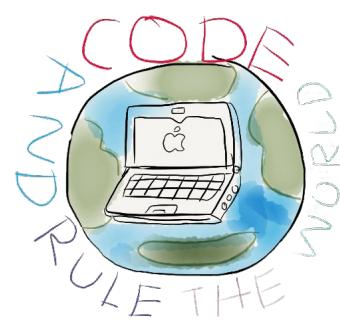


PROCODE^{CG}



PROCODECG

DYCODEX ESPECTRO

TUTORIAL

Notes:

Ver 0.8 – 20 March 2017: adding pin out espectro ver 3

Ver 0.7 – 18 March 2017: adding more codes. adding traffic light FSM

Ver 0.6 – 17 March 2017: adding traffic light mqtt

Ver 0.5 – 10 Feb 2017: adding screenshots of connecting to mqtt. Adding table of contents

Ver 0.4: adding potentiometer and LDR

Table of Contents

1	<u>GETTING STARTED</u>	4
1.1	SETTING UP	5
1.2	BLINKING LED ONBOARD	11
1.3	GETTING KNOW MORE ABOUT ESPECTRO BOARD	13
1.4	ESPECTRO LIBRARY - BLINK	14
1.5	ESPECTRO LIBRARY – BLINK WITHOUT DELAY	16
2	<u>EXPLORING ESPECTRO</u>	19
2.1	ESPECTRO LIBRARY – BUTTON	19
2.2	ESPECTRO LIBRARY – NEOPIXEL	20
2.2.1	SIMPLE EXAMPLE	20
2.2.2	RGBW STRAND TEST	22
2.3	PROCODECG LED	26
2.3.1	ALTERNATE LED	26
2.3.2	LED METER	27
2.3.3	ENTER EXIT	28
2.3.4	KNIGHT RIDER	29
3	<u>CONNECTING TO CLOUD</u>	32
3.1	CONNECTING TO WIFI	32
3.2	CONNECTING TO MQTT	33
3.2.1	ESPECTRO LIBRARY – BASIC ESP8266 EXAMPLE	33
3.2.2	ESPECTRO LIBRARY – LEDBYCLOUD	43
3.3	TRAFFIC LIGHT MQTT	43
3.4	TRAFFIC LIGHT FSM	47
4	<u>ESPECTRO BASE</u>	50
4.1	POTENTIOMETER	50
4.2	LDR	51

1 GETTING STARTED

HARDWARE REQUIRED:

- DyCodeX ESPectro
- A connector cable to computer USB port

DRIVERS:

- <http://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers>

SOFTWARE AND LIBRARY REQUIRED:

- Arduino Software (downloaded from <https://www.arduino.cc>)
or
- CLION (<https://www.jetbrains.com/clion/>)
- Platform IO IDE (<http://platformio.org/>)
- FTDI driver (<http://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers>).
- EspX library. (<https://github.com/andriyadi/EspX>)
- PubSubClient.
- Adafruit BMP Library. (<https://github.com/adafruit/Adafruit-BMP085-Library>)
- MQTT.fx. (<http://www.mqttfx.org/>)
- AzureIoTHubMQTT (<https://github.com/andriyadi/AzureIoTHubMQTTClient>)

WEBSITES & REFERENCES:

- <https://learn.sparkfun.com/tutorials/esp8266-thing-hookup-guide/installing-the-esp8266-arduino-addon>

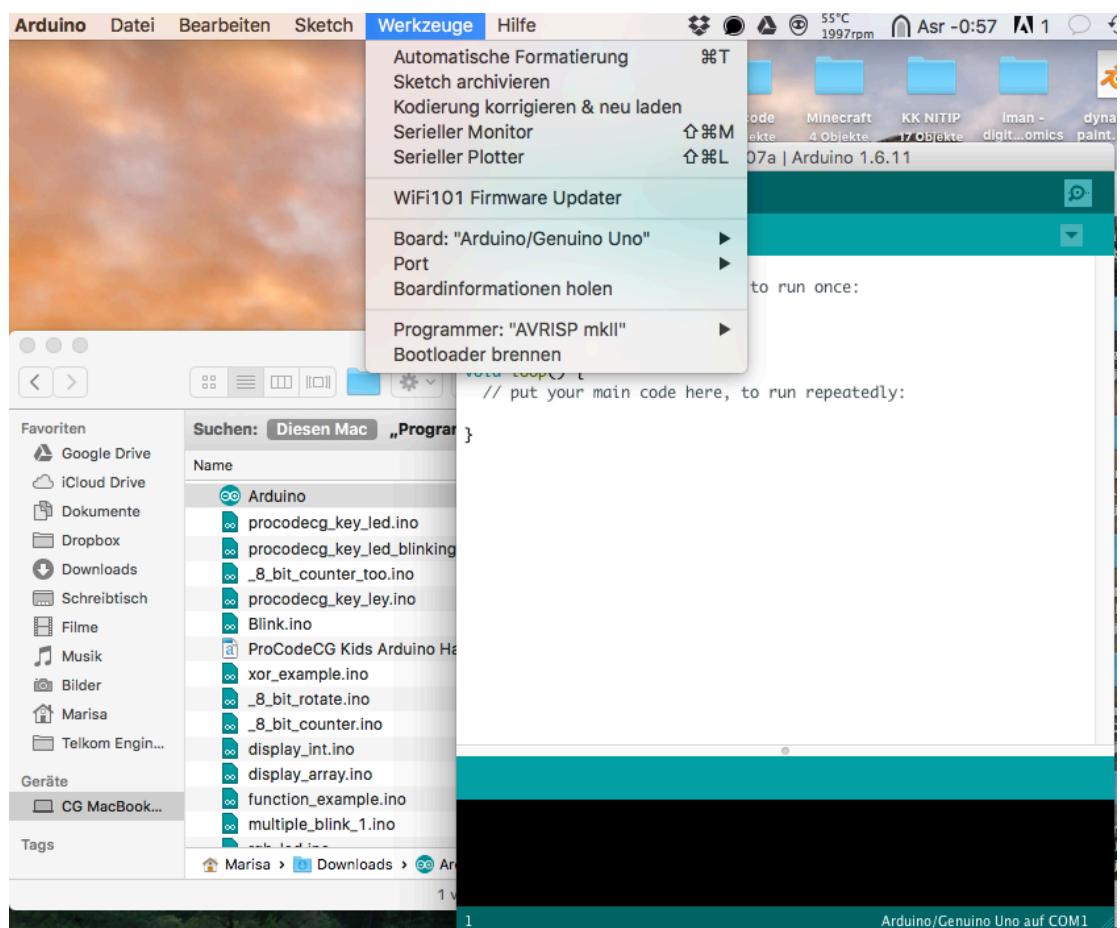
- <https://shop.dycodex.com/type/development-board/espectro/>
- <https://www.youtube.com/watch?v=XtSIW9wVSaU&feature=share>

GITHUB REPOSITORIES:

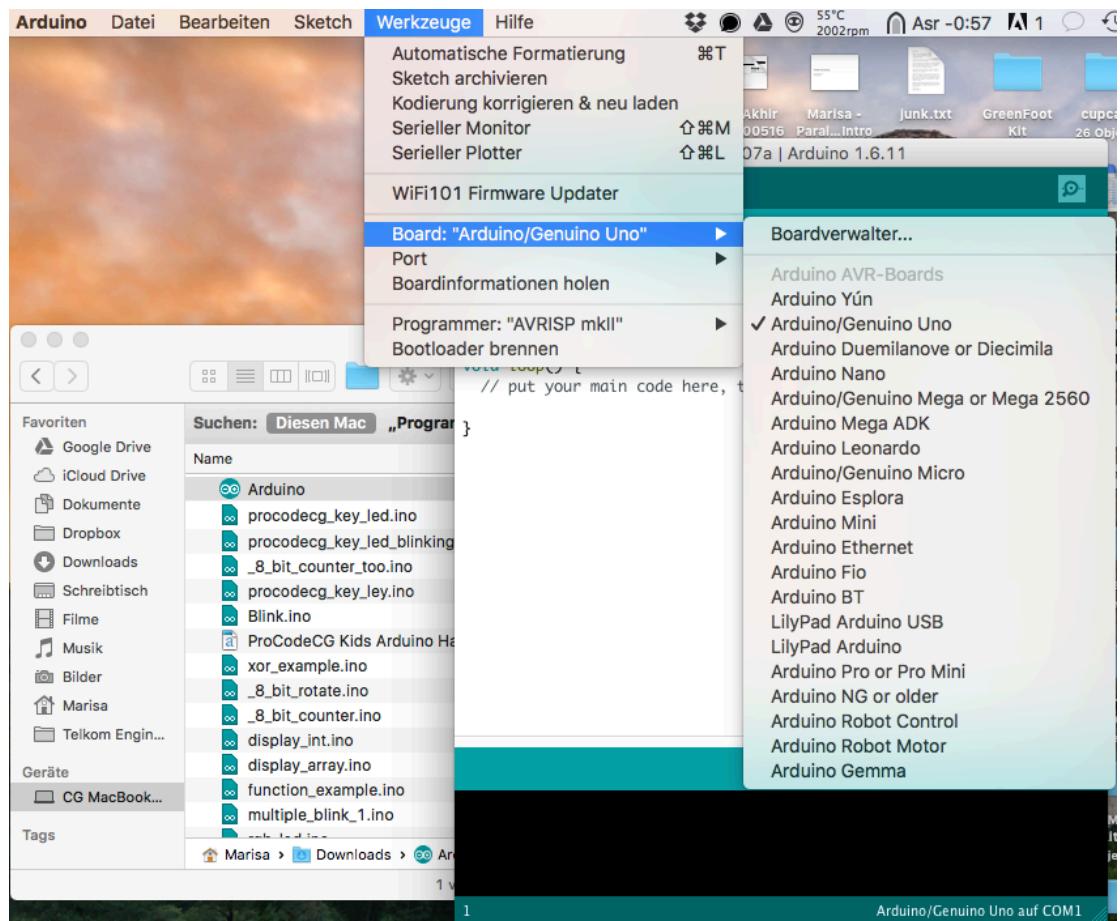
- <https://github.com/andriyadi/EspX>
- <https://github.com/knolleary/pubsubclient>
- <https://github.com/cryptocodecg/CG-IoT>
- <https://github.com/rahard/BRiot-stuff>

1.1 SETTING UP

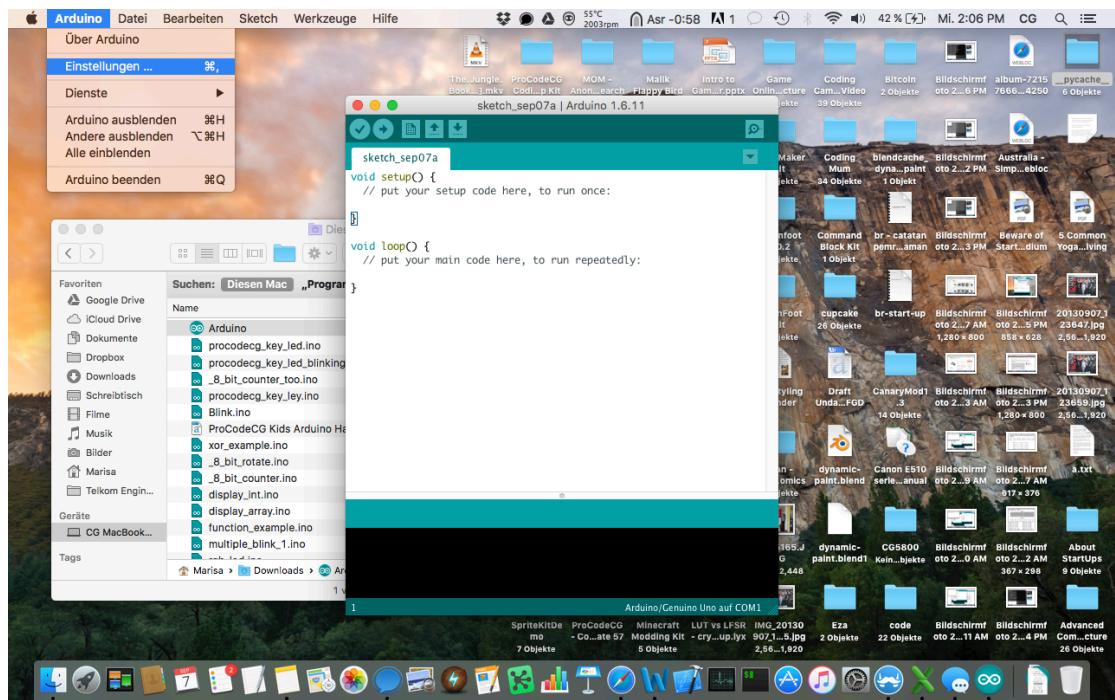
- Open “Tools”



- Choose “Board”. If we don’t have options other than Arduino, that means we have to download the library



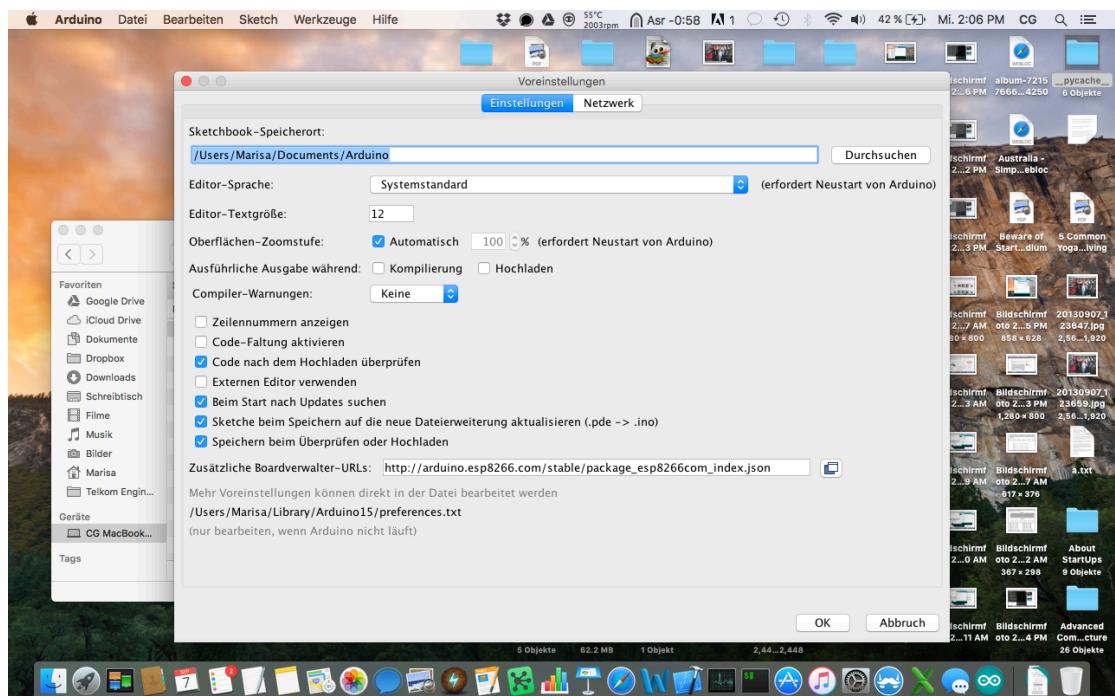
- Open “Setting”



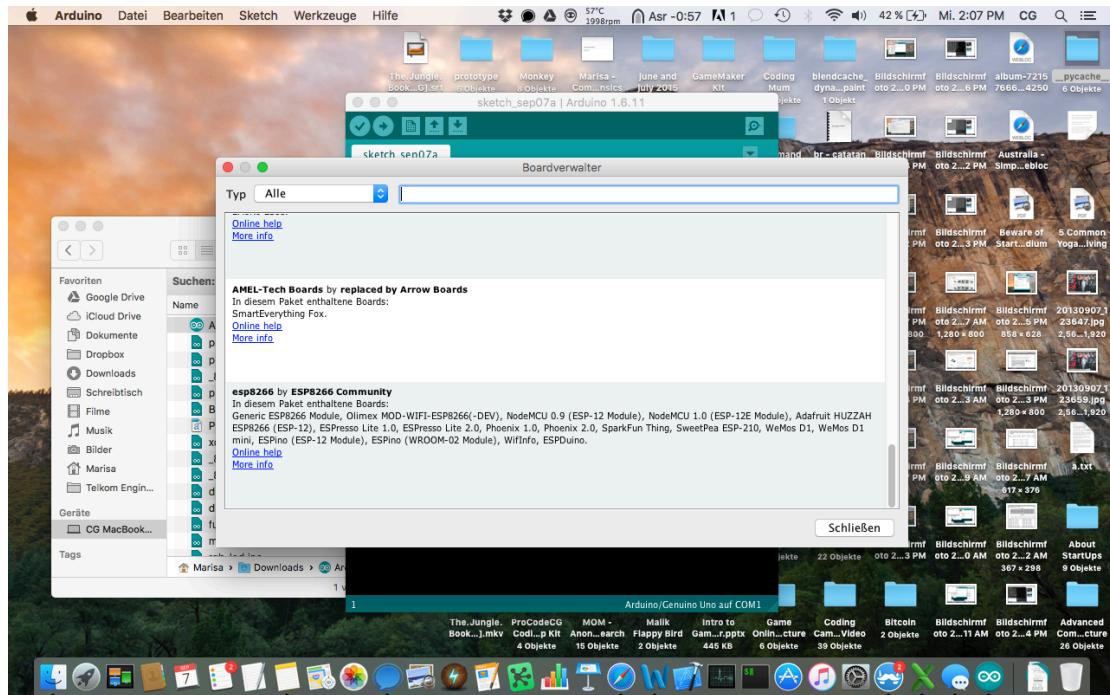
- Copy this URL:

http://arduino.esp8266.com/stable/package_esp8266com_index.json

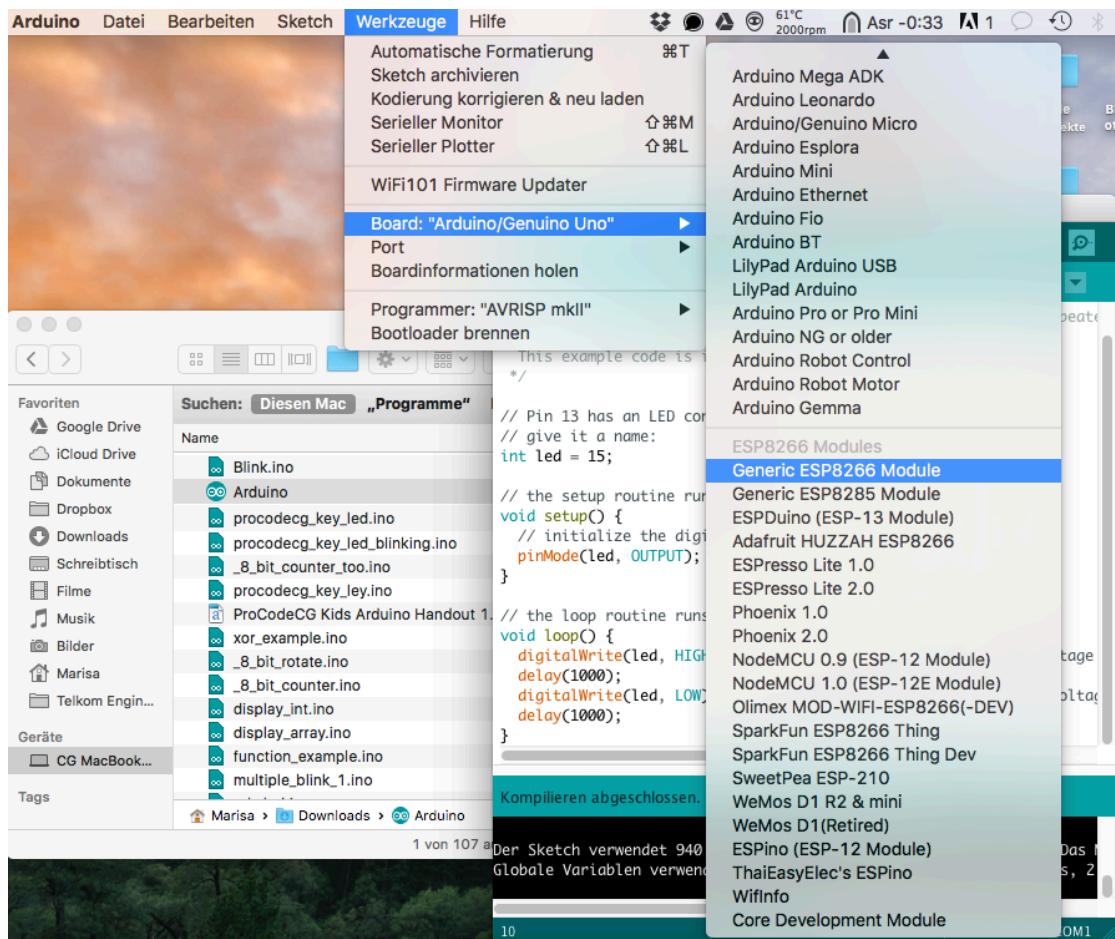
to the "Additional Board Manager URLs" text box. Press OK.



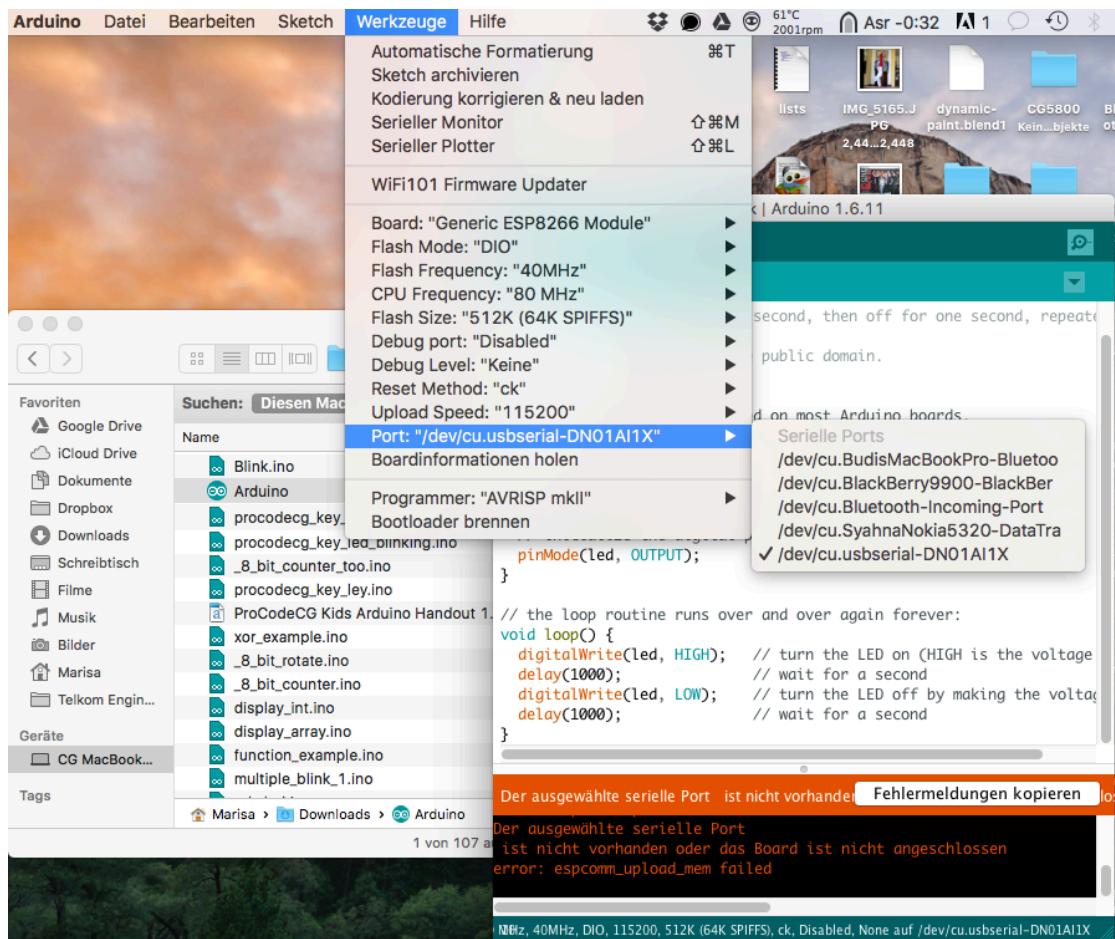
- Navigate to Board Manager by going to **Tools > Boards > Boards Manager**. There should be a couple new entries in addition to the standard Arduino boards. Look for **esp8266**. Click on that entry, then select **Install**.



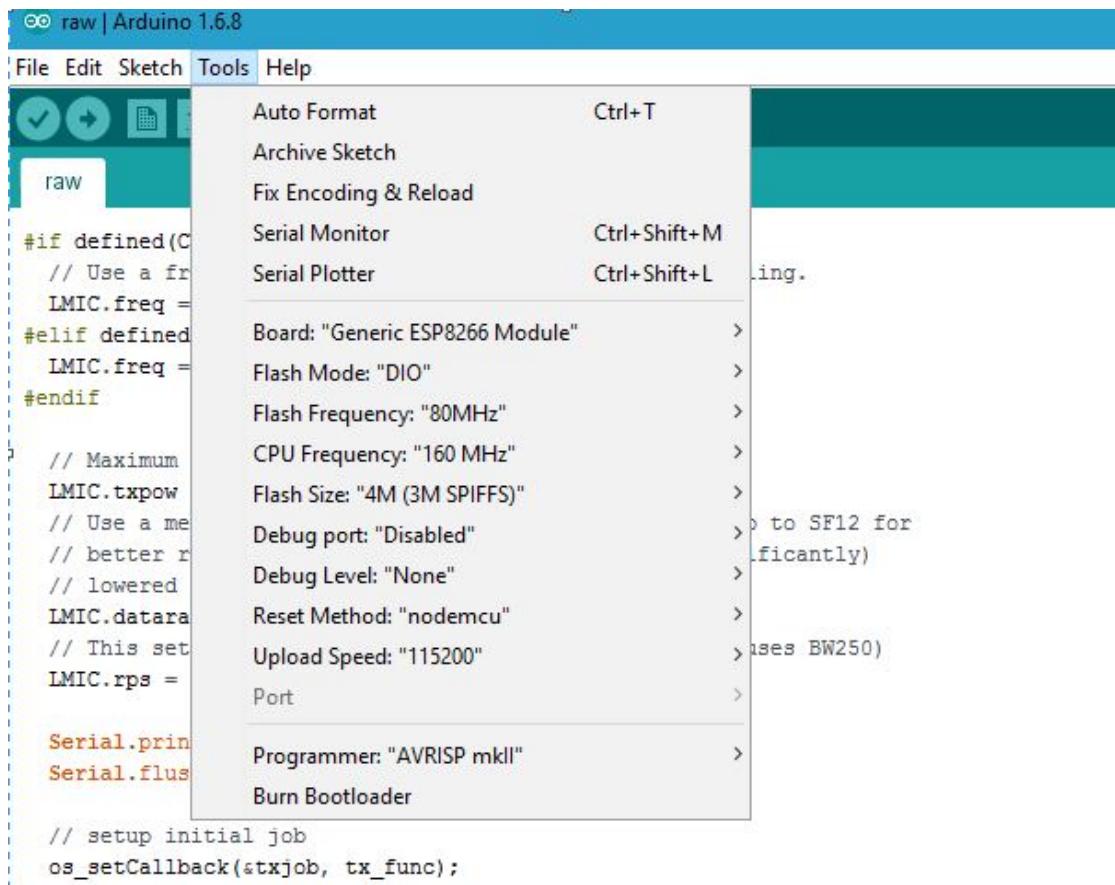
- Now we should have more board options. Select “ESP8266 Thing” from the **Tools > Boards** menu.



- Choose port



- Make sure the configuration is exactly like shown below



1.2 BLINKING LED ONBOARD

When you first unbox your ESPectro Board, first thing to do is to test it. ESPectro has LED (and resistor) connected to port 15. We will go through the following steps to make it blinking:

- Download the Arduino Software <https://www.arduino.cc/en/Main/Software>
- Install the Arduino Software on your computer
- Connect your ESPectro to your computer via the USB cable.
- On the ESPectro board, there is a resistor next to pin 15 and an LED next to it. That LED works just as if it were connected between pin 15 and the ground (GND) pin next to it. If the LED is NOT on your board, just connect an LED between 15 and GND. You don't need to do anything else since a resistor is already built in and limits the current through the LED so you don't put your board at risk of a short circuit.
- Open the blink example from the software kit: “File | Sketchbook |

Examples | Digital". The onboard LED (or the one you added) should blink on and off after you upload the **Blink** "sketch" (as Arduino projects are called) to the board (**File | Upload**).

- When you write programs for your Arduino, you will normally do much of your debugging in the software development kit by doing a Verify/Compile before uploading.

THE CODE:

```
/*
Blink
Turns on an LED on for one second, then off for one second,
repeatedly.

Most Arduinos have an on-board LED you can control. On the Uno and
Leonardo, it is attached to digital pin 13. If you're unsure what
pin the on-board LED is connected to on your Arduino model, check
the documentation at http://www.arduino.cc

This example code is in the public domain.

modified 8 May 2014
by Scott Fitzgerald
*/



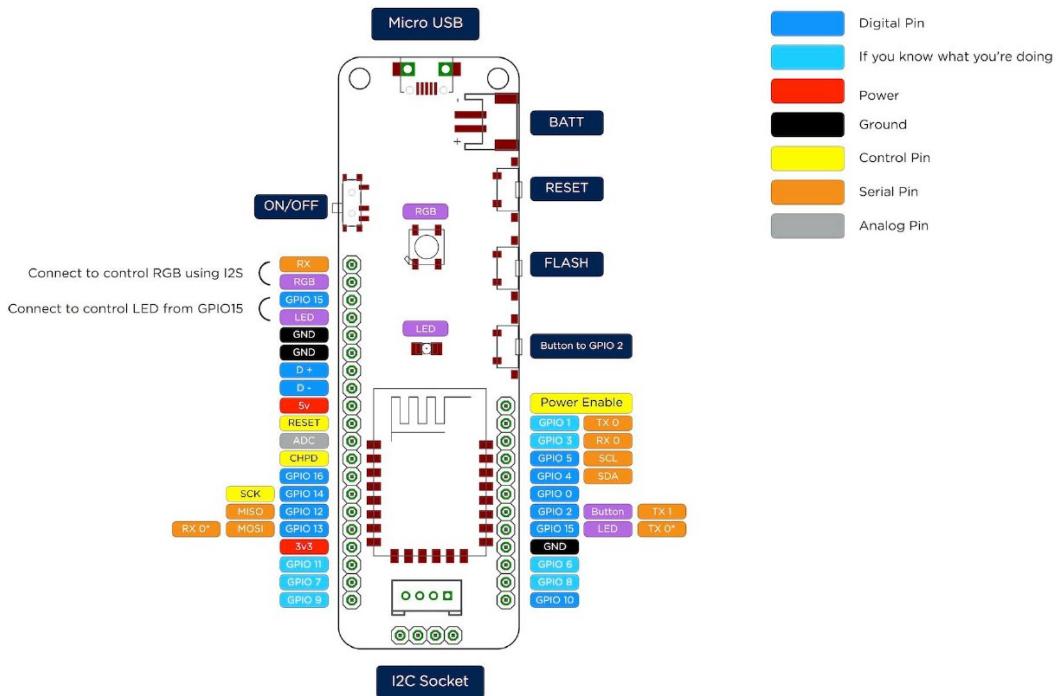
// the setup function runs once when you press reset
// or power the board
void setup() {
    // initialize digital pin 13 as an output.
    pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
    digitalWrite(13, HIGH); // turn LED on (HIGH is the voltage level)
    delay(1000);           // wait for a second
    digitalWrite(13, LOW); // turn the LED off by making the voltage LOW
    delay(1000);           // wait for a second
}
```

Now that you have a blinking LED onboard, so you are good to do more things.

1.3 GETTING KNOW MORE ABOUT ESPectro BOARD

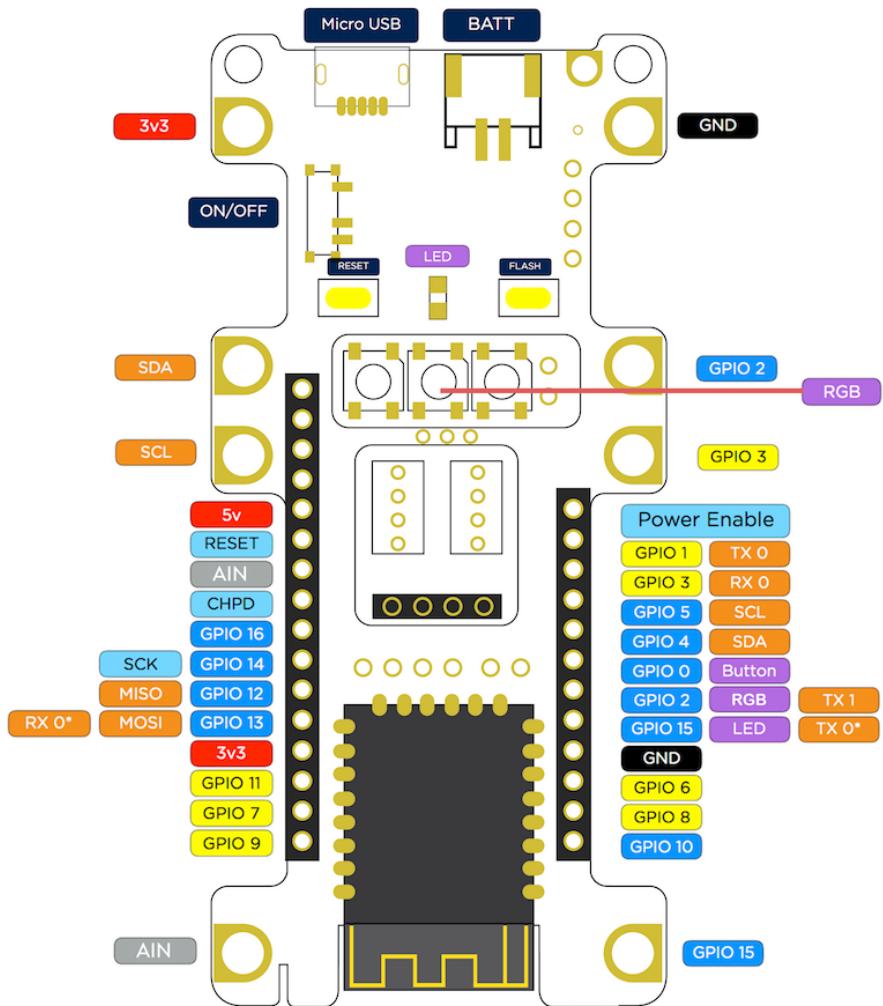
Below is the detail hardware layout of ESPectro. To connect it to breadboard and other components you have to refer to the picture. Next we will connect the board to a LED on a breadboard, using GPIO, Power and Ground port.



GPIO 2 is active low (connected to push button). It uses pull up resistor to VCC 3,3 V. GPIO 15 is active high. Uses pull down resistor. Power enable pin can be set up to 1 to activate all the circuit on the board. All ground is connected. There is only one analog port (ADC) but we can use I2C Socket. ADC 1V (1024) -> needs voltage divider. No charger on board so charger module should be added.

RESET is for resetting the whole program. To be able to upload the program to the board, FLASH mode should be in ON position. Press FLASH and hold, RESET then release RESET and then release FLASH.

ESPectro Ver 3 Pin Out



Restart Arduino. Load Blink from ESPectro folder.

```
/*
Blink
Turns on an LED on for one second, then off for one second,
repeatedly.

Most Arduinos have an on-board LED you can control. On the UNO,
MEGA and ZERO
it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN
takes care
of use the correct LED pin whatever is the board used.
If you want to know what pin the on-board LED is connected to on
your Arduino model, check
the Technical Specs of your board at
https://www.arduino.cc/en/Main/Products

This example code is in the public domain.

modified 8 May 2014
by Scott Fitzgerald

modified 2 Sep 2016
by Arturo Guadalupi

modified 22 Sep 2016
by Andri Yadi
*/
/*
* ESPectro LED_BUILTIN = 15;
*/
// the setup function runs once when you press reset or power the
board
void setup() {
    // initialize digital pin LED_BUILTIN as an output.
    pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
```

```

void loop() {
    digitalWrite(LED_BUILTIN, HIGH);      // turn the LED on (HIGH is the
voltage level)
    delay(1000);                      // wait for a second
    digitalWrite(LED_BUILTIN, LOW);     // turn the LED off by making
the voltage LOW
    delay(1000);                      // wait for a second
}

```

1.5 ESPectro LIBRARY – BLINK WITHOUT DELAY

/* Blink without Delay

Turns on and off a light emitting diode (LED) connected to a digital pin, without using the `delay()` function. This means that other code can run at the same time without being interrupted by the LED code.

The circuit:

- * LED attached from pin 13 to ground.
- * Note: on most Arduinos, there is already an LED on the board that's attached to pin 13, so no hardware is needed for this example.

created 2005
by David A. Mellis
modified 8 Feb 2010
by Paul Stoffregen
modified 11 Nov 2013
by Scott Fitzgerald

This example code is in the public domain.

<http://www.arduino.cc/en/Tutorial/BlinkWithoutDelay>
*/

```

// constants won't change. Used here to set a pin number :
const int ledPin = LED_BUILTIN;      // the number of the LED pin

```

```

// Variables will change :
int ledState = LOW; // ledState used to set the LED

// Generally, you should use "unsigned long" for variables that hold
time
// The value will quickly become too large for an int to store
unsigned long previousMillis = 0; // will store last time LED
was updated

// constants won't change :
const long interval = 1000; // interval at which to blink
(milliseconds)

void setup() {
    // set the digital pin as output:
    pinMode(ledPin, OUTPUT);
}

void loop() {
    // here is where you'd put code that needs to be running all the
time.

    // check to see if it's time to blink the LED; that is, if the
    // difference between the current time and last time you blinked
    // the LED is bigger than the interval at which you want to
    // blink the LED.
    unsigned long currentMillis = millis();

    if (currentMillis - previousMillis >= interval) {
        // save the last time you blinked the LED
        previousMillis = currentMillis;

        // if the LED is off turn it on and vice-versa:
        if (ledState == LOW) {
            ledState = HIGH;
        } else {
            ledState = LOW;
        }

        // set the LED with the ledState of the variable:
        digitalWrite(ledPin, ledState);
}

```

```
 }  
 }
```

2 EXPLORING ESPECTRO

2.1 ESPectro LIBRARY – BUTTON

```
/*
Button

Turns on and off a light emitting diode(LED) connected to digital
pin 13, when pressing a pushbutton attached to pin 2.

The circuit:
* LED attached from pin 13 to ground
* pushbutton attached to pin 2 from +5V
* 10K resistor attached to pin 2 from ground

* Note: on most Arduinos there is already an LED on the board
attached to pin 13.

created 2005
by DojoDave <http://www.0j0.org>
modified 30 Aug 2011
by Tom Igoe

This example code is in the public domain.

http://www.arduino.cc/en/Tutorial/Button
 */

// constants won't change. They're used here to
// set pin numbers:
const int buttonPin = BUTTON_BUILTIN;      // the number of the
pushbutton pin
const int ledPin = LED_BUILTIN;           // the number of the LED pin
```

```

// variables will change:
int buttonState = 0;           // variable for reading the pushbutton
status

void setup() {
    // initialize the LED pin as an output:
    pinMode(ledPin, OUTPUT);
    // initialize the pushbutton pin as an input:
    pinMode(buttonPin, INPUT);
}

void loop() {
    // read the state of the pushbutton value:
    buttonState = digitalRead(buttonPin);

    // check if the pushbutton is pressed.
    // if it is, the buttonState is HIGH:
    if (buttonState == HIGH) {
        // turn LED on:
        digitalWrite(ledPin, HIGH);
    } else {
        // turn LED off:
        digitalWrite(ledPin, LOW);
    }
}

```

2.2 ESPectro LIBRARY – NeoPixel

GITHUB REPOSITORIES:

- https://github.com/adafruit/Adafruit_NeoPixel

Don't forget to shift the jumper to the center

2.2.1 Simple Example

```

// NeoPixel Ring simple sketch (c) 2013 Shae Erisson
// released under the GPLv3 license to match the rest of the AdaFruit
NeoPixel library

#include <Adafruit_NeoPixel.h>
#endif __AVR__

```

```

#include <avr/power.h>
#endif

// Which pin on the Arduino is connected to the NeoPixels?
// On a Trinket or Gemma we suggest changing this to 1
#define PIN          15

// How many NeoPixels are attached to the Arduino?
#define NUMPIXELS    1

// When we setup the NeoPixel library, we tell it how many pixels,
// and which pin to use to send signals.

// Note that for older NeoPixel strips you might need to change the
// third parameter--see the strandtest
// example for more information on possible values.

Adafruit_NeoPixel pixels = Adafruit_NeoPixel(NUMPIXELS, PIN, NEO_GRB
+ NEO_KHZ800);

int delayval = 500; // delay for half a second

void setup() {
    // This is for Trinket 5V 16MHz, you can remove these three lines
    if you are not using a Trinket
#if defined (__AVR_ATtiny85__)
    if (F_CPU == 16000000) clock_prescale_set(clock_div_1);
#endif
    // End of trinket special code

    pixels.begin(); // This initializes the NeoPixel library.
}

void loop() {

    // For a set of NeoPixels the first NeoPixel is 0, second is 1, all
    // the way up to the count of pixels minus one.

    for(int i=0;i<NUMPIXELS;i++) {

        // pixels.Color takes RGB values, from 0,0,0 up to 255,255,255
        pixels.setPixelColor(i, pixels.Color(0,150,0)); // Moderately
        bright green color.
}

```

```

    pixels.show(); // This sends the updated pixel color to the
hardware.

delay(delayval); // Delay for a period of time (in milliseconds).

}
}

```

2.2.2 RGBW Strand Test

```

#include <Adafruit_NeoPixel.h>

#ifndef __AVR__
#include <avr/power.h>
#endif

#define PIN 15

// Parameter 1 = number of pixels in strip
// Parameter 2 = Arduino pin number (most are valid)
// Parameter 3 = pixel type flags, add together as needed:
//   NEO_KHZ800  800 KHz bitstream (most NeoPixel products w/WS2812
LEDs)
//   NEO_KHZ400  400 KHz (classic 'v1' (not v2) FLORA pixels, WS2811
drivers)
//   NEO_GRB      Pixels are wired for GRB bitstream (most NeoPixel
products)
//   NEO_RGB      Pixels are wired for RGB bitstream (v1 FLORA pixels,
not v2)
//   NEO_RGBW     Pixels are wired for RGBW bitstream (NeoPixel RGBW
products)

Adafruit_NeoPixel strip = Adafruit_NeoPixel(60, PIN, NEO_GRB +
NEO_KHZ800);

// IMPORTANT: To reduce NeoPixel burnout risk, add 1000 uF capacitor
across
// pixel power leads, add 300 - 500 Ohm resistor on first pixel's
data input
// and minimize distance between Arduino and first pixel. Avoid
connecting
// on a live circuit...if you must, connect GND first.

```

```

void setup() {
    // This is for Trinket 5V 16MHz, you can remove these three lines
    // if you are not using a Trinket
    #if defined (__AVR_ATtiny85__)
        if (F_CPU == 16000000) clock_prescale_set(clock_div_1);
    #endif
    // End of trinket special code

    strip.begin();
    strip.show(); // Initialize all pixels to 'off'
}

void loop() {
    // Some example procedures showing how to display to the pixels:
    colorWipe(strip.Color(255, 0, 0), 50); // Red
    colorWipe(strip.Color(0, 255, 0), 50); // Green
    colorWipe(strip.Color(0, 0, 255), 50); // Blue
    //colorWipe(strip.Color(0, 0, 255), 50); // White RGBW
    // Send a theater pixel chase in...
    theaterChase(strip.Color(127, 127, 127), 50); // White
    theaterChase(strip.Color(127, 0, 0), 50); // Red
    theaterChase(strip.Color(0, 0, 127), 50); // Blue

    rainbow(20);
    rainbowCycle(20);
    theaterChaseRainbow(50);
}

// Fill the dots one after the other with a color
void colorWipe(uint32_t c, uint8_t wait) {
    for(uint16_t i=0; i<strip.numPixels(); i++) {
        strip.setPixelColor(i, c);
        strip.show();
        delay(wait);
    }
}

void rainbow(uint8_t wait) {
    uint16_t i, j;
}

```

```

        for(j=0; j<256; j++) {
            for(i=0; i<strip.numPixels(); i++) {
                strip.setPixelColor(i, Wheel((i+j) & 255));
            }
            strip.show();
            delay(wait);
        }
    }

// Slightly different, this makes the rainbow equally distributed
throughout

void rainbowCycle(uint8_t wait) {
    uint16_t i, j;

    for(j=0; j<256*5; j++) { // 5 cycles of all colors on wheel
        for(i=0; i< strip.numPixels(); i++) {
            strip.setPixelColor(i, Wheel(((i * 256 / strip.numPixels()) +
j) & 255));
        }
        strip.show();
        delay(wait);
    }
}

//Theatre-style crawling lights.

void theaterChase(uint32_t c, uint8_t wait) {
    for (int j=0; j<10; j++) { //do 10 cycles of chasing
        for (int q=0; q < 3; q++) {
            for (uint16_t i=0; i < strip.numPixels(); i=i+3) {
                strip.setPixelColor(i+q, c);      //turn every third pixel on
            }
            strip.show();

            delay(wait);

            for (uint16_t i=0; i < strip.numPixels(); i=i+3) {
                strip.setPixelColor(i+q, 0);      //turn every third pixel
off
            }
        }
    }
}

```

```

    }

//Theatre-style crawling lights with rainbow effect
void theaterChaseRainbow(uint8_t wait) {
    for (int j=0; j < 256; j++) {      // cycle all 256 colors in the
wheel
        for (int q=0; q < 3; q++) {
            for (uint16_t i=0; i < strip.numPixels(); i=i+3) {
                strip.setPixelColor(i+q, Wheel( (i+j) % 255));      //turn
every third pixel on
            }
            strip.show();

            delay(wait);

            for (uint16_t i=0; i < strip.numPixels(); i=i+3) {
                strip.setPixelColor(i+q, 0);                  //turn every third pixel
off
            }
        }
    }
}

// Input a value 0 to 255 to get a color value.
// The colours are a transition r - g - b - back to r.
uint32_t Wheel(byte WheelPos) {
    WheelPos = 255 - WheelPos;
    if(WheelPos < 85) {
        return strip.Color(255 - WheelPos * 3, 0, WheelPos * 3);
    }
    if(WheelPos < 170) {
        WheelPos -= 85;
        return strip.Color(0, WheelPos * 3, 255 - WheelPos * 3);
    }
    WheelPos -= 170;
    return strip.Color(WheelPos * 3, 255 - WheelPos * 3, 0);
}

```

2.3 PROCODECG LED

2.3.1 ALTERNATE LED

```
/*
This is an alternating blinking LEDs demo
1 0 1 0 1 0
0 1 0 1 0 1

Done for ESPectro board, but can be used by other boards.
Just change the pin definition

This code is in public domain
26 Sep 2016
Budi Rahardjo (@rahard)
*/

// pin definition of ESPectro board
// I am connecting this to ProcodeCG LED board
// thus, this pin configuration
uint8_t BRD1 = 15;
uint8_t BRD2 = 2;
uint8_t BRD3 = 0;
uint8_t BRD4 = 4;
uint8_t BRD5 = 5;
uint8_t BRD6 = 3;

int TUNGGU = 500;

// the setup function runs once when you press reset or power the
board
void setup() {
    pinMode(BRD1, OUTPUT);
    pinMode(BRD2, OUTPUT);
    pinMode(BRD3, OUTPUT);
    pinMode(BRD4, OUTPUT);
    pinMode(BRD5, OUTPUT);
    pinMode(BRD6, OUTPUT);
}

void loop()
```

```

digitalWrite(BRD1, HIGH);
digitalWrite(BRD2, LOW);
digitalWrite(BRD3, HIGH);
digitalWrite(BRD4, LOW);
digitalWrite(BRD5, HIGH);
digitalWrite(BRD6, LOW);
delay(TUNGGU);

digitalWrite(BRD1, LOW);
digitalWrite(BRD2, HIGH);
digitalWrite(BRD3, LOW);
digitalWrite(BRD4, HIGH);
digitalWrite(BRD5, LOW);
digitalWrite(BRD6, HIGH);
delay(TUNGGU);
}

```

2.3.2 LED METER

```

/*
LED Meter

Public Domain
September 2016
Budi Rahardjo (@rahard)

*/

// pin configuration for DycodeX ESPectro board
int led[] = {15, 2, 0, 4, 5, 3};
// pin configuration for Arduino UNU board
// int led[] = {13, 12, 11, 10, 9, 8};
int NUMLED = 6; // number of LEDs
int WAITFOR = 100;
int i;
int LEDON=0;

void setup() {
    // setup pins for output
    for (i=0; i<NUMLED; i++) { pinMode(led[i], OUTPUT); }
}

```

```

void display(int n) {
    // protection
    if (n < 0) { n = 0; }
    if (n > NUMLED) { n = NUMLED; }

    // turn off all LED first
    for (i=0; i<NUMLED; i++) { digitalWrite(led[i], LOW); }

    // turn on LED from 1 to n; 0 means do not display
    for (i=1; i<=n; i++) {
        digitalWrite(led[i-1], HIGH);
    }
}

void loop() {
    int j;
    // increment
    for (j=0; j < NUMLED; j++) {
        display(j);
        delay(WAITFOR);
    }

    for (j=NUMLED; j >= 0; j--) {
        display(j);
        delay(WAITFOR);
    }
}

```

2.3.3 ENTER EXIT

```

/*
Enter - Exit

Public Domain
September 2016
Budi Rahardjo (@rahard)

*/

// pin configuration for DycodeX ESPectro board
int led[] = {15, 2, 0, 4, 5, 3};
// pin configuration for Arduino UNU board
// int led[] = {13, 12, 11, 10, 9, 8};

```

```

int x[] = {0, 0, 0, 0, 0, 0};
int NUMLED = 6; // number of LEDs
int WAITFOR = 100;
int i;

void setup() {
    // setup pins for output and reset low
    for (i=0; i<NUMLED; i++) { pinMode(led[i], OUTPUT); }
    for (i=0; i<NUMLED; i++) { digitalWrite(led[i], LOW); }
}

void display() {
    for (i=0; i < NUMLED ; i++) {
        digitalWrite(led[i], x[i]);
    }
}

void loop() {
    int j;
    // populate LED
    for (j=0; j<NUMLED; j++) {
        x[j]=1;
        display();
        delay(WAITFOR);
    }
    // unpopulate LED
    for (j=0; j<NUMLED; j++) {
        x[j]=0;
        display();
        delay(WAITFOR);
    }
}

```

2.3.4 KNIGHT RIDER

```

/*
Knight Rider LED

Public Domain
September 2016

```

```
Budi Rahardjo (@rahard)
```

```
*/  
  
// pin configuration for DycodeX ESPectro board  
// int led[] = {15, 2, 0, 4, 5, 3};  
// pin configuration for Arduino UNU board  
int led[] = {13, 12, 11, 10, 9, 8};  
int NUMLED = 6; // number of LEDS  
int WAITFOR = 10;  
int i;  
int LEDON=0;  
int direction=1;  
  
void setup() {  
    // setup pins for output  
    for (i=0; i<NUMLED; i++) { pinMode(led[i], OUTPUT); }  
}  
  
void loop() {  
  
    // turn off all LED  
    for (i=0; i<NUMLED; i++) { digitalWrite(led[i], LOW); }  
    // turn on LED with current counter  
    digitalWrite(led[LEDON], HIGH);  
  
    delay(WAITFOR);  
  
    LEDON = LEDON + direction;  
  
    // if already hit the largest number, reverse direction  
    if (LEDON >= NUMLED) {  
        direction = -1;  
        LEDON = NUMLED - 1;  
    }  
  
    // if already hit the lowest number  
    if (LEDON < 0) {  
        direction = 1;  
        LEDON = 0;  
    }  
}
```

```
}
```

3 CONNECTING TO CLOUD

3.1 CONNECTING TO WIFI

```
// Example to connect Espectro to WiFi
// code taken from places
// public domain
// Budi Rahardjo @rahard 2016

#include <ESP8266WiFi.h>
#include <ESP8266WiFiMulti.h>

ESP8266WiFiMulti WiFiMulti;

int z=0;

void setup() {
    Serial.begin(9600);
    delay(1000);

    WiFiMulti.addAP("procodecg LG", "12345678");

    while(WiFiMulti.run() != WL_CONNECTED) {
        Serial.print(".");
        delay(500);
        // just to limit the tries ... so that I doesnt loop forever
        z++;
        if (z>100) {
            Serial.println("Fail connecting. Exiting");
            return;
        }
    }
}
```

```

void loop() {

    // yayyy we are connected
    Serial.println("");
    Serial.println("WiFi connected");
    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());

    delay(500);
}

```

3.2 CONNECTING TO MQTT

REFERENCES:

1. <http://stackoverflow.com/questions/26716279/how-to-test-the-mosquitto-server>
2. <http://www.hivemq.com/hivemq/>
3. <http://www.hivemq.com/resources/getting-started/>
4. <http://www.mqtt-dashboard.com/index.html>

3.2.1 ESPectro LIBRARY – Basic ESP8266 Example

```

/*
Basic ESP8266 MQTT example

This sketch demonstrates the capabilities of the pubsub library in
combination
with the ESP8266 board/library.

It connects to an MQTT server then:
- publishes "hello world" to the topic "outTopic" every two seconds
- subscribes to the topic "inTopic", printing out any messages
  it receives. NB - it assumes the received payloads are strings
not binary
- If the first character of the topic "inTopic" is an 1, switch ON
the ESP Led,

```

```

else switch it off

It will reconnect to the server if the connection is lost using a
blocking
reconnect function. See the 'mqtt_reconnect_nonblocking' example for
how to
achieve the same result without blocking the main loop.

To install the ESP8266 board, (using Arduino 1.6.4+):
- Add the following 3rd party board manager under "File ->
Preferences -> Additional Boards Manager URLs":
http://arduino.esp8266.com/stable/package\_esp8266com\_index.json
- Open the "Tools -> Board -> Board Manager" and click install for
the ESP8266"
- Select your ESP8266 in "Tools -> Board"

*/



#include <ESP8266WiFi.h>
#include <PubSubClient.h>

// Update these with values suitable for your network.

const char* ssid = "DyWare-AP2";
const char* password = "957DaegCen";
const char* mqtt_server = "iothub.id";

WiFiClient espClient;
PubSubClient client(espClient);
long lastMsg = 0;
char msg[50];
int value = 0;

void setup_wifi() {

    delay(10);
    // We start by connecting to a WiFi network
    Serial.println();
    Serial.print("Connecting to ");
    Serial.println(ssid);
}

```

```

WiFi.begin(ssid, password);

while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
}

randomSeed(micros());

Serial.println("");
Serial.println("WiFi connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());
}

void callback(char* topic, byte* payload, unsigned int length) {
    Serial.print("Message arrived [");
    Serial.print(topic);
    Serial.print("] ");
    for (int i = 0; i < length; i++) {
        Serial.print((char)payload[i]);
    }
    Serial.println();

    // Switch on the LED if an 1 was received as first character
    if ((char)payload[0] == '1') {
        digitalWrite(BUILTIN_LED, LOW);      // Turn the LED on (Note that
        // LOW is the voltage level
        // but actually the LED is on; this is because
        // it is active low on the ESP-01)
    } else {
        digitalWrite(BUILTIN_LED, HIGH);    // Turn the LED off by making
        // the voltage HIGH
    }
}

void reconnect() {
    // Loop until we're reconnected
    while (!client.connected()) {

```

```

Serial.print("Attempting MQTT connection...");
// Create a random client ID
String clientId = "ESP8266Client-";
clientId += String(random(0xffff), HEX);
// Attempt to connect
if (client.connect(clientId.c_str(), "procodecg",
"procodeiothubCG")) {
    Serial.println("connected");
    // Once connected, publish an announcement...
    //client.publish("procodecg/procodecgproject/data", "hello
world");
    // ... and resubscribe
    client.subscribe("inTopic");
} else {
    Serial.print("failed, rc=");
    Serial.print(client.state());
    Serial.println(" try again in 5 seconds");
    // Wait 5 seconds before retrying
    delay(5000);
}
}

void setup() {
pinMode(BUILTIN_LED, OUTPUT);      // Initialize the BUILTIN_LED pin
as an output
Serial.begin(115200);
setup_wifi();
client.setServer(mqtt_server, 1883);
client.setCallback(callback);
}

void loop() {

if (!client.connected()) {
    reconnect();
}
client.loop();

long now = millis();
if (now - lastMsg > 2000) {

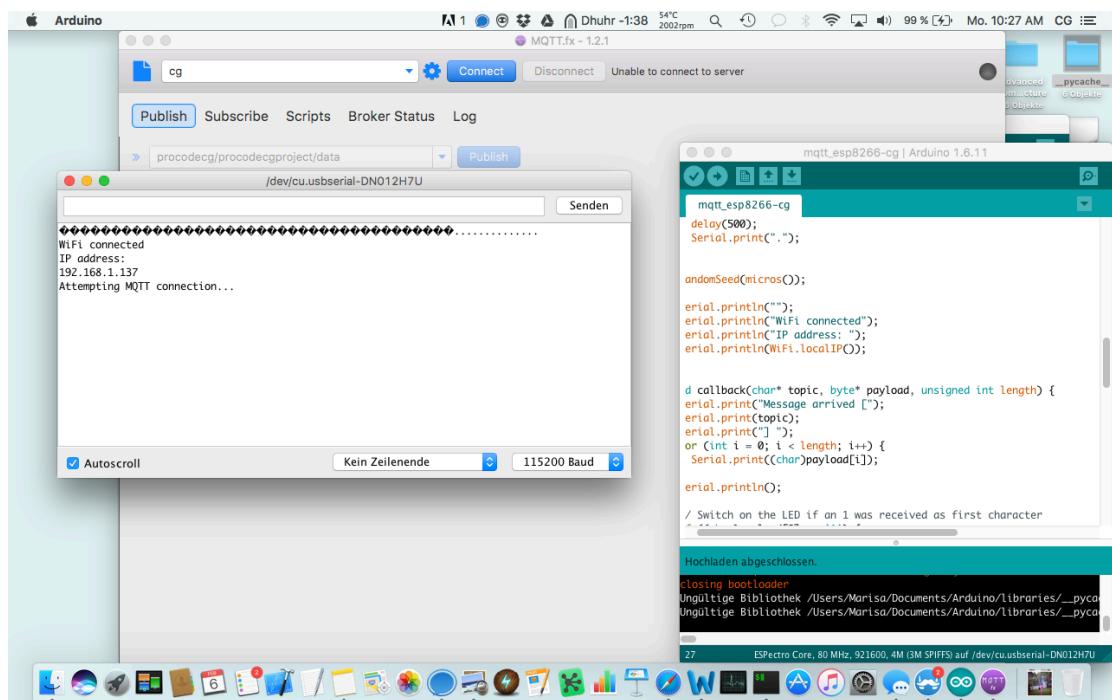
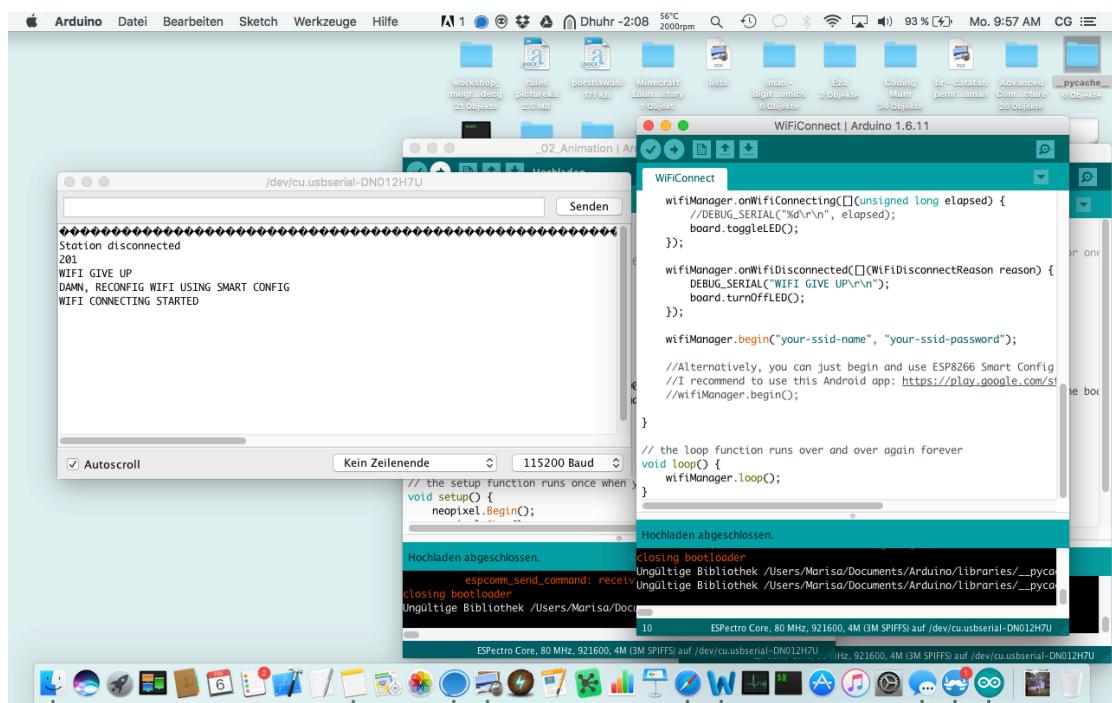
```

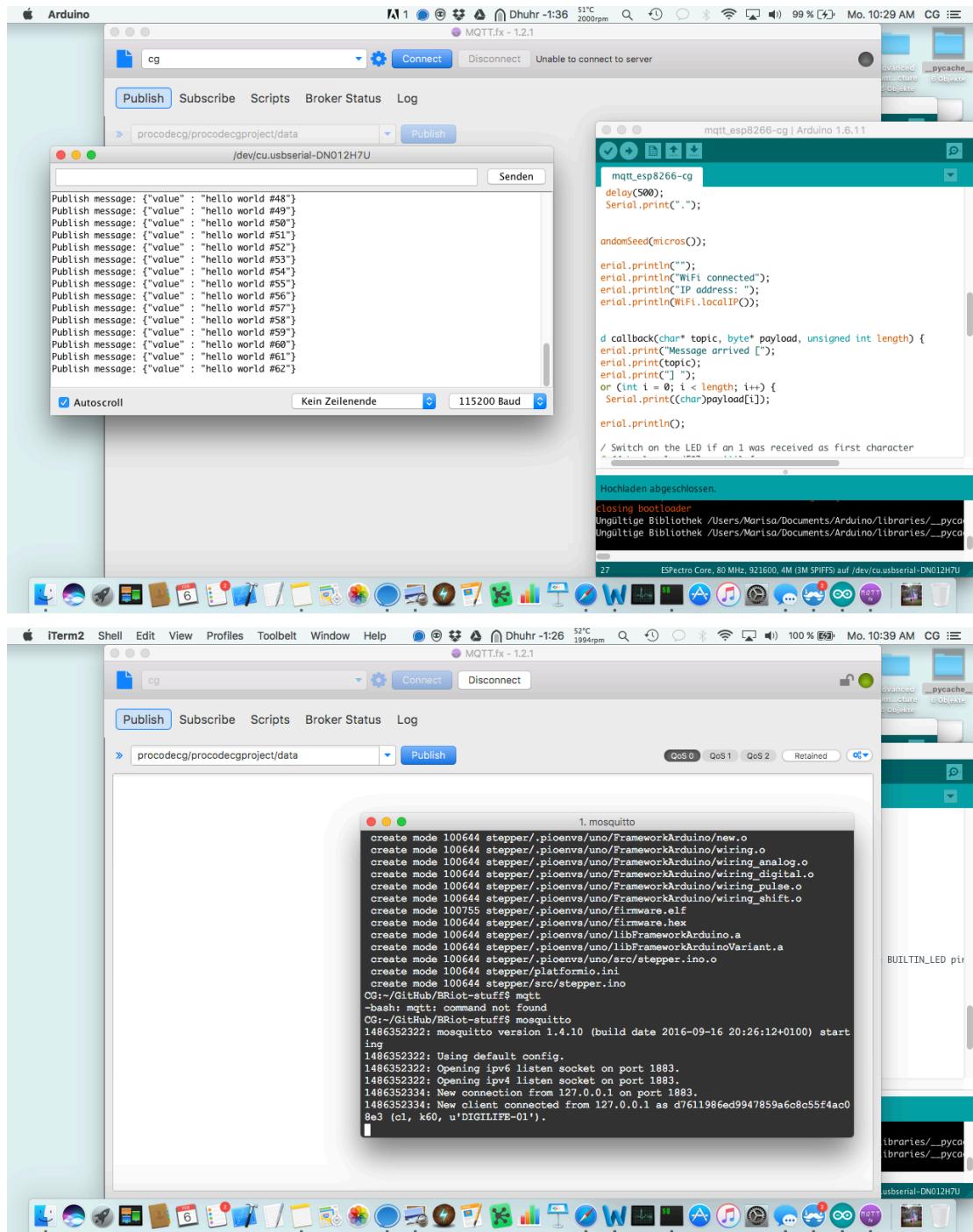
```

lastMsg = now;
++value;

snprintf (msg, 75, "{\"value\" : \"hello world #%ld\"", value);
Serial.print("Publish message: ");
Serial.println(msg);
client.publish("procodecg/procodecgproject/data", msg);
}
}

```





16 Is there any way to test the Mosquitto server?

4. mosquito_sub

```
Last login: Mon Feb 6 10:43:47 on ttys002
CG:~$ mosquitto_sub -v -t 'test/cg'
test/cg hello cg
test/cg hello cg
test/cg hello cg
[...]
add a comment.
```

2. bash

```
Last login: Mon Feb 6 10:14:19 on ttys000
CG:~$ mosquitto_pub -t 'test/topic' -m 'helloWorld'
CG:~$ mosquitto_pub -t 'test/cg' -m 'hello cg'
CG:~$ Start the command line subscriber.
mosquitto_sub -v -t 'test/topic'
```

3. Publish test message with the command line publisher:

```
mosquitto_pub -t 'test/topic' -m 'helloWorld'
```

As well as seeing both the subscriber and publisher connection messages in the broker terminal the following should be printed in the subscriber terminal:

```
test/topic helloWorld
```

share improve this answer

answered Nov 3 '14 at 14:37 by hardillb

0 MQTT implementation using Arduino and Mosquitto

Related

MQTT.fx - 1.2.1

MQTT.fx

cg Connect Disconnect

Publish Subscribe Scripts Broker Status Log

procodecg/procodecgproject/data Publish

2. mosquito_sub

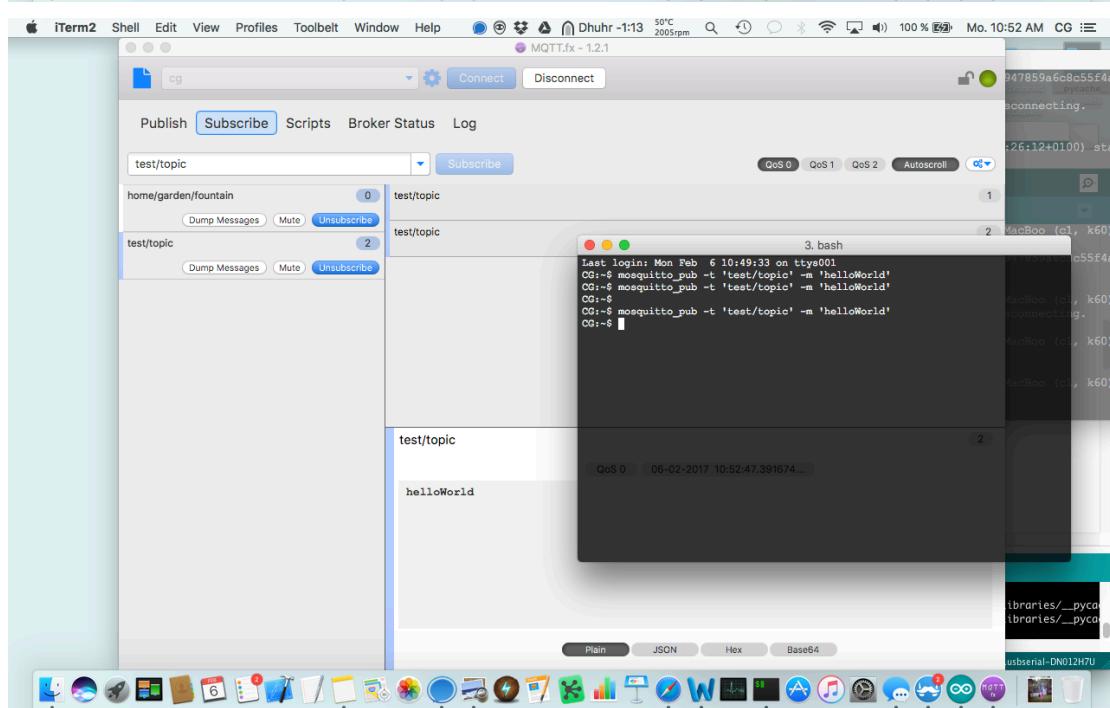
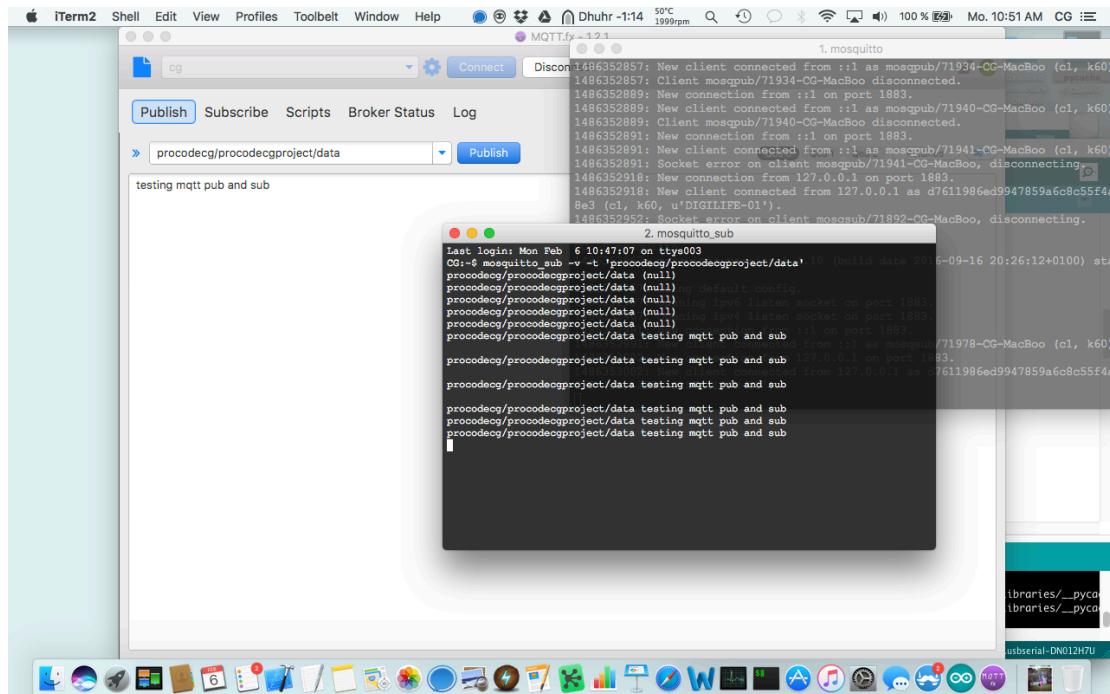
```
Last login: Mon Feb 6 10:47:07 on ttys003
CG:~$ mosquitto_sub -v -t 'procodecg/procodecgproject/data' | cat
procodecg/procodecgproject/data (null) serial.println();
procodecg/procodecgproject/data (null) serial.print("Connecting to ");
procodecg/procodecgproject/data (null) serial.println(ssid);
procodecg/procodecgproject/data (null) WiFi.begin(ssid, password);
[...]
while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
}
randomSeed(micros());
Serial.println("");
Serial.println("WiFi connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());
```

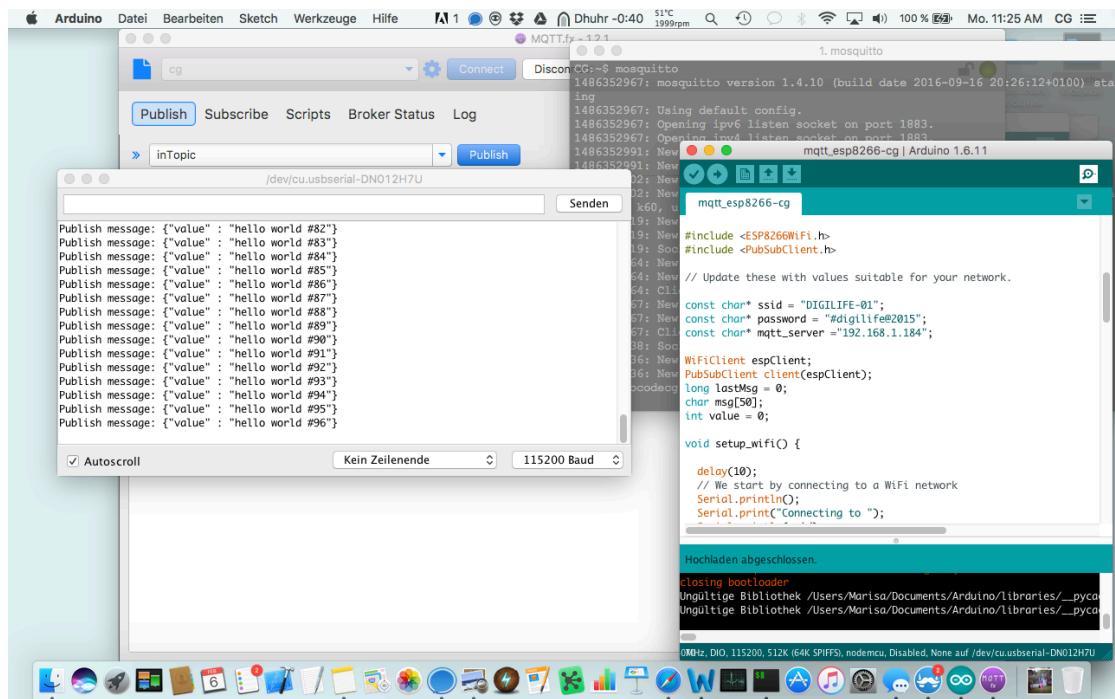
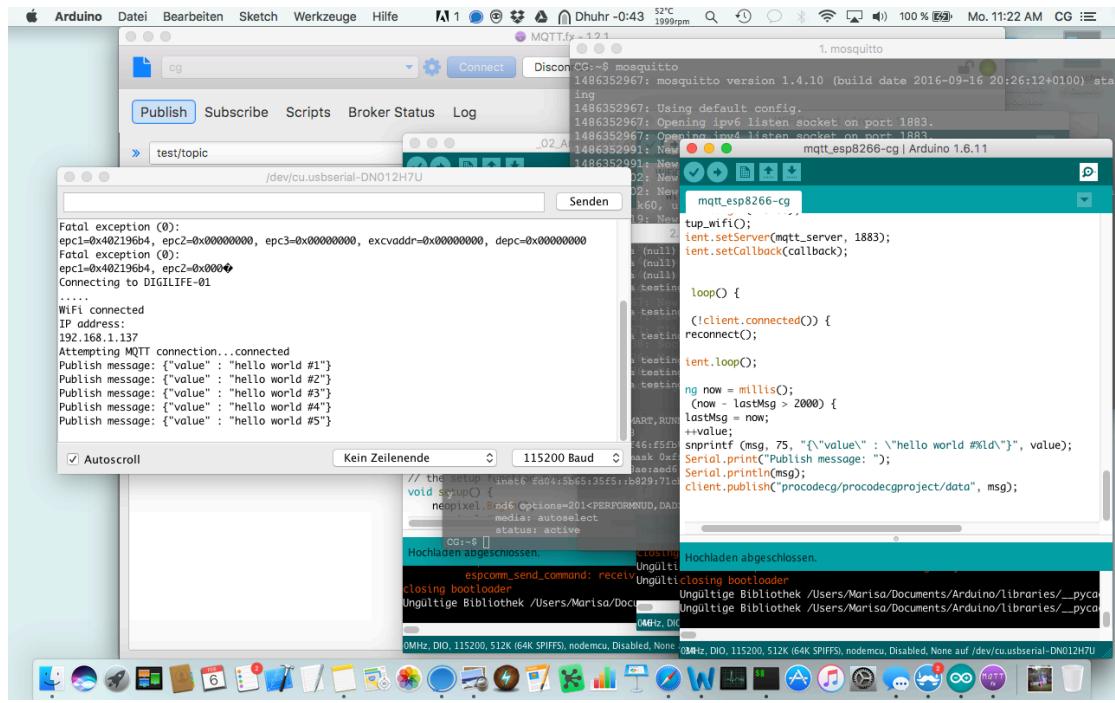
Hochladen abgeschlossen.

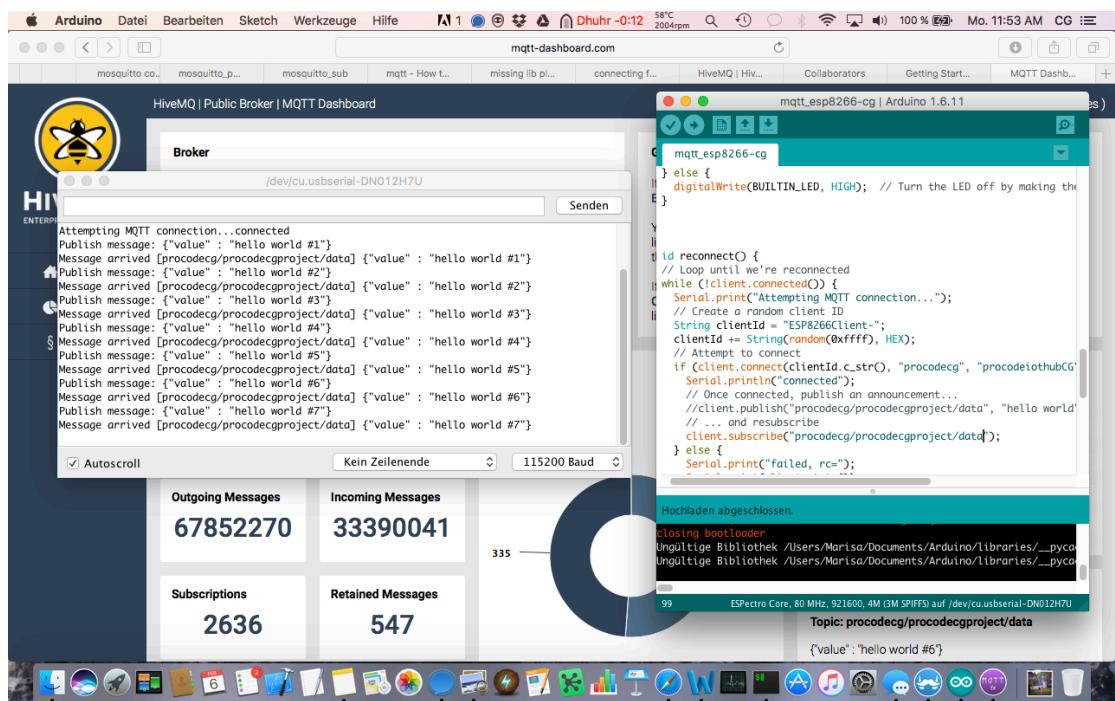
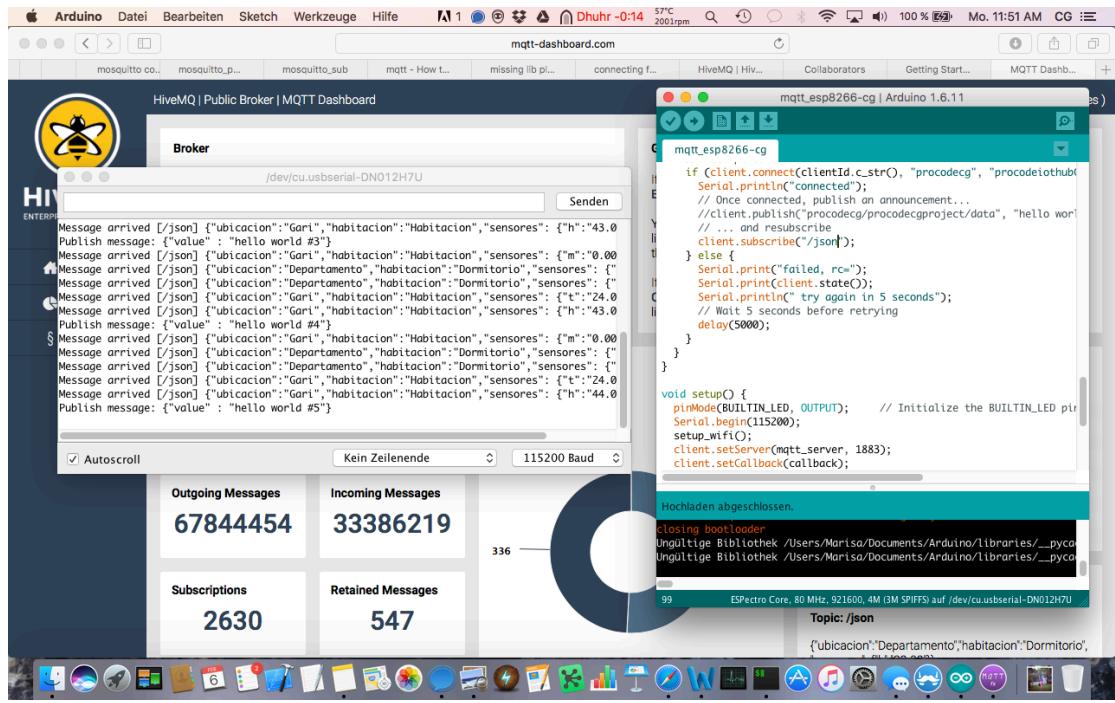
closing bootloader

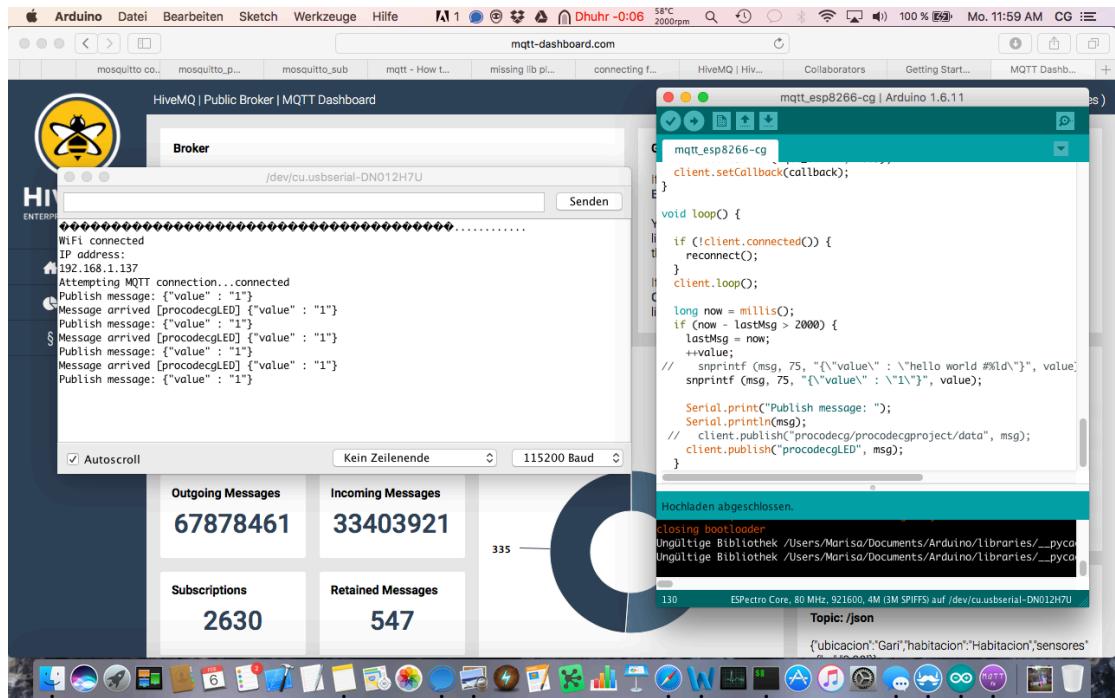
Ungültige Bibliothek /Users/Marisa/Documents/Arduino/libraries/_pycodecg
Ungültige Bibliothek /Users/Marisa/Documents/Arduino/libraries/_pycodecg

60 ESP8266 Core, 80 MHz, 921600, 4M (3M SPIFFS) auf /dev/cu.usbserial-DN012H7U









Testing MQTT Server

In separate terminal windows do the following:

1. Start the broker:

```
mosquitto
```

2. Start the command line subscriber:

```
mosquitto_sub -v -t 'test/topic'
```

3. Publish test message with the command line publisher:

```
mosquitto_pub -t 'test/topic' -m 'helloWorld'
```

As well as seeing both the subscriber and publisher connection messages in the broker terminal the following should be printed in the subscriber terminal:

```
test/topic helloWorld
```

3.2.2 ESPectro LIBRARY – LEDByCloud

3.3 TRAFFIC LIGHT MQTT

```
#include <Adafruit_NeoPixel.h>
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

#define PIN 15
#define NUMPIXELS 1
```

```

const char* ssid = "procodecg";
const char* password = "12345678";
//const char* mqtt_server = "broker.mqtt-dashboard.com";
//const char* mqtt_server = "192.168.1.118";
const char* mqtt_server = "192.168.1.101";

Adafruit_NeoPixel pixels = Adafruit_NeoPixel(NUMPIXELS, PIN, NEO_GRB
+ NEO_KHZ800);
int delayval = 1000;
int i=0;

WiFiClient espClient;
PubSubClient client(espClient);
long lastMsg = 0;
char msg[50];
int value = 0;

void setup_wifi() {

    delay(10);
    // We start by connecting to a WiFi network
    Serial.println();
    Serial.print("Connecting to ");
    Serial.println(ssid);

    WiFi.begin(ssid, password);

    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print("x");
    }

    randomSeed(micros());

    Serial.println("");
    Serial.println("WiFi connected");
    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());
}

```

```

void callback(char* topic, byte* payload, unsigned int length) {
    Serial.print("Message arrived [");
    Serial.print(topic);
    Serial.print("] ");
    for (int i = 0; i < length; i++) {
        Serial.print((char)payload[i]);
    }
    Serial.println();

    // Switch on the LED if an 1 was received as first character
    if ((char)payload[0] == '0') {
        digitalWrite(BUILTIN_LED, LOW);      // Turn the LED on (Note that
        LOW is the voltage level
        // but actually the LED is on; this is because
        // it is active low on the ESP-01)
    } else {
        digitalWrite(BUILTIN_LED, HIGH);     // Turn the LED off by making
        the voltage HIGH
    }
}

void reconnect() {
    // Loop until we're reconnected
    while (!client.connected()) {
        Serial.print("Attempting MQTT connection...");
        // Create a random client ID
        String clientId = "ESP8266Client-";
        clientId += String(random(0xffff), HEX);
        // Attempt to connect
        if (client.connect(clientId.c_str())) {
            Serial.println("connected");
            // Once connected, publish an announcement...
//            client.publish("outTopic", "hello world");
            // ... and resubscribe
            client.subscribe("led1");
        } else {
            Serial.print("failed, rc=");
            Serial.print(client.state());
            Serial.println(" try again in 5 seconds");
            // Wait 5 seconds before retrying
        }
    }
}

```

```

        delay(5000);
    }
}
}

void setup() {
    pixels.begin(); // This initializes the NeoPixel library.
    Serial.begin(9600);
    setup_wifi();
    client.setServer(mqtt_server, 1883);
    client.setCallback(callback);
}

void loop() {
    if (!client.connected()) {
        reconnect();
    }
    client.loop();

    // red
    client.publish("led1", "red");
    pixels.setPixelColor(0, pixels.Color(63,0,0));
    pixels.show();
    delay(3000);

    client.publish("led1", "yellow");
    for (i=0; i<5; i++) {
        // yellow
        pixels.setPixelColor(0, pixels.Color(63,63,0)); pixels.show();
    delay(500);
        // blank
        pixels.setPixelColor(0, pixels.Color(0,0,0)); pixels.show();
    delay(500);
    }

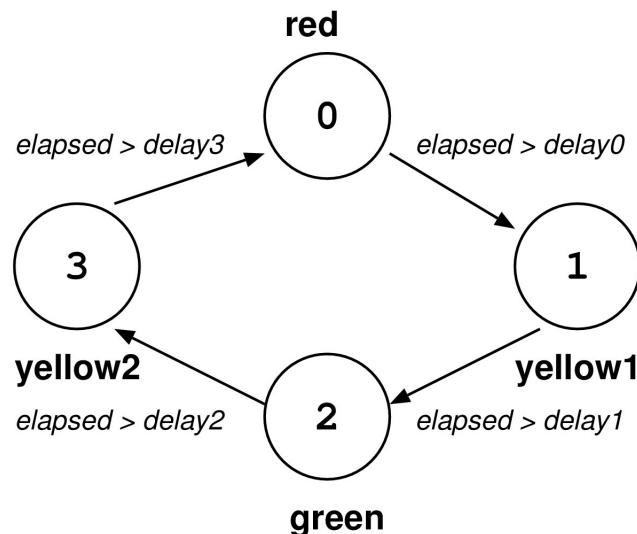
    // green
    client.publish("led1", "green");
    pixels.setPixelColor(0, pixels.Color(0,63,0));
}

```

```
pixels.show();
delay(3000);

client.publish("led1", "yellow");
for (i=0; i<3; i++) {
    // orange
    pixels.setPixelColor(0, pixels.Color(63,40,0)); pixels.show();
delay(500);
    // blank
    pixels.setPixelColor(0, pixels.Color(0,0,0)); pixels.show();
delay(500);
}
}
```

3.4 TRAFFIC LIGHT FSM



```
/* traffic light with FSM */
/* Budi Rahardjo @rahard 2017 */

#include <Adafruit_NeoPixel.h>
#define PIN 15
#define NUMPIXELS 1

Adafruit_NeoPixel pixels = Adafruit_NeoPixel(NUMPIXELS, PIN, NEO_GRB
+ NEO_KHZ800);
int state=0; // 0=red, 1=yellow1, 2=green; 3=yellow2
int delay0=3000;
int delay1=1000;
int delay2=3000;
int delay3=2000;
unsigned long lastTime=0;
unsigned long elapsed=0;

void setup() {
    pixels.begin();
    state=1; // start with red
    lastTime = millis(); //
    Serial.begin(9600);
}

void loop() {
    elapsed = millis() - lastTime;
    // if red -> yellow1
```

```

if ((state==0) && (elapsed > delay0)) {
    state=1;
    lastTime = millis();
    elapsed=0;
    pixels.setPixelColor(0, 63, 30, 0);
    pixels.show();
    Serial.println("yellow1");
}

// yellow1 -> green
if ((state==1) && (elapsed > delay1)) {
    state=2;
    lastTime = millis();
    elapsed=0;
    pixels.setPixelColor(0, 0, 63, 0);
    pixels.show();
    Serial.println("green");
}

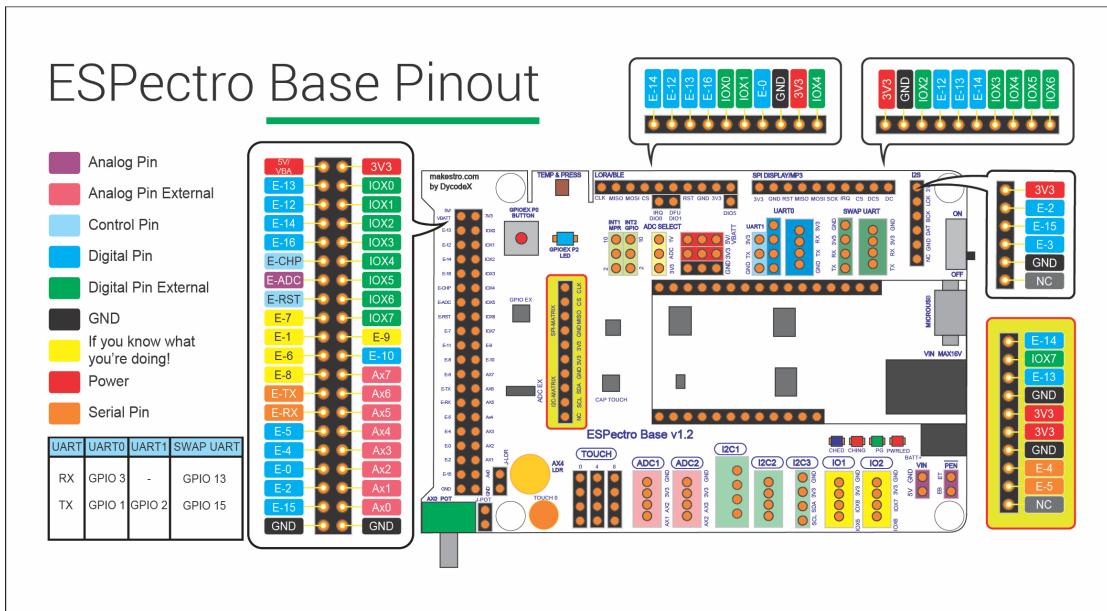
// green -> yellow2
if ((state==2) && (elapsed > delay2)) {
    state=3;
    lastTime = millis();
    elapsed=0;
    pixels.setPixelColor(0, 63, 30, 0);
    pixels.show();
    Serial.println("yellow2");
}

// yellow2 -> red
if ((state==3) && (elapsed > delay3)) {
    state=0;
    lastTime = millis();
    elapsed=0;
    pixels.setPixelColor(0, 63, 0, 0);
    pixels.show();
    Serial.println("red");
}

Serial.println("unknown state");
}

```

4 ESPECTRO BASE



4.1 POTENTIOMETER

CODE:

```
/*
Getting started with on-board ADC on ESPectro Base.

This sketch demos how to read analog value from on-board
potentiometer

Created by 22 Sep 2016
by Andri Yadi
*/

#include <ESPectroBase.h>

ESPectroBase base;

int lastAdcPot = 0;

// the setup function runs once when you press reset or power the
board

void setup() {
    Serial.begin(115200);
    //Wait Serial to be ready
}
```

```

while (!Serial);

    //Should explicitely call this to activate ADC
    base.beginADC();
}

// the loop function runs over and over again forever
void loop() {
    int pot = base.analogRead(ESPECTRO_BASE_ADC_POT_CHANNEL);
    if (pot != -1 && abs(pot - lastAdcPot) > 10) {
        lastAdcPot = pot;
        Serial.printf("Pot value: %d\r\n", pot);
    }

    delay(500);
}

```

4.2 LDR

CODE:

```

/*
Getting started with on-board ADC on ESPectro Base.
This sketch demoes how to read analog value from on-board LDR

Created by 22 Sep 2016
by Andri Yadi
*/

#include <ESPectro.h>
#include <ESPectroBase.h>

ESPectro board;
ESPectroBase base;

int lastAdcLdr = 0;
const int LED_THRESHOLD = 500;

```

```

// the setup function runs once when you press reset or power the
board

void setup() {
    Serial.begin(115200);
    //Wait Serial to be ready
    while (!Serial);

    //Should explicitely call this to activate ADC
    base.beginADC();
}

// the loop function runs over and over again forever
void loop() {
    int ldr = base.analogRead(ESPECTRO_BASE_ADC_LDR_CHANNEL);
    if (ldr != -1 && abs(ldr - lastAdcLdr) > 10) {
        lastAdcLdr = ldr;
        Serial.printf("Pot value: %d\r\n", ldr);

        if (ldr < LED_THRESHOLD) {
            board.turnOnLED();
        }
        else {
            board.turnOffLED();
        }
    }

    delay(500);
}

```