Entrada-Saída no RPi - GPIO

PSI2653 – Meios Eletrônicos Interativos 1

Prof. Sergio Takeo Kofuji

1º. Semestre 2017

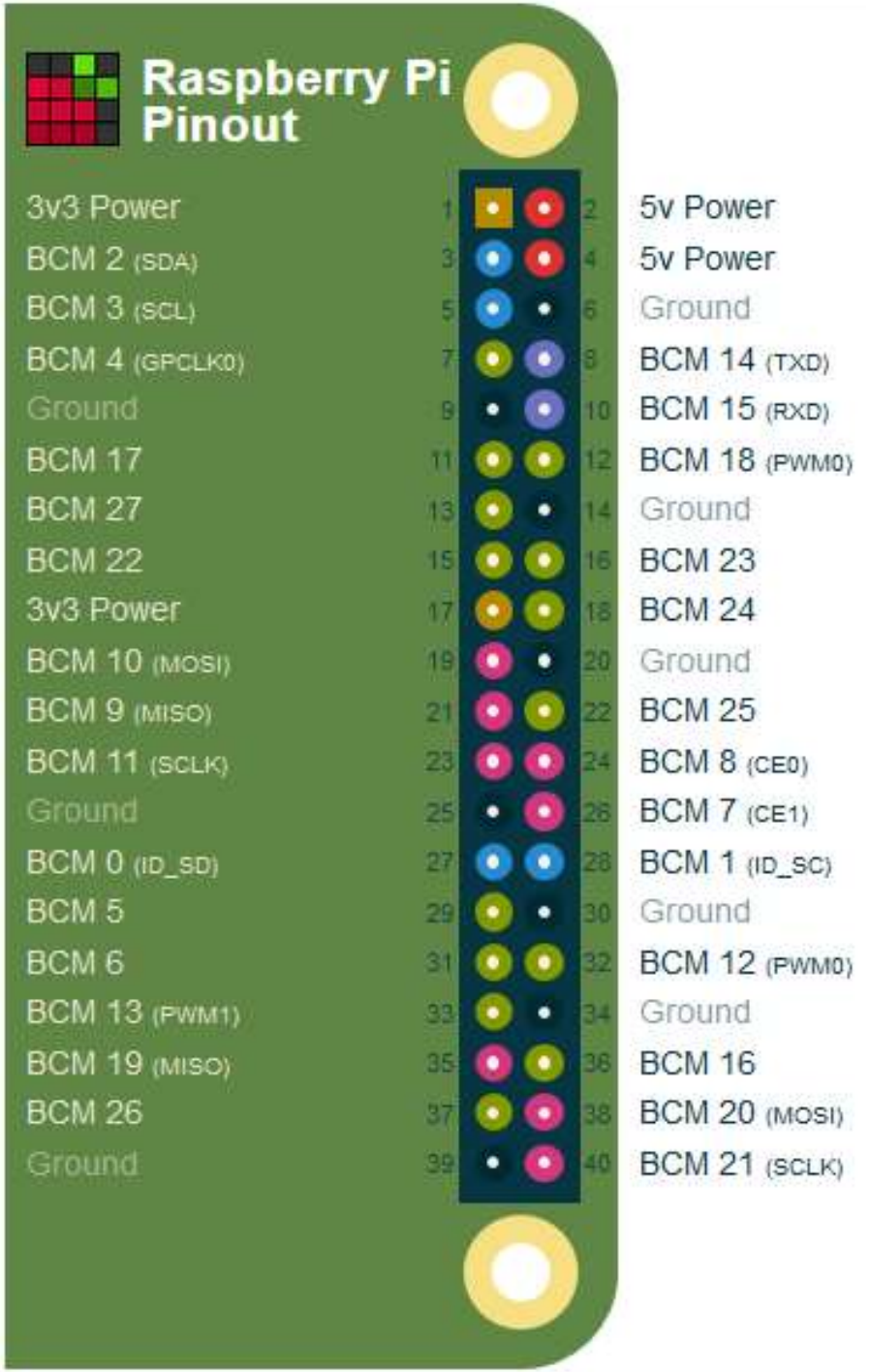
Sumário

- O GPIO
- Acessando o GPIO através do SysFS
- Acessando o GPIO através da biblioteca WiringPi

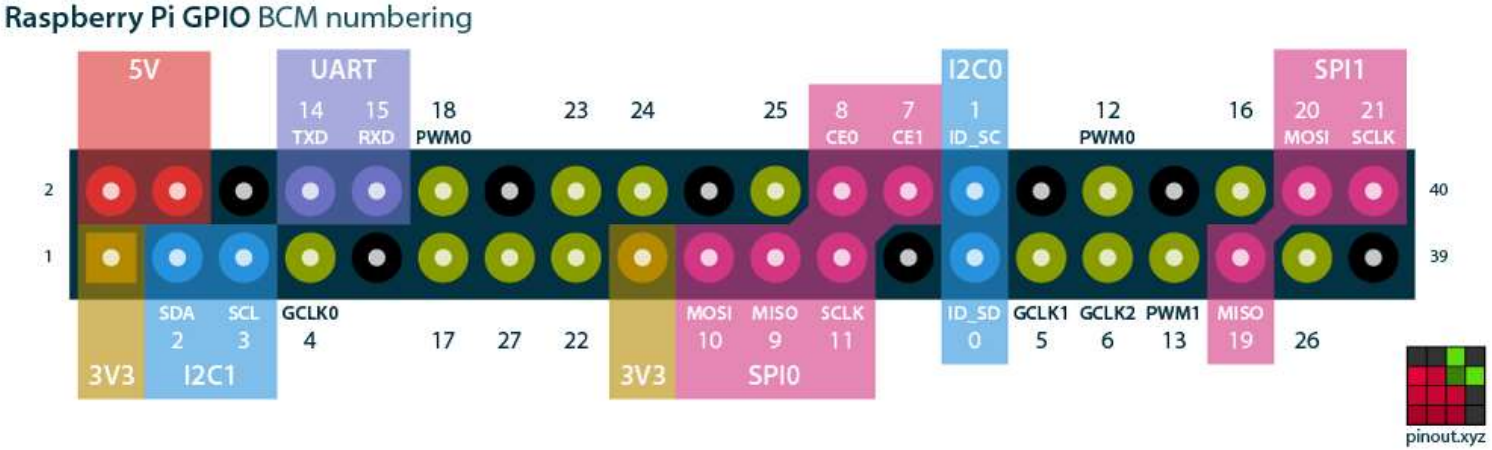
O GPIO

SoC utilizados nos Raspberry PI

- BMC2835 – Raspberyy PI A, B, B+
- BMC2836 – Raspberry PI 2 B
  - Arquitetura ARMv7-A
  - CORTEX A7
    - http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0464f/index.html
- BMC2837 – Raspberry PI 3 B
  - Arquitetura ARMv8, 64 bits
  - CORTEX A53, 4 núcleos
    - https://developer.arm.com/products/processors/cortex-a/cortex-a53
    - http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0500e/index.html



Portal pinout: mostrando a correspondência dos pinos do conector principal (40 pinos) com os pinos do chip Broadcom BCM - Broadcom



Fonte: https://pinout.xyz/#

Alternate Function							Alternate Function
	3.3V PWR	1		2	5V PWR		
I2C1 SDA	GPIO 2	3		4	5V PWR		
I2C1 SCL	GPIO 3	5		6	GND		
	GPIO 4	7		8	UART0 TX		
	GND	9		10	UART0 RX		
	GPIO 17	11		12	GPIO 18		
	GPIO 27	13		14	GND		
	GPIO 22	15		16	GPIO 23		
	3.3V PWR	17		18	GPIO 24		
SPI0 MOSI	GPIO 10	19		20	GND		
SPI0 MISO	GPIO 9	21		22	GPIO 25		
SPI0 SCLK	GPIO 11	23		24	GPIO 8		SPI0 CS0
	GND	25		26	GPIO 7		SPI0 CS1
	Reserved	27		28	Reserved		
	GPIO 5	29		30	GND		
	GPIO 6	31		32	GPIO 12		
	GPIO 13	33		34	GND		
SPI1 MISO	GPIO 19	35		36	GPIO 16		SPI1 CS0
	GPIO 26	37		38	GPIO 20		SPI1 MOSI
	GND	39		40	GPIO 21		SPI1 SCLK

Acessando o GPIO através do SysFS

GPIO

- Há diversas formas de acessar o GPIO, usando diversas linguagens de programação, como mostrado na página http://elinux.org/RPi_GPIO_Code_Samples#sysfs
  - C, C++, C#, Ruby, Perl, Python, Scratch, Java, Shell,

Usando o Sistema de Arquivos

- Criar e executar o seguinte shell script (blink.sh) usando sudo:
  - sudo ./blink.sh

```
#!/bin/sh
echo 17 > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio17/direction
while true
do
        echo 1 > /sys/class/gpio/gpio17/value
        sleep 1
        echo 0 > /sys/class/gpio/gpio17/value
        sleep 1
done
```

Make the pin available for other applications using with the command: echo 17 > /sys/class/gpio/unexport

Detalhando…

- Criar o shell script usando um editor, como o ***nano***:
  - ***nano blink.sh***
    - Cut and paste the previous slide to nano window
    - ***Ctrl-w*** to save then ***Ctrl-x*** to exit nano
  - Mude as permissões do blink.sh: ***chmod 755 blink.sh***
  - Execute o blink.sh: ***sudo ./blink.sh*** (no diretório do blink.sh)
- Após a execução do script, o LED vai ficar piscando indefinidamente. Para abortar, execute o comando ***Ctrl-c***
- Todos os commandos do script podem ser emitidos, um por vez, através da linha de comandos, começando pelo comando ***sudo -i para executar o*** a root shell---observe a mudança do prompt…
- Examine os arquivos e seus conteúdos no diretório /sys/class/gpio/ e seus subdiretórios

/sys/class/gpio/

- Em Linux todo dispositivo é arquivo: /dev/ttyUSB0, /sys/class/net/eth0/address, /dev/mmcblk0p2,…
- sysfs permite acesso ao dispositivo em /sys/class
  - usuário (ou Código no *user-space*) acessa dispositivos gerenciados pelo system (kernel)
- Vantagens/ Desvantagens
  - Permite acesso convencional no userspace aos pinos do dispositivo
  - mode switch to kernel -> action in kernel -> mode switch to use -> and could have a context switch
  - Muito mais lento que o digitalWrite()/digitalRead() do Arduino

SysFs

- Fonte: https://www.embarcados.com.br/gpio-da-raspberry-pi-linguagem-c/
- Alguns diretórios

  do SysFs

```
pi@raspberrypi:/sys $ pwd
/sys
pi@raspberrypi:/sys $ ls -la
total 4
dr-xr-xr-x 12 root root    0 Nov 25 11:37 .
drwxr-xr-x 21 root root 4096 Sep 13 03:08 ..
drwxr-xr-x  2 root root    0 Nov 25 12:48 block
drwxr-xr-x 18 root root    0 Nov 25 12:48 bus
drwxr-xr-x 48 root root    0 Nov 25 11:37 class
drwxr-xr-x  4 root root    0 Nov 25 12:48 dev
drwxr-xr-x  9 root root    0 Nov 25 12:48 devices
drwxr-xr-x  3 root root    0 Nov 25 12:48 firmware
drwxr-xr-x  5 root root    0 Nov 25 12:48 fs
drwxr-xr-x  8 root root    0 Nov 25 12:48 kernel
```

Alguns diretórios do SysFs

- Block
- Bus
- Class
- Devices
  - Devices
  - Drivers
- Firmware
- Module

Diretório /sys/class/gpio

- Interfaces de controle usadas para permitir ao espaço de usuário o controle do pino de GPIO
- GPIOs
- Instâncias de Controle do GPIO

Figura 2 - Estrutura do diretório /sys/class/gpio/

Export e Unexport

- Para que possamos controlar um pino do GPIO, precisamos que este controle, feito no núcleo do S.O, exportado para o espaço do usuário
  - export: O programa no espaço de usuário solicita ao kernel o controle do GPIO no espaço de usuário.
  - unexport: Reverte as ações efetuadas pelo export

GPIO – programação do pino como entrada ou saída

- Através do arquivo value, é possível realizar a leitura ou escrita no pino de acordo com as configurações definidas no método **direction**.
- O arquivo edge nos permite verificar se a tensão sofreu uma borda de subida, descida ou ambas, já o método **active_low**, permite a inversão dos níveis lógicos de leitura e/ou gravação

Programas C

- Acesse o repositório
  - https://github.com/leal-freitas/rpi-gpio

Exportando o pino

```
1  bool export_gpio(int pin)
2  {
3       arquive = open ("/sys/class/gpio/export", O_WRONLY);
4       if (arquive==-1)
5       {
6               printf("Arquivo abriu incorretamente\n");
7               return false;
8       }
9       snprintf(buffer, 3, "%d", pin);
10      if(write(arquive, buffer, 3) == -1)
11      {
12              close(arquive);
13              return false;
14      }
15
16
17      close(arquive);
18
19
20      return true;
21 }
```

Direção do Pino

```
1  bool direction_gpio(int pin, int direction)
2  {
3       arquive=0;
4       snprintf(path, 35, "/sys/class/gpio/gpio%d/direction", pin);
5       arquive = open (path, O_WRONLY);
6       if (arquive==-1)
7       {
8               return false;
9       }
10      snprintf(buffer, 3, "%d", pin);
11      if (write( arquive, ((direction == INPUT)?"in":"out"), 3 )==-1)
12      {
13              close(arquive);
14              return false;
15      }
16      close(arquive);
17      return true;
18 }
```

Efetuando a Leitura

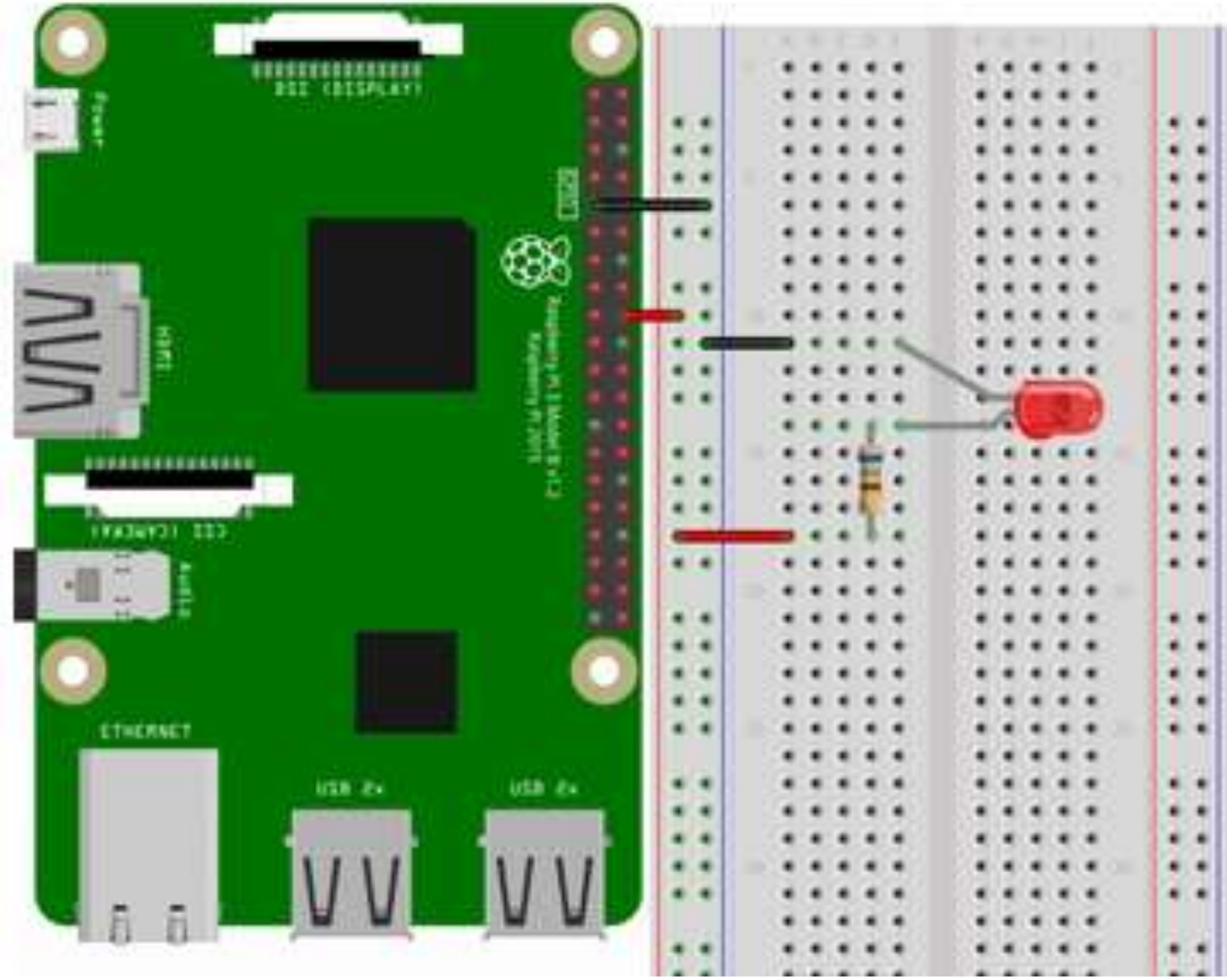
```
1  int value_in_gpio(int pin, int value)
2  {
3       arquive=0;
4       char retorno[3];
5       snprintf(path, 35, "/sys/class/gpio/gpio%d/value", pin);
6       arquive = open(path, O_RDONLY);
7       //printf("Descritor do arquivo: %d \n", arquive);
8       if (arquive == -1)
9       {
10              return false;
11      }
12      if (read(arquive, retorno, 3) == -1)
13      {
14              close(arquive);
15              return false;
16      }
17      close(arquive);
18      printf("Valor do pino: %c \n", retorno[0]);
19
20
21      return atoi(retorno);
22 }
```

Escrita no Pino

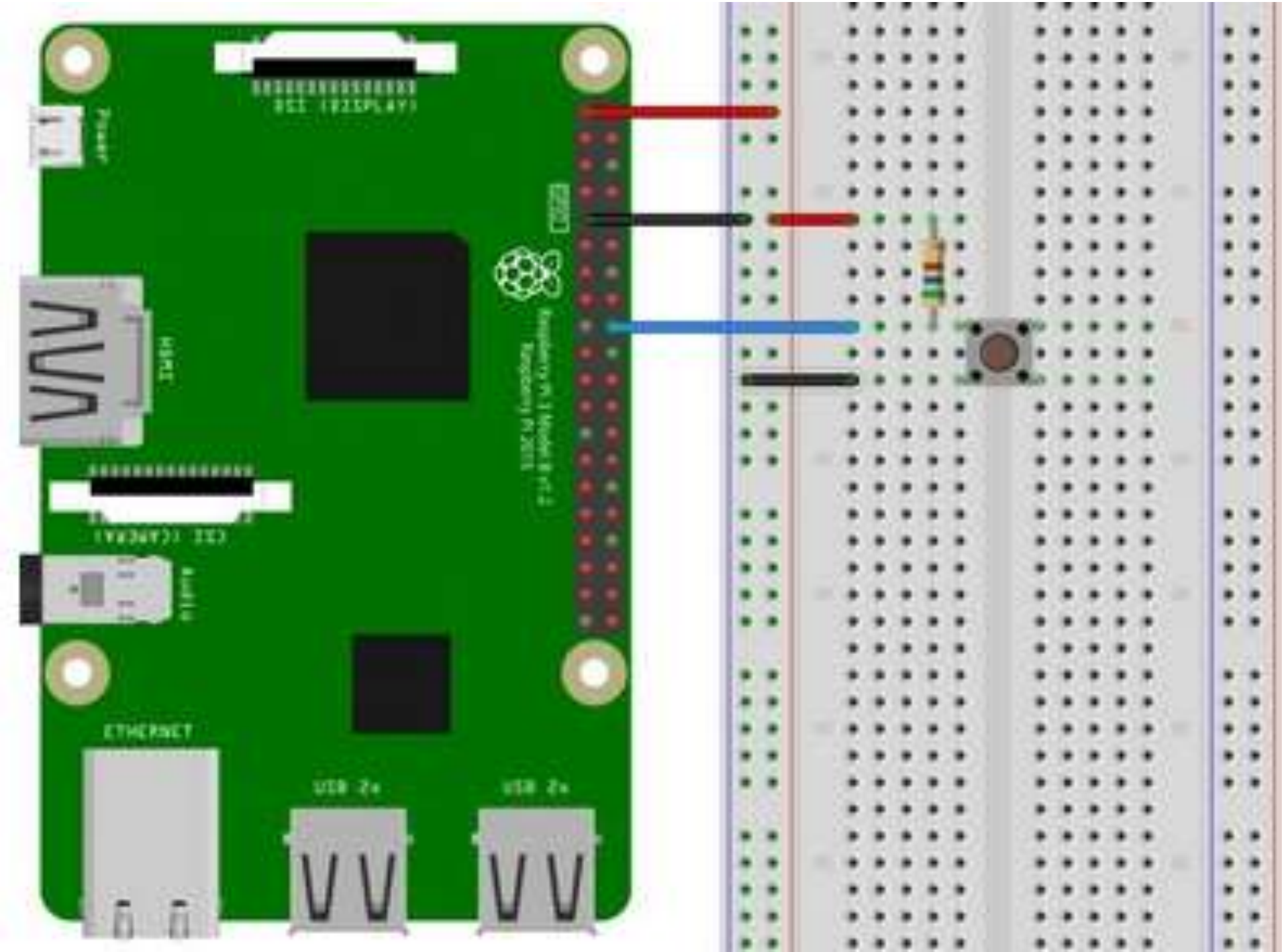
```
1  bool value_gpio(int pin, int value)
2  {
3       arquive=0;
4       snprintf(path, 35, "/sys/class/gpio/gpio%d/value", pin);
5       arquive = open(path, O_WRONLY);
6       if (arquive == -1)
7       {
8               return false;
9       }
10      if (write (arquive, ((value == HIGH)?"1":"0"), 1) == -1)
11      {
12              close(arquive);
13              return false;
14      }
15      close(arquive);
16      return true;
17 }
```

Unexport do Pino

```
1  bool unexport_gpio(int pin)
2  {
3       arquive = open ("/sys/class/gpio/unexport", O_WRONLY);
4       if (arquive==-1)
5       {
6            printf("Arquivo abriu incorretamente\n");
7            return false;
8       }
9       if(write(arquive, buffer, 3) == -1)
10      {
11           close(arquive);
12           return false;
13      }
14      return true;
15 }
```

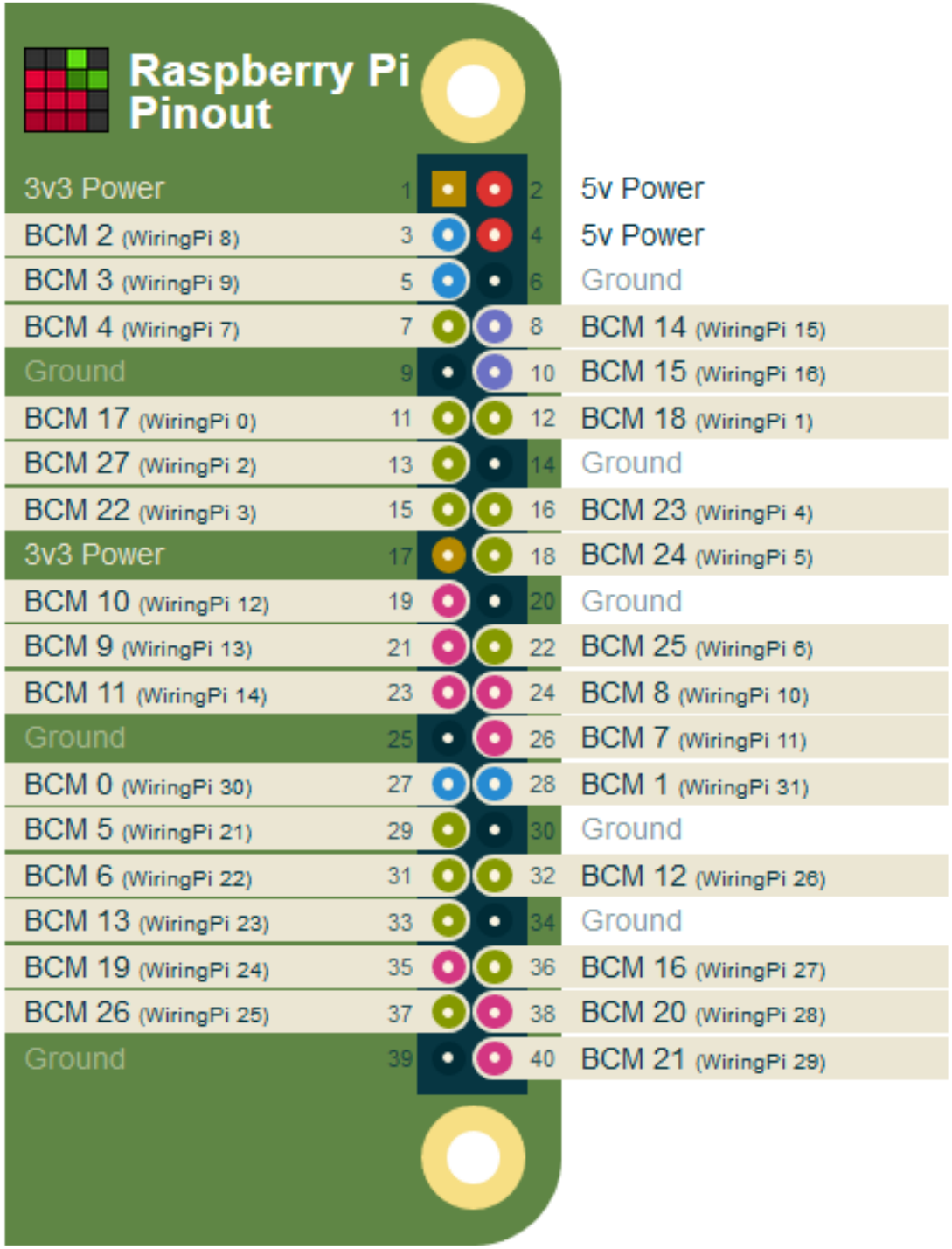


SAÍDA


DSI (DISPLAY)
Power
HDMI
CSI (CAMERA)
Audio
ETHERNET
USB 2x
USB 2x

ENTRADA

WiringPi



Portal pinout.: mostrando o mapeamento dos pinos para o WiringPi

WiringPi is a PIN based GPIO access library written in C for the BCM2835 used in the Raspberry Pi. It's released under the GNU LGPLv3 license and is usable from C, C++ and RTB (BASIC) as well as many other languages with suitable wrappers (See below) It's designed to be familiar to people who have used the Arduino "wiring" system.

WiringPi includes a command-line utility gpio which can be used to program and setup the GPIO pins. You can use this to read and write the pins and even use it to control them from shell scripts.

WiringPi library

- Biblioteca de acesso ao GPIO escrito em C para o BCM2835
  - Writes/reads the base address of the memory allocated to the GPIO
- Similar ao Wiring library do Arduino
- Características:
  - command-line utility ***gpio***
  - supports analog reading and writing
  - More
- Instalação: instruções

Blinking lights com Wiring

```
#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 () ;                  // ← note the setup method chosen
  pinMode (LED, OUTPUT) ;

  for (;;) {
    digitalWrite (LED, HIGH) ;  // On
    delay (500) ;                    // mS
    digitalWrite (LED, LOW) ;   // Off
    delay (500) ;
  }
  return 0 ;
}
```

Executando blink

- Compile and run the blink program

  gcc -Wall -o blink blink.c -lwiringPi ← compile

  sudo ./blink ← run

- Runs forever---kill with the command ***ctrl-c ctrl-c***
- Note: One of the four wiring ***setup*** functions **must** be called at the start of your program or your program will not work correctly

Usando o utilitário gpio

- O programa ***gpio*** pode ser usado em scripts para manipular os pinos do GPIO
- The ***gpio*** command is designed to be called by a normal user without using the sudo command or logging in as root
- Try at the command line:

  gpio mode 0 out

  gpio write 0 1

- Sets pin 0 as output and then sets the pin to high
- More info on the gpio utility

Exercícios

- Escreva um programa em linguagem C para controle de luminosidade de lâmpada LED por PWM
- Escreva um Programa para leitura de luminosidade através de carga e descarga de um capacitor

Bibliografia

- https://www.embarcados.com.br/gpio-da-raspberry-pi-linguagem-c/

Perguntas?