**MCU**

función Read\_esp32()

Si UART\_DATA\_Ready es verdadero

Mientras RCIF es falso realizar loop infinito

Establecer PORTD a RCREG;

\_\_delay\_ms(100);

función Write\_esp32()

Establecer TXREG a dato\_a\_enviar;

Mientras TRMT es falso realizar loop infinito

\_\_delay\_ms(100);

función Read\_sensor()

call I2C\_Master\_Start();

call I2C\_Master\_Stop();

configuración setup()

establecer TRISD a 0;

establecer PORTD a 0;

establecer TRISC a 0;

establecer PORTC a 0;

principal main()

call setup();

call I2C\_Master\_Init()

UART\_Init();

loop principal

call Read\_sensor();

call Write\_esp32();

call Read\_esp32();

**UART\_Library**

función UART\_Init()

establecer SPBRG a 25;

establecer BRGH a 1;

establecer SPBRGH a 0;

establecer SYNC a 0;

esatablecer SPEN a 1;

establecer TX9 a 0;

establecer TRISC7 a 1;

establecer TRISC6 a 0;

establecer CREN a 1;

establecer TXEN a 1;

función UART\_DATA\_Ready()

retornar RCIF;

**I2C\_Library**

función I2C\_Master\_Init()

establecer SSPADD;

establecer SSPSTAT a 0;

establecer SSPCON a 0b00101000;

establecer TRISC3 a 1;

establecer TRISC4 a 1;

función I2C\_Master\_Wait()

Mientras (SSPSTAT & 0b00000100) || (SSPCON2 & 0b00011111)

// loop infinito

función I2C\_Master\_Start()

call I2C\_Master\_Wait();

establecer SEN a 1;

función I2C\_Master\_Stop()

call I2C\_Master\_Wait();

establecer PEN a 1;

**pubsup\_adafruit**

establecer Adafruit\_Feed \*P1 = io.feed(“P1”);

establecer Adafruit\_Feed \*P1 = io.feed(“P2”);

establecer Adafruit\_Feed \*SENSOR = io.feed(“SENSOR”);

principal configuracion\_principal

Serial.begin(115200);

establecer parametros de Serial2;

call io.connetc(); // conectar con io.adafruit.com

asignar un onMessage a P1

asignar un onMessage a P2

asignar un onMessage a SENSOR

Mientras io.status() sea menor que AIO\_CONNECTED

Imprimir io.StatusText();

delay(500);

P1->get();

P2->get();

SENSOR->get();

Loop principal

Mientras Serial2.available sea verdadero

Establecer Read\_pic a Serial2.read();

Write\_pic = State\_p1 + State\_p2;

Call Serial2.Write y enviar Write\_pic;

Si millis() es mayor que (lastUpdate + IO\_LOOP\_DELAY)

Guardar Read\_pic en el feed SENSOR;

Establecer lastUpdate a millis();

función handleSensor()

imprimir “received <-”;

imprimir value del feed SENSOR;

función handleP1()

Si data->isTrue()

Establecer state\_p1 a 1;

Si no

Establecer state\_p1 a 0;

función handleP2()

Si data->isTrue()

Establecer state\_p2 a 2;

Si no

Establecer state\_p2 a 0;