

TRABAJO PRACTICO FINAL

Shields v 1.0



Docentes:

- Jorge Morales

-C. Gonzales Vera

Alumno:

Ernesto Alejandro Canio

Grupo 5



WS2812

Inteligente controlado
integrado luz fuente

Características y ventajas

- El circuito de control y el chip RGB están integrados en un paquete de 5050 componentes, forman un control completo del punto de píxel.
- El circuito de remodelación de la señal incorporado, después de la remodelación de la onda al siguiente conductor, asegura que la distorsión de la forma de la onda no se acumule.
- Circuito de rearme eléctrico incorporado y circuito de rearme por pérdida de potencia.
- Cada píxel de los tres colores primarios puede lograr una visualización de 256 brillos, completando 16777216 colores a todo color, y una frecuencia de exploración no inferior a 400Hz/s.
- Señal de transmisión de puerto en cascada por línea única.
- Cualquier dos puntos la distancia más de 5m señal de transmisión sin ningún circuito de aumento.
- Cuando la tasa de refresco es de 30fps, el número de cascadas del modelo de baja velocidad no es inferior a 512 puntos, el modo de alta velocidad no es inferior a 1024 puntos.
- Envía datos a velocidades de 800Kbps.
- El color de la luz era muy consistente, rentable. **Aplicaciones** Módulo a todo color, Luces suaves a todo color una tira de lámparas. Iluminación decorativa LED, Pantalla irregular de vídeo LED para interior/exterior.

Descripción general

WS2812 es una fuente de luz LED de control inteligente que el circuito de control y el chip RGB están integrados en un paquete de 5050 componentes. Incluye un puerto digital inteligente para el bloqueo de datos y un circuito de amplificación de la señal. También incluye un oscilador interno de precisión y una corriente constante programable de 12V.

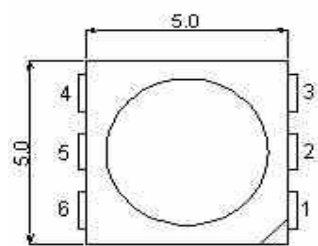
-nt parte de control, asegurando efectivamente la altura de color de la luz del punto de píxel consistente.

El protocolo de transferencia de datos utiliza el modo de comunicación NZR simple. Tras el reinicio del píxel, el puerto DIN recibe los datos del controlador, el primer píxel recoge los datos iniciales de 24 bits y los envía a la memoria de datos interna, los otros datos que se reforman mediante el circuito de amplificación de la señal interna se envían al siguiente píxel en cascada a través del puerto DO. Después de la transmisión para cada píxel, la señal para reducir 24bit. píxel adoptar auto resha

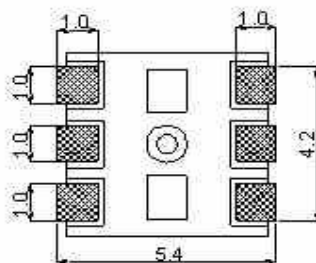
-Tecnología de transmisión, haciendo que el número de píxeles en cascada no esté limitado la transmisión de la señal, sólo depende de la velocidad de transmisión de la señal.

LED con bajo voltaje de conducción, la protección del medio ambiente y el ahorro de energía, alto brillo, el ángulo de dispersión es grande, buena consistencia, de baja potencia, larga vida y otras ventajas. El chip de control integrado en el LED por encima de convertirse en circuito más simple, pequeño volumen, la instalación conveniente.

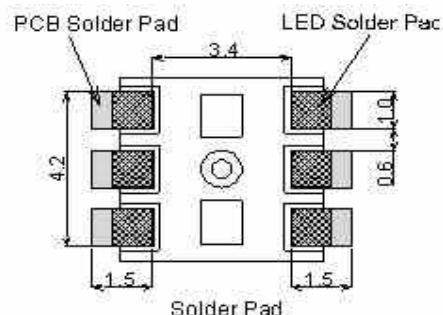
Dimensiones mecánicas



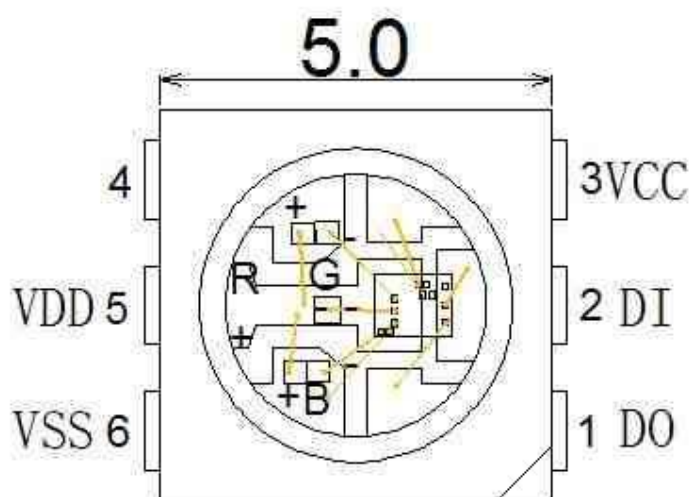
Top View



Back View



Configuración del PIN



Función PIN

NO.	Símbolo	Descripción de la función
1	DOUT	Salida de la señal de control de datos
2	DIN	Entrada de la señal de control
3	VCC	Circuito de control de la fuente de alimentación
4	NC	
5	VDD	LED de alimentación
6	VSS	Tierra

Valores máximos absolutos

Prameter	Símbolo	Clasificaciones	Unidad
Tensión de alimentación	VCC	+6.0~+7.0	V
Tensión de alimentación	VDD	+6.0~+7.0	V
Tensión de entrada	V _I	-0,5~VDD+0,5	V
Temperatura de unión de funcionamiento	T _{opt}	-25~+80	°C
Rango de temperatura de almacenamiento	T _{stg}	-55~+150	°C

Características eléctricas ($T_A = -20 \sim +70^\circ\text{C}$, $V_{DD} = 4,5 \sim 5,5\text{V}$, $V_{SS} = 0\text{V}$, a menos que se especifique lo contrario)

Prameter	Smybol	condiciones	Min	Tpy	Max	Unidad
Salida de baja tensión actual	IOL	ROUT	--	18.5	--	mA
	I _{dout}	V _O =0,4V, D _{OUT}	10	--	--	mA
Corriente de entrada	I _I	V _I =VDD/VSS	--	--	±1	μA
Nivel de tensión de entrada	V _{IH}	D _{IN} , SET	0.7V _{DD}	--	--	V
	V _{IL}	D _{IN} , SET	--	--	0.3 V _{DD}	V
Tensión de histéresis	V _H	D _{IN} , SET	--	0.35	--	V

Características de conmutación ($T_A = -20 \sim +70^\circ\text{C}$, $V_{DD} = 4,5 \sim 5,5\text{V}$, $V_{SS} = 0\text{V}$, a menos que se especifique lo contrario)

Prameter	Símbolo	Condición	Min	Tpy	Max	Unidad
Frecuencia de funcionamiento	Fosc2	--	--	800	--	KHz
Tiempo de retardo de la transmisión	tPLZ	CL=15pF, DIN→DOUT, RL=10KΩ	--	--	300	ns
Tiempo de otoño	tTHZ	CL=300pF, OUTR/OUTG/OUTB	--	--	120	μs
Velocidad de transmisión de datos	F _{MAX}	Relación de derechos 50%	400	--	--	Kbps
Capacidad de entrada	C _I	--	--	--	15	pF

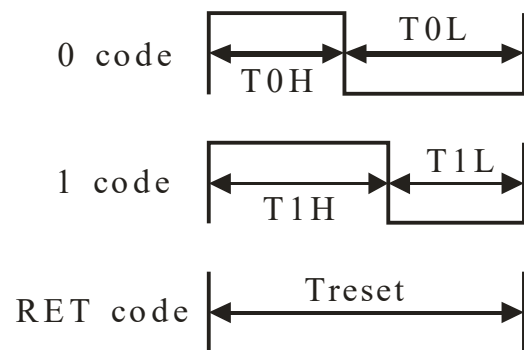
Parámetro característico del LED

Color de emisión	Longitud de onda(nm)	Intensidad luminosa (mcd)	Corriente (mA)	Tensión (V)
Rojo	620-630	550-700	20	1.8-2.2
Verde	515-530	1100-1400	20	3.0-3.2
Azul	465-475	200-400	20	3.2-3.4

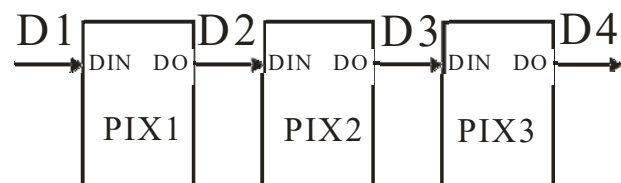
Tiempo de transferencia de datos(TH+TL=1,25μs±600ns)

T0H	0 código, tiempo de alta tensión	0,35us	±150ns
T1H	1 código, tiempo de alta tensión	0,7us	±150ns
T0L	0 código , tiempo de baja tensión	0,8us	±150ns
T1L	1 código, tiempo de baja tensión	0,6us	±150ns
RES	tiempo de baja tensión	Por encima de 50μs	

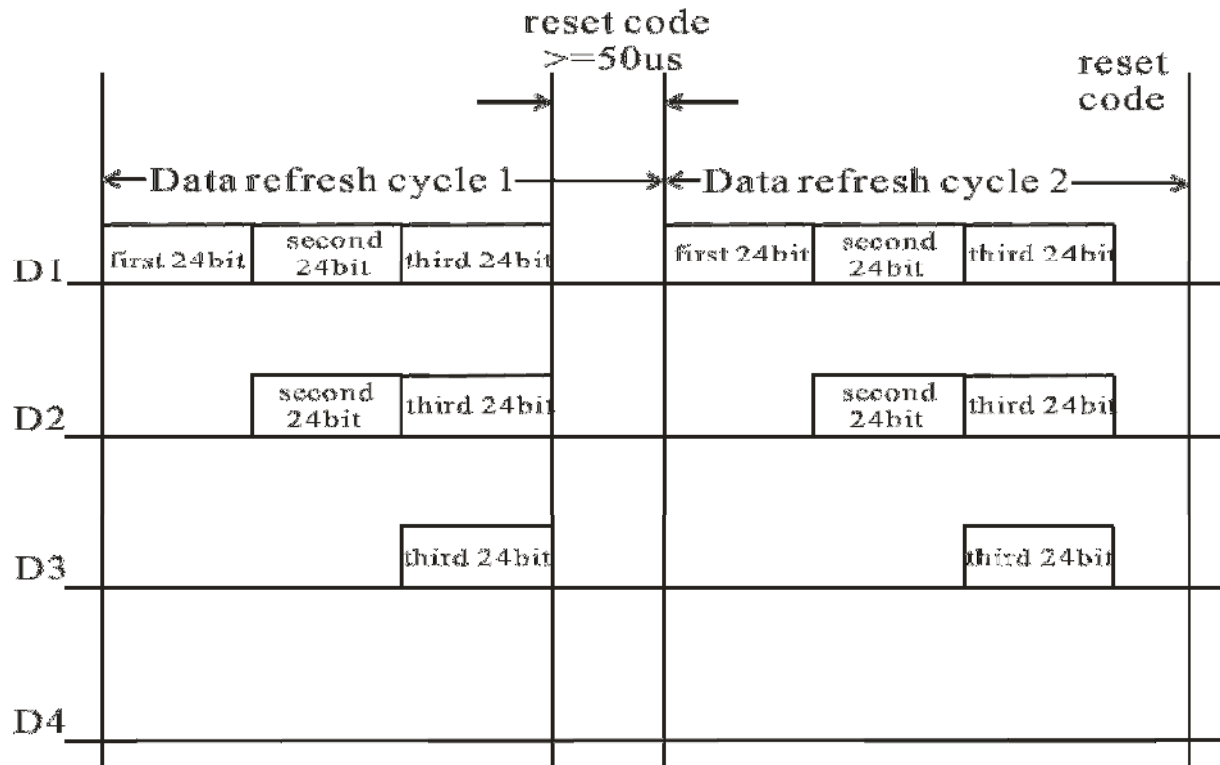
Cuadro de secuencias:



Cascada método:



Método de transmisión de datos :



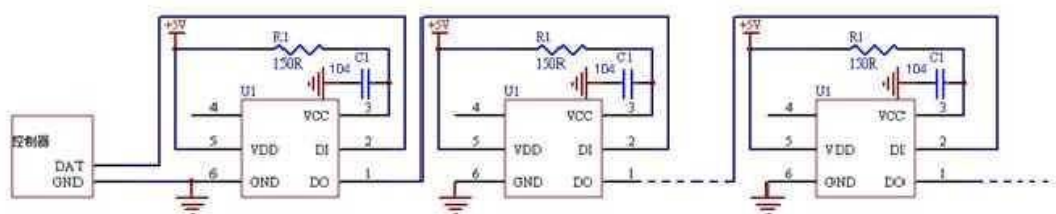
Nota: Los datos de D1 son enviados por la MCU, y D2, D3, D4 a través de la amplificación de remodelación interna del píxel para transmitir.

Composición de los datos de 24 bits:

G7	G6	G5	G4	G3	G2	G1	G0	R7	R6	R5	R4	R3	R2	R1	R0	B7	B6	B5	B4	B3	B2	B1	B0
----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

Nota: Siga el orden de GRB para enviar los datos y el bit alto enviado en primer lugar.

Circuito típico de aplicación:



Conectar y controlar LEDs WS2812B a ESP8266 con ESPEasy

Los LEDs WS2812 (también llamados NeoPixels) son fantásticos y dan posibilidades y flexibilidad increíble.

Se presentan tanto de forma individual como en tiras, pero mientras que una tira normal RGB se enciende toda a la vez de un color, en este caso podemos controlar cada LED independientemente.

Los podemos conectar en paralelo, uniendo tantos como queramos con solo tres cables y la gran ventaja es que los podemos controlar de uno en uno. Pueden encender unos y apagar otros, poner cada uno del color que queramos y regular su intensidad independientemente.

Estos LEDs son de los llamados RGB porque cada uno de ellos tiene los tres colores básicos (rojo, verde y azul). Lo que quiere decir que podemos conseguir muchísimos colores combinando esos tres básicos (incluso el blanco, si encendemos los tres juntos con la misma intensidad).

Además, se alimentan a 5 voltios, lo que facilita mucho los montajes, al no necesitar fuentes de alimentación adicionales. Podemos utilizar la misma alimentación de un NodeMCU o Wemos D1 Mini o poner un cargador de móvil independiente para alimentarlos (u otra fuente de alimentación mayor si hay muchos LEDs).

Solamente hay que tener en cuenta una cosa: cada color de cada led, encendido al 100% consume 20mA. Cada LED tiene tres colores. Esto significa que un LED puede consumir 60mA si encendemos sus tres colores a máxima potencia al mismo tiempo.

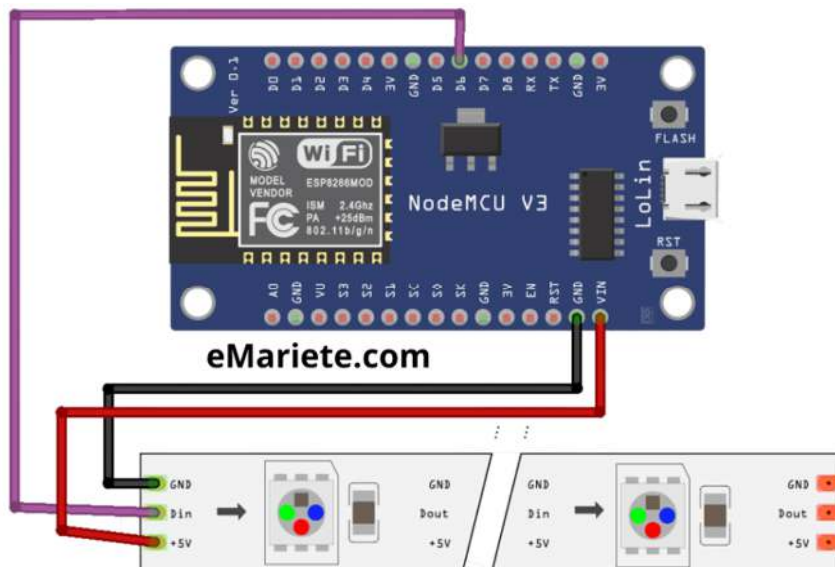
Si ponemos una tira de 10 LEDs WS2812 consumirá un máximo de 600mA. Hay que tenerlo en cuenta a la hora de pensar en el alimentador a usar.

Los alimentadores de celular suelen proporcionar entre 500 y 2000mA, con estos datos se puede calcular cuántos LEDs se podrían conectar a este tipo de fuentes para no exceder su capacidad.

Diagrama con las conexiones del NodeMCU a los LEDs WS2812 (Neopixel)

Como te decía, la conexión no puede ser más fácil. **¡Solamente tres cables y a funcionar!**

En el ejemplo he utilizado tres LED WS2812B pero, una de las grandes ventajas de este tipo de LED, es que puedes utilizar los que quieras. **Eso sí, ten en cuenta el consumo, cada LED puede consumir hasta 60 mA.**



Configuración ESPEasy

Una vez hechas las conexiones, hay que configurar el ESPEasy para que reconozca los dispositivos e incluir unas sencillas reglas para que se comporten de la forma que se desee.

Como ejemplo tenemos el medidor de CO₂ casero de eMariete, pero se puede utilizar exactamente igual con cualquier otro proyecto.

Básicamente hay que hacer lo siguiente:

Activar el motor de reglas de ESPEasy

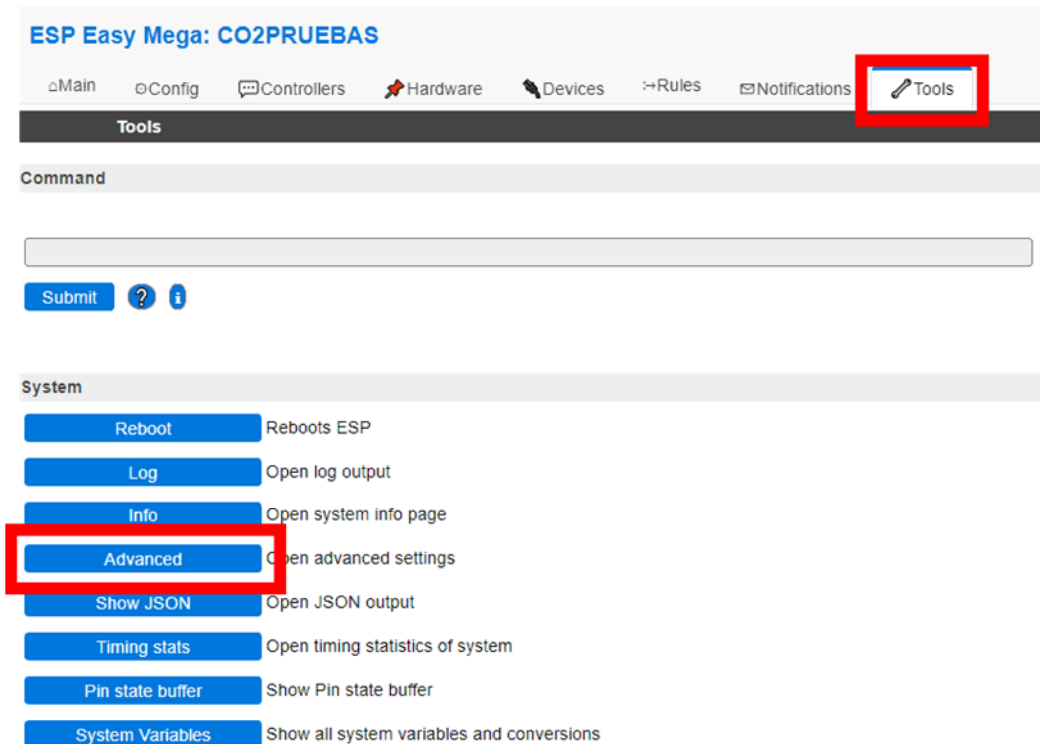
Añadir una regla para que cuando se encienda el ESP8266 lo haga con los LEDS apagados

Añadir una regla para que cambie el color de los LEDS dependiendo del nivel de CO₂

No es necesario que des de alta los LEDs como dispositivo en la pestaña «Devices».

Activar el motor de reglas de ESPEasy

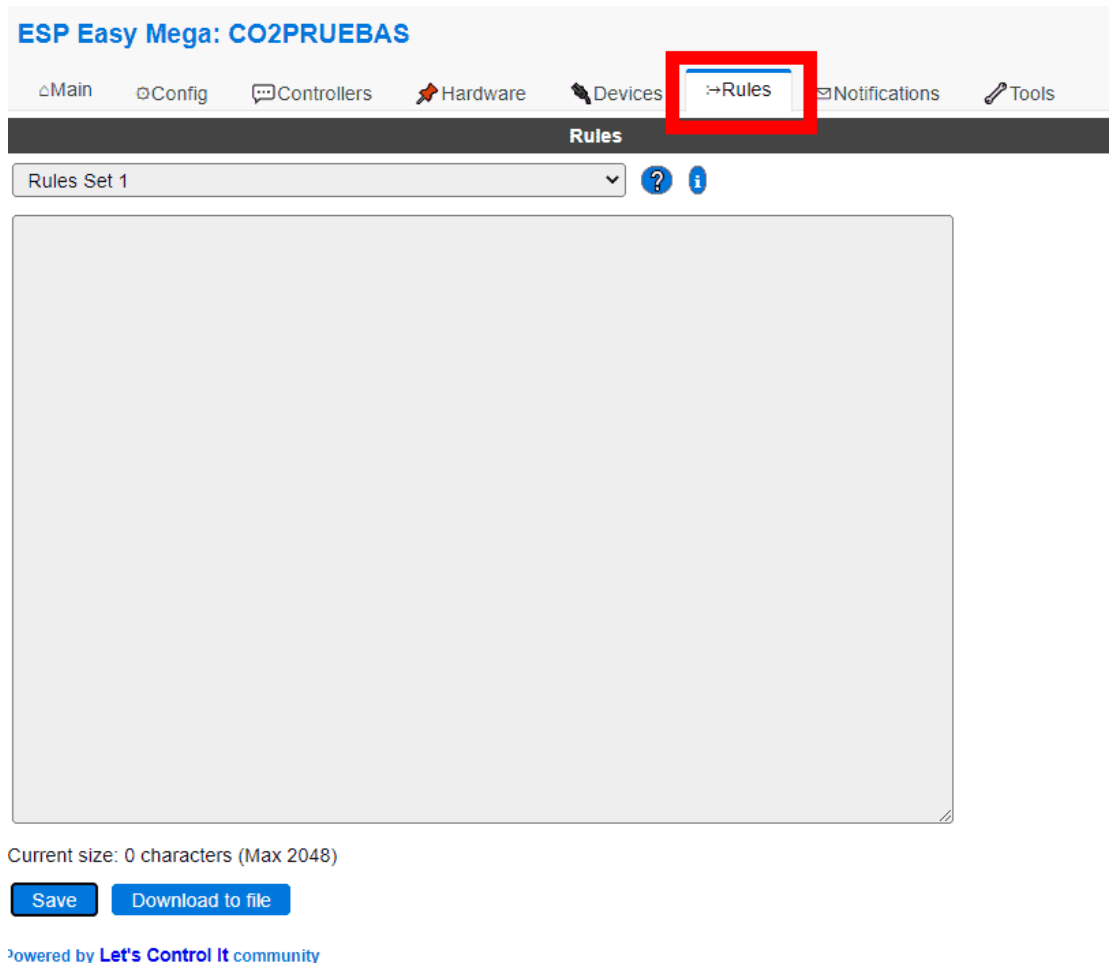
Lo primero que debes hacer, es activar el motor de reglas de ESPEasy, si no lo has hecho ya. Para ello, vete a la pestaña “Tools” (Herramientas) y pulsa el botón “Advanced” (Avanzado):



Se abrirá la ventana con las opciones de configuración avanzadas de ESPEasy. Allí debes activar las opciones “Rules” (Reglas) y “Old Engine” (Viejo Motor):

Añadir una regla para que cuando se encienda lo haga con los LEDS apagados

Ahora tienes que escribir las reglas, en la pestaña “Rules” (Reglas). Esta pestaña no aparecerá hasta que no habilites el motor de reglas, como has visto en el punto anterior:



En el editor de reglas que aparece en ESPEasy, escribes las siguientes reglas (te recomiendo copiar y pegar, para evitar errores y dejar los comentarios – líneas que empiezan por // – que ESPEasy ignorará).

Deberás sustituir Senseair#PPM por el nombre de dispositivo y valor que hayas elegido (en la pestaña “Devices” – Dispositivos). En esta captura de pantalla el nombre del dispositivo es CO2, por lo que en vez de Senseair#PPM escribiríamos CO2#PPM:

ESP Easy Mega: CO2PRUEBAS								
<div> Main Config Controllers Hardware Devices Rules Notifications Tools </div>								
	Task	Enabled	Device	Name	Port	Ctr (IDX)	GPIO	Values
Edit	1	✓	Gases - CO2 Senseair	SenseairCO2	HW Serial0	1	RX: GPIO-3 (D9) TX: GPIO-1 (D10)	PPM: 2387 TEMP: 32
Edit	2	✓	Display - OLED SSD1306/SH1106 Framed	Display	I2C		SDA: GPIO-4 (D2) SCL: GPIO-5 (D1)	
Edit	3	✓	Output - NeoPixel (Basic)	LEDS			GPIO-12 (D6)	
Add	4							
Add	5							
Add	6							
Add	7							
Add	8							
Add	9							
Add	10							
Add	11							
Add	12							

La línea `NeoPixelAll,0,0,0` ordena a ESPEasy que apague todos los LEDS.

Realmente lo que le dice esta línea es que ponga el Rojo a 0, el Verde a 0 y el Azul a 0.

Al estar la línea `NeoPixelAll,0,0,0` entre «un bloque», formado por la línea `On System#Boot do` y `EndOn`, le indicamos a ESPEasy que lo ejecute al arrancar el NodeMCU (cualquier cosa que metas dentro de ese bloque se ejecutará cada vez que el NodeMCU se encienda).

Añadir una regla para que cambie el color de los LEDS dependiendo del nivel de CO2

Ahora añade este otro bloque a las reglas:

```

On SenseairCO2#PPM do
if [SenseairCO2#PPM]>2000
NeoPixelAll,32,0,0 // LEDS ROJO
Elseif [SenseairCO2#PPM] > 1500
NeoPixelAll,16,9,0 // LEDS NARANJA
Else
NeoPixelAll,0,32,0 // LEDS VERDE
Endif
Endon

```

Este bloque ordena a ESPEasy que cada vez que reciba un nuevo valor desde el sensor SenseairCO2 (`On SenseairCO2#PPM do`), compruebe lo siguiente:

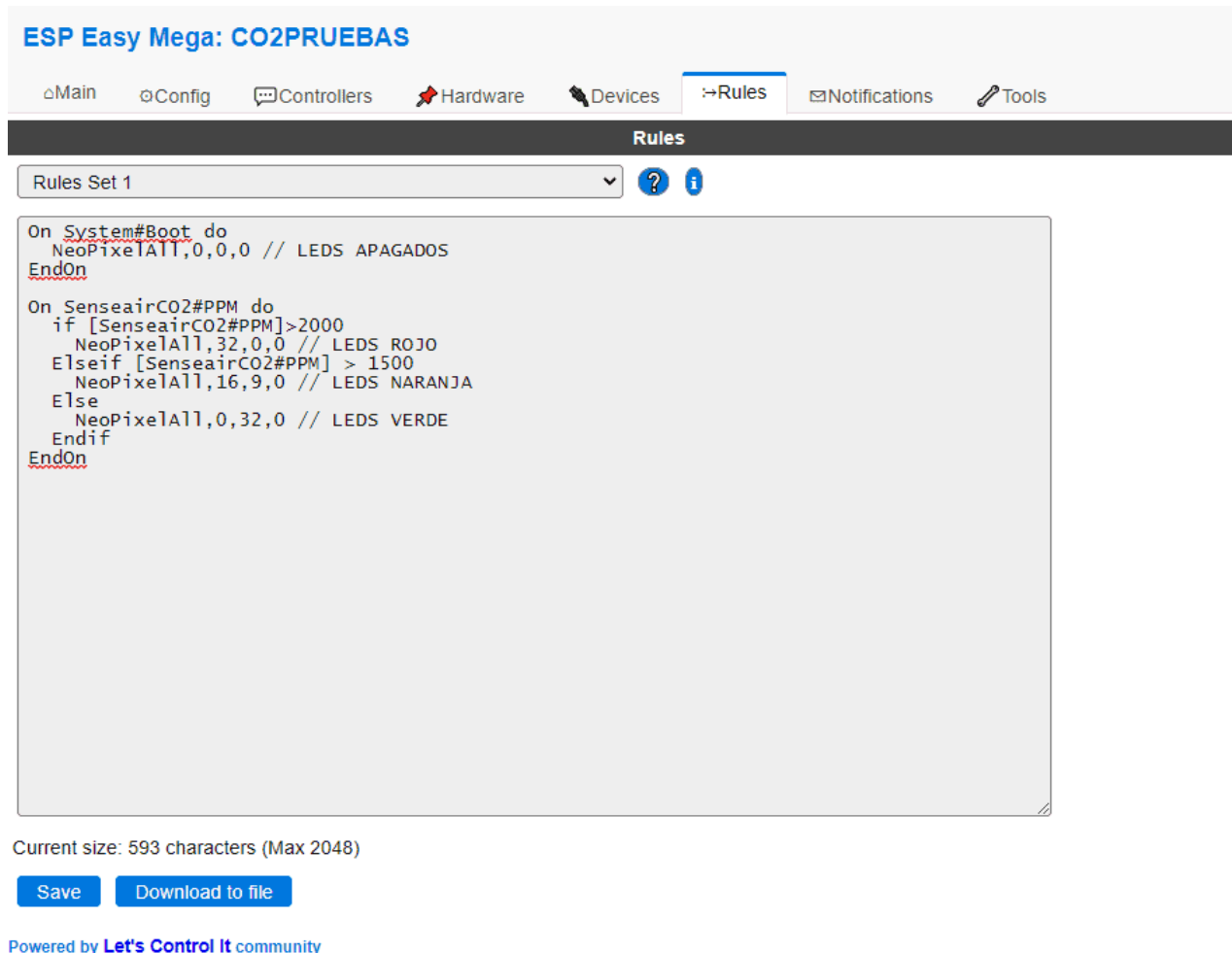
Si el valor de `SenseairCO2#PPM` es superior a 2000 y, de ser así, encienda todos los LEDS de la tira con los siguientes valores: Rojo a 32, el Verde a 0 y el Azul a 0.

si no se cumple lo anterior (que el valor de `SenseairCO2#PPM` sea superior a 2000), comprobará si el valor es superior a 1500 y, si se cumple, encienda todos los LEDS de la tira con los siguientes valores: Rojo a 16, el Verde a 9 y el Azul a 0 (un amarillo/naranja).

Si no se cumple lo anterior (que el valor de SenseairCO2#PPM sea superior a 1500), y sin hacer más comprobaciones, encenderá todos los LEDs de la tira con los siguientes valores: Rojo a 0, el Verde a 32 y el Azul a 0.

la línea *endon* termina el bloque de comprobaciones que empezó con *On SenseairCO2#PPM do*.

El resultado final, debería ser similar a este:



The screenshot shows the 'Rules' tab in the ESP Easy Mega: CO2PRUEBAS interface. The rule set is named 'Rules Set 1'. The code defines a rule that triggers on system boot and another that triggers on a change in the SenseairCO2#PPM sensor value. The rule logic is as follows:

```
On System#Boot do
  NeoPixelAll,0,0,0 // LEDS APAGADOS
EndOn

On SenseairCO2#PPM do
  if [SenseairCO2#PPM]>2000
    NeoPixelAll,32,0,0 // LEDS ROJO
  ElseIf [SenseairCO2#PPM] > 1500
    NeoPixelAll,16,9,0 // LEDS NARANJA
  Else
    NeoPixelAll,0,32,0 // LEDS VERDE
  Endif
EndOn
```

Current size: 593 characters (Max 2048)

Buttons: Save, Download to file

Powered by Let's Control It community

Controlar el encendido y apagado de los LEDs con un interruptor (switch) y qué sean intermitentes y que se apaguen a ciertas horas.

Las posibilidades de las reglas de ESP-Easy son enormes, y **nos permiten hacer casi cualquier cosa...** ¿Y si quisieras que el encendido de los LEDs fuera intermitente y que además estuvieran apagados entre determinadas horas, para que no molesten, por ejemplo, y además controlar con un interruptor (o un pulsador) que se enciendan o no (bonus: y que además si tenemos una pantalla OLED conectada también se apague)?

Podríamos utilizar esta variación de las reglas del punto anterior en las que incluiríamos unos temporizadores para crear la intermitencia y algunas comprobaciones extras de las horas y de un Switch que conectaríamos (entre el pin D7 y GND, en este caso):

```

On Rules#Timer=1 do
  if [Switch#State]=1 and [var#1]=0 // Si el switch está encendido y no es hora de dormir
  if [co2#co2]>1000
    NeoPixelAll,32,0,0 // LEDS ROJO
  Elseif [co2#co2]>750
    NeoPixelAll,16,9,0 // LEDS NARANJA
  Elseif [co2#co2]>400
    NeoPixelAll,0,32,0 // LEDS VERDE
  Else
    NeoPixelAll,0,0,0 // LEDS APAGADOS
  Endif
Endif
  looptimerset_ms,2,100 //encendido 100ms cada 2 seg
Endon

```

```

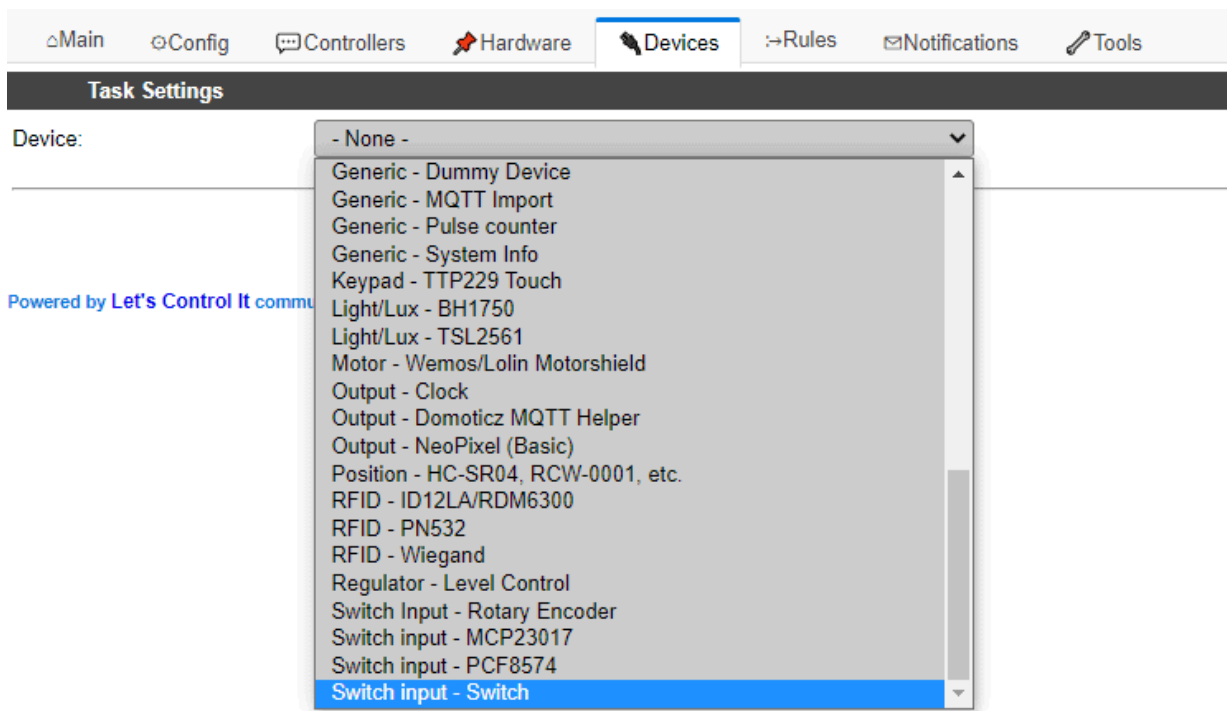
On rules#Timer=2 do
  NeoPixelAll,0,0,0 //led off
  looptimerset_ms,2,0
  timerset,1,2
endon
On System#Boot do
  NeoPixelAll,0,0,0 // LEDS APAGADOS
  Let,1,0 // Si está a 1 = hora de dormir, leds apagados!
  timerSet,1,2 //cambiar el 2 por 60seg y se encenderán los leds con la primera medida
EndOn
on Switch#State do
  if [Switch#State]=0
    oledframedcmd,display,off
    NeoPixelAll,0,0,0 // APAGAR LEDS
  else
    oledframedcmd,display,on
    Let,1,0 // Si está a 0 = no es hora de dormir, leds encendidos!
  endif
endon
on Clock#Time=All,22:00 do
  Let,1,1 // Si está a 1 = hora de dormir, leds apagados!
  NeoPixelAll,0,0,0
  oledframedcmd,display,off
endon
on Clock#Time=All,08:00 do
  Let,1,0 // Si está a 0 = no es hora de dormir, leds encendidos!
  oledframedcmd,display,on
endon

```

En este ejemplo, los LEDs y la pantalla OLED permanecerían apagados cada día desde las 22:00 hasta las 08:00.

Puedes «cancelar» ese *tiempo de dormir* apagando y volviendo a encender el apagado desde el interruptor.

Solamente te quedaría conectar el switch (en el pin D7 en este ejemplo, aunque podrías usar otros), añadirlo como un nuevo dispositivo de tipo «Switch»:



y configurarlo:

ESP Easy Mega: AZUL

△Main ⚙️Config 💬Controllers 🛠️Hardware 🖱️Devices ➡️Rules 📧Notifications 🔧Tools

Task Settings

Device: Switch input - Switch ? ⓘ

Name: switch

Enabled: ☒

Sensor

Internal PullUp: ☒

Inversed Logic: ☒

Note: Will go into effect on next input change.

GPIO ⇄: GPIO-13 (D7)

Switch Type: Switch

Switch Button Type: Normal Switch

Send Boot state: ☐

Advanced event management

De-bounce (ms): 0

Doubleclick event: Disabled

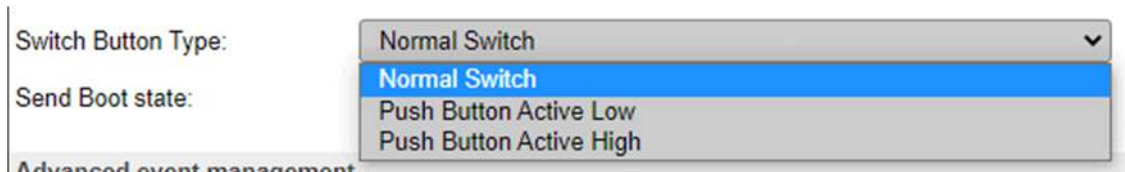
Doubleclick max. interval (ms): 1000

Longpress event: Disabled

Longpress min. interval (ms): 500

Use Safe Button (slower): ☐

Si lo prefieres puedes utilizar un pulsador en vez de un interruptor. Solamente tendrías que cambiar el tipo de botón y decir que es un pulsador activo a nivel bajo (que se activa cuando se une el pin D7 a GND):



Añadir una regla para hacer un semáforo de LEDs

Con los LEDs WS2812 es posible crear, por ejemplo, un efecto de semáforo, en el que, con una tira de tres LED, colocados de manera vertical, podremos encender el primer LED de la tira en verde, el segundo en naranja y el tercero en rojo, dependiendo del nivel de CO2 o de cualquier otro valor.

El comando a enviar para encender un led determinado de la tira es:

NeoPixel, NumeroLed, CantidadDeRojo, CantidadDeVerde, CantidadDeAzul

Siendo *NumeroLed* la posición que ocupa el LED que queremos controlar en la tira, y la cantidad de rojo, verde y azul un número entre 0 (apagado) y 255 (encendido al 100% de intensidad).

Para encender el primer LED en rojo (al 50% de intensidad, por ejemplo, ya que el 100% es muy fuerte para mi gusto) utilizaríamos el comando:

NeoPixel,1,128,0,0

Para encender el segundo LED en ámbar (al 50% de intensidad), lo podríamos hacer sumando un 25% de rojo más un 12% de verde. Para conseguir esto utilizaríamos el comando:

NeoPixel,2,64,32,0

Para encender el tercer LED en verde (al 50% de intensidad), utilizaríamos el comando:

NeoPixel,3,0,64,0

La regla que deberíamos incluir sería esta:

```
On SenseairCO2#PPM do  
if [SenseairCO2#PPM]>900  
NeoPixelAll,0,0,0 // APAGAMOS LOS LED  
NeoPixel,1,128,0,0 // ENCENDEMOS EL PRIMER LED EN ROJO  
Elseif [SenseairCO2#PPM] > 800  
NeoPixelAll,0,0,0 // APAGAMOS LOS LED  
NeoPixel,2,64,32,0 // ENCENDEMOS EL SEGUNDO LED EN NARANJA  
Else  
NeoPixelAll,0,0,0 // APAGAMOS LOS LED  
NeoPixel,3,0,64,0 // ENCENDEMOS EL TERCER LED EN VERDE  
Endif  
Endon
```

Se ha escogido los niveles de CO2 para que el semáforo se ponga rojo cuando se superen las 900 ppm (según la normativa RITE IDA 2 para aulas de enseñanza, adecuado para aulas de colegios). Siéntete libre de sustituir los valores de 900 y 700 por los que más te convengan.

Dependiendo de a qué distancia se vaya a ver y de la luminosidad de la sala, puedes ajustar la intensidad de los LEDs (uno de estos LEDS encendido al máximo, valor de 255, se ve mucho).

Como verás antes de encender cada led primero apagamos la tira completa, utilizando el comando *NeoPixelAll,0,0,0* porque, si no lo hiciéramos así, en los cambios de color (por ejemplo, al pasar de verde a naranja), se quedarían encendidos los dos LEDS.

Este tipo de LEDS es perfecto para hacer semáforos y otros indicadores. Mejor que las tiras porque se pueden utilizar unidos o separar de forma individual y dan mucha más flexibilidad.