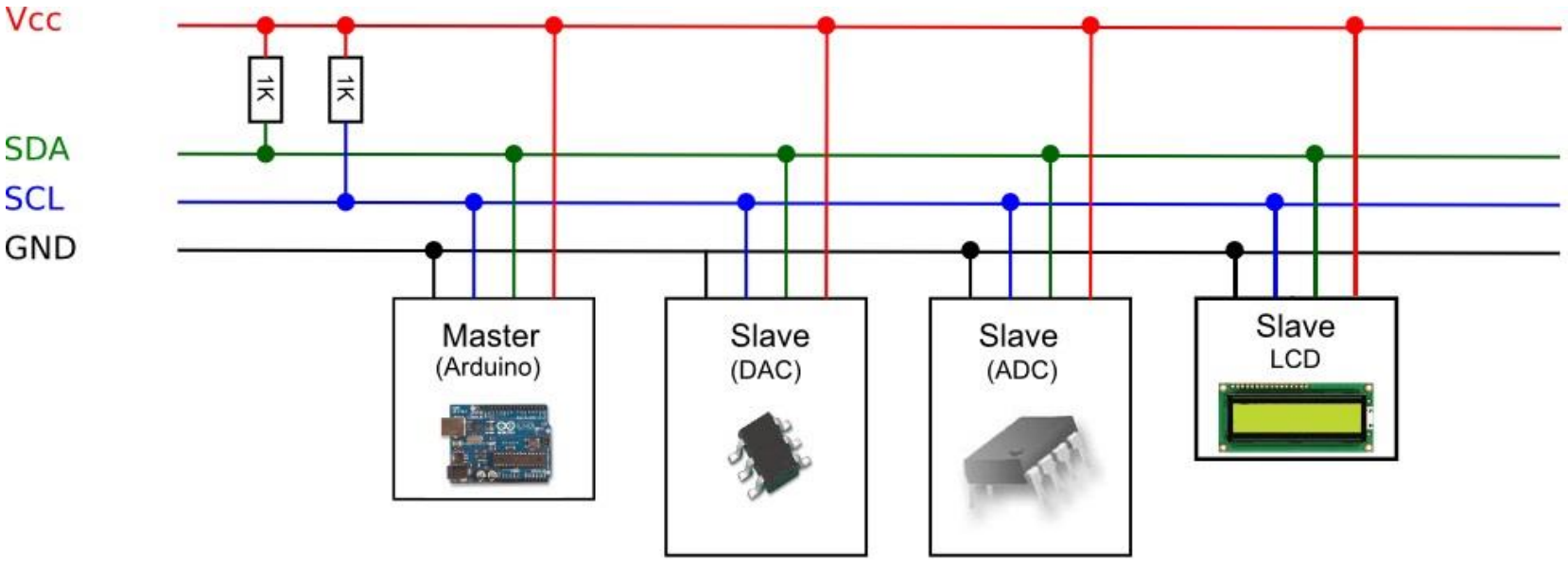
Curso de Arduino avanzado

Rama de estudiantes del IEEE de la universidad de Málaga

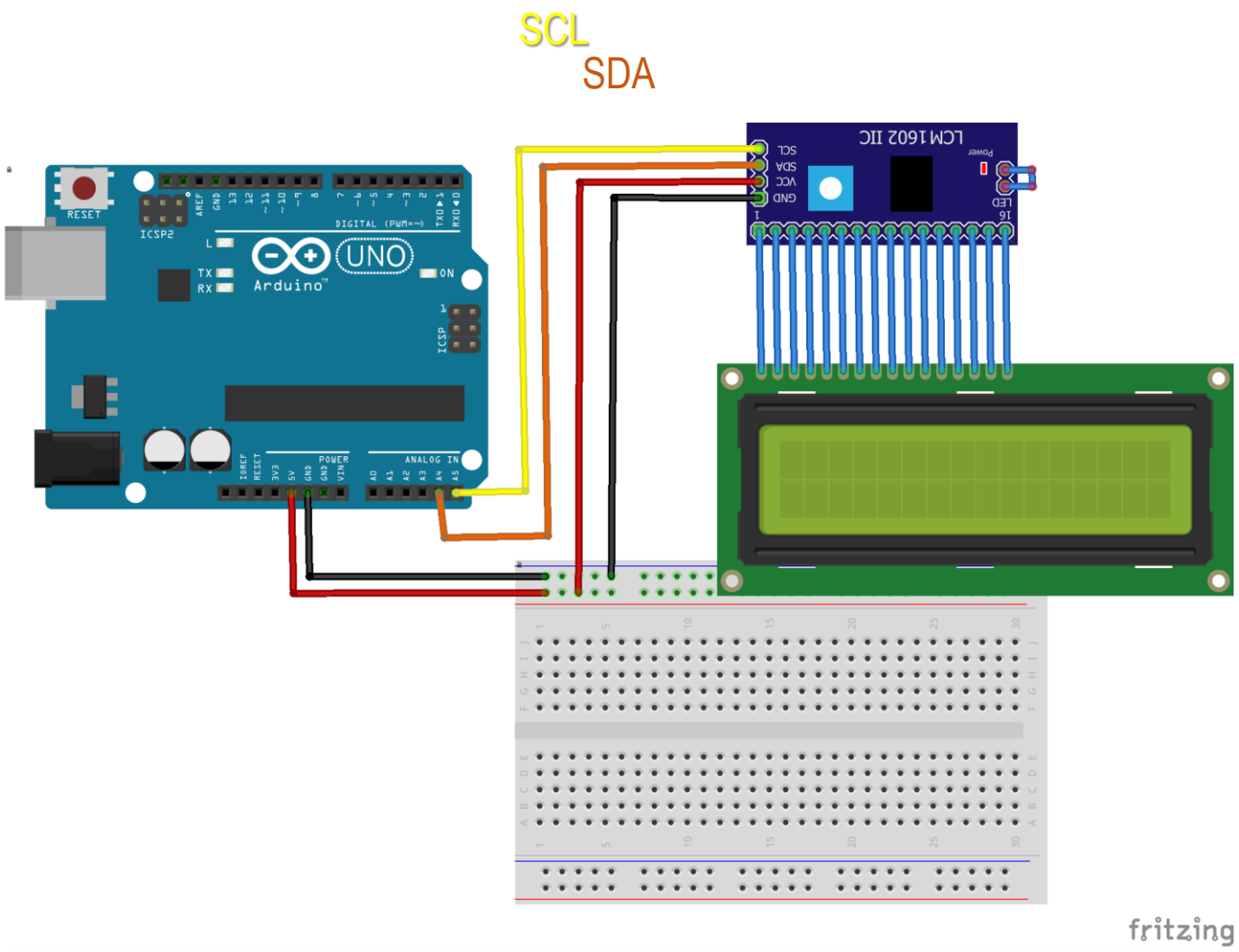
Bus I2C

Es un bus serie , síncrono, multi-maestro, multi-esclavo, enrutamiento por paquetes.

Nodo maestro → Genera la señal de reloj e inicia la comunicación con los esclavos.
Nodo esclavo → Recibe el reloj y responde cuando el maestro se dirige a él.



Conectar LCD 16x2 I2C



LCD: Hello World

```
LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);  // Set the LCD I2C address


void setup()
{
  lcd.begin(16,2);               // initialize the lcd 16x2

  lcd.home ();                   // go home
  lcd.print("Hello, ARDUINO ");
  lcd.setCursor ( 0, 1 );        // go to the next line
  lcd.print (" WORLD!  ");
}
```

LCD: Imprimir datos puerto serie

```
Serial.begin(9600);
```

Inicializa el Puerto serie hardware con una velocidad de 9600 bits/seg

```
Serial.available()
```

Devuelve número de bytes en el buffer de RX del Puerto Serial

```
// clear the screen
lcd.clear();
```

LCD: Desplazar por la pantalla

```
// scroll one position right:
lcd.scrollDisplayLeft();

// scroll one position right:
lcd.scrollDisplayRight();
```

LCD: Crear caracteres

```
// Creat a set of new characters
const uint8_t charBitmap[][8] = {
   { 0x00, 0x11, 0x00, 0x00, 0x11, 0x0E, 0x00}
};

int charBitmapSize=(sizeof(charBitmap )/sizeof(charBitmap[0]));


for ( int i = 0; i < charBitmapSize; i++ )
{
   lcd.createChar ( i, (uint8_t *)charBitmap[i] );
}
lcd.write(byte(0));
```

Temporizadores: Timer1

Contador de 16 bit
Modos: Cuenta, Overflow, captura-comparación, generador de PWM

Bit	7	6	5	4	3	2	1	0	
	TCNT1[15:8]								TCNT1H
	TCNT1[7:0]								TCNT1L
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Modo Overflow:

- El registro Contador TCNT1 cuenta de 0 a 65536.
- Cuando llega al máximo el flag TOV1 del registro TIFR se pone a ‘1’ y TCNT1 es puesto a 0 automáticamente.
-

Bit	7	6	5	4	3	2	1	0	
	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0	TIFR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Temporizadores: Timer1

¡Problema!

La cuenta se incrementa en 1 por cada flanco de reloj del Sistema. Con un reloj de 16MHz como en nuestro caso, él máximo tiempo possible es:

$$T\_\max = 65536 \times \frac{1}{F\_Osc} \approx 4\,ms$$

En el mundo real es necesario medir tiempos más grandes, para ello el Timer posee divisores de frecuencia llamados prescarlers que nos permiten aumentar el tiempo entre incrementos de la cuenta. Este preescaler se puede configurar en los siguientes modos.

CS12	CS11	CS10	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	$clk_{I/O}$/1 (No prescaling)
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	$clk_{I/O}$/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)
1	1	0	External clock source on T1 pin. Clock on falling edge.
1	1	1	External clock source on T1 pin. Clock on rising edge.

Temporizadores: Timer1

Si configuramos el prescaler a 1024 con un cristal de 16MHz el tiempo máximo aumenta hasta:

$$T\_\max = 65536 \times \frac{1024}{F\_Osc} \approx 4.19\ s$$

En nuestro caso vamos a hacer un reloj, por lo que necesitamos poder contra segundos. Para ello vamos a configurar un preescaler de 1024 y vamos a configurar el Timer1 para que llegue al overflow cada 100ms. De esta manera cada 10 veces que llegue al overflow habrá pasado un segundo.

Bit	7	6	5	4	3	2	1	0	
	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	TCCR1B
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Para ello hay que poner a ‘1’ el bit CS12 del registro de control TCCR1B

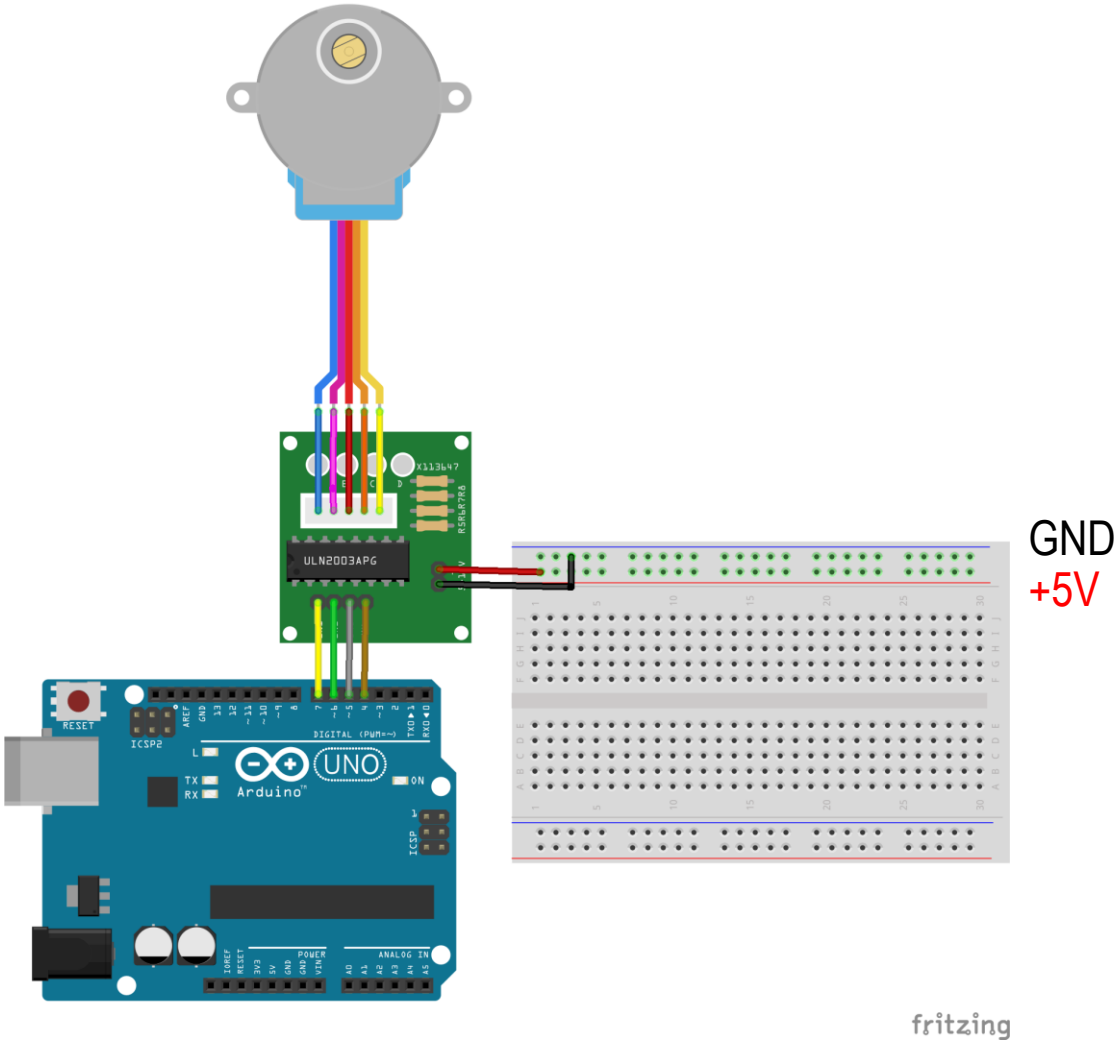
Temporizadores: Timer1

Vamos a calculary que valor hay que cargar en el contador del Timer1 para que llegue al overflow cada 100ms, mediante la siguiente expression.

$$TCNT = 2^{16} - (Tseg) \times \left(\frac{F_{Osc}}{prescaler}\right)$$

$$TCNT = 2^{16} - (100 \times 10^{-3}) \times \left(\frac{16 \times 10^{6}}{1024}\right) = 63973{,}5$$

Conectar Motor PaP y driver



Librería Stepper.h

Constructor de la clase

```
Stepper motor(stepsPerRev, 7, 6, 5, 4);
```

Velocidad

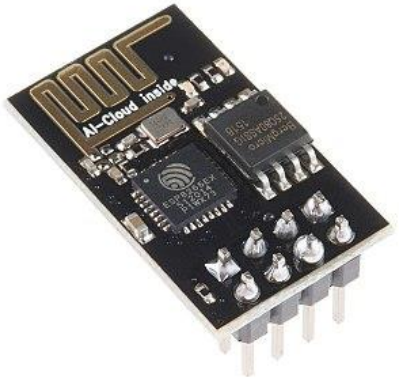
```
motor.setSpeed(RPMs);
```

Pasos

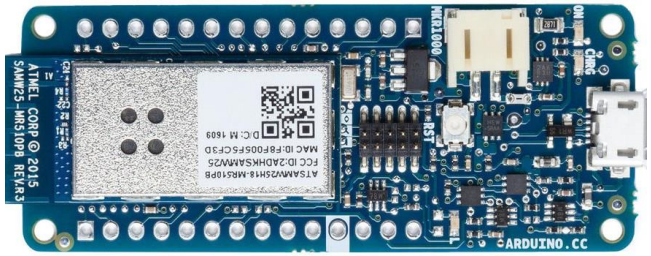
```
motor.step(steps);
```

La revolución del IoT

- Hoy en día cualquier microcontrolador puede conectarse fácilmente a una red gracias a los nuevos chips the ethernet con la pila de TCP/IP integrada.



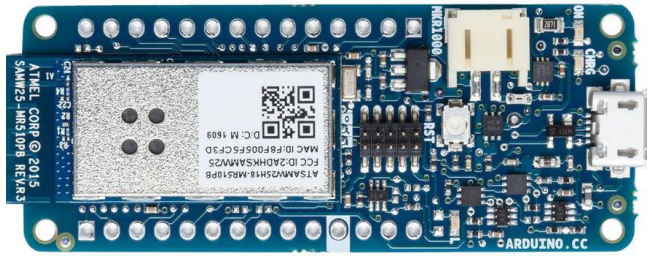
- Diferentes dispositivos



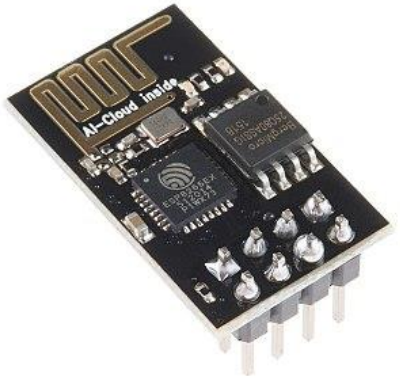
La revolución del IoT

- Uno entre todos marca la diferencia y ha supuesto una revolución.

69€



31€



~2€

ESP8266

- Chip de ultra-bajo coste.
- Microprocesador RISC (Parecido a ARM) de 32 bit 80 MHz.
- Flash de 4MB.
- Compatible con IEEE 802.11 b/g/n Wi-Fi. Autentificación WEP (con limitaciones), WPA y WPA2.
- 16 GPIO pins.
- ADC de 10-bit por aproximaciones sucesivas.
- Temperatura de operación de -40º a 125º nos permite hacer dispositivos preparados para intemperie.
- Ultra-Bajo consume hasta 20uA utilizando un reloj externo por el pin 16

ESP8266: NodeMCU

¿Por que usar un Arduino externo para controlar el ESP, cuando este tiene un microprocesador mucho más potente que el propio Arduino?

NodeMCU es una plataforma abierta de IoT. Incluye:

- Un firmware que corre sobre el ESP8266 basado en Lua
- Diferentes placas de desarrollo

ESP8266

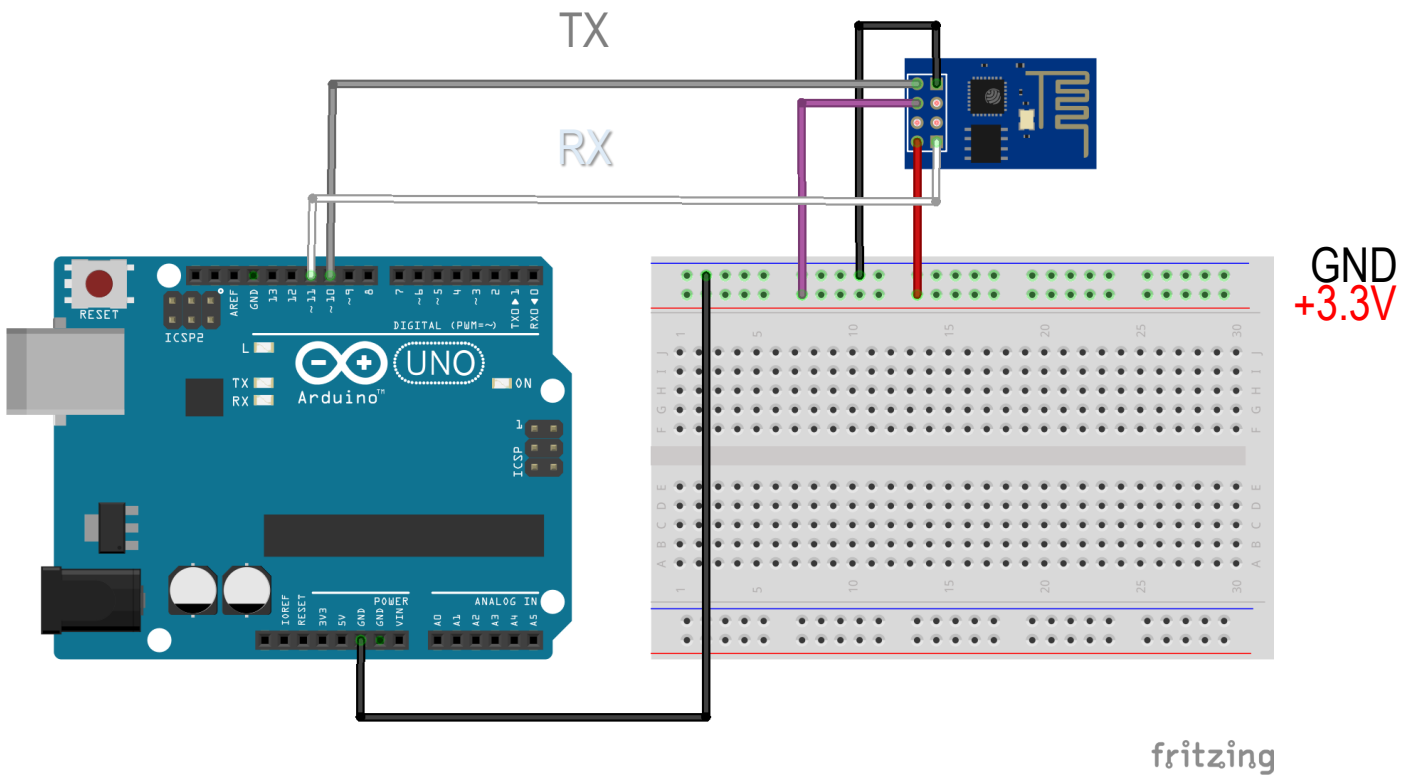
A modo de introducción al ESP8266 vamos a utilizar el modulo ESP-01.

- ✓ Es el modulo más sencillo y barato.
- ✓ Perfecto para introducirse en el mundo de ESP8266
- ✗ Está limitado el número de salidas, y por lo tanto sus aplicaciones.

En nuestro caso programaremos el Arduino, y este le mandará commandos al ESP8266.

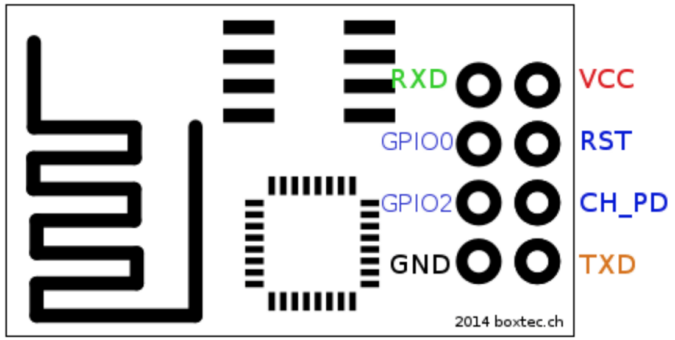
Conectar ESP8266

Arduino
TX: 11
RX: 10



Comandos AT

- Utiliza comandos de estilo Hayes (comandos AT)
  - AT+CWJAP_CUR
- Existen librerías para Arduino
  - … pero no las vamos a usar



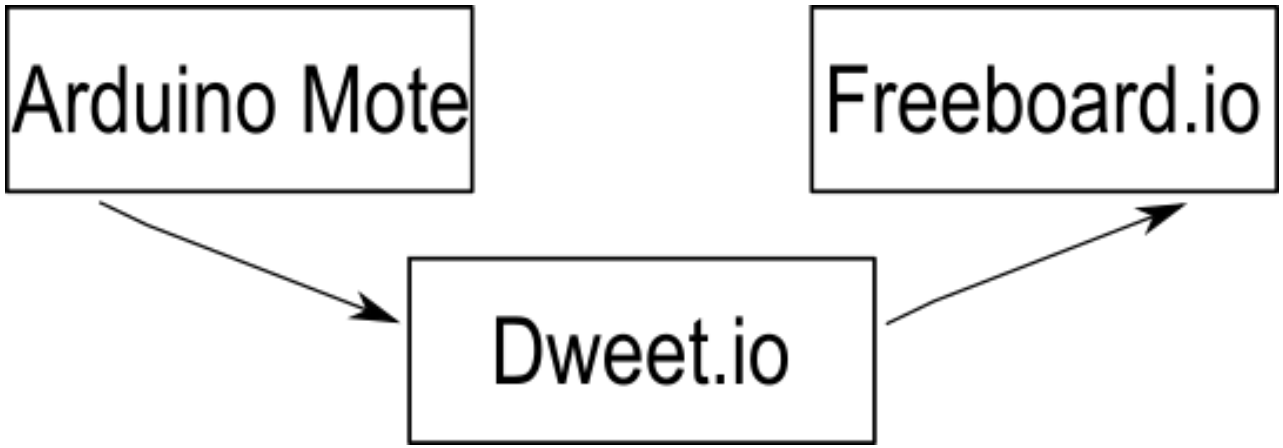
Comandos AT

- Hay alrededor de 100
- Los más básicos que usaremos son:

- AT
- AT+RST
- AT+CWMODE_CUR (y _DEF)
- AT+CWJAP_CUR (y _DEF)
- AT+CWSAP_CUR (y _DEF)
- AT+CIPSERVER
- AT+CIPSTART
- AT+CIPSEND
- AT+CIPCLOSE

Construye tu mota: DweetMote

- Arduino Mote
- Dweet
- Freeboard



Construye tu mota: Servidor

- FALTA COMPLETAR ESTA PRÁCTICA
- Mote
- Navegador