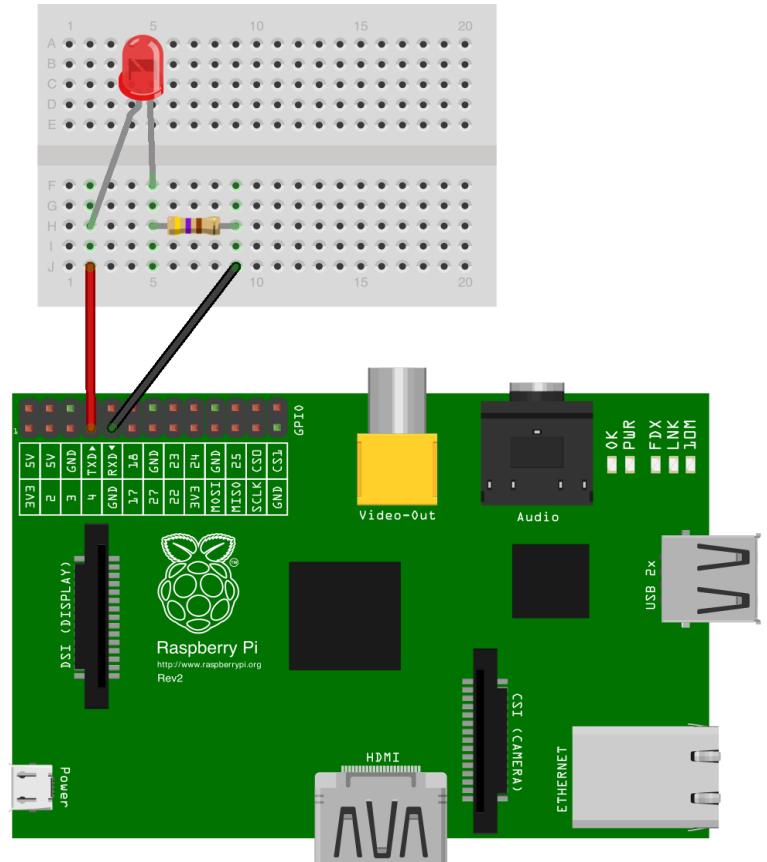
Prueba de salida de la GPIO

Lista de Materiales

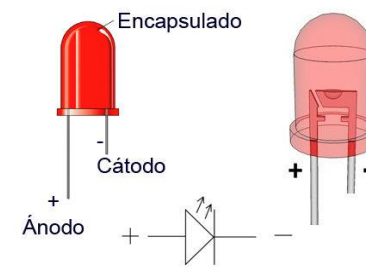
- Raspberry Pi 2 (etiquetada)
- Caja Transparente (etiquetada)
- Fuente de alimentación 5V 2A
- Micro SD 8 GB con Raspbian precargado (etiquetada)
- 2 Adaptadores USB WiFi
- Placa de montaje
- Cables de conexión (se suele usar rojo para la señal y negro para tierra/GND)
- 1 LEDs
- 1 pulsador
- dos resistencias de $470\ \Omega$ y una de $1K\Omega$
- Tira de 9 pines



fritzing



Made with Fritzing.org



Prueba de salida de la GPIO

Encender un led conectado al pin 7

1. Desde Raspbian

```
~$ echo 4 > /sys/class/gpio/export
```

```
~$ echo out >
```

```
    /sys/class/gpio/gpio4/direction
```

```
~$ echo {1/0} > /sys/class/gpio/gpio4/value
```

2. Desde un programa en C

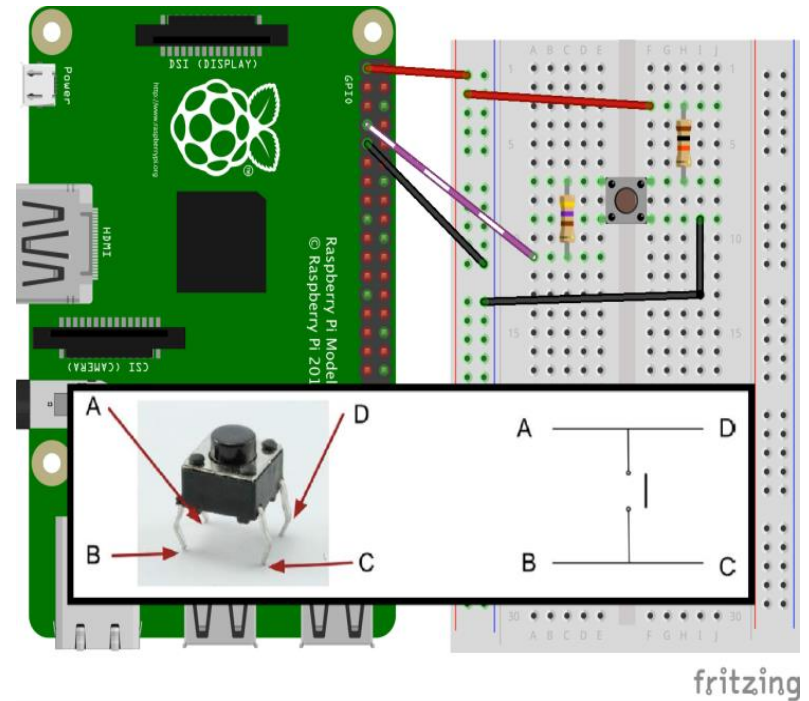
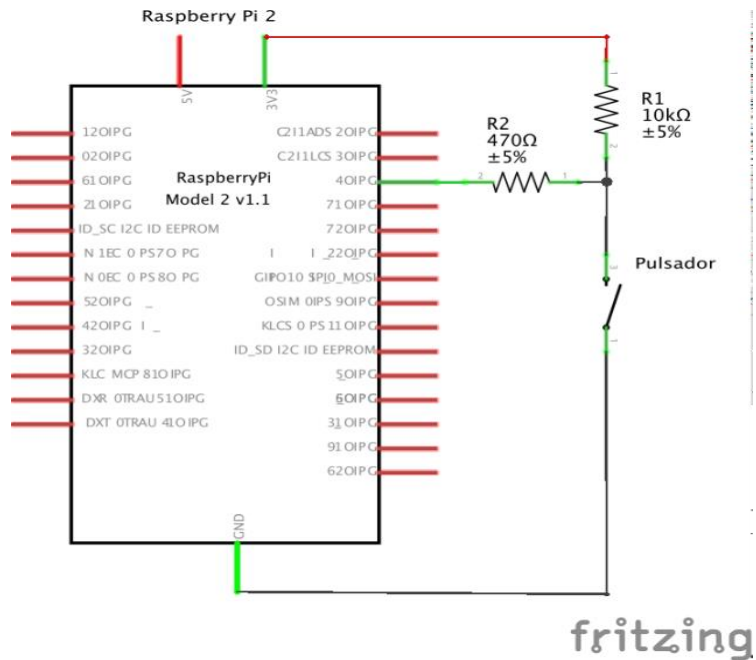
- usar la función `int system("comando")` que permite ejecutar comandos del Shell.

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

```
..
```

```
int system("comando");
```

Prueba de entrada por la GPIO



Prueba de entrada por la GPIO

Leer el estado de un pulsador conectado al pin 5 (correspondiente al GPIO3)

1. Desde Raspbian

- Programar el PIO4 con los comandos

```
~$ echo 3 > /sys/class/gpio/export
~$ echo in > /sys/class/gpio/gpio3/direction
```
- y verificar el funcionamiento del pulsador con el comando:

```
~$ cat /sys/class/gpio/gpio3value
```

2. Desde un Programa en C

- ¿Puedes hacer un programa en C que obtenga el valor del pulsador con una llamada?

```
system (cat /sys/class/gpio/gpio3value);
```
- En ese caso, modifica el programa del apartado 1.4.3.2 para que el LED empiece a parpadear a intervalos regulares de 0,5s cuando se pulse el pulsador y se apague completamente al pulsar de nuevo el pulsador

Librería Wiring Pi

Usaremos la librería Wiring Pi para acceder desde C/C++ a los pines del puerto J8 de la Raspberry Pi 2: <http://wiringpi.com/>

Instalación de Wiring Pi

- Instalar GIT

```
~$ sudo apt-get install  
git-core
```

- Update & Upgrade Raspberry Pi

```
~$ sudo apt-get update  
~$ sudo apt-get upgrade
```

Descargar el proyecto WiringPi

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

```
~$ git clone  
https://github.com/WiringPi/Wi  
ringPi
```

- Entrar en la carpeta y actualizar el contenido

```
~$ cd wiringPi  
~$ git pull origin
```

- Compilar las librerías

```
~$ ./build
```

- Comprobar que la librería funciona

```
~$ gpio -v
```

- ccv

Librería Wiring Pi

La orden:

```
~$ gpio readall
```

devuelve la siguiente tabla, con la programación actual de los pines, donde:

- BCM es el código asignado por la placa BCM GPIO
- wPi: código asignado por Wiring Pi
- Name: función del pin
- Mode: cómo está programado actualmente (IN/OUT)
- V: su estado lógico (1: high/true/, 0, low/false)
- Physical: posición en le conector (impares a la izquierda y pares a la derecha)

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	
Pi 2											

Uso de la librería Wiring Pi

- Wiring Pi 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 incluir referencias a los archivos .h que definen las funciones a usar:
`#include <wiringPi.h>`
`#include <softPwm.h>` (para usar PWM)
- Algunas funciones que usaremos en el laboratorio de robótica:

Entrada/salida	
Función	Funcionamiento
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: Param1: Identificador del pin; Param2: INPUT / OUTPUT
digitalWrite (Param1, Param2)	Establece el valor de un pin configurado de salida: Param1: Identificador del pin; Param2: HIGH / LOW
int digitalRead(Param)	Adquiere el valor de un pin configurado de entrada: Param: identificador del pin; Return: 1 - High / 0 - Low
delay(Param);	Pausa la ejecución de un programa. Param: Número de milisegundos
softPwmCreate(Param1, Param2, Param3);	Configura un pin de salida para el uso de la modulación de anchura de pulso (PWM): 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: Param1: Identificador del pin; Param2: Valor del PWM respecto al rango

Compilación de programas con WiringPi en Raspbian

- 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
```

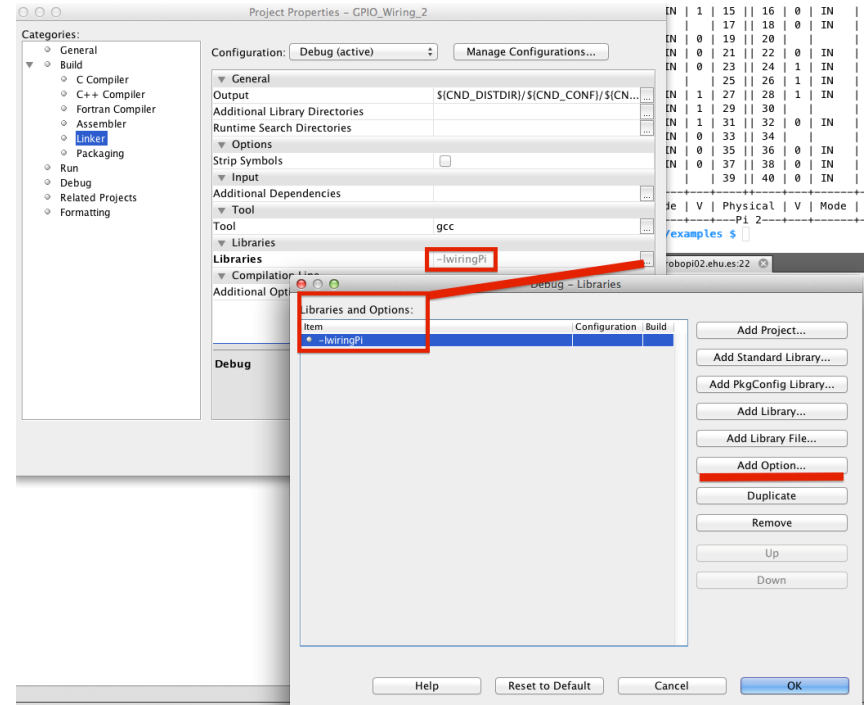
- Si se usa la funcionalidad de PWM también hay que incluir la opción `-lpthread`

- Hay que ejecutar el programa como administrador:

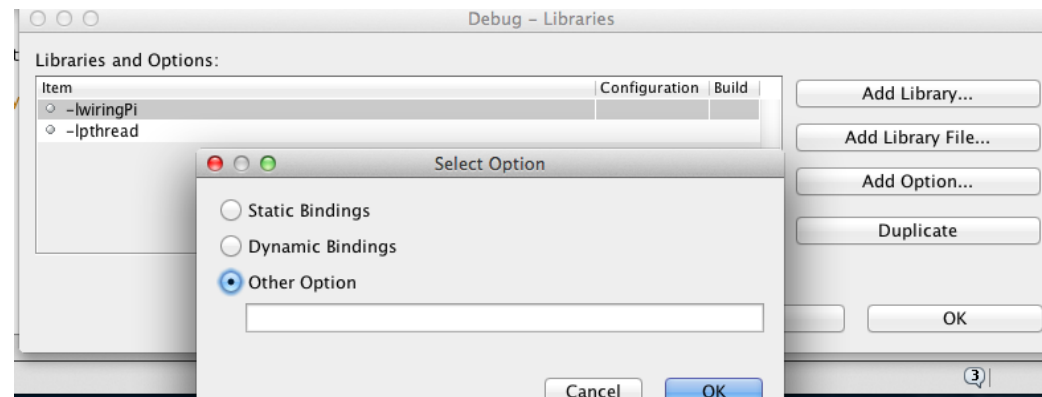
```
~$ sudo ./programa
```

Compilación de programas con WiringPi en NetBeans

- Es necesario añadir las librerías en “propiedades del proyecto” > “Linker” de forma manual



- Escribir en la ventana "Other Option"
`-lwiringPi`



Gestión de interrupciones en Wiring Pi

- La consulta por encuesta mantiene al procesador ocupado aunque no se esté procesando nada.
- Wiring Pi permite establecer la espera por interrupción asociada a un pin de entrada
- cuando se produzca la interrupción se ejecutará una rutina de atención.

Interrupciones	
Función	Funcionamiento
<code>int wiringPiISR (int pin, int edgeType, void (*function)(void)) ;</code>	Llama a la rutina de atención asociada al pin especificado. edgeType: INT_EDGE_FALLING, INT_EDGE_RISING, INT_EDGE_BOTH, o INT_EDGE_SETUP

Lectura de una entrada por interrupción

```
/* Programa interrupcion.c
Enciende un LED conectado al pin físico 7
(pin lógico Wiring Pi: 7) y espera a la
interrupción generada por el pulsador
(normalmente en alto) conectado al pin
físico 5 (pin lógico Wiring Pi: 9)*/
```

```
#include <stdio.h>
#include <wiringPi.h>
#include <softPwm.h>
#define LED_PIN 7
#define BUTTON_PIN 9
static volatile int fin=0;

/*Rutina de atención a la interrupción*/
void esperaInterrupcion(void)
{
    fin=1;
    printf("Se ha pulsado el pulsador: %d
    \n", fin);
}
```

```
/*Programa principal*/
int main()
{
    int i, x, led;
    wiringPiSetup();
    pinMode(LED_PIN,OUTPUT);

    /*Activa la espera por interrupción*/
    wiringPiISR(BUTTON_PIN, INT_EDGE_BOTH,
        &esperaInterrupcion);

    /*Enciende el LED mientras fin sea falso*/
    printf("LED encendido. Esperando
    interrupción\n ");
    while (fin==0)
    {
        digitalWrite(LED_PIN,HIGH);
        delay (900);
        digitalWrite(LED_PIN,LOW);
        delay (100);
    }
    return 0;
}
```

Multithreading en Wiring Pi

- Wiring Pi permite desencadenar threads o hilos que se ejecutan concurrentemente con el programa principal.
- Además aporta instrucciones de bloqueo para implementar acceso a las variables y recursos compartidos en exclusión mutua.

Threads: concurrencia, comunicación y sincronización	
Función	Funcionamiento
<code>int piHiPri (int prioridad);</code>	Establece la prioridad del programa/hilo [0-90] (si se ejecuta como root)
<code>int piThreadCreate (nombre);</code>	Crea un subproceso cuyo c es una función previamente declarada usando PI_THREAD
<code>piLock (int keyNum) ;</code>	Bloquea el acceso a una variable. keyNum: [1-3]
<code>piUnlock (int keyNum);</code>	Desbloquea el acceso a una variable. keyNum: [1-3]

Uso de threads

```
/*Programa test threads.c :Crea 2 threads  
que se ejecutan en paralelo con el main*/
```

```
//librerias necesarias para el programa  
#include <wiringPi.h>  
#include <softPwm.h>  
#include <stdio.h>
```

```
//Código de cada thread
```

```
PI_THREAD (thread1) {
```

```
{  
    while(true)  
    {  
        ...  
    }  
}
```

```
PI_THREAD (thread2) {
```

```
{  
    while(true)  
    {  
        ...  
    }  
}
```

```
int main(void)
```

```
{  
    //creación de los threads  
    printf("Main: creando threads\n");  
    int r,v;  
    /* @piThreadCreate: creación de un  
    thread. Param: nombre del thread;  
    Return: 0 si no falla */  
    r = piThreadCreate(thread1);  
    v = piThreadCreate(thread2);  
    if((r != 0) || (v != 0))  
    {  
        printf("Error al crear un thread!\n");  
        return 1;  
    }  
    for(;;)//Prog. de control: no termina  
    {  
        ...  
    }  
    return 0;  
}
```

Materiales

- Putty
 - Cliente Secure SHell, linea de comandos remota
 - <http://www.putty.org/>
- NetBeans 8.0.2
 - Compilador C/C++
 - <https://netbeans.apache.org/download/>
 - Manuales
 - RaspberriPI para robótica
 - Presentación de la práctica