

Programing RaspberryPi

Tassadaq Hussain

Assistant Professor: Riphah International University

Research Collaborations: Microsoft Barcelona Supercomputing Center

University of Valenciennes, France (CNRS UMR)

UCERD Pvt Ltd Pakistan

Programming Languages

C/C++, Python, Java, PHP, bash scripting, AWK, Assembly language, LUA, JS scripting, Haskell, Erlang, Forth, Pascal, TeX, SQL, Ruby, Smalltalk and so on..

it's by no means a complete list.

GCC Compiler

```
gcc -lib -lib filename.c -o name
```

CodeBlocks

The screenshot displays the Code::Blocks IDE with the following components:

- Menu Bar:** File, Edit, View, Search, Project, Build, Debug, Tools, Plugins, Settings, Help.
- Toolbar:** Standard IDE icons for file operations, editing, and execution. The 'Debug' dropdown menu is currently selected.
- Project Manager:** Shows a workspace named 'timer' with a sub-folder 'Sources'.
- Code Editor:** Displays the following C++ code in `main.cpp`:

```
24 struct timespec start, end;
25 int i;
26
27 /* measure monotonic time */
28 clock_gettime(CLOCK_MONOTONIC, &start); /* mark start time */
29 sleep(1); /* do stuff */
30 clock_gettime(CLOCK_MONOTONIC, &end); /* mark the end time */
31
32 diff = BILLION * (end.tv_sec - start.tv_sec) + end.tv_nsec - start.tv_nsec;
33 printf("elapsed time = %llu nanoseconds\n", (long long unsigned int) diff);
34
35 /* now re-do this and measure CPU time */
36 /* the time spent sleeping will not count (but there is a bit of overhead */
37 clock_gettime(CLOCK_PROCESS_CPUTIME_ID, &start); /* mark start time */
38 sleep(1); /* do stuff */
39 clock_gettime(CLOCK_PROCESS_CPUTIME_ID, &end); /* mark the end time */
40
41 diff = BILLION * (end.tv_sec - start.tv_sec) + end.tv_nsec - start.tv_nsec;
42 printf("elapsed process CPU time = %llu nanoseconds\n", (long long unsigned int) diff);
43
44 exit(0);
45 }
46
47
```
- Logs & others:** Shows tabs for Code::Blocks, Search results, Build log, Build messages, and Debugger. The Build messages tab is active, but it is currently empty.
- Status Bar:** Shows the file path `/home/thussain/Dropbox/R_n_D/training/IN_House_Training/respbarry_pi/labs/timer/timer/main.cpp`, encoding (Unix (LF)), font size (UTF-8), cursor position (Line 1, Column 1), and mode (Insert).

Demo: LED Programming

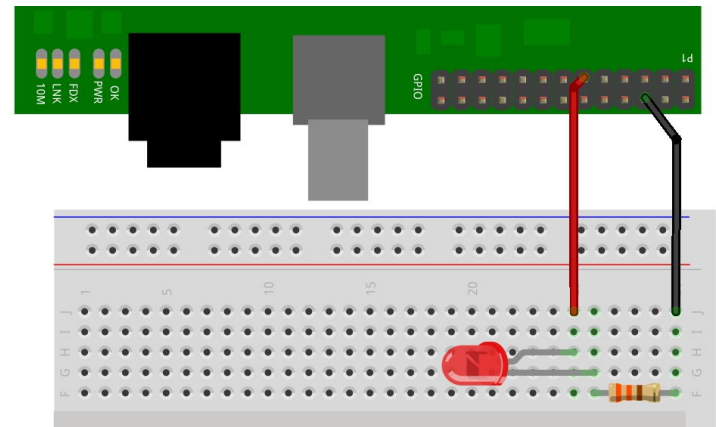
```
#include <stdio.h>
#include <wiringPi.h>
// LED Pin - wiringPi pin 0 is BCM_GPIO 17.
#define LED 0
int main (void)
{ printf ("Raspberry Pi blink\n");
  wiringPiSetup ();
  pinMode (LED, OUTPUT);
  for (;;)
  {
    digitalWrite (LED, HIGH); // On
    delay (500); // mS
    digitalWrite (LED, LOW); // Off
    delay (500);
  } return 0 ; }
```

Raspberry Pi2 GPIO Header

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

Rev. 1
28/01/2014

<http://www.element14.com>



Programming Demo

```

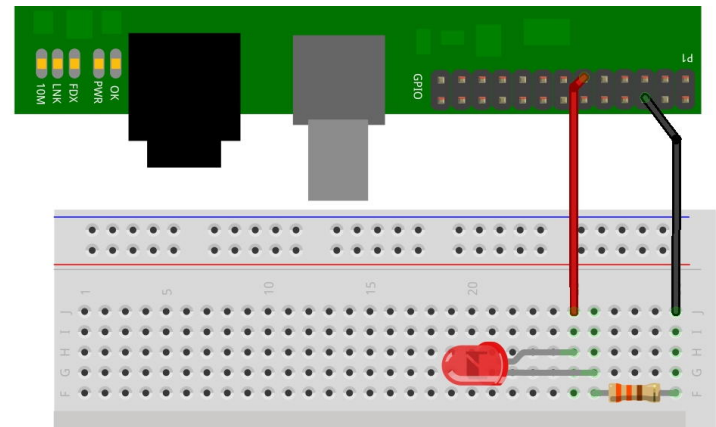
#include <stdio.h>
#include <wiringPi.h>
// LED Pin - wiringPi pin 0 is BCM_GPIO 17.
#define LED 0
int main (void)
{ printf ("Raspberry Pi blink\n");
  wiringPiSetup ();
  pinMode (LED, OUTPUT);
  for (;;)
  {
    digitalWrite (LED, HIGH); // On
    delay (500); // mS
    digitalWrite (LED, LOW); // Off
    delay (500);
  } return 0 ; }

```

Raspberry Pi2 GPIO Header

Pin#	NAME	NAME	Pin#
01	3.3v DC Power	DC Power 5v	02
03	GPIO02 (SDA1 , I2C)	DC Power 5v	04
05	GPIO03 (SCL1 , I2C)	Ground	06
07	GPIO04 (GPIO_GCLK)	(TXD0) GPIO14	08
09	Ground	(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)	(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)	Ground	14
15	GPIO22 (GPIO_GEN3)	(GPIO_GEN4) GPIO23	16
17	3.3v DC Power	(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)	Ground	20
21	GPIO09 (SPI_MISO)	(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)	(SPI_CE0_N) GPIO08	24
25	Ground	(SPI_CE1_N) GPIO07	26
27	ID_SD (I2C ID EEPROM)	(I2C ID EEPROM) ID_SC	28
29	GPIO05	Ground	30
31	GPIO06	GPIO12	32
33	GPIO13	Ground	34
35	GPIO19	GPIO16	36
37	GPIO26	GPIO20	38
39	Ground	GPIO21	40

Rev. 1
28/01/2014 <http://www.element14.com>



Compiling

```
gcc -libraries -c filename.c -o output.out
```

```
gcc -lwiringPi -lwiringPiDev -c blinking.c -o blink.out
```


Arduino: LED Blinking

```
int led0 = 22;  
int led1 = 23;  
int led2 = 24;  
int led3 = 25;
```

```
// the setup routine runs once when you  
// press reset:
```

```
void setup() {  
    // initialize the digital pin as an output.  
    pinMode(led0, OUTPUT);  
    pinMode(led1, OUTPUT);  
    pinMode(led2, OUTPUT);  
    pinMode(led3, OUTPUT);  
}
```

```
// the loop routine runs over and over again forever:
```

```
void loop() {  
    digitalWrite(led0, HIGH); // turn the LED on (HIGH is  
    // the voltage level)  
    digitalWrite(led1, LOW); // turn the LED off (Low is the  
    // voltage level)  
    digitalWrite(led2, LOW);  
    digitalWrite(led3, LOW);  
    delay(1000); // wait for a second  
    digitalWrite(led0, LOW);  
    digitalWrite(led1, HIGH);  
    digitalWrite(led2, LOW);  
    digitalWrite(led3, LOW);  
    delay(1000); // wait for a second  
    digitalWrite(led0, LOW);  
    digitalWrite(led1, LOW);  
    digitalWrite(led2, HIGH);  
    digitalWrite(led3, LOW);  
    delay(1000); // wait for a second  
    digitalWrite(led0, LOW);  
    digitalWrite(led1, LOW);  
    digitalWrite(led2, LOW);  
    digitalWrite(led3, HIGH);  
    delay(1000); // wait for a second  
}
```


Soft and USB Serial Comm

```
#include <SoftwareSerial.h>
```

```
SoftwareSerial mySerial(0, 1); // RX, TX  
char serial_data='c';
```

```
void setup()  
{  
    // set the data rate for the SoftwareSerial port  
    mySerial.begin(9600);  
    mySerial.println("Serial Print");  
}
```

```
void loop() // run over and over  
{  
    serial_data=mySerial.read();  
    if(serial_data !='c')  
        mySerial.write(serial_data);  
}
```

Stepper Motor Controller

```
#include <stdio.h>
#include <stdlib.h>
#include <wiringPi.h>
#define LED0 0 // pin number 11
#define LED1 2 // pin number 13
#define LED2 3 // pin number 15
#define LED3 1 // pin number 10
int main (void)
{ printf ("Raspberry Pi blinking LEDs\n");
  wiringPiSetup ();
  pinMode (LED0, OUTPUT);
  pinMode (LED1, OUTPUT);
  pinMode (LED2, OUTPUT);
  pinMode (LED3, OUTPUT);
  for (;;)
  {
    digitalWrite (LED0, HIGH); // On
    digitalWrite (LED1, LOW); // off
    digitalWrite (LED2, LOW); // off
    digitalWrite (LED3, LOW); // off
    delay (500); // mS
    digitalWrite (LED0, LOW); // off
    digitalWrite (LED1, HIGH); // On
    digitalWrite (LED2, HIGH); // On
    digitalWrite (LED3, LOW); // off
    delay (500); // mS
    digitalWrite (LED0, LOW); // off
    digitalWrite (LED1, LOW); // off
    digitalWrite (LED2, LOW); // off
    digitalWrite (LED3, HIGH); // On
    delay (500); // mS
  } return 0 ; }
```

Serial Comm

```
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <wiringPi.h>
#include <wiringSerial.h>
int main ()
{
    int fd ;
    int count ;
    unsigned int nextTime ;
    if ((fd = serialOpen ("/dev/ttyAMA0", 115200)) < 0)
    {
        fprintf (stderr, "Unable to open serial device: %s\n",
                strerror (errno)) ;
        return 1 ;
    }
    if (wiringPiSetup () == -1)
    {
```

```
        fprintf (stdout, "Unable to start wiringPi: %s\n", strerror (errno)) ;
        return 1 ;
    }
    nextTime = millis () + 300 ;
    for (count = 0 ; count < 256 ; )
    {
        if (millis () > nextTime)
        {
            printf ("\nOut: %3d: ", count) ;
            fflush (stdout) ;
            serialPutchchar (fd, count) ;
            nextTime += 300 ;
            ++count ;
        }

        delay (3) ;
        while (serialDataAvail (fd))
        {
            printf (" -> %3d", serialGetchar (fd)) ;
            fflush (stdout) ;
        }
    }
    printf ("\n") ;
    return 0 ;
}
```