03-2 – Blink an LED the JAVAScript Way

Much of this is from

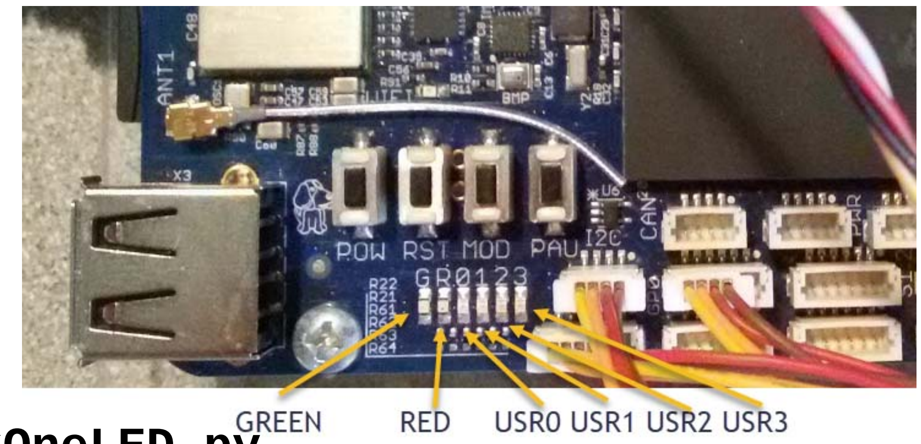
BeagleBone Cookbook

BoneScript

- http://beagleboard.org/Support/BoneScript/
- BoneScript is a Node.js library specifically optimized for the Beagle family and featuring familiar *Arduino* function calls.
- http://nodejs.org/
- Node.js® is a platform built on Chrome's JavaScript runtime for easily building fast, scalable network applications.
- Node.js uses an event-driven, non-blocking I/O model.
- V8 is Google's open source JavaScript engine.
- V8 is written in C++.

Blink an LED

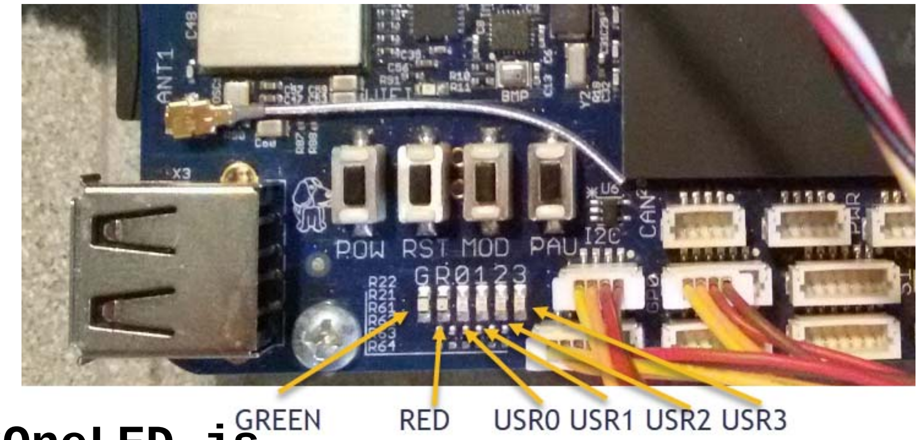
```
#!/usr/bin/env python3
import Adafruit_BBIO.GPIO as GPIO
import time

LED = "USR0"
delay = 0.25

GPIO.setup(LED, GPIO.OUT)

while True:
    GPIO.output(LED, 1)
    time.sleep(delay)
    GPIO.output(LED, 0)
    time.sleep(delay)
```



exercises/displays/blue/blinkOneLED.py

Blink an LED - JavaScript

```
#!/usr/bin/env node
var b = require('bonescript');
var LED   = 'USR0';
var state = 0;     // Initial state
b.pinMode(LED, b.OUTPUT);

setInterval(flash, 500);    // Change state every 500 ms

function flash() {
    b.digitalWrite(LED, state);
    if(state === 1) {
        state = 0;
    } else {
        state = 1;
    }
}
```



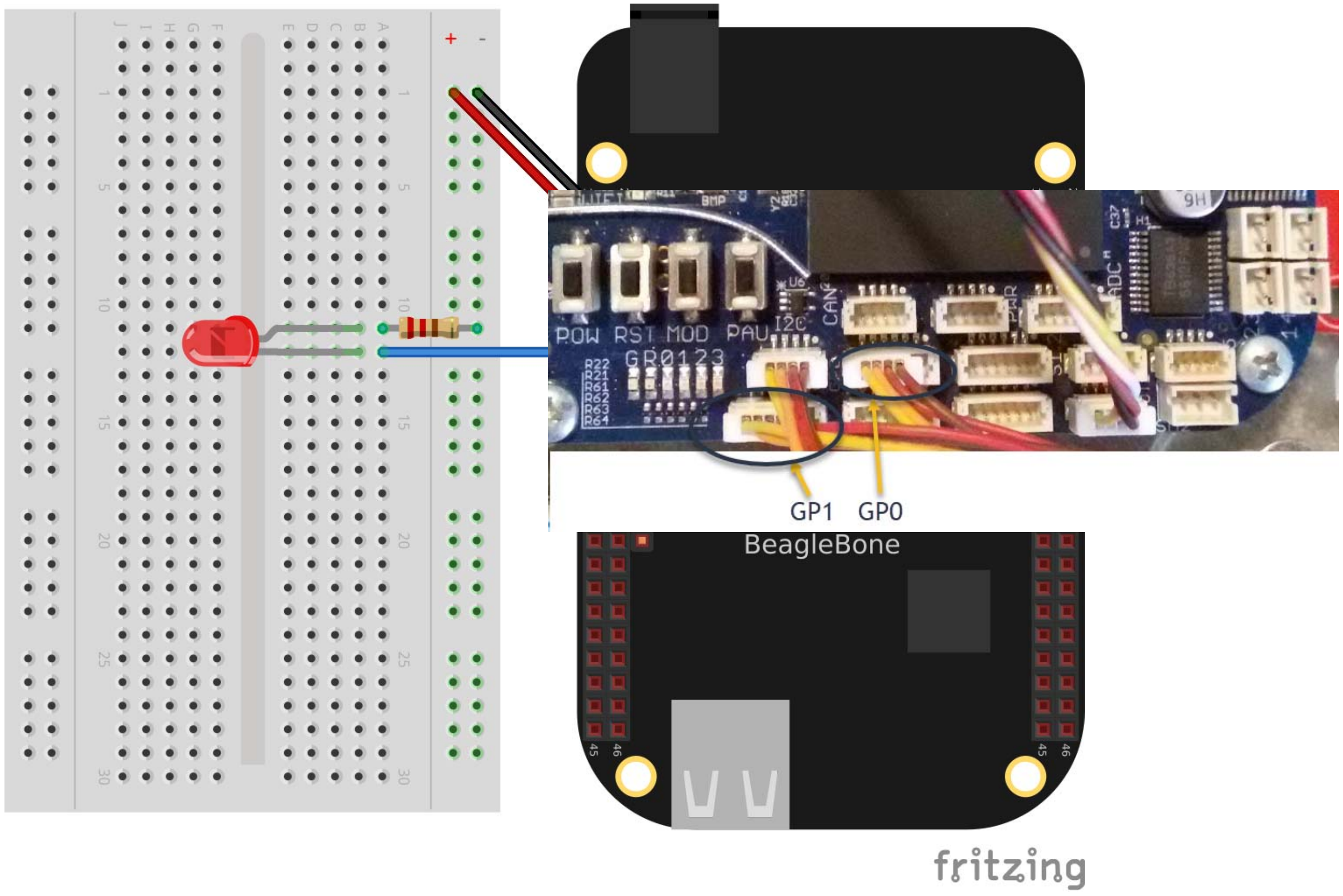
exercises/displays/blue/blinkOneLED.js

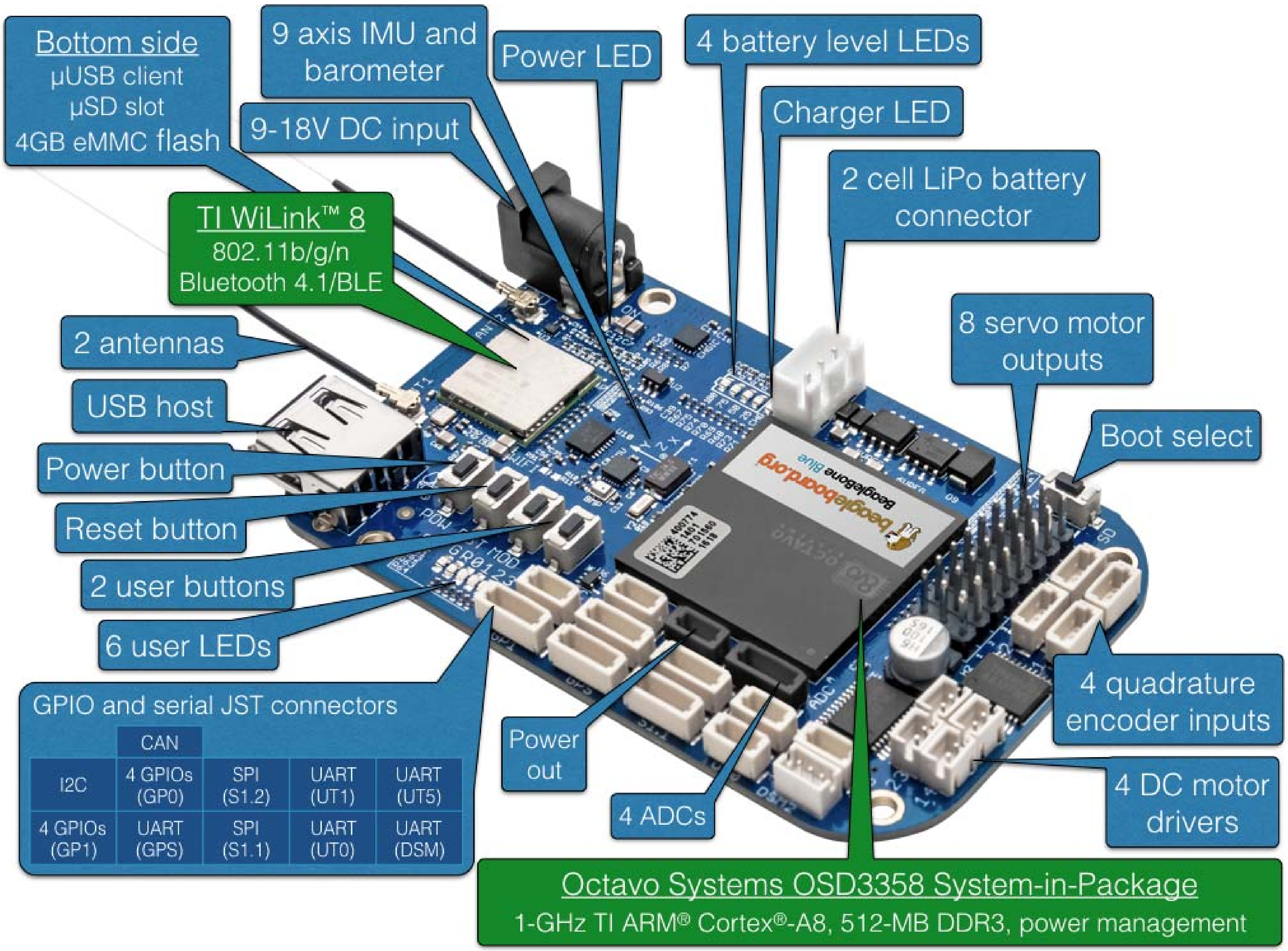
Running js

- Use Cloud9 debugger
- From command line
- If the first line is: **`#!/usr/bin/env node`**

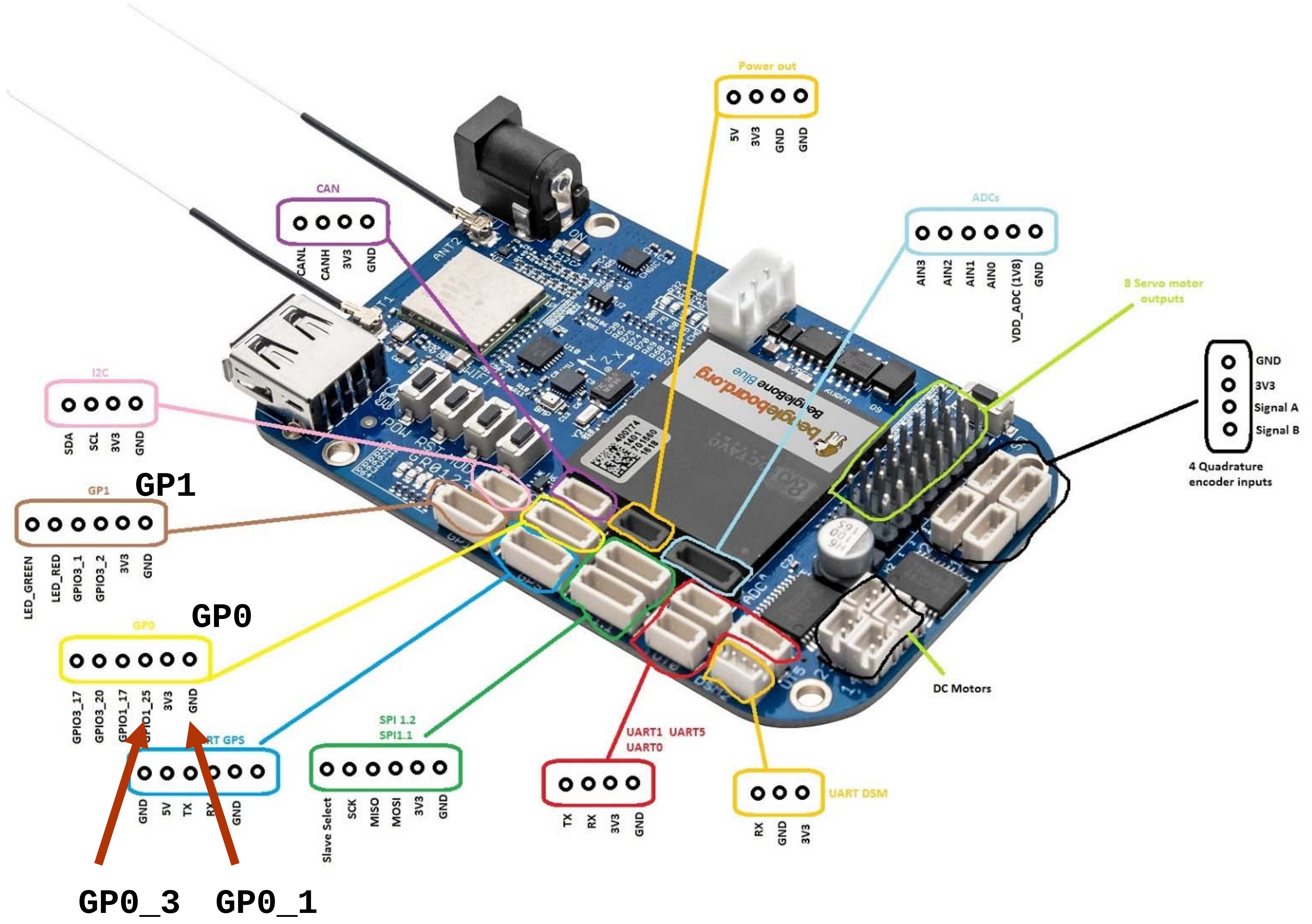
```
bone$ ./blinkOneLED.js
```

External LED





	CAN			
I2C	4 GPIOs (GP0)	SPI (S1.2)	UART (UT1)	UART (UT5)
4 GPIOs (GP1)	UART (GPS)	SPI (S1.1)	UART (UT0)	UART (DSM)

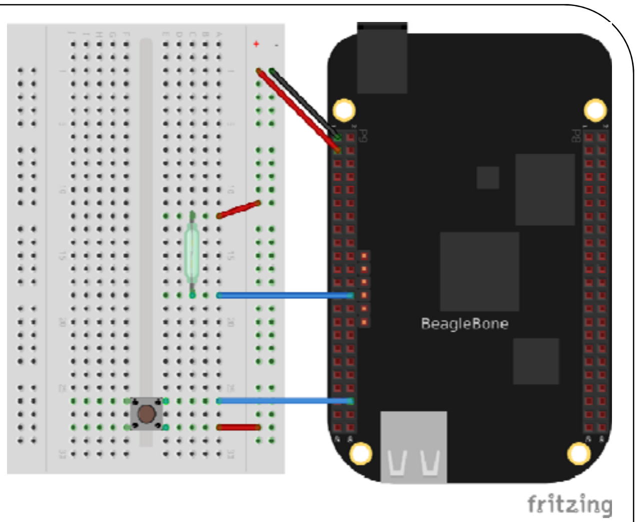

Power out
5V
3V3
GND
GND
CAN
CANL
CANH
3V3
GND
ADCs
AIN3
AIN2
AIN1
AIN0
VDD_ADC (1V8)
GND
8 Servo motor outputs
GND
3V3
Signal A
Signal B
4 Quadrature encoder inputs
I2C
SDA
SCL
3V3
GND
GP1
GP1
LED_GREEN
LED_RED
GPIO3_1
GPIO3_2
3V3
GND
GP0
GP0
GPIO3_17
GPIO3_20
GPIO1_17
GPIO1_25
3V3
GND
UART GPS
GND
5V
TX
RX
GND
SPI 1.2
SPI1.1
Slave Select
SCK
MISO
MOSI
3V3
GND
UART1 UART5
UART0
TX
RX
3V3
GND
UART DSM
RX
GND
3V3
DC Motors
GP0_3
GP0_1

Button - Events

```
#!/usr/bin/env python3
import Adafruit_BBIO.GPIO as GPIO
import time

buttonP="PAUSE"  # PAUSE or MODE
buttonM="MODE"

LEDp   ="RED"
LEDm   ="GREEN"

# Set the GPIO pins:
GPIO.setup(LEDp,    GPIO.OUT)
GPIO.setup(LEDm,    GPIO.OUT)
GPIO.setup(buttonP, GPIO.IN)
GPIO.setup(buttonM, GPIO.IN)

# Turn on both LEDs
GPIO.output(LEDp, 1)
GPIO.output(LEDm, 1)
```

```
# Map buttons to LEDs
map = {buttonP: LEDp, buttonM: LEDm}

def updateLED(channel):
    print("channel = " + channel)
    state = GPIO.input(channel)
    GPIO.output(map[channel], state)
    print(map[channel] + " Toggled")

print("Running...")

GPIO.add_event_detect(buttonP, GPIO.BOTH, callback=updateLED)
# RISING, FALLING or BOTH
GPIO.add_event_detect(buttonM, GPIO.BOTH, callback=updateLED)

try:
    while True:
        time.sleep(100)   # Let other processes run

except KeyboardInterrupt:
    print("Cleaning Up")
    GPIO.cleanup()
GPIO.cleanup()
```

Button via Interrupts



```
#!/usr/bin/env node
var b = require('bonescript');
var button = 'PAUSE';

b.pinMode(button, b.INPUT, 7, 'pulldown');

b.attachInterrupt(button, true,
     b.CHANGE, printStatus);

function printStatus(x) {
    console.log('x.value = ' + x.value);
    console.log('x.err   = ' + x.err);
}
```

callback

Button via Read

```
#!/usr/bin/env node
var b = require('bonescript');
var button = ‘PAUSE';
var state;      // State of pushbutton

b.pinMode(button, b.INPUT, 7, 'pulldown');

state = b.digitalRead(button);
console.log('button state = ' + state);
```