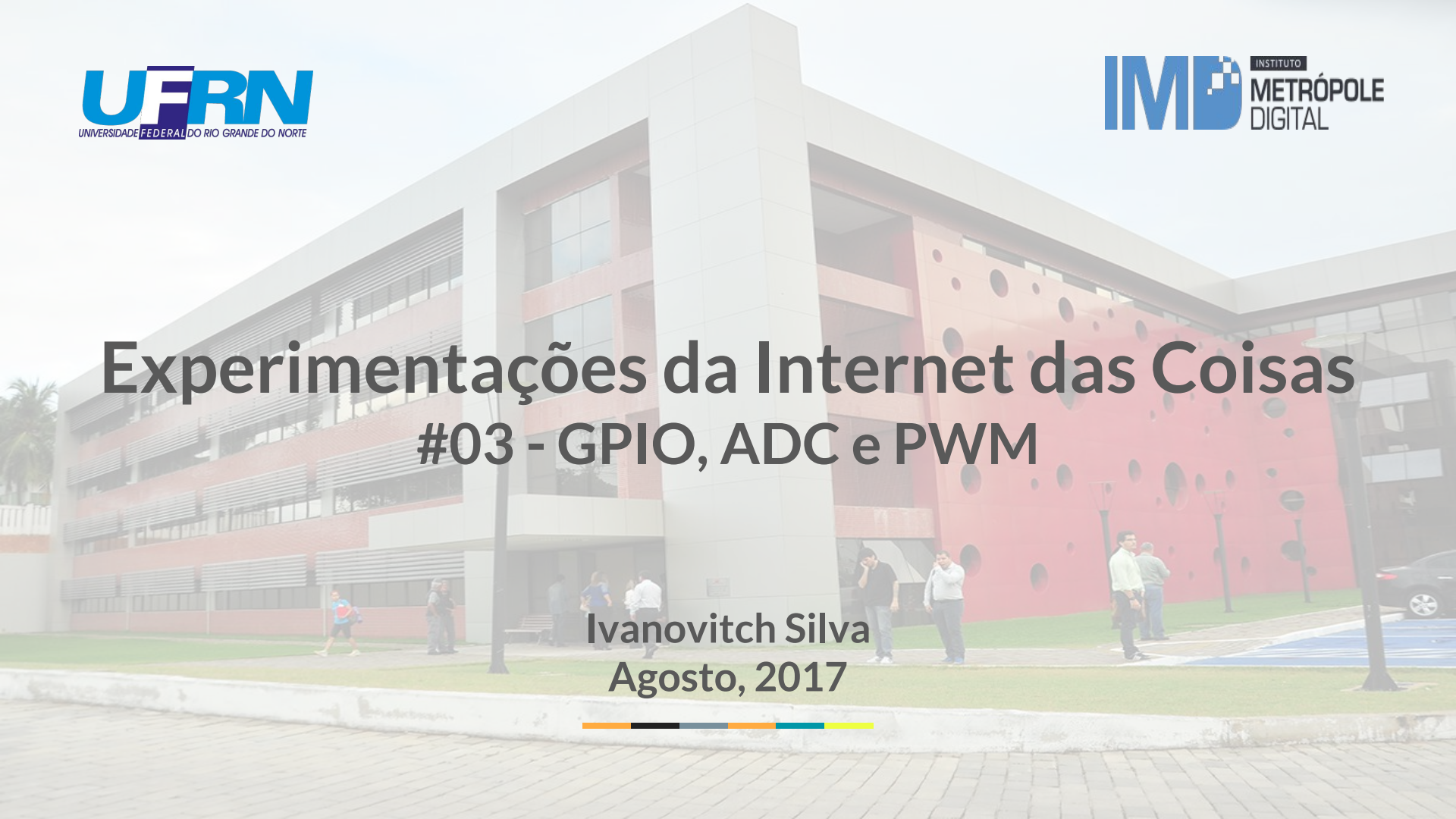


Experimentações da Internet das Coisas

#03 - GPIO, ADC e PWM

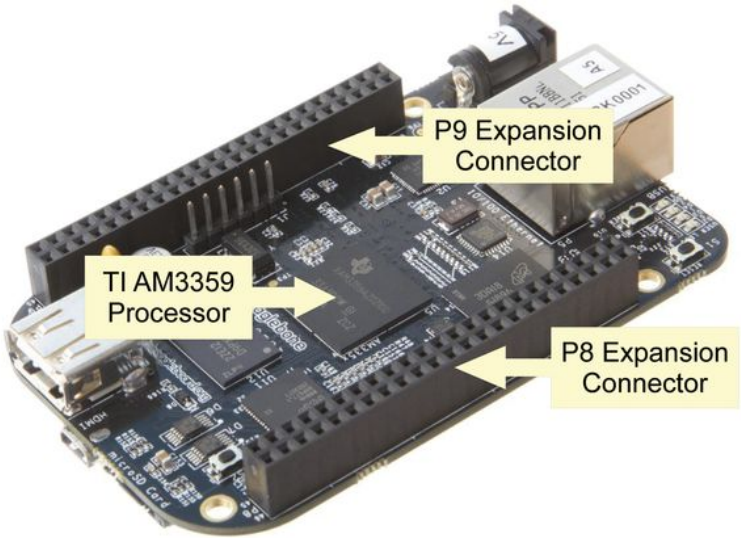
Ivanovitch Silva

Agosto, 2017

General Purpose Input/Output (GPIO)

Comunicação com os Headers P8 e P9

- Saída digital
  - controlar um LED (on|off)
- Entrada digital
  - ler o status de um *swtich*
- Entrada analógica
  - 7 ADC
  - ler dados de um sensor
- Saída analógica
  - usar PWM para controlar motores

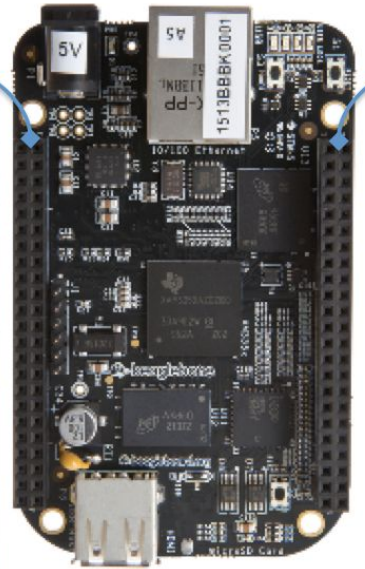


Cape Expansion Headers

Configuração Padrão

P9

DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3
VDD_5V	5	6	VDD_5V
SYS_5V	7	8	SYS_5V
PWR_BUT	9	10	SYS_RESETn
UART4_RXD	11	12	GPIO_60
UART4_TXD	13	14	EHRPWM1A
GPIO_48	15	16	EHRPWM1B
SPI0_CS0	17	18	SPI0_D1
I2C2_SCL	19	20	I2C2_SDA
SPI0_D0	21	22	SPI0_SCLK
GPIO_49	23	24	UART1_TXD
GPIO_117	25	26	UART1_RXD
GPIO_115	27	28	SPI1_CS0
SPI1_D0	29	30	GPIO_112
SPI1_SCLK	31	32	VDD_ADC
AIN4	33	34	GNDA_ADC
AIN6	35	36	AIN5
AIN2	37	38	AIN3
AIN0	39	40	AIN1
GPIO_20	41	42	ECAPPWM0
DGND	43	44	DGND
DGND	45	46	DGND



LEGEND

- Power/Ground/Reset
- Available Digital
- Available PWM
- Shared I2C Bus
- Reconfigurable Digital
- Analog Inputs (1.8V)

P8

DGND	1	2	DGND
MMC1_DAT6	3	4	MMC1_DAT7
MMC1_DAT2	5	6	MMC1_DAT3
GPIO_66	7	8	GPIO_67
GPIO_69	9	10	GPIO_68
GPIO_45	11	12	GPIO_44
EHRPWM2B	13	14	GPIO_26
GPIO_47	15	16	GPIO_46
GPIO_27	17	18	GPIO_65
EHRPWM2A	19	20	MMC1_CMD
MMC1_CLK	21	22	MMC1_DAT5
MMC1_DAT4	23	24	MMC1_DAT1
MMC1_DAT0	25	26	GPIO_61
LCD_VSYNC	27	28	LCD_PCLK
LCD_HSYNC	29	30	LCD_AC_BIAS
LCD_DATA14	31	32	LCD_DATA15
LCD_DATA13	33	34	LCD_DATA11
LCD_DATA12	35	36	LCD_DATA10
LCD_DATA8	37	38	LCD_DATA9
LCD_DATA6	39	40	LCD_DATA7
LCD_DATA4	41	42	LCD_DATA5
LCD_DATA2	43	44	LCD_DATA3
LCD_DATA0	45	46	LCD_DATA1

http://beagleboard.org/support/bone101

Cada pino apresenta 8 modos diferentes de funcionamento, um desses modos é o GPIO

65 possible digital I/Os

4 Controladoras de GPIO (gpio0,gpio1,gpio2,gpio3)
- offset 32
- Número da GPIO (gpio1[18])
  - 1*32 + 18 = 50 (pino 14 de P9)

P9

DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3
VDD_5V	5	6	VDD_5V
SYS_5V	7	8	SYS_5V
PWR_BUT	9	10	SYS_RESETn
GPIO_30	11	12	GPIO_60
GPIO_31	13	14	GPIO_50
GPIO_48	15	16	GPIO_51
GPIO_5	17	18	GPIO_4
I2C2_SCL	19	20	I2C2_SDA
GPIO_3	21	22	GPIO_2
GPIO_49	23	24	GPIO_15
GPIO_117	25	26	GPIO_14
GPIO_115	27	28	GPIO_113
GPIO_111	29	30	GPIO_112
GPIO_110	31	32	VDD_ADC
AIN4	33	34	GNDA_ADC
AIN6	35	36	AIN5
AIN2	37	38	AIN3
AIN0	39	40	AIN1
GPIO_20	41	42	GPIO_7
DGND	43	44	DGND
DGND	45	46	DGND

P8

DGND	1	2	DGND
GPIO_38	3	4	GPIO_39
GPIO_34	5	6	GPIO_35
GPIO_66	7	8	GPIO_67
GPIO_69	9	10	GPIO_68
GPIO_45	11	12	GPIO_44
GPIO_23	13	14	GPIO_26
GPIO_47	15	16	GPIO_46
GPIO_27	17	18	GPIO_65
GPIO_22	19	20	GPIO_63
GPIO_62	21	22	GPIO_37
GPIO_36	23	24	GPIO_33
GPIO_32	25	26	GPIO_61
GPIO_86	27	28	GPIO_88
GPIO_87	29	30	GPIO_89
GPIO_10	31	32	GPIO_11
GPIO_9	33	34	GPIO_81
GPIO_8	35	36	GPIO_80
GPIO_78	37	38	GPIO_79
GPIO_76	39	40	GPIO_77
GPIO_74	41	42	GPIO_75
GPIO_72	43	44	GPIO_73
GPIO_70	45	46	GPIO_71

http://beagleboard.org/support/bone101

Os pinos apresentam uma tolerância max. de 8mA (input), 4mA - 6mA (saída) e 3.3V
ADC são tolerantes a 1.8V

Trabalhar com GPIO requer atenção!!!

GPIO como saída

Antes de conectar qualquer pino na BBB VERIFIQUE se a DIREÇÃO é a correta!!!

P9

DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3
VDD_5V	5	6	VDD_5V
SYS_5V	7	8	SYS_5V
PWR_BUT	9	10	SYS_RESETn
GPIO_30	11	12	GPIO_60
GPIO_31	13	14	GPIO_50
GPIO_48	15	16	GPIO_51
GPIO_5	17	18	GPIO_4
I2C2_SCL	19	20	I2C2_SDA
GPIO_3	21	22	GPIO_2
GPIO_49	23	24	GPIO_15
GPIO_117	25	26	GPIO_14
GPIO_115	27	28	GPIO_113
GPIO_111	29	30	GPIO_112
GPIO_110	31	32	VDD_ADC
AIN4	33	34	GNDA_ADC
AIN6	35	36	AIN5
AIN2	37	38	AIN3
AIN0	39	40	AIN1
GPIO_20	41	42	GPIO_7
DGND	43	44	DGND
DGND	45	46	DGND

P8

DGND	1	2	DGND
GPIO_38	3	4	GPIO_39
GPIO_34	5	6	GPIO_35
GPIO_66	7	8	GPIO_67
GPIO_69	9	10	GPIO_68
GPIO_45	11	12	GPIO_44
GPIO_47	15	16	GPIO_46
GPIO_27	17	18	GPIO_65
GPIO_22	19	20	GPIO_63
GPIO_62	21	22	GPIO_37
GPIO_36	23	24	GPIO_33
GPIO_32	25	26	GPIO_61
GPIO_86	27	28	GPIO_88
GPIO_87	29	30	GPIO_89
GPIO_10	31	32	GPIO_11
GPIO_9	33	34	GPIO_81
GPIO_8	35	36	GPIO_80
GPIO_78	37	38	GPIO_79
GPIO_76	39	40	GPIO_77
GPIO_74	41	42	GPIO_75
GPIO_72	43	44	GPIO_73
GPIO_70	45	46	GPIO_71

manipular GPIO requer privilégios

$sudo su

Melhorando a UX

```
root@beaglebone:/sys/class/gpio# cat ~/.bashrc
# ~/.bashrc: executed by bash(1) for non-login shells.

# Note: PS1 and umask are already set in /etc/profile. You should not
# need this unless you want different defaults for root.
# PS1='${debian_chroot:+($debian_chroot)}\h:\w\$ '
# umask 022

# You may uncomment the following lines if you want `ls' to be colorized:
# export LS_OPTIONS='--color=auto'
# eval "`dircolors`"
# alias ls='ls $LS_OPTIONS'
# alias ll='ls $LS_OPTIONS -l'
# alias l='ls $LS_OPTIONS -lA'
#
```

Após editar o arquivo (e.g usando "nano" como editor) digite $bash

Acessando o GPIO - Diretório de Trabalho

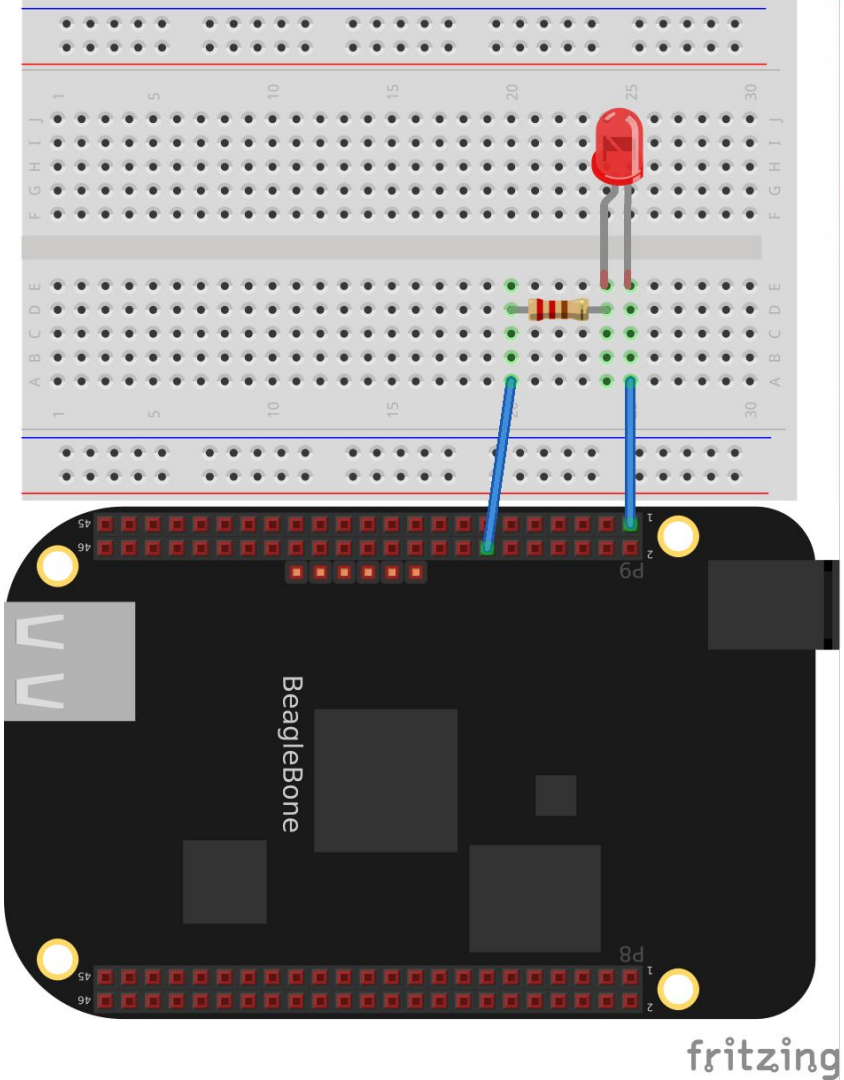
```
root@beaglebone:/sys/class/gpio# ls
export    gpio14  gpio23  gpio31  gpio47  gpio51  gpio67     gpiochip32
gpio112   gpio15  gpio26  gpio4   gpio48  gpio60  gpio68     gpiochip64
gpio114   gpio2   gpio27  gpio44  gpio49  gpio61  gpio69     gpiochip96
gpio115   gpio20  gpio3   gpio45  gpio5   gpio65  gpio7      unexport
gpio116   gpio22  gpio30  gpio46  gpio50  gpio66  gpiochip0
```

Propriedades da porta GPIO

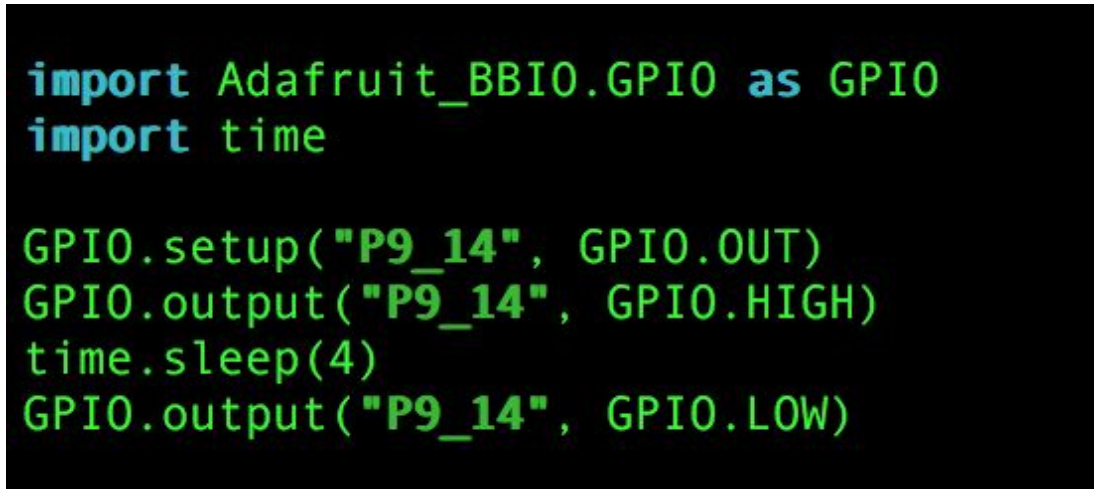
```
[root@beaglebone:/sys/class/gpio/gpio50# ls
 active_low  device  direction  edge  power  subsystem  uevent  value
[root@beaglebone:/sys/class/gpio/gpio50# cat direction
 in
[root@beaglebone:/sys/class/gpio/gpio50# echo out > direction
[root@beaglebone:/sys/class/gpio/gpio50# cat direction
 out
[root@beaglebone:/sys/class/gpio/gpio50# cat value
 0
[root@beaglebone:/sys/class/gpio/gpio50# echo 1 > value
[root@beaglebone:/sys/class/gpio/gpio50# cat value
 1
 root@beaglebone:/sys/class/gpio/gpio50#
```

GPIO como saída - Exemplo

Utilize a porta GPIO50 na direção “out”
Ligue e desligue o LED



Exemplo - Pisca Led



```
import Adafruit_BBIO.GPIO as GPIO
import time

GPIO.setup("P9_14", GPIO.OUT)
GPIO.output("P9_14", GPIO.HIGH)
time.sleep(4)
GPIO.output("P9_14", GPIO.LOW)
```

$sudo python piscaled.py

GPIO como entrada

Antes de conectar qualquer pino na BBB VERIFIQUE se a DIREÇÃO é a correta!!!

O pino GPIO115 possui um resistor interno PULL-DOWN (conectado ao GND).

P9

DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3
VDD_5V	5	6	VDD_5V
SYS_5V	7	8	SYS_5V
PWR_BUT	9	10	SYS_RESETN
GPIO_30	11	12	GPIO_60
GPIO_31	13	14	GPIO_50
GPIO_48	15	16	GPIO_51
GPIO_5	17	18	GPIO_4
I2C2_SCL	19	20	I2C2_SDA
GPIO_3	21	22	GPIO_2
GPIO_49	23	24	GPIO_15
GPIO_117	25	26	GPIO_14
GPIO_115	27	28	GPIO_113
GPIO_111	29	30	GPIO_112
GPIO_110	31	32	VDD_ADC
AIN4	33	34	GNDA_ADC
AIN6	35	36	AIN5
AIN2	37	38	AIN3
AIN0	39	40	AIN1
GPIO_20	41	42	GPIO_7
DGND	43	44	DGND
DGND	45	46	DGND

P8

DGND	1	2	DGND
GPIO_38	3	4	GPIO_39
GPIO_34	5	6	GPIO_35
GPIO_66	7	8	GPIO_67
GPIO_69	9	10	GPIO_68
GPIO_45	11	12	GPIO_44
GPIO_23	13	14	GPIO_26
GPIO_47	15	16	GPIO_46
GPIO_27	17	18	GPIO_65
GPIO_22	19	20	GPIO_63
GPIO_62	21	22	GPIO_37
GPIO_36	23	24	GPIO_33
GPIO_32	25	26	GPIO_61
GPIO_86	27	28	GPIO_88
GPIO_87	29	30	GPIO_89
GPIO_10	31	32	GPIO_11
GPIO_9	33	34	GPIO_81
GPIO_8	35	36	GPIO_80
GPIO_78	37	38	GPIO_79
GPIO_76	39	40	GPIO_77
GPIO_74	41	42	GPIO_75
GPIO_72	43	44	GPIO_73
GPIO_70	45	46	GPIO_71

Como identificar as conf. padrões dos pinos?

Pinagem
http://exploringbeaglebone.com/wp-content/uploads/resources/BBBP8Header.pdf
http://exploringbeaglebone.com/wp-content/uploads/resources/BBBP9Header.pdf

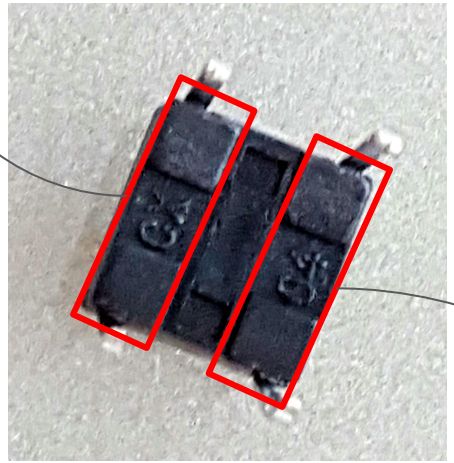
Como identificar as conf. padrões dos pinos?

BIT	AM335x FIELD	DESCRIPTION
0,1,2	mmode	The multiplexer mode: Using the three least-significant bits you can select a mode between 0 and 7, e.g., 000=0, 111=7. This enables each pin to have up to eight different modes.
3	puden	Enable internal pull-up/pull-down resistor: Enable=0, Disable=1.
4	putypesel	Select internal pull-up or pull-down form: Pull-down=0, Pull-up=1.
5	rxactive	*Input Active*: Receiver disabled=0, Input enabled=1. If this bit is set high, the pin will be an input; otherwise, it will be an output.
6	slewctrl	*Slew Control*: Fast=0, Slow=1. *Slew rate* provides control over the rise/fall time of an output. You would only set this value to slow if you were using long interconnects, such as on I^2C buses.

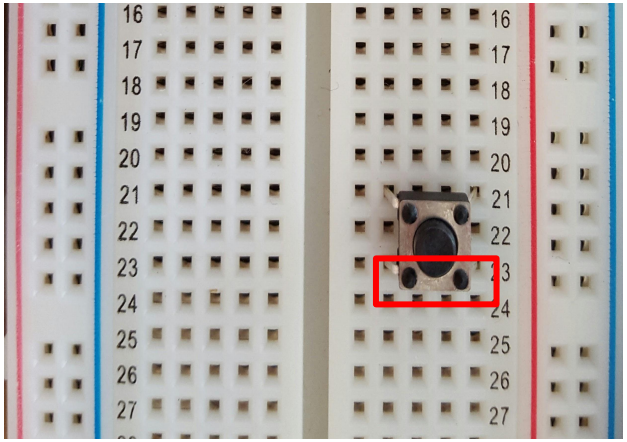
0x27 (0100111) Fast, Input, Pull-Down, Enabled and Mux Mode 7
0x37 (0110111) Fast, Input, Pull-Up, Enabled, Mux Mode 7

Configurando o SWITCH

Devem ficar conectadas

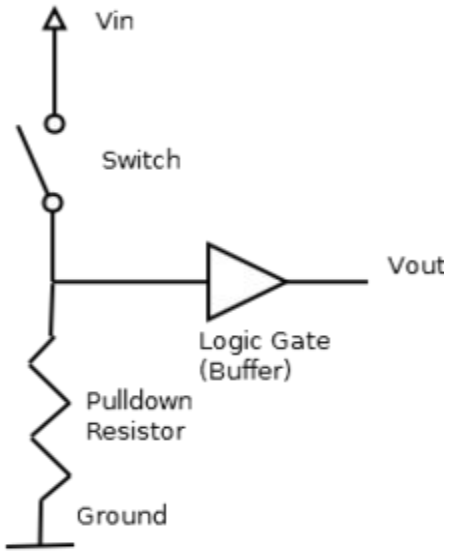


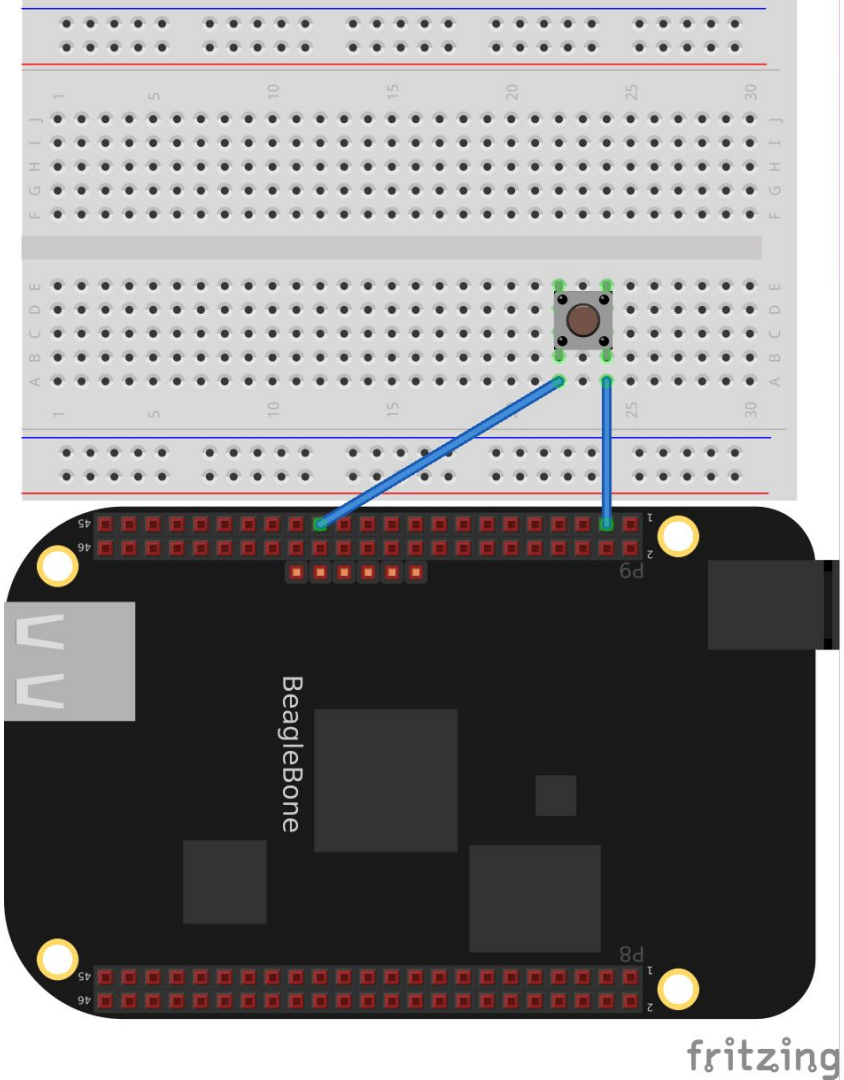
Devem ficar conectadas



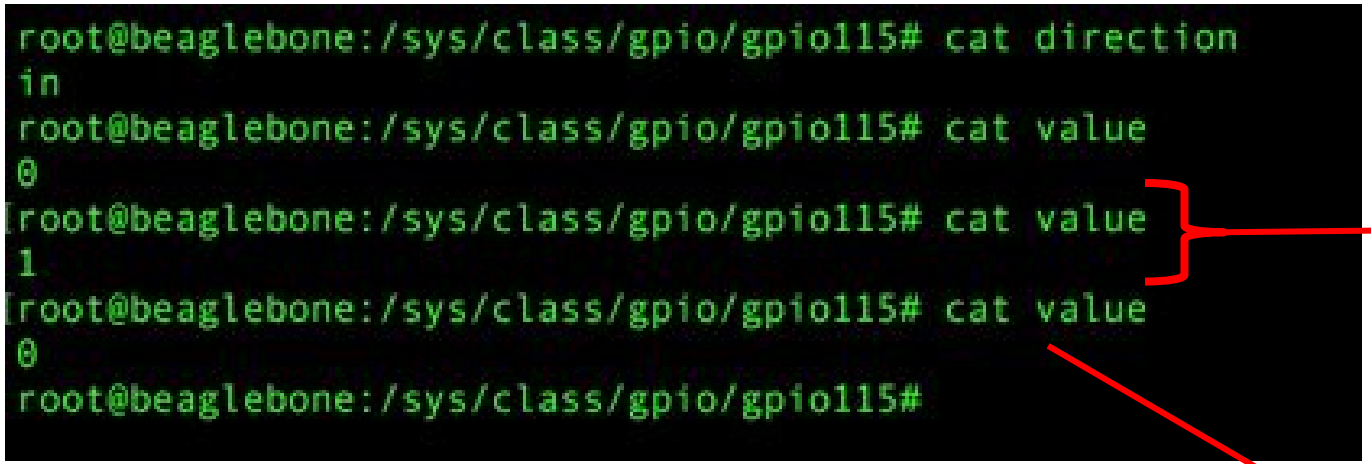
Configurando o SWITCH

Lembre-se que o GPIO115 possui um pull-down nativo





GPIO como entrada



Exemplo - Pressionando o botão

```
import Adafruit_BBIO.GPIO as GPIO
import time

GPIO.setup("P9_27", GPIO.IN)

while True:
        if GPIO.input("P9_27"):
                print("HIGH")
        else:
                print("LOW")
        time.sleep(1)
```

$sudo python botao.py

Desafio I - Dificuldade 1

- Modificar o programa anterior
  - Ficar em um Loop Infinito
  - Verifica se o botão está pressionado
    - Sim? Liga o LED
    - Não? Desliga o LED

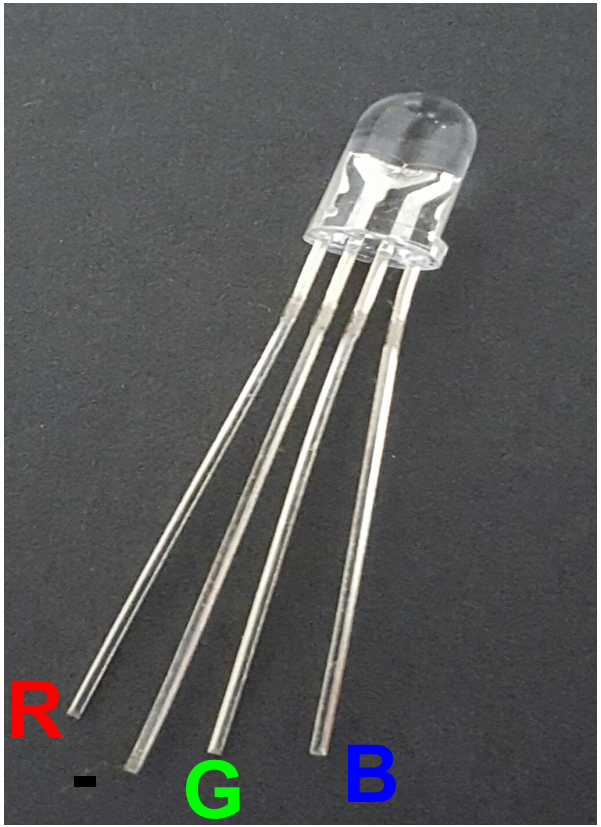
LED RGB

P9

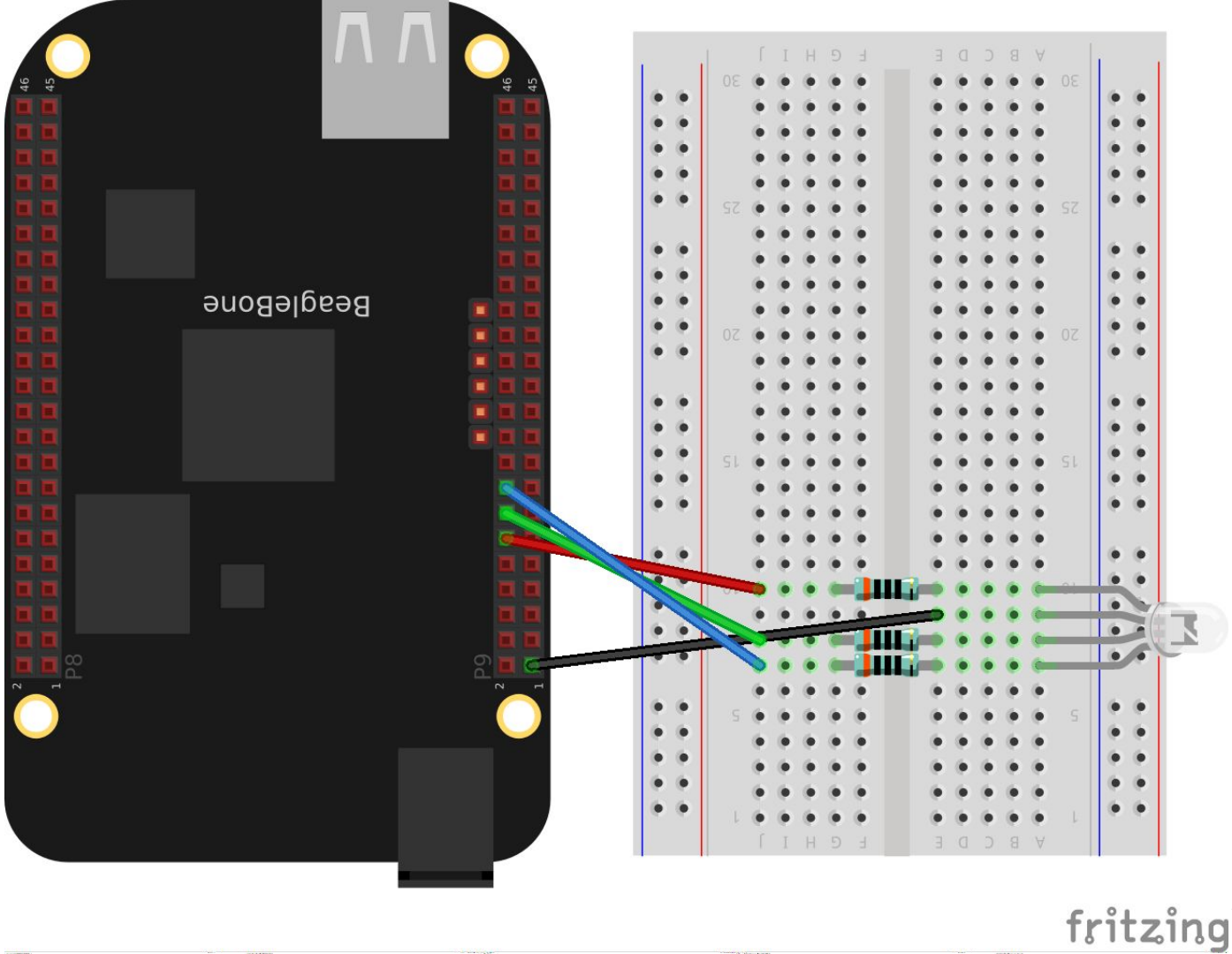
DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3
VDD_5V	5	6	VDD_5V
SYS_5V	7	8	SYS_5V
PWR_BUT	9	10	SYS_RESETN
GPIO_30	11	12	GPIO_60
GPIO_31	13	14	GPIO_50
GPIO_48	15	16	GPIO_51
GPIO_5	17	18	GPIO_4
I2C2_SCL	19	20	I2C2_SDA
GPIO_3	21	22	GPIO_2
GPIO_49	23	24	GPIO_15
GPIO_117	25	26	GPIO_14
GPIO_115	27	28	GPIO_113
GPIO_111	29	30	GPIO_112
GPIO_110	31	32	VDD_ADC
AIN4	33	34	GNDA_ADC
AIN6	35	36	AIN5
AIN2	37	38	AIN3
AIN0	39	40	AIN1
GPIO_20	41	42	GPIO_7
DGND	43	44	DGND
DGND	45	46	DGND

P8

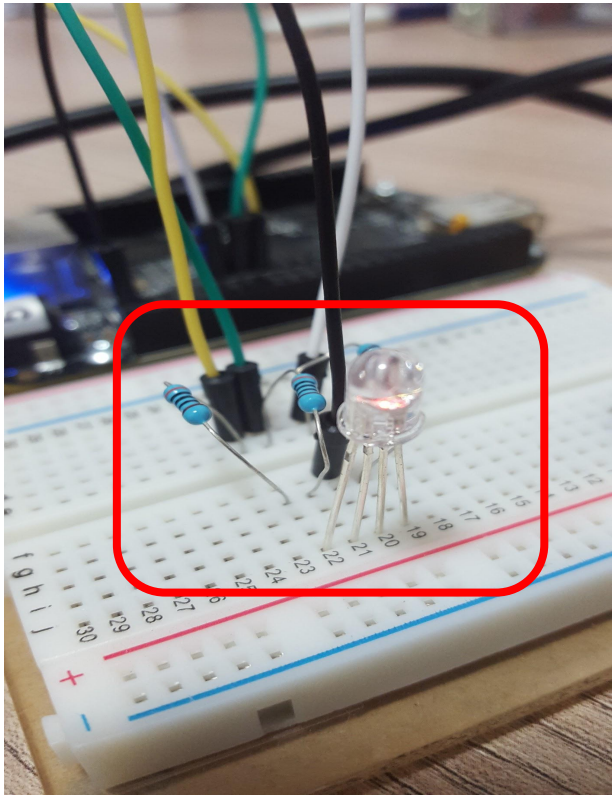
DGND	1	2	DGND
GPIO_38	3	4	GPIO_39
GPIO_34	5	6	GPIO_35
GPIO_66	7	8	GPIO_67
GPIO_69	9	10	GPIO_68
GPIO_45	11	12	GPIO_44
GPIO_23	13	14	GPIO_26
GPIO_47	15	16	GPIO_46
GPIO_27	17	18	GPIO_65
GPIO_22	19	20	GPIO_63
GPIO_62	21	22	GPIO_37
GPIO_36	23	24	GPIO_33
GPIO_32	25	26	GPIO_61
GPIO_86	27	28	GPIO_88
GPIO_87	29	30	GPIO_89
GPIO_10	31	32	GPIO_11
GPIO_9	33	34	GPIO_81
GPIO_8	35	36	GPIO_80
GPIO_78	37	38	GPIO_79
GPIO_76	39	40	GPIO_77
GPIO_74	41	42	GPIO_75
GPIO_72	43	44	GPIO_73
GPIO_70	45	46	GPIO_71



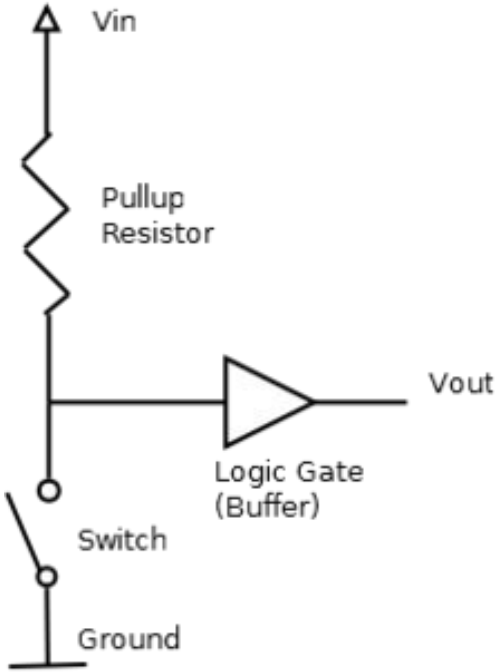
Esquemático



Por que isso está acontecendo?



GPIO 60 tem um resistor Pullup



Exemplo 03 - Liga Led RGB (versão 1.0)

```
import Adafruit_BBIO.GPIO as GPIO
import time

GPIO.setup("P9_12",GPIO.OUT)
GPIO.setup("P9_14",GPIO.OUT)
GPIO.setup("P9_16",GPIO.OUT)

GPIO.output("P9_16",GPIO.HIGH)
time.sleep(4)
GPIO.output("P9_16",GPIO.LOW)
```

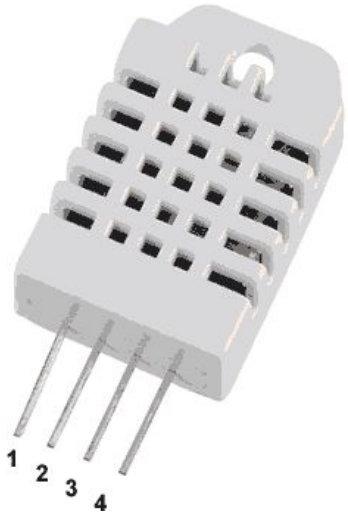
Exemplo 03 - Liga Led RGB (versão 2.0)

```
import Adafruit_BBIO.GPIO as GPIO
import time
import sys

GPIO.setup("P9_12",GPIO.OUT)
GPIO.setup("P9_14",GPIO.OUT)
GPIO.setup("P9_16",GPIO.OUT)

if len(sys.argv) < 4:
        print "Program requires: python name.py HIGH LOW LOW"
else:
        GPIO.output("P9_12",eval("GPIO.%s" % (sys.argv[1])))
        GPIO.output("P9_14",eval("GPIO.%s" % (sys.argv[2])))
        GPIO.output("P9_16",eval("GPIO.%s" % (sys.argv[3])))
        time.sleep(4)
        GPIO.output("P9_12",GPIO.LOW)
        GPIO.output("P9_14",GPIO.LOW)
        GPIO.output("P9_16",GPIO.LOW)
```

Sensor de temperatura e umidade

- DHT22 ou AM2302
- Tensão: 3.3V a 6V dc
- Umidade: 0 a 100%
- Temperatura: -40ºC a 80ºC
- Taxa de atualização: 2s
- API: https://goo.gl/tmPHbS

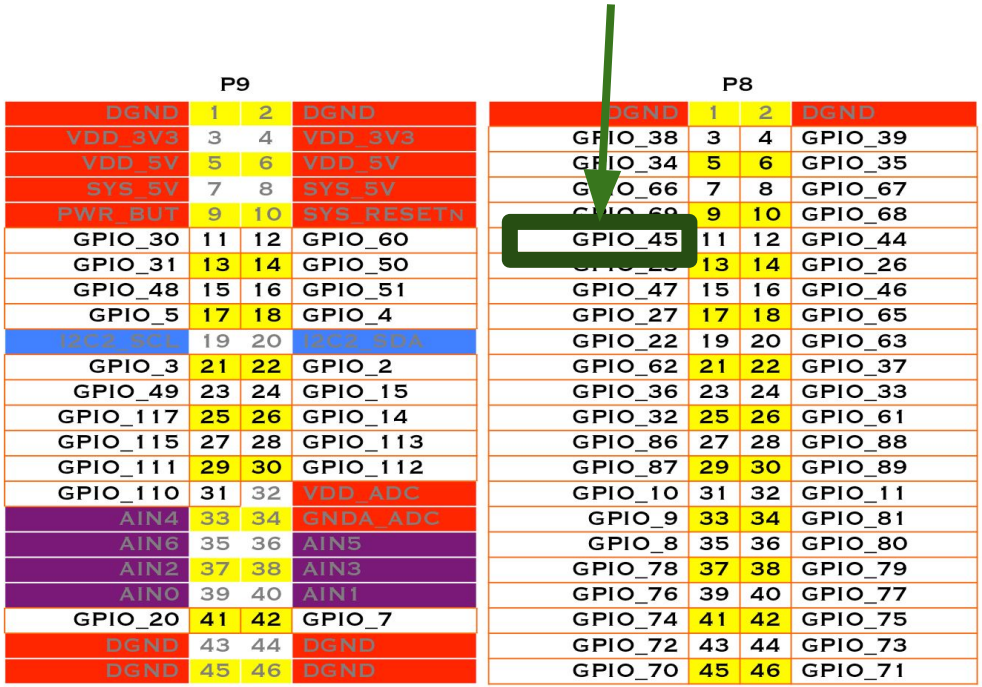
DHT22 pins	
1	VCC
2	DATA
3	NC
4	GND



Instalar a biblioteca

```
git clone https://github.com/adafruit/Adafruit_Python_DHT.git

cd Adafruit_Python_DHT

sudo python setup.py install

sudo rm -rf Adafruit_Python_DHT
```

Esquemático



P9

DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3
VDD_5V	5	6	VDD_5V
SYS_5V	7	8	SYS_5V
PWR_BUT	9	10	SYS_RESETn
GPIO_30	11	12	GPIO_60
GPIO_31	13	14	GPIO_50
GPIO_48	15	16	GPIO_51
GPIO_5	17	18	GPIO_4
I2C2_SCL	19	20	I2C2_SDA
GPIO_3	21	22	GPIO_2
GPIO_49	23	24	GPIO_15
GPIO_117	25	26	GPIO_14
GPIO_115	27	28	GPIO_113
GPIO_111	29	30	GPIO_112
GPIO_110	31	32	VDD_ADC
AIN4	33	34	GNDA_ADC
AIN6	35	36	AIN5
AIN2	37	38	AIN3
AIN0	39	40	AIN1
GPIO_20	41	42	GPIO_7
DGND	43	44	DGND
DGND	45	46	DGND

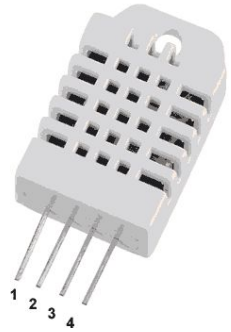
P8

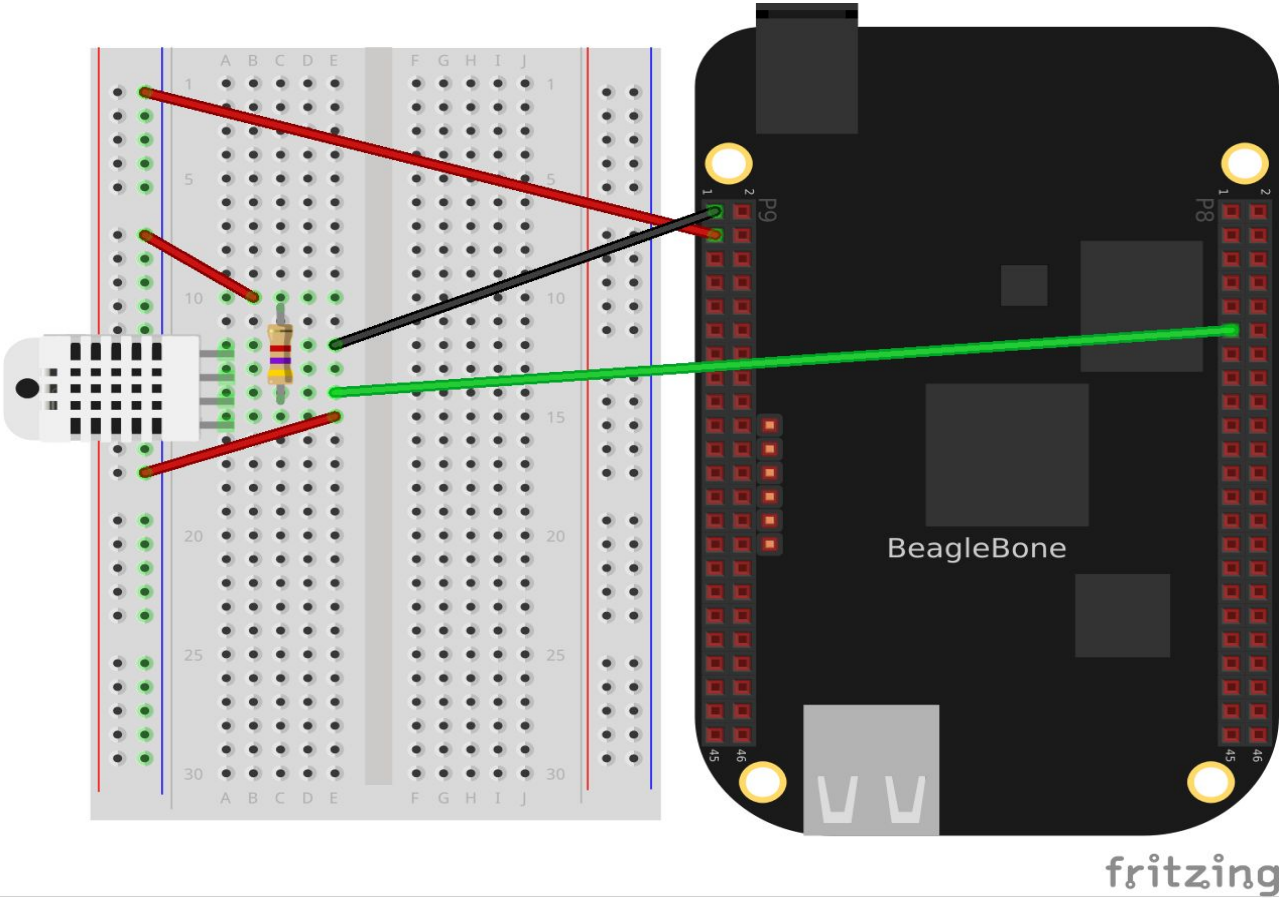
DGND	1	2	DGND
GPIO_38	3	4	GPIO_39
GPIO_34	5	6	GPIO_35
GPIO_66	7	8	GPIO_67
GPIO_69	9	10	GPIO_68
GPIO_45	11	12	GPIO_44
GPIO_23	13	14	GPIO_26
GPIO_47	15	16	GPIO_46
GPIO_27	17	18	GPIO_65
GPIO_22	19	20	GPIO_63
GPIO_62	21	22	GPIO_37
GPIO_36	23	24	GPIO_33
GPIO_32	25	26	GPIO_61
GPIO_86	27	28	GPIO_88
GPIO_87	29	30	GPIO_89
GPIO_10	31	32	GPIO_11
GPIO_9	33	34	GPIO_81
GPIO_8	35	36	GPIO_80
GPIO_78	37	38	GPIO_79
GPIO_76	39	40	GPIO_77
GPIO_74	41	42	GPIO_75
GPIO_72	43	44	GPIO_73
GPIO_70	45	46	GPIO_71

http://exploringbeaglebone.com/wp-content/uploads/resources/BBBP8Header.pdf

Esquemático

DHT22 pins	
1	VCC
2	DATA
3	NC
4	GND





Medindo a temperatura e umidade

```
import Adafruit_DHT as dht

sensor = dht.DHT22
pin = "P8_11"

umidade, temperatura = dht.read_retry(sensor,pin)

if umidade is not None and temperatura is not None:
        print("Temp.={0:0.1f}C Umidade={1:0.1f}%"
                .format(temperatura, umidade))
else:
        print("Erro na leitura, tente novamente")
```

$sudo python lerTemperaturaUmidade.py

Entrada Analógica

Entradas Analógicas

P9

DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3
VDD_5V	5	6	VDD_5V
SYS_5V	7	8	SYS_5V
PWR_BUT	9	10	SYS_RESETN
GPIO_30	11	12	GPIO_60
GPIO_31	13	14	GPIO_50
GPIO_48	15	16	GPIO_51
GPIO_5	17	18	GPIO_4
I2C2_SCL	19	20	I2C2_SDA
GPIO_3	21	22	GPIO_2
GPIO_49	23	24	GPIO_15
GPIO_117	25	26	GPIO_14
GPIO_115	27	28	GPIO_113
GPIO_111	29	30	GPIO_112
GPIO_110	31	32	VDD_ADC
AIN4	33	34	GNDA_ADC
AIN6	35	36	AIN5
AIN2	37	38	AIN3
AIN0	39	40	AIN1
GPIO_20	41	42	GPIO_7
DGND	43	44	DGND
DGND	45	46	DGND

P8

DGND	1	2	DGND
GPIO_38	3	4	GPIO_39
GPIO_34	5	6	GPIO_35
GPIO_66	7	8	GPIO_67
GPIO_69	9	10	GPIO_68
GPIO_45	11	12	GPIO_44
GPIO_23	13	14	GPIO_26
GPIO_47	15	16	GPIO_46
GPIO_27	17	18	GPIO_65
GPIO_22	19	20	GPIO_63
GPIO_62	21	22	GPIO_37
GPIO_36	23	24	GPIO_33
GPIO_32	25	26	GPIO_61
GPIO_86	27	28	GPIO_88
GPIO_87	29	30	GPIO_89
GPIO_10	31	32	GPIO_11
GPIO_9	33	34	GPIO_81
GPIO_8	35	36	GPIO_80
GPIO_78	37	38	GPIO_79
GPIO_76	39	40	GPIO_77
GPIO_74	41	42	GPIO_75
GPIO_72	43	44	GPIO_73
GPIO_70	45	46	GPIO_71

Utilizadas para para conectar sensores

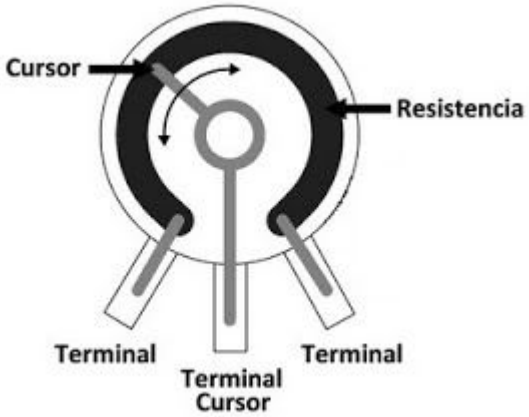
Sinais analógicos são sinais continuos que realizam a medição de algum fenômeno físico

- 7 entradas analógicas [AIN0...AIN6]
- Limitação 1.8V [usar VDD ADC]

Conversor Analógico Digital (ADC) na BBB

- 7 ADC de 12 bits na faixa de 0 a 1.8V
- 2^{12} = 4096 representações discretas
  - 0V equivale a 0
  - 1.8V equivale a 4095
  - Cada um dos 4096 valores equivale a 0.44mV (1.8/4096)

Potenciômetro

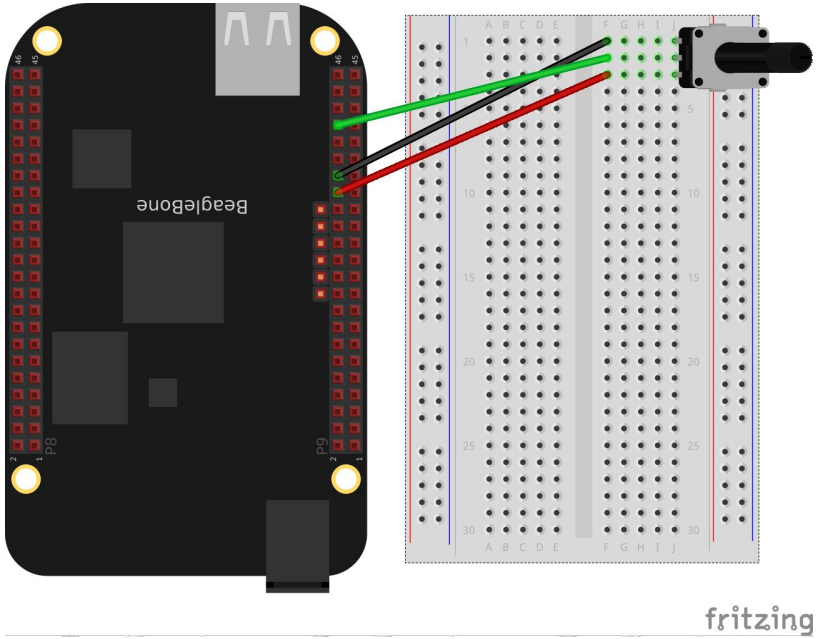


Teste com o multimetro

Potenciômetro (AIN1)

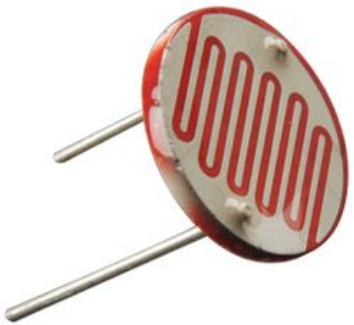
```
import Adafruit_BBIO.ADC as ADC
import time

ADC.setup()

while True:
        valor = ADC.read_raw("P9_40")
        print(valor)
        time.sleep(1)
```

$sudo python potenciometro.py

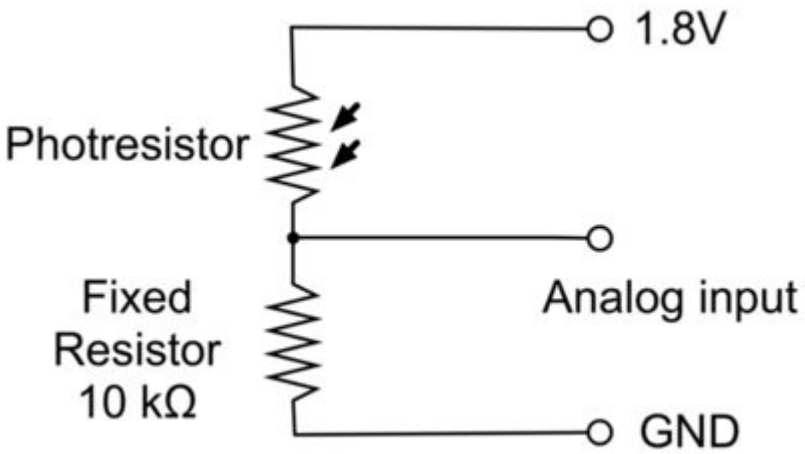


Fotoresistor

Light-Dependent Resistor (LDR)



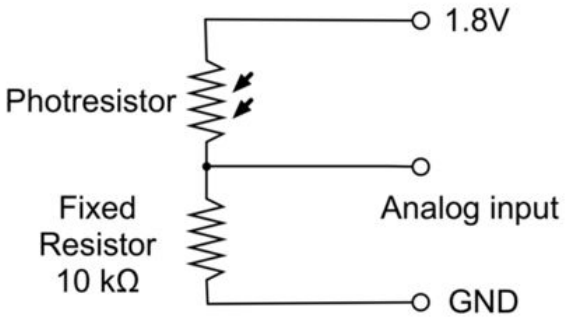
http://goo.gl/sdlPyn

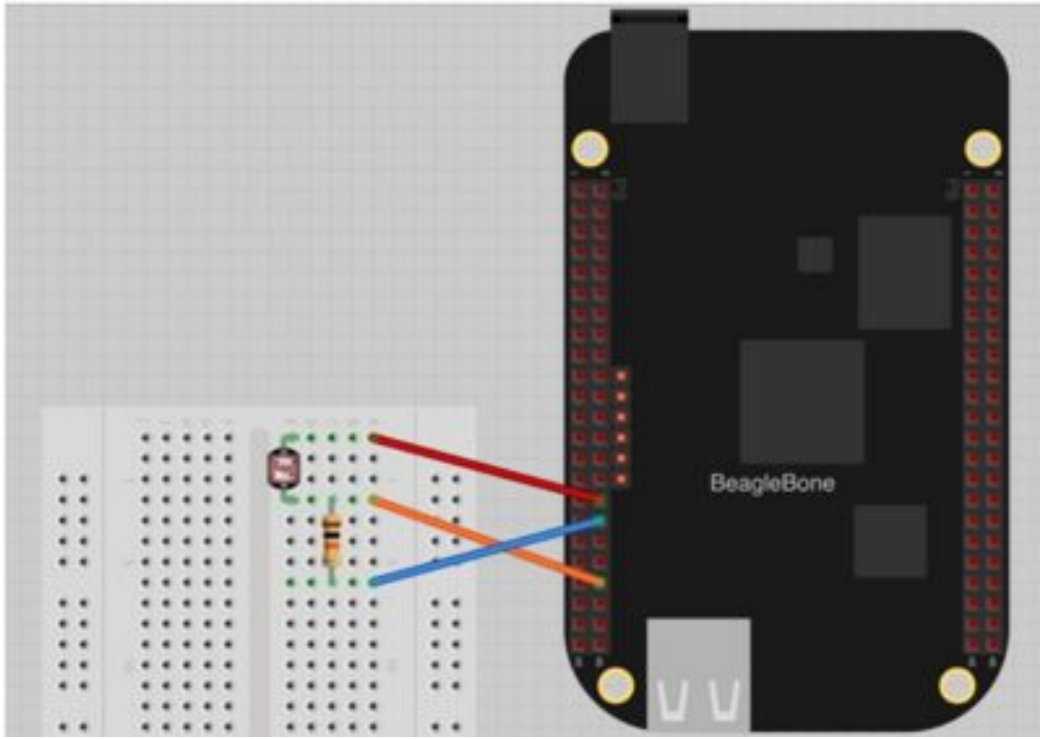


- Iluminado
  - Baixa resistência
  - Tensão em AIN próximo de 1.8 V
- Escuro
  - Alta resistência
  - Tensão em AIN próximo de 0 V

Fotoresistor (ler os dados do LDR)

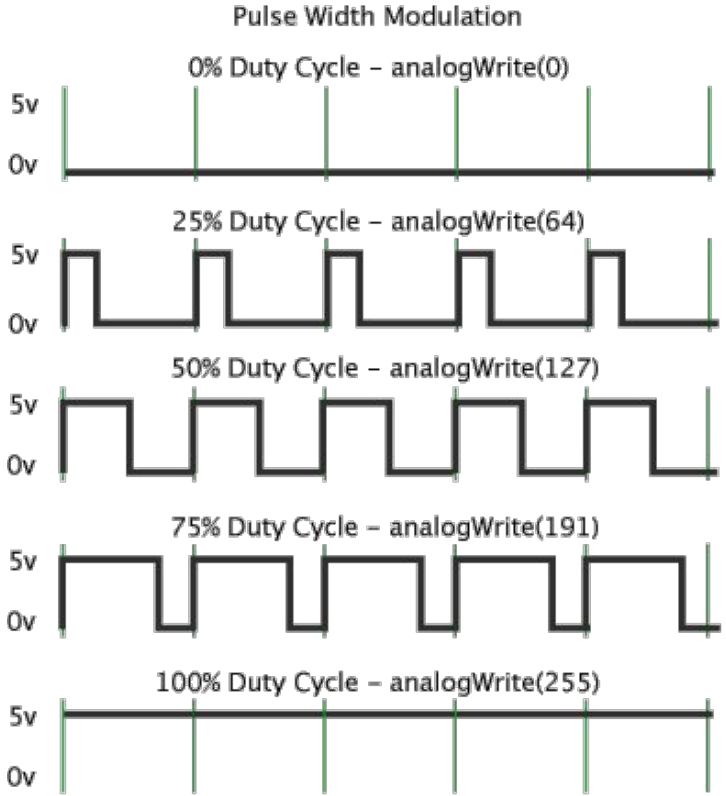
http://goo.gl/sdlPyn





PWM (Pulse Width Modulation)

Ideia



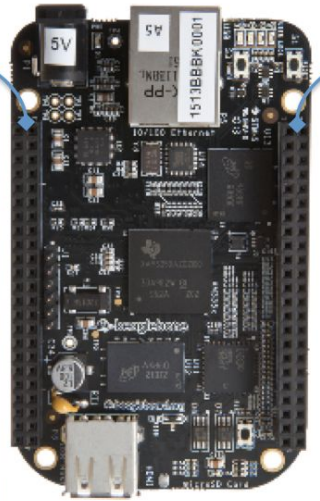
https://www.arduino.cc/en/Tutorial/PWM

Esquemático (pwm1 e pwm2)

Cape Expansion Headers

P9

DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3
VDD_5V	5	6	VDD_5V
SYS_5V	7	8	SYS_5V
PWR_BUT	9	10	SYS_RESETn
UART4_RXD	11	12	GPIO_60
UART4_TXD	13	14	EHRPWM1A
GPIO_48	15	16	EHRPWM1B
SPI0_CS0	17	18	SPI0_D1
I2C2_SCL	19	20	I2C2_SDA
SPI0_D0	21	22	SPI0_SCLK
GPIO_49	23	24	UART1_TXD
GPIO_117	25	26	UART1_RXD
GPIO_115	27	28	SPI1_CS0
SPI1_D0	29	30	GPIO_112
SPI1_SCLK	31	32	VDD_ADC
AIN4	33	34	GNDA_ADC
AIN6	35	36	AIN5
AIN2	37	38	AIN3
AIN0	39	40	AIN1
GPIO_20	41	42	ECAPPWM0
DGND	43	44	DGND
DGND	45	46	DGND



LEGEND

- Power/Ground/Reset
- Available Digital
- Available PWM
- Shared I2C Bus
- Reconfigurable Digital
- Analog Inputs (1.8V)

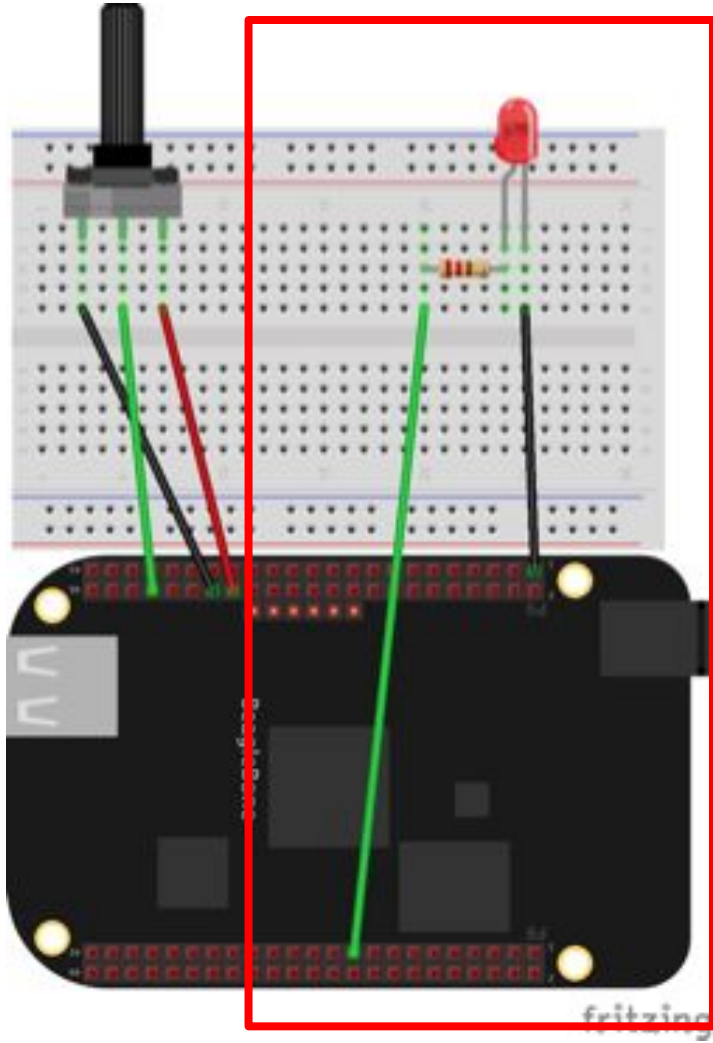
P8

DGND	1	2	DGND
MMC1_DAT6	3	4	MMC1_DAT7
MMC1_DAT2	5	6	MMC1_DAT3
GPIO_66	7	8	GPIO_67
GPIO_69	9	10	GPIO_68
GPIO_45	11	12	GPIO_44
EHRPWM2B	13	14	GPIO_26
GPIO_47	15	16	GPIO_46
GPIO_27	17	18	GPIO_65
EHRPWM2A	19	20	MMC1_CMD
MMC1_CLK	21	22	MMC1_DAT5
MMC1_DAT4	23	24	MMC1_DAT1
MMC1_DAT0	25	26	GPIO_61
LCD_VSYNC	27	28	LCD_PCLK
LCD_HSYNC	29	30	LCD_AC_BIAS
LCD_DATA14	31	32	LCD_DATA15
LCD_DATA13	33	34	LCD_DATA11
LCD_DATA12	35	36	LCD_DATA10
LCD_DATA8	37	38	LCD_DATA9
LCD_DATA6	39	40	LCD_DATA7
LCD_DATA4	41	42	LCD_DATA5
LCD_DATA2	43	44	LCD_DATA3
LCD_DATA0	45	46	LCD_DATA1

Utilizando PWM

```
import Adafruit_BBIO.PWM as PWM
import time

PWM.start("P8_19", 10)
PWM.set_duty_cycle("P8_19", 50.0)
PWM.set_frequency("P8_19", 2000)
time.sleep(2)
PWM.stop("P8_19")
PWM.cleanup()
```

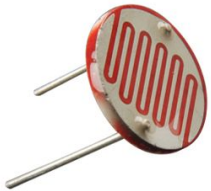
$sudo config-pin P8.19 pwm
$sudo python pwm.py

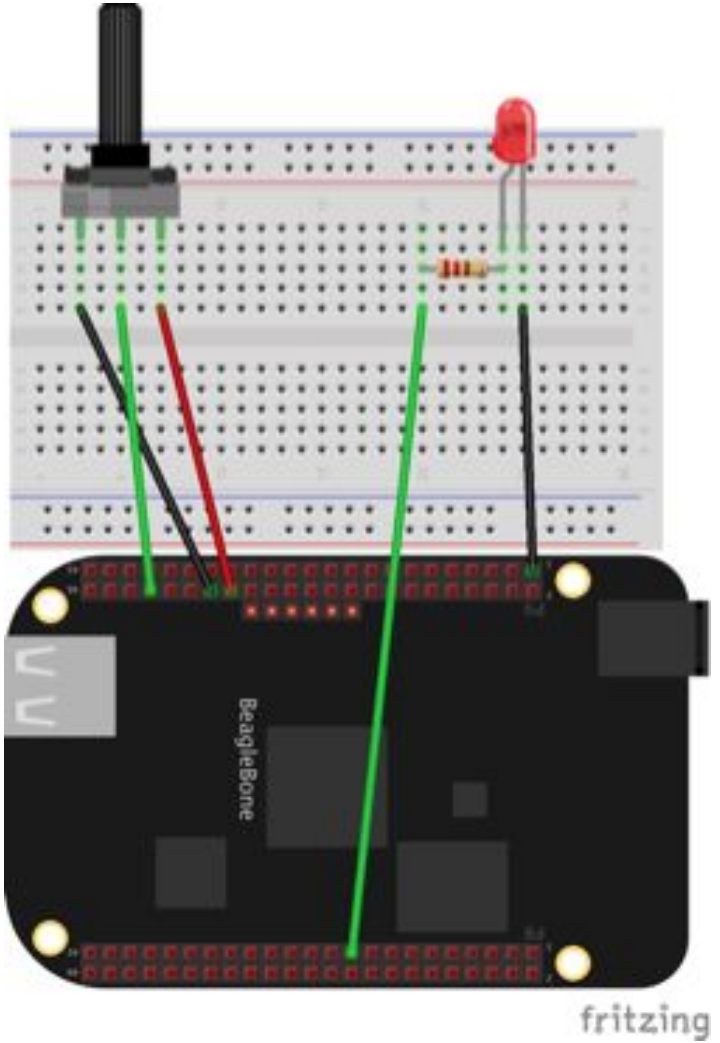


Desafio 2 - Dificuldade 3

```
long map(long x, long in_min, long in_max, long out_min, long out_max)
{
  return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min;
}
```

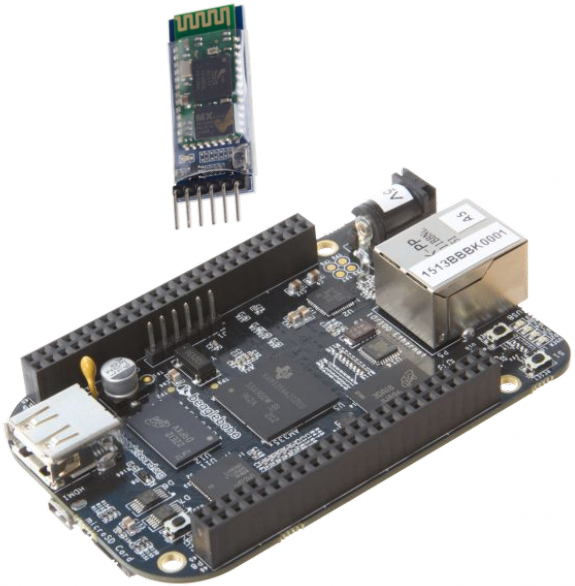
https://www.arduino.cc/en/Reference/Map





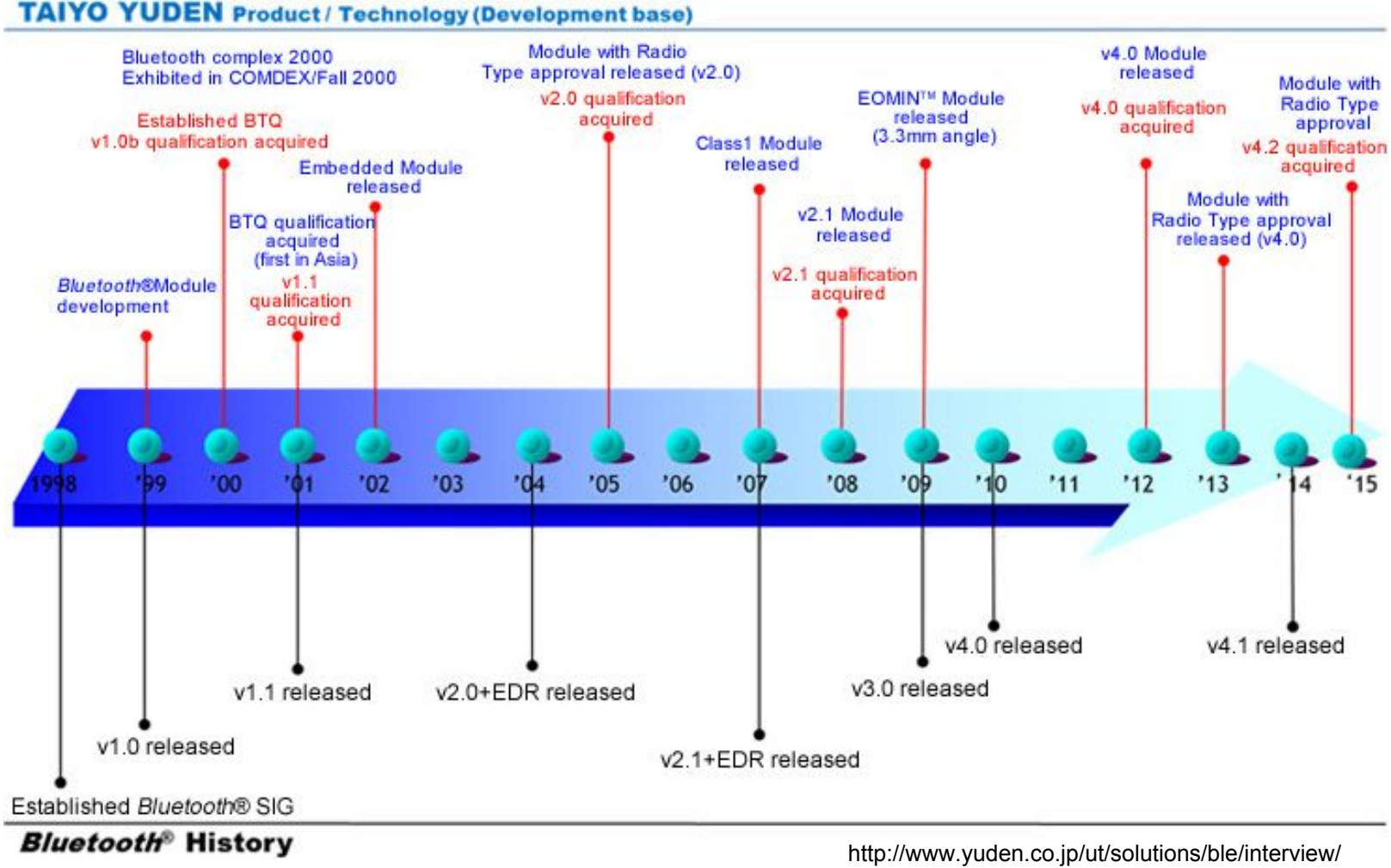
Quais os próximos passos?

HC Family



App genéricas
"Bluetooth Terminal"

Sensores &
Atuadores



http://www.yuden.co.jp/ut/solutions/ble/interview/

Fim da aula #3