MANUAL KIT
ARDUINO

Coordinador del Proyecto:

- Johnny Valencia
- Felipe Aguilar

Apoyo Logístico y Laboratorio:

- Gabriel Retamales Muñoz
- Anahi Ovando
- Christopher Belanger
- Claudia Hernández
- Braulio Catrileo

Universidad de Aysén

MANUAL KIT
ARDUINO

Ministerio de Educación
Gobierno de Chile

Universidad de Aysén
PROYECTOS DE VINCULACIÓN ESCOLAR CIENTÍFICA (VEC)
DESCUBRIENDO LA CIENCIA EN EL MUNDO ESCOLAR

Índice

Tabla de contenido

Resumen

El kit de arduino es un set de partes y piezas que permiten la creación de diferentes tipos de proyectos, como por ejemplo la medición de diferentes variables ambientales controladas por un sistema capaz de ser monitoreado en algún notebook o algún otro equipo compatible con Arduino. Este Kit Arduino trae consigo una variedad de piezas y partes, además de sensores de bajo costo que permiten llevar a cabo las mediciones correspondientes, como por ejemplo sensores de gases tales como el metano , monóxido de carbono y calidad del aire. Como muchos gases no son perceptibles a los sentidos humanos esta variedad de sensores pueden ayudar a precisar los niveles presentes en el aire. Además incluye un sensor de medición de temperatura y humedad relativa de muy buena calidad con respecto a la precisión sus mediciones.

Por otra parte, el objetivo principal de este manual entregará las herramientas básicas para que cada estudiante pueda realizar por sí mismo una configuración y conexión de sensores ambientales capaces de medir temperatura, humedad, calidad de aire, monóxido de carbono y GLP, etc. Esto con el fin de que cada estudiante pueda adquirir conocimiento básico de nuevas tecnologías y además adentrarse un poco más en la problemática actual de contaminación que aqueja al planeta y más específicamente a la ciudad de coyhaique.

Finalmente este manual viene estructurado de tal forma que sea sumamente sencilla su comprensión, por lo cual solo es necesario seguir paso a paso cada instrucción para así poder obtener un correcta ejecución y funcionamiento del proyecto final. Además este manual presenta la descripción de cada una de las componentes necesarias para el ensamblaje, también incluye esquemas gráficos que indican cómo llevar a cabo las conexiones correspondientes. Finalmente, este manual también incluye los códigos necesarios para cada ejercicio, dichos códigos resultan imprescindibles para el funcionamiento de cualquier tipo proyecto en Arduino.

1. Introducción

La contaminación del aire se ha convertido en los últimos años en un problema de carácter mundial, siete millones de personas mueren cada año a causa de la contaminación atmosférica existente dentro y fuera de los hogares. Este tipo de contaminación se compone de partículas sólidas y líquidas suspendidas en el aire que se combinan formando partículas de 10 micrómetros o menos (MP10) y partículas finas de hasta 2,5 micrómetros (MP2,5), siendo estas últimas las más nocivas para la salud, debido a que pueden inhalarse y penetrar en vías respiratorias, así como también en el torrente sanguíneo. Lo cual puede desencadenar diferentes tipos de enfermedades.

Las principales causas de contaminación atmosférica son el uso ineficiente de energía en los hogares y sectores industriales, agricultura, transporte y centrales eléctricas alimentadas con carbón. La Organización Mundial para la Salud (OMS) ha establecido en su guía de calidad del aire los niveles máximos de material particulado recomendados para la salud de la población, estos establecen 10 $\mu g/m^3$ de media anual y 25 $\mu g/m^3$ como media de 24 horas para MP2,5 y de 20 $\mu g/m^3$ como media anual y de 50 $\mu g/m^3$ para media de 24 horas para MP10.

Los niveles máximos de material particulado aceptados en Chile los establece la Norma Primaria de Calidad del Aire, para MP2,5 se fija en 20 $\mu g/m^3$ para concentración anual y 50 $\mu g/m^3$ para concentración de 24 horas, por otro lado, para MP10 fija un límite de 50 $\mu g/m^3$ para la concentración anual y de 150 $\mu g/m^3$ para una concentración de 24 horas. En Chile se consideran cinco umbrales para establecer la calidad de aire, estos corresponden a una concentración Buena; Regular; Alerta; Preemergencia y Emergencia. Para saber el estado de la calidad del aire se utiliza el promedio móvil de las últimas 24 horas de concentraciones de MP2,5, con el fin de pronosticar la situación futura respecto de la actual. La calidad del aire es monitoreada por el Sistema Nacional de Calidad del Aire (SINCA[1]), perteneciente al Ministerio del Medio Ambiente (MMA), este sistema se encarga de monitorear e informar al respecto

La contaminación atmosférica en Chile corresponde en su mayoría a fuentes móviles como automóviles y a fuentes fijas como industrias y sistemas de calefacción a leña. Esta última, es la principal causa de la contaminación existente en las ciudades del sur del país. El mayor consumo de este recurso, según la Corporación de Desarrollo Tecnológico en un estudio de 2015 realizado para el MMA, se concentra en la Región de Aysén tanto en sectores públicos como privados. Coyhaique, capital de la Región de Aysén, es actualmente la ciudad con mayor índice de contaminación atmosférica de América según la base de datos publicada por la OMS durante 2018.

El gran crecimiento demográfico de nuestra ciudad ha aumentado la superficie de todas las zonas urbanas presentes en la comuna, generando un aumento en las instalaciones utilizadas para calefacción y cocina por cada vivienda nueva construida, las cuales utilizan leña como combustible. El uso indiscriminado de leña para suplir las necesidades de calefacción en los sectores residenciales ha convertido a Coyhaique en una ciudad que constantemente se encuentra bajo episodios críticos

de contaminación atmosférica durante los últimos años. Si bien, Coyhaique cuenta desde el 2016 con un plan de descontaminación atmosférica (PDA) para MP2,5, que abarca una superficie de 111 km^2 alrededor de la zona urbana, donde se establecen metas y procedimientos para los días declarados críticos, este recién se encuentra en fase de implementación, al mismo tiempo que la ciudad ha proyectado un crecimiento urbano para los próximos años que no ha sido considerado dentro de este plan.

Actualmente, existen dos estaciones certificadas capaces de medir material particulado en la ciudad de Coyhaique, por lo cual la implementación de este proyecto y el producto final que se busca obtener en este manual serán de mucha ayuda para poder recopilar más información al respecto.

TALLER 1
“Prepara tu Arduino”

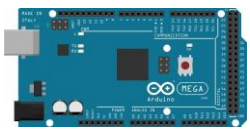
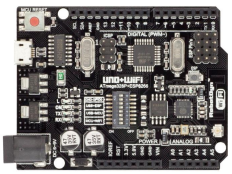
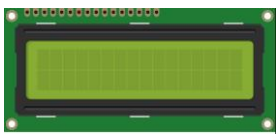
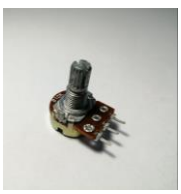
2. Componentes del kit Arduino

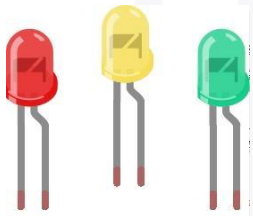
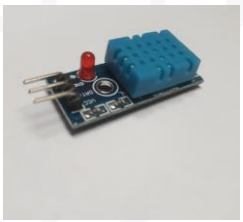
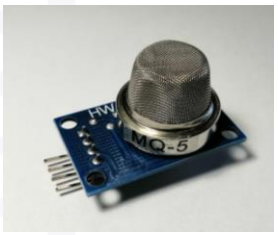
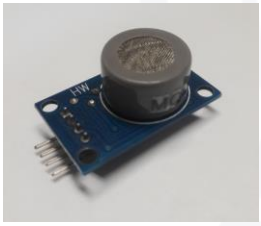
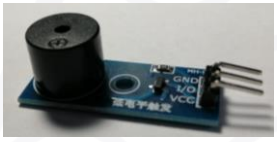
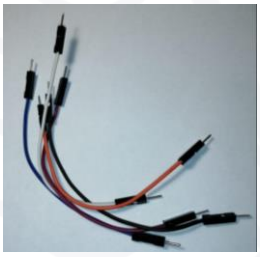
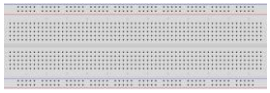
2.1. Material Físico

Las componentes incluidos en el kit Arduino para la elaboración del producto final son las siguientes:

a) 1 Arduino MEGA 2560
b) 1 Protoboard
c) 1 Pantalla LCD
d) 1 Potenciómetro
e) Led´s verde, amarillo y rojo
f) 1 Sensor digital de temperatura y humedad DHT11
g) 1 Sensor de calidad del aire MQ135
h) 1 Sensor de gas licuado de petróleo (GLP) MQ5
i) 1 Sensor de monóxido de carbono MQ7
j) 1 Alarma Buzzer
k) 33 cables dupont macho macho
l) 1 Cable usb arduino-pc
m) 5 resistencias de 1k ohm
n) 5 resistencias de 10k ohm.
o) 5 resistencias de 220k ohm.

- **Las componentes se muestran de forma ilustrativa en la siguiente figura:**

 Arduino MEGA 2560		 Pantalla LCD	 Potenciómetro

	WeMos UNO+ESP8266		
 Led´s verde, amarillo y rojo	 Sensor digital de temperatura y humedad DHT11	 Sensor de calidad del aire MQ135	 Sensor de Gas Licuado de Petróleo (GLP) MQ5
 Sensor de monóxido de carbono MQ7	 Alarma Buzzer	 Cables dupont macho macho unidades	 Cable usb arduino-pc
 Caja “kit”	 Mini protoboard	 Protoboard	 Sensor Grove

			HM3101

Figura 1: Componentes kit Arduino.

2.2. Material Digital.

Además de material físico, el kit de Arduino cuenta con material digital complementario para poder realizar los ejercicios de este manual. Estos son:

- Manual Kit Arduino
- Video Tutorial para resolver problemas de Puerto USB.
- Capsulas resumen de cada taller.
- Blink_Example.ino
- Buzzer.ino
- Encender_LED.ino
- DHT11.ino
- Prueba_Pantalla_LCD.ino
- Clase_4.ino
- Clase_5.ino
- Clase_6.ino
- ESP8266_GroveHM3101.ino
- UNO_GroveHM3101.ino

2.3. Resistencias

2.3.1. Origen y descripción

La resistencia corresponde a una componente de los circuitos, la cual se encarga de limitar la cantidad de corriente que puede pasar a través de este mismo, esto con el fin de proteger sensores u otros aparatos los cuales funcionan con menor intensidad a la que atraviesa por el circuito. Las resistencias entonces sirven para disminuir el voltaje de algunos circuitos y así reducir las posibilidades que quemar sensores u algún otro aparato por un exceso de voltaje.

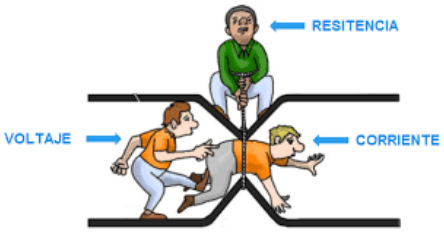


Figura 2: Concepto gráfico de voltaje, resistencia y corriente.

El origen del concepto resistencia se remonta hacia el año 1827, gracias al físico alemán Georg Simon Ohm, quien descubrió el principio que ahora lleva su nombre, la **Ley de Ohm**, la cual se representa con la letra griega **omega (Ω)** en el sistema de unidades internacional.

2.3.2. Barras de colores y el concepto de resistencia (ohm).

Entendemos la **Ley de Ohm** como el flujo de **corriente** en amperios (A) que circula por circuito eléctrico cerrado, es directamente proporcional al voltaje (V) aplicado, e inversamente proporcional a la **resistencia** en **ohm**.

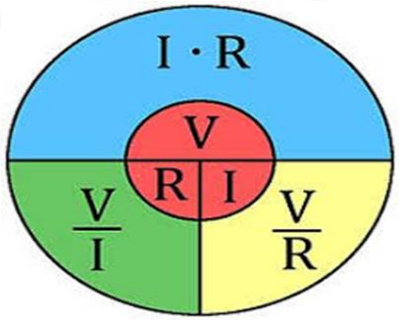


Figura 3: Imágen explicativa Ley de Ohm.

Entonces se define como **ohmio** como la resistencia eléctrica que existe entre dos puntos de un conductor, cuando al aplicar 1 voltio **(V)**, y luego de pasar por la resistencia se produce una corriente de intensidad de 1 amperio **(A)**.

Pues bien, entendiendo estos conceptos las resistencias buscan disminuir los riesgos de dañar los sensores por exceso de voltaje, para lo siguiente se ha diseñado un método que nos permite saber qué tipo de resistencias debemos utilizar según los diferentes tipos de sensores u otros aparatos existentes, este sistema se conoce como la **Tabla de Colores de Resistencias**.

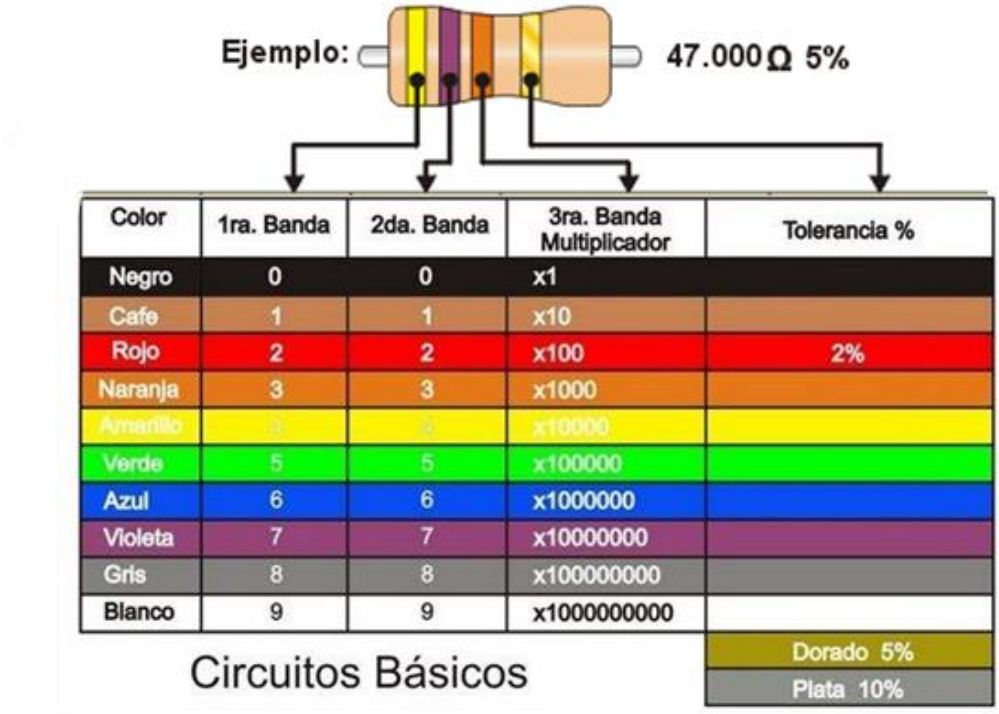


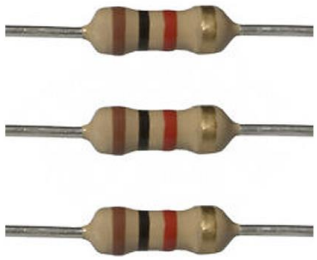
Figura 4: Tabla de colores para las resistencias.

La primera banda nos indica el primer dígito del valor de resistencia. La segunda banda, nos da el segundo dígito. Los dos dígitos de las primeras dos bandas nos dan un número que puede variar entre 0 y 99 (En este caso 4 y 7, unidos forman 47).

La tercera banda es el multiplicador, un factor con el cual debemos multiplicar el número de las dos primeras bandas. Por ejemplo, si el valor de las primeras bandas es 47 y el multiplicador es 1000 (1k) el valor de resistencia será de 47.000 ohms (47k).

2.3.3. Resistencias Kit Arduino

Las resistencias que incluye este Kit se muestran en la figura 5.



Resistencias 1k ohm

Resistencia 10k ohm

Resistencia de 220k ohm

Figura 5: Tipos de resistencias incluidas en el kit.

3. Comencemos de cero

3.1. Te invitamos a conocer Arduino

- Arduino nos permite desarrollar diferentes tipos de proyectos a partir del ensamblaje de piezas y sensores y de la ejecución de los diferentes tipos de códigos que se pueden crear.

- Las actividades prácticas en este manual fueron diseñadas con el fin de iniciar al interesado desde los conceptos más básicos hasta los de mayor dificultad cuyo resultado es la instalación de un kit de medición de gases.

- Antes de comenzar, se debe tener en cuenta algunos aspectos relevantes de Arduino para el correcto funcionamiento, sigue los pasos.

- **Paso 1:** Enciende tu computador o notebook, Antes de realizar cualquier conexión asegúrate de comprobar que tu placa Arduino funciona correctamente. Para ello solo basta con conectar el Arduino con el cable USB a cualquier computador o notebook y ver la luz del Arduino parpadear.

- **Paso 2:** El siguiente paso es instalar el software de Arduino, para ello es necesario entrar a la página **https://www.arduino.cc/en/main/software**, a continuación es necesario seleccionar el sistema operativo que corresponda al PC con el que se trabajara.

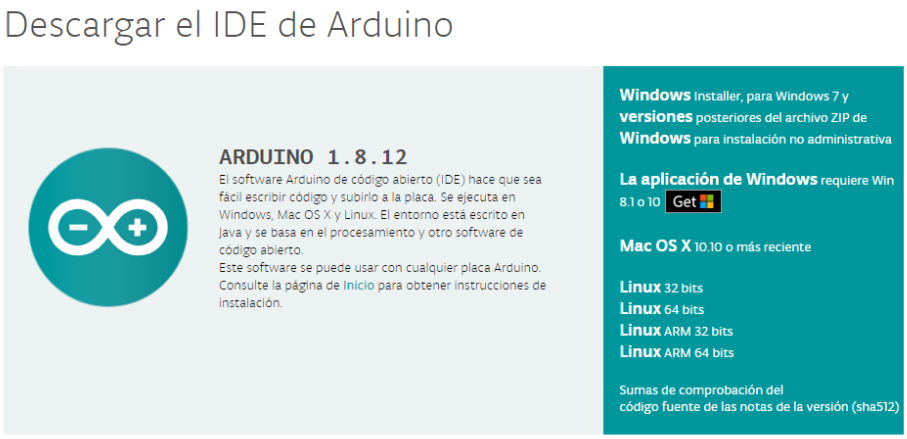


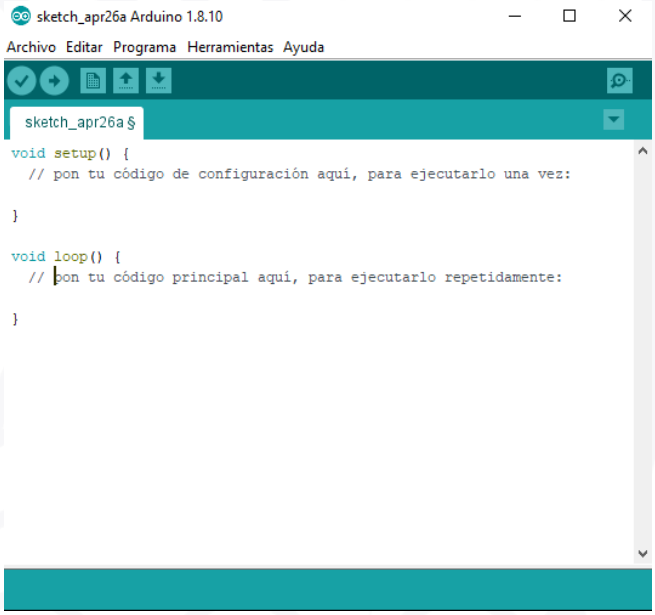
Figura 6: Pantalla de selección de sistema operativo.

Una vez seleccionado lo anterior, comenzara automáticamente la descarga del software, al completarse esta, en el escritorio del PC, aparecerá un nueva carpeta de nombre Arduino, seguido por la versión que se ha descargado (la última versión existente). Al abrir la carpeta se podrán ver otras carpetas y algunos iconos, sin embargo solo es importante el icono llamado Arduino.

Figura 7: Icono Arduino.

Una vez identificado este icono, es posible extraerlo de la carpeta y moverlo hacia el escritorio del computador, a modo de crear un acceso directo al software. Una vez hecho esto, se da doble click para comenzar a trabajar con el software.

- **Paso 3:** Una vez hechos los pasos anteriormente descritos**,** asegúrate de ver la siguiente pantalla en tu computador o notebook.



-

Figura 8: Pantalla inicio software.

Ahora es necesario establecer la comunicación del Arduino con el software, para ello es necesario hacer click en la parte que dice “Herramientas”, una vez hecho esto se desplegará la siguiente ventana.

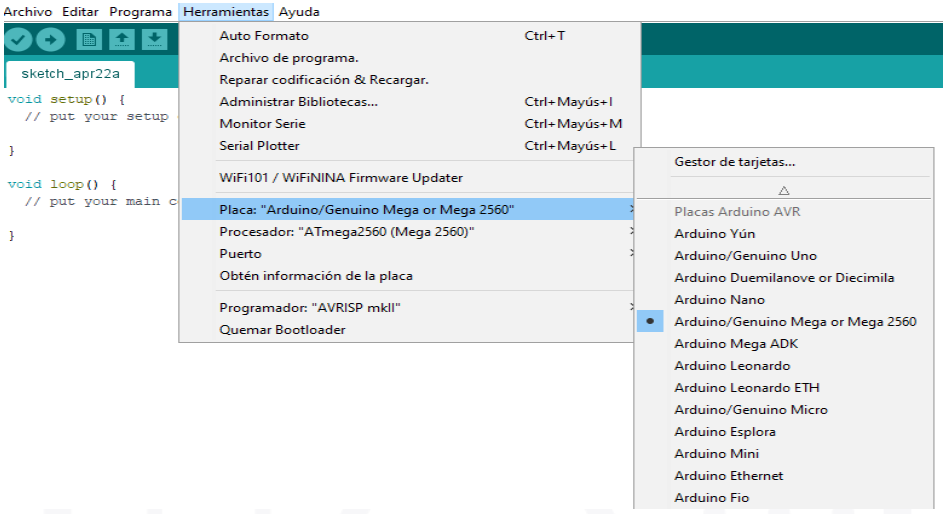


Figura 9: Opción para seleccionar el Arduino a utilizar.

Ahora se debe posicionar el cursor donde dice “Placa”, al hacer esto se desplegará una nueva ventana donde se encuentran todos los Arduino que el software puede reconocer, se debe buscar el Arduino del cual disponemos, en este caso el **Arduino/Genuino Mega or Mega 2560.**

Una vez hecho esto se debe establecer la conexión con el Puerto COM, para ello se debe posicionar el cursor sobre la parte que dice “Puerto”, esto desplegara otra ventana donde se observa el puerto COM disponible.

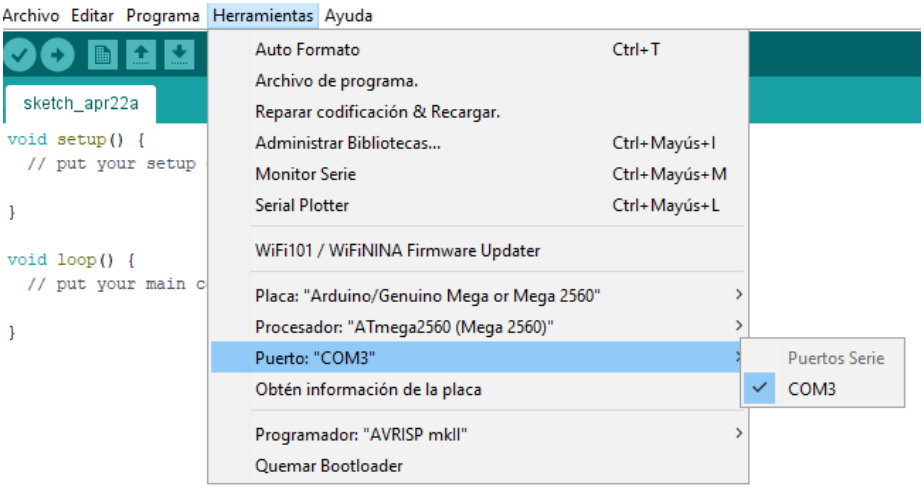


Figura 10: Opción para seleccionar puerto COM.

A continuación, se debe seleccionar el único COM disponible (el número que acompaña al puerto puede variar al momento de desplegar la ventana). Una vez hecho esto el Arduino se encuentra completamente comunicado con el software.

- **Paso 4:** Para el paso final es necesario comprender que la pantalla del software se divide en 2 partes, **void setup** y **void loop**.

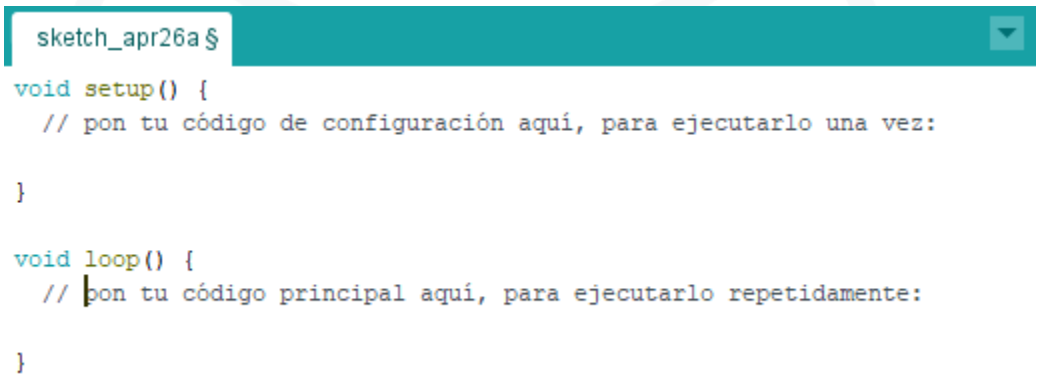


Figura 11: Opciones de ejecución de códigos.

Pues bien, dentro del **void setup**, se nombran y declaran partes constantes de nuestro proyecto, por ejemplo se declaran los LED's y a que pin están conectados.

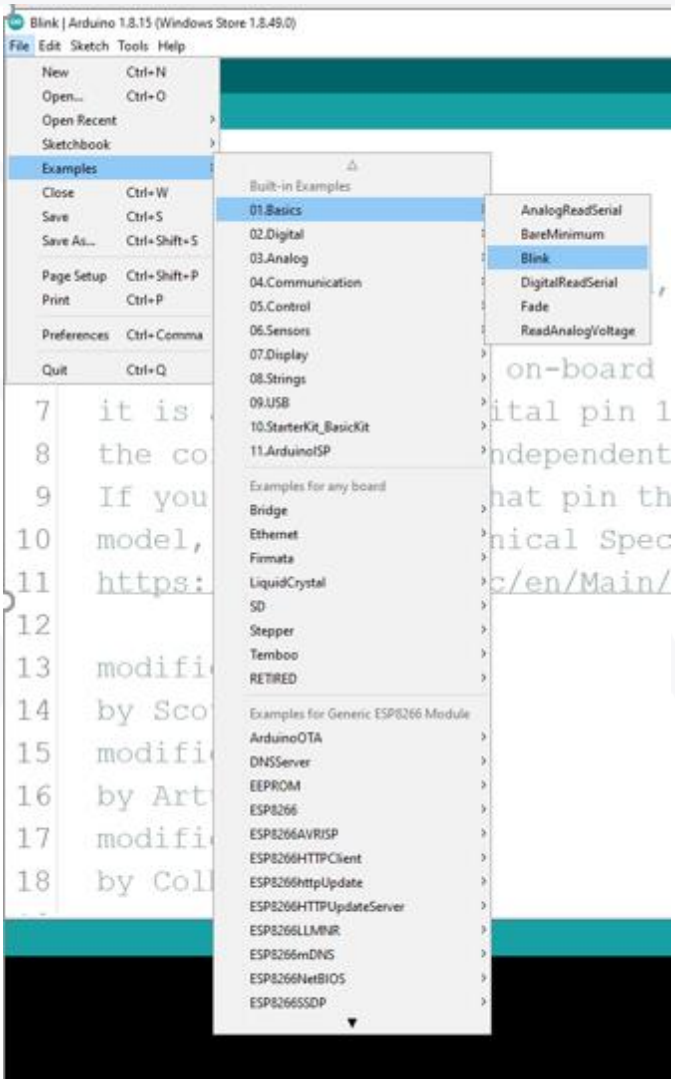
Mientras que en la parte de **void loop** se indican las funciones que el software debe realizar. Los comandos que se ejecutarán mientras la placa Arduino esté conectada. Esto comienza con el primer comando, el microcontrolador irá hasta el final y saltará inmediatamente al principio para repetir la misma secuencia. Y así un número infinito de veces

- **Ejemplo:**

Rutina en Arduino	Comentario de Rutina
void setup(){ **Serial.begin(9600);** **pinMode(13, OUTPUT);** **}**	• Serial.begin corresponde a la velocidad en que serán transmitidos los datos. • En este caso, se declara que algún componente del Arduino, los cuales serán utilizados en la ejecución del código (en este caso un LED) está conectado al pin número 13.
void loop(){ **digitalWrite(13, HIGH);** **delay(1000);** **digitalWrite(13, LOW);** **delay(1000);** **}**	• En este punto se indica que el LED se encenderá por por un segundo (delay), y luego se apagará por un segundo, repitiendo esto mismo una y otra vez. • La unidad del delay corresponde a milisegundos, por lo tanto 1000 corresponde a 1 segundo.

3.2. Test de conexión

En el siguiente ejercicio, haremos una simple prueba para verificar que nuestro Arduino pueda comunicarse correctamente con el ordenador. Para esto, utilizaremos los ejemplos predeterminados que trae el Arduino en su pestaña de ejemplos, específicamente, seleccionaremos el ejemplo "Blink". El ejemplo se encuentra en ***Archivo > Ejemplos > 0.1 Basics > Blink***.



Una ves seleccionado, tendremos un código prescrito en nuestra interfaz, el cual subiremos a la placa controladora y verificaremos que esta, encienda y apague la luz led integrada, simulando un pestañeo (*blink*) de luz. Si la prueba fue exitosa, podremos realizar todos los ejercicios que tendremos por delante.

TALLER 2
“Sensores Arduino”

3.2. Enciende la luz del conocimiento.

3.2.1. ¿Qué es una luz LED?

Una luz LED es una luz que tiene una conexión eléctrica donde existen dos filamentos uno largo Ánodo (polo positivo) y otro corto el cátodo (polo negativo) (Figura 12).

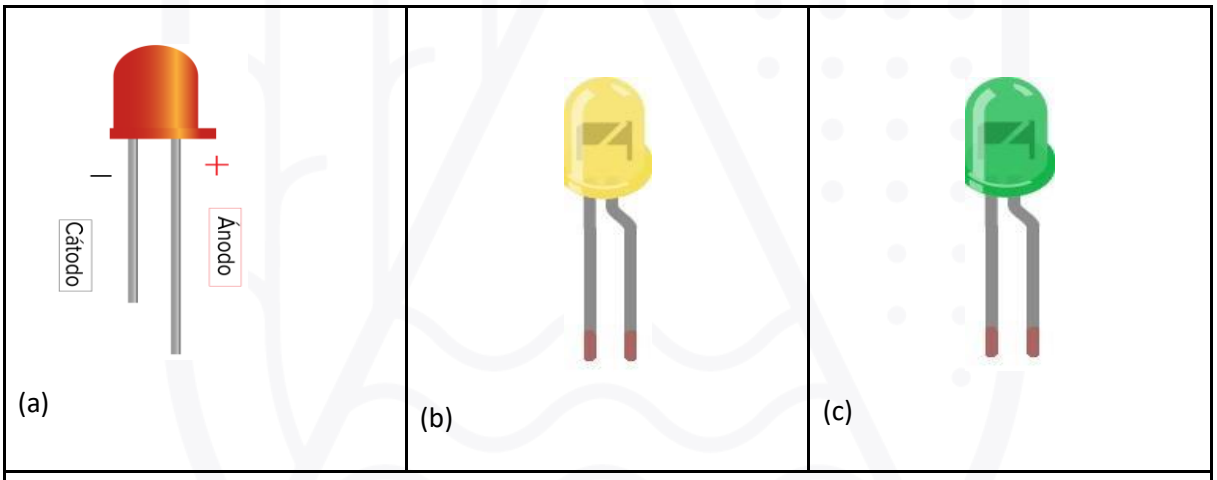


Figura 12. Diferentes LED's (a) se visualiza el ánodo (parte más larga) y cátodo (parte más corta), (b) LED color amarillo y (c) LED color verde.

3.2.2. ¿Qué materiales necesito?

Los materiales que se necesitan son:

- ***Materiales***

  1 Placa Arduino Mega 2560

  1 LED de cada color (Azul, rojo, verde)

3.2.3. ¿Cómo empezar?

Paso 1: Conecta el ánodo del LED rojo en el PIN 13 de la placa arduino y el cátodo en el PIN que señala la sigla (GND).

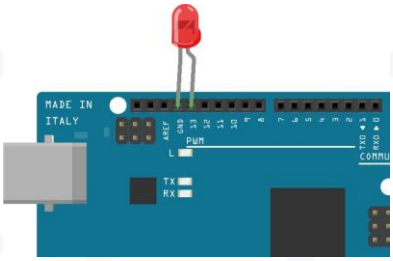


Figura 13. LED color rojo conectado a la placa arduino MEGA.

Paso 2*:* Introduce el código adjunto en la siguiente figura (fig. 14).

Figura 14: Código para un solo LED.

Paso 3: Cambia el paso de luces para que puedas visualizar diferentes parpadeos, para esto cambia el número de delay, que se encuentra entre paréntesis.

TAREA: ¿Qué pasaría con los LED de cada color? ¿Qué visualizas?

3.2.4. ¿Cómo puedo conectar más LED a mi arduino?

- ***Materiales***

  1 Placa Arduino Mega 2560

  1 LED de cada color (amarillo, verde)

  1 Protoboard

  1 Cable Rojo

  1 Cable Negro

  3 Cables de distintos colores

Para la conexión de más luces en nuestro arduino, realizaremos los siguientes pasos:

Paso 1: Conecta el GND de la placa Arduino hacia la barra negativa de la protoboard (azul) (Ver Fig16(a))

Paso 2: A continuación conecta un LED verde, la conexión del ánodo irá conectado al PIN 4 de la placa Arduino. (Ver Fig16(b)) .

Paso 3: Inserte la resistencia desde el cátodo del LED hacia la barra azul de la protoboard(Ver Fig16(c)).

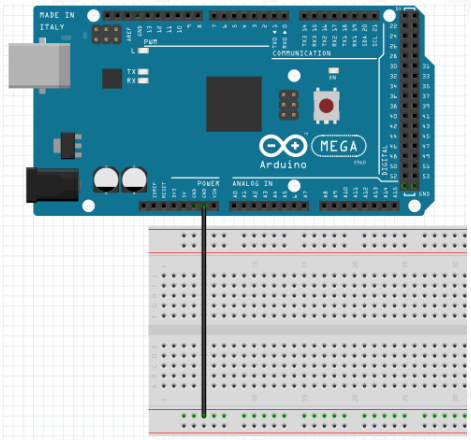
Paso 4: De la misma manera del LED verde conectaremos el LED amarillo, éste irá conectado al pin 8(Ver Fig16(d)).

Paso 5: Carga el siguiente código adjunto en la siguiente figura.(figura 15)

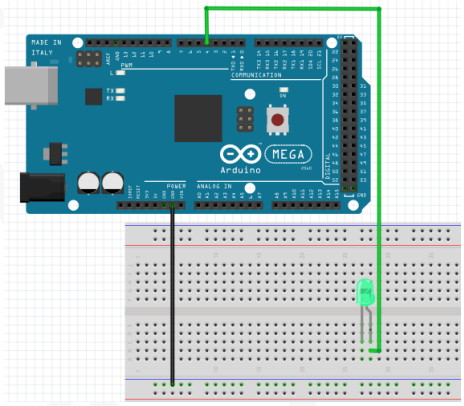

```
void setup() {
  pinMode(4, OUTPUT); // pin 4 como salida verde
  pinMode(8, OUTPUT); // pin 8 como salida amarilla
}

void loop() {
 digitalWrite(4,HIGH);    //encender luz verde
 delay(2000);             // esperar 2 segundos
 digitalWrite(4, LOW);  // apagar luz verde
 delay(2000);            // esperar 2 segundos

 digitalWrite(8,HIGH);    //encender luz verde
 delay(2000);             // esperar 2 segundos
 digitalWrite(8, LOW);  // apagar luz verde
 delay(2000);            // esperar 2 segundos
}
```

Figura 15: Código para 2 LED’s

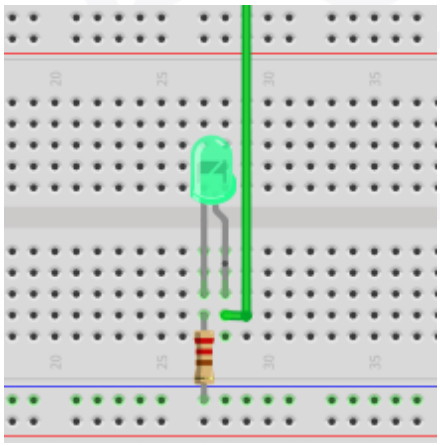
a)



b)



c)



d)

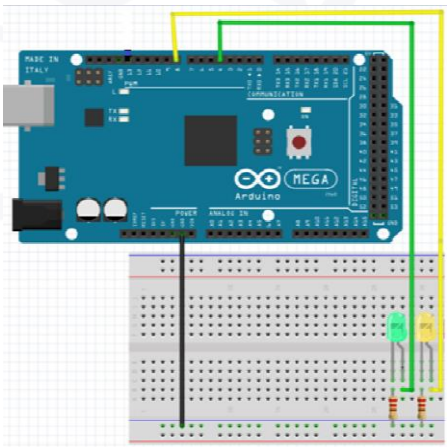


Figura 16: (a)Conexión GND a Protoboard, (b)Conexión LED verde a pin 4, (c)incorporación de resistencia a LED verde y (d) Conexión de los 2 LED´s con sus resistencias.

3.2.5. ¿Cómo configurar un semáforo?

Un semáforo es un dispositivo que cuenta generalmente de 3 luces, verde, amarilla y roja, este normalmente se usa para dar indicaciones de tráfico en la vía pública las cuales podemos replicar con Arduino, A continuación, se presentan los materiales que utilizaremos y la conexión para encender 3 luces LED´s en forma de semáforo.

- Materiales

1 Placa Arduino Mega 2560

1 LED de cada color (Amarillo, rojo, verde)

1 Protoboard

1 Cable Negro

3 Cables de distintos colores

3 resistencias 220Ω.

Para realizar la configuración de un semáforo, es necesario seguir los siguientes pasos:

Paso 1: Conecta el GND de la placa Arduino hacia la barra negativa de la protoboard (azul) (Ver Fig18(a))

Paso 2: A continuación, conecta un LED verde, la conexión del ánodo irá conectado al PIN 4 de la placa Arduino. (Ver Fig18(b)).

Paso 3: Inserte la resistencia desde el cátodo del LED hacia la barra azul de la protoboard (Ver Fig18(c)).

Paso 4: De la misma manera del LED verde conectaremos los LED´s amarillo y rojo, al Pin 8 y 13 respectivamente (Ver Fig18(d)).

Paso 5: Carga el código que se encuentra en la siguiente figura (Fig17).

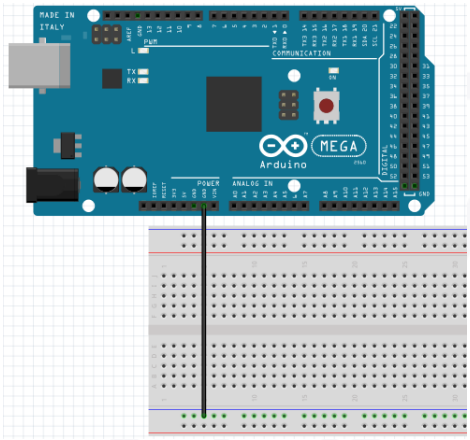
Archivo Editar Programa Herramientas Ayuda

sketch_apr26a §

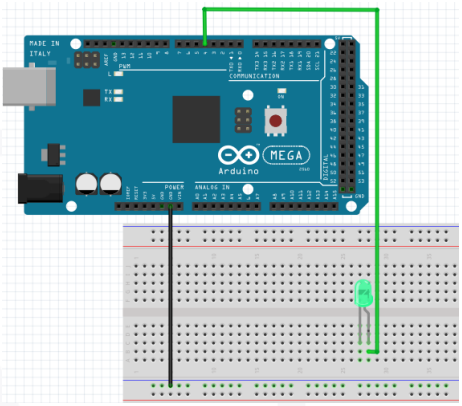
```
void setup() {
  pinMode(13,OUTPUT); // pin 13 como salida roja
  pinMode(8,OUTPUT);  // pin 8 como salida amarilla
  pinMode(4,OUTPUT);  // pin 4 como salida verde
}

void loop() {
digitalWrite(13,HIGH);    // encender luz roja
delay(2000);              // esperar 1 segundo
digitalWrite(13,LOW);      // apagar luz roja
delay(2000);               // eseprar 1 segundo

digitalWrite(8,HIGH);    // encender luz amarilla
delay(2000);              // esperar 1 segundo
digitalWrite(8,LOW);      // apagar luz amarilla
delay(2000);               // eseprar 1 segundo

digitalWrite(4,HIGH);    // encender luz verde
delay(2000);              // esperar 1 segundo
digitalWrite(4,LOW);      // apagar luz verde
delay(2000);               // eseprar 1 segundo
}
```

Figura 17: Código para tres Led`s juntos.

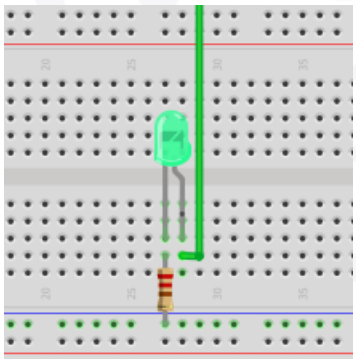
(a)



(b)



(c)



(d)

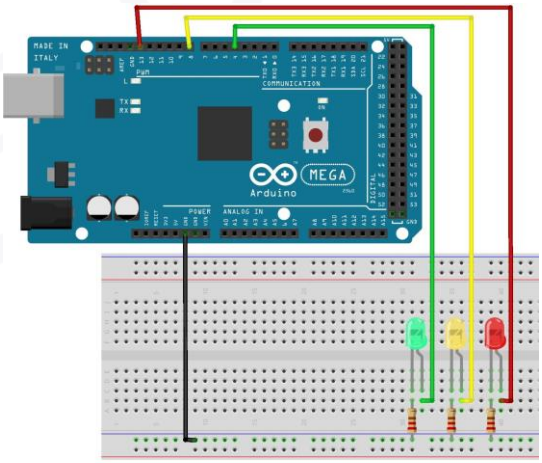


Figura 18: (a) Conexión GND a Protoboard, (b)Conexión LED verde a pin 4, (c)incorporación de resistencia a led verde, (d) Conexión de los 3 LED's con sus resistencias.

4. Sensores

Otra de las componentes importantes que forman parte del Kit Arduino son los **sensores,** estos son dispositivos que captan magnitudes físicas (variaciones de luz, temperatura, sonido, etc.) u otras alteraciones de su entorno, y son capaces de transformar dichas magnitudes física en magnitudes digitales, lo cual permite poder visualizar dicha información.

4.1 Hacer sonar Buzzer (Alarma).

¿Qué es un Buzzer?

Los Buzzer o zumbador son dispositivos de altavoz, que transforman la energía eléctrica a ondas de sonido. Existen dos tipos:

- **Buzzer Pasivos:** Son ausentes de electrónica interna, por lo cual se debe añadir una señal eléctrica para que se obtenga un sonido deseado.

- **Buzzer Activos:** Posee un oscilador interno, por ende, cada vez que obtenga una señal eléctrica, este emitirá sonido.

En el kit se encuentra un Buzzer activo, por lo cual posee una frecuencia fija de sonido. Estos instrumentos son ideales para ocuparlos como **alarmas,** ya que el sonido emitido por estos resulta similar a una sirena de bomberos, lo que en un hipotético caso podría indicar que es necesaria la evacuación de un lugar específico debido a un exceso de temperatura o una alta concentración de gases lo que podría resultar letal.

Para que la alarma buzzer funcione de manera correcta se necesita lo siguiente:

Materiales

- 1 Placa arduino mega 2560
- 1 Protoboard

- 1 Buzzer
- 1 Cable largo de cualquier color
- 2 Cable corto de cada color (rojo, Negro)

A continuación, seguiremos los siguientes pasos para la instalación del Buzzer:

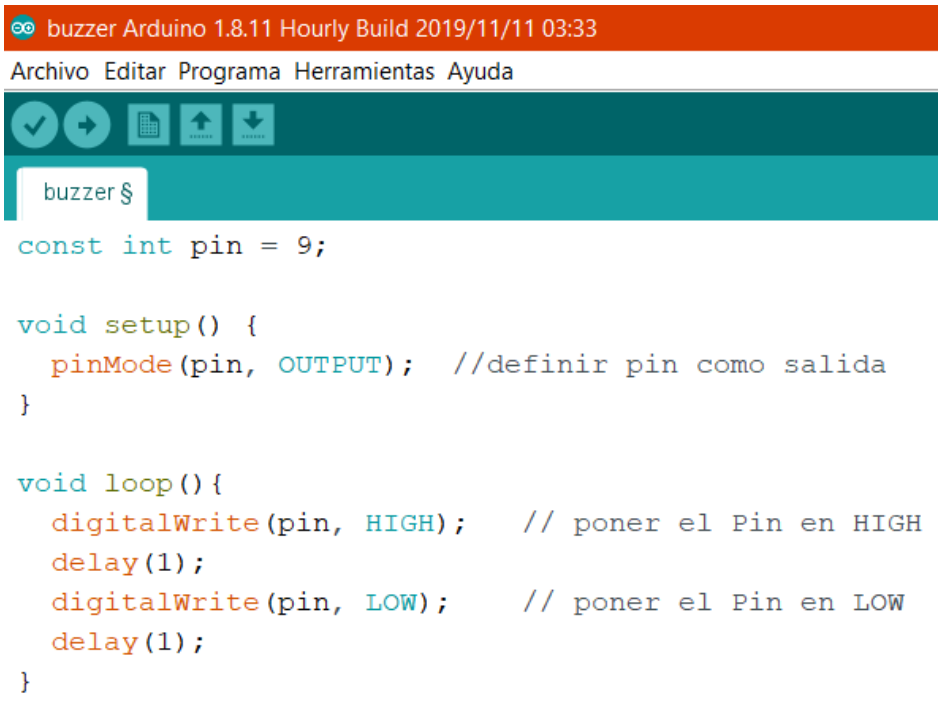
Paso 1: Realizar la conexión de carga negativa desde la barra azul de la protoboard hacia el buzzer (Ver Fig. 26(a))

Paso 2: Realizar la conexión de carga positiva desde la barra roja de la protoboard hacia el buzzer (Ver Fig. 26(b))

Paso 3: Realizar la conexión desde el pin N°9 de la placa arduino hacia la protoboard (Ver Fig. 26(c))

Paso 4: Insertar la alarma buzzer en la protoboard, cuidando que la conexión vcc indicada en el buzzer debe ir con la conexión al positivo, la conexión I/O al pin N°9 y el GND a la carga negativa (Ver Fig. 26(d)).

Paso 5: Carga el siguiente código:



Tarea: Cambia el delay (número entre paréntesis). ¿Qué sucede?

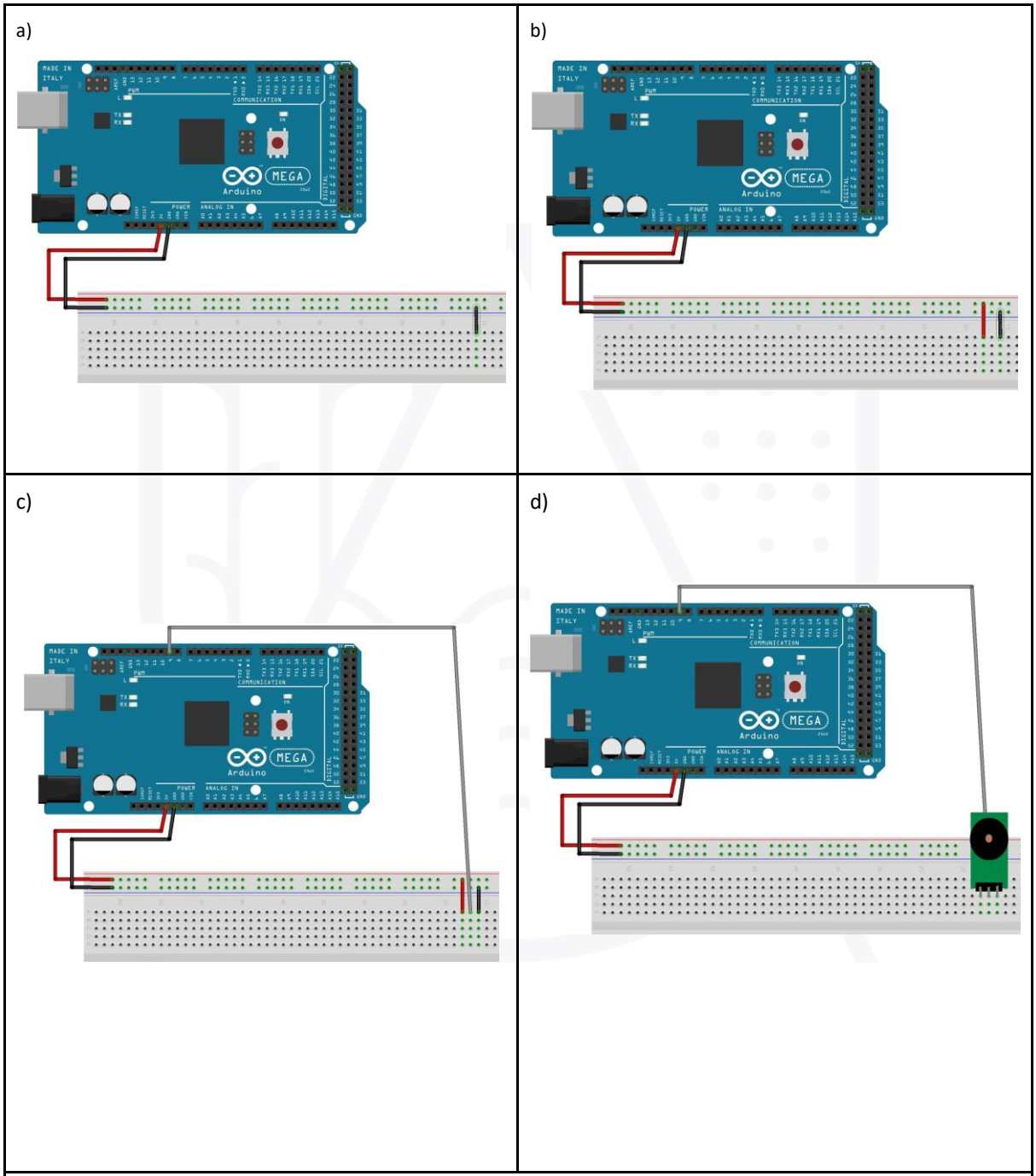


Figura 26: *(a)* Incorporación de conexión de la carga negativa (GND) de la barra color azul de la protoboard hacia el sensor. (b) Incorporación de conexión de la carga positiva (5V) de la barra color roja de la protoboard hacia el sensor. (c) Incorporación de conexión del PIN N°9. (d) Alarma BUZZER incrustada en la protoboard (Nota: El buzzer está por encima de los cables rojo, gris y negro).

TALLER 3
“Estación Meteorológica”
Parte 1

4.2. Sensor de temperatura DHT11 y mediciones.

El sensor DHT11 es un sensor que mide temperatura (°C) y humedad relativa (%) de forma simultánea. Este funciona en un rango de temperatura 0 a 50°C, con un rango de precisión ±2.0 °C. La humedad relativa (RH) trabaja con valores entre 20 y 90 %, con una precisión de ±4 %.

4.2.1. ¿Qué es la temperatura del aire?

Esta se define como el grado de calor específico del aire en un lugar y momento determinado. Además es posible encontrar otros conceptos importantes tales como:

- **Temperatura Máxima**: Valor máximo de temperatura del aire que alcanza un determinado lugar en un determinado período de tiempo. Puede ser horario, diario, mensual, anual, etc.
- **Temperatura Mínima:** Valor mínimo de temperatura del aire que alcanza un determinado lugar en un determinado período de tiempo. Puede ser horario, diario, mensual, anual, etc.
- **Temperatura Media:** Valor medio de temperatura del aire que alcanza un determinado lugar en un determinado período de tiempo. Puede ser horario, diario, mensual, anual, etc.

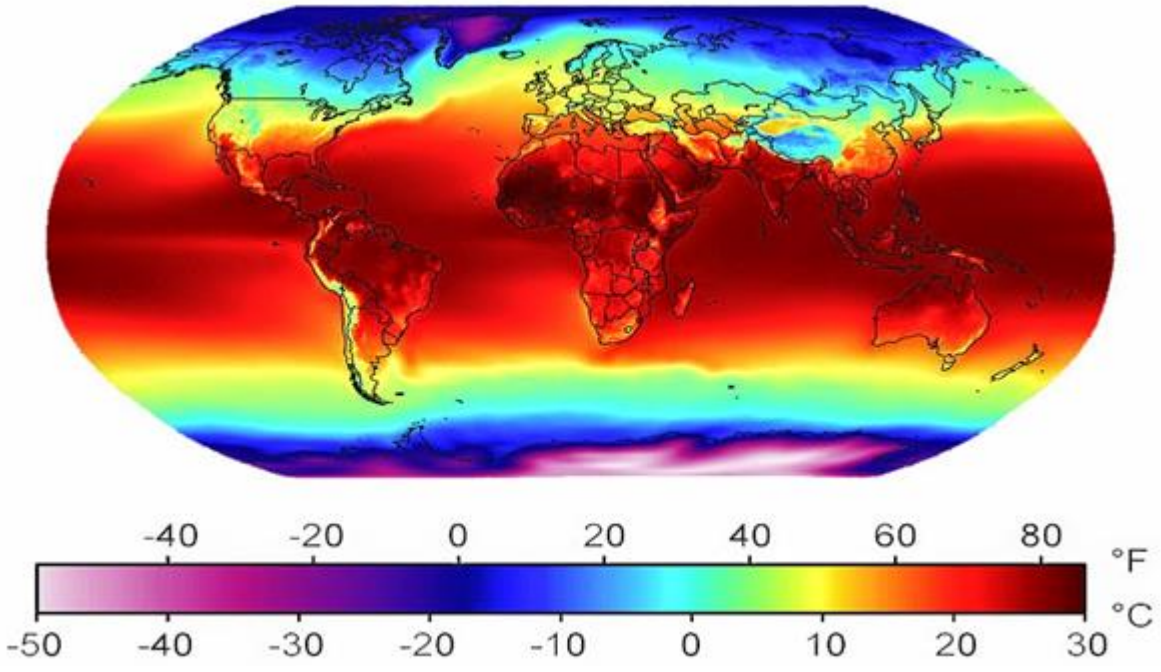


Figura 21: Imagen de las variaciones de la temperatura del aire en la tierra.

Por otra parte, los sensores nos permiten obtener datos, en este caso datos de temperatura del aire, dicha información permite elaborar modelos predictivos para poder conocer acerca de las condiciones climáticas futuras, además de poder estudiar las **variaciones** (Oscilación Natural y recurrente del clima, además son temporales) y **cambios** (Tendencia progresiva en el largo plazo de una condición del clima Se afecta la variabilidad extrema haciéndola frecuente.) climáticos que se han desarrollado durante los últimos años.

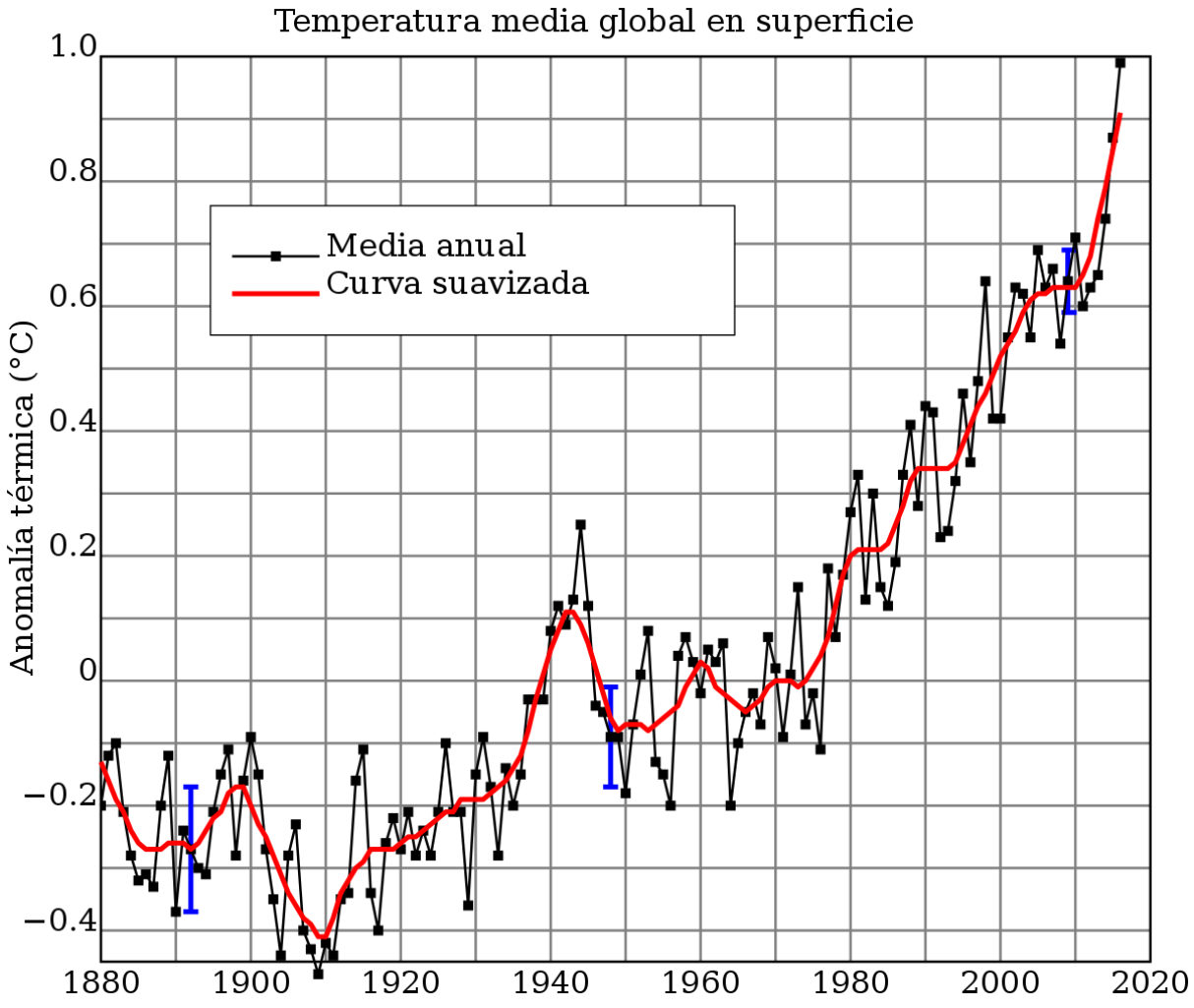


Figura 22: Variación de temperatura (1880-2020)

Pues bien, los datos recolectados por estos sensores a lo largo de los años, ha permitido estudiar las temperaturas de la tierra. Como se puede observar en la figura x, los datos obtenidos permiten ver

las variaciones de la temperatura de superficie en ° C desde 1880 hasta el año 2020, por otra parte, en la figura anterior se puede observar la tendencia de aumento de la temperatura durante estos últimos años.

4.2.2. ¿En qué escalas se mide la temperatura?

La temperatura se puede medir en distintas escalas siendo comúnmente las más usadas:

- **Fahrenheit(°F):** Esta es la escala más antigua que se usa actualmente, esta fue determinada por el físico holandés-alemán Gabriel Daniel Fahrenheit en el año 1724, y como podemos ver esta lleva su nombre, esta escala está determinada por la diferencia entre los puntos Fusión y ebullición del agua y la divide en 180 partes iguales. Esta escala es comúnmente usada en Estados Unidos, es importante destacar que 100°F equivalen a 37.7° Celsius

- **Celsius (°C):** Esta es la escala más usada en la actualidad, fue determinada por el astrónomo sueco Andrés Celsius en el año 1742 y está basada en la división del punto de ebullición del agua en 100 partes iguales, por lo que también se le conoce como escala centígrada.
- **Kelvin (K):** Esta escala determina el 0 absoluto, un punto donde se encuentra una completa ausencia de energía calórica, fue determinada por el físico británico William Thomson Kelvin. En esta escala 0°K equivalen a -273,5°C.

- **Rankin(R):** Esta escala determina el 0 absoluto en relación a los °F, por lo tanto, fue determinada por el ingeniero escocés William John MacQuorn Rankine en el año 1848. En esta escala el 0°R equivalen a -458, 67°F, es importante destacar que en la actualidad el uso de esta escala está prácticamente extinto.

4.2.3. ¿Cómo podemos medir temperatura con Arduino?

Materiales:

- 1 Placa arduino mega 2560
- 1 Protoboard
- 1 Sensor DHT11
- 1 Cable largo de cada color (Negros, Rojo, color a elección)
- 1 Cable corto de cada color (rojo, Negro)

A continuación, seguiremos los siguientes pasos para instalar nuestro sensor DHT11:

Paso 1: Conectamos el GND y los 5V de la placa arduino al protoboard, a la barra azul y roja correspondientemente. además, en este paso agregaremos una conexión con cable corto desde la barra azul hacia los pines de la protoboard como se ve en la figura 23(a)

Paso 2: Conectamos la barra roja con los pines de la protoboard como se ve en la figura 23(b).

Paso 3: Con un cable largo del color que elijamos conectaremos el pin 2 de la placa Arduino con la protoboard, este cable irá conectado entre los dos cables que conectamos anteriormente tal como se ve en la figura 23(c).

Paso 4: Conectamos nuestro sensor DHT11 en la protoboard, en los pines inmediatamente inferiores donde conectamos los cables, aquí hay tener cuidado y conectar el VCC con el cable rojo, el GND con el cable negro y el DATA con el cable que está conectado al pin 2, esto lo podemos observar en sensor donde cada una de sus patas tiene el nombre. Ver figura 23(d).

Paso 5: Para utilizar este sensor deberemos descargar una librería, en la plataforma Arduino, hacemos click en programa, incluir librería y administrar bibliotecas, tal como se muestra en la figura

(e). Una vez se nos abra la ventana buscaremos la librería que utilizaremos, para ello escribiremos en la barra de búsqueda lo siguiente “DHT sensor library” (recuadro rojo)y presionamos *enter*, nos saldrán varias versiones, instalaremos la que es By Adafruit(recuadro amarillo). Ver figura 23(f).

Paso 6: Una vez que realizamos todas las conexiones, y descargamos la librería necesaria Cargaremos el siguiente código:

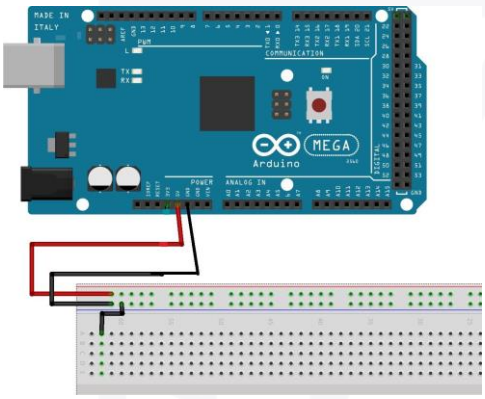
cofigo_DHT11 §

```
#include "DHT.h"

DHT dht(2, DHT11);

void setup() {
  Serial.begin(9600);
  Serial.println("Prueba del sensor DHT11");

  // CARGAMOS LA LIBRERIA PAA NUESTRO SENSOR
  dht.begin();
}

void loop() {
  // TIEMPO ENTRE MEDICIONES
  delay(2000);

  // REALIZAR LECTURA DE HUMEDAD
  float h = dht.readHumidity();
  // REALIZAR LECTURA DE TEMPERATURA
  float t = dht.readTemperature();

  // REVISAR QUE LOS RESULTADOS SEAN VALORES NUMERICOS VALIDOS, INDICANDO QUE LA COMUNICACION ES CORRECTA
  if (isnan(h) || isnan(t)) {
    Serial.println("Falla al leer el sensor DHT11!");
    return;
  }

  // VER RESULTADOS EN MONITOR SERIE
  Serial.print("Humedad: ");
  Serial.print(h);
  Serial.print(" % ");
  Serial.print("Temperatura: ");
  Serial.print(t);
  Serial.println(" *C");
}
```

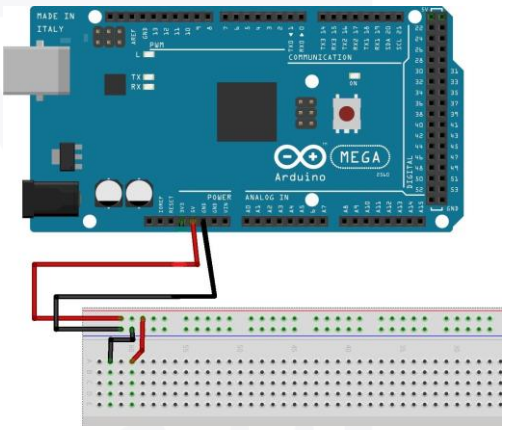
Paso 7: Una vez realizada las conexiones y la carga del código podremos observar los valores obtenidos en el monitor serie que se encuentra en la sección herramientas de la plataforma

Arduino. Este nos mostrará un gráfico con las mediciones que estamos haciendo, la barra roja con indica la variación de la humedad, y la barra morada nos indica la variación de la temperatura Ver figura 23(g) y (h).

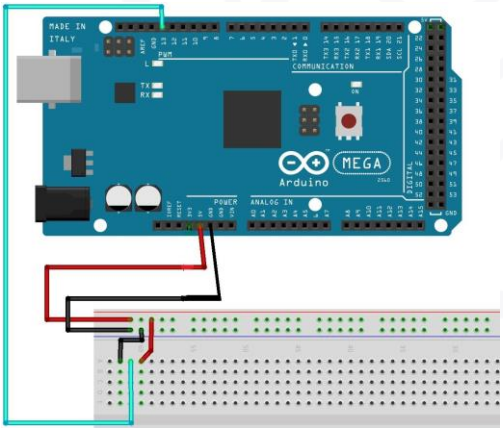
(a)



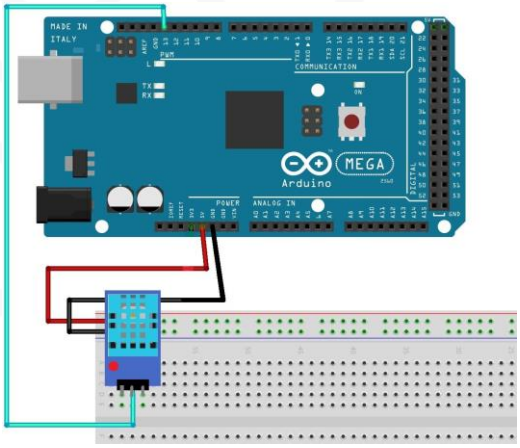
(b)



(c)



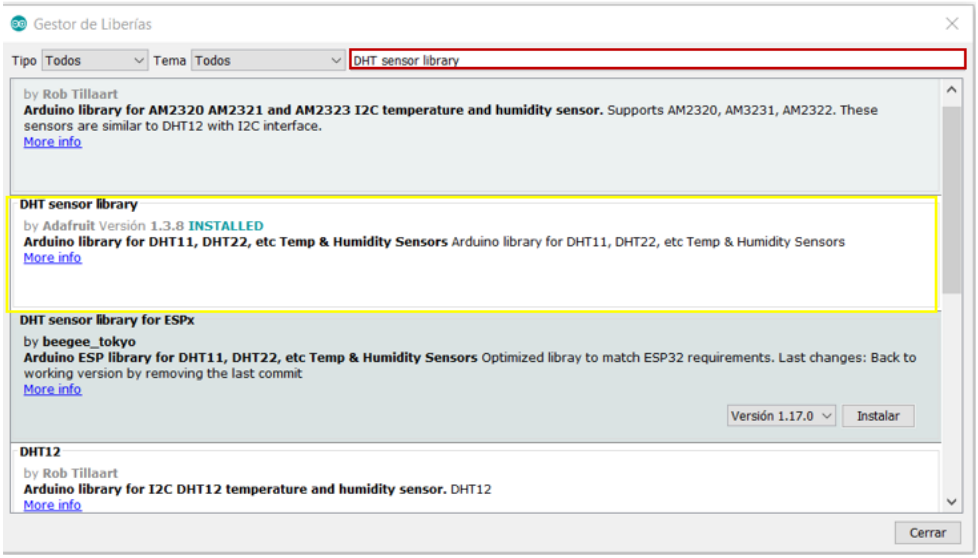
(d)



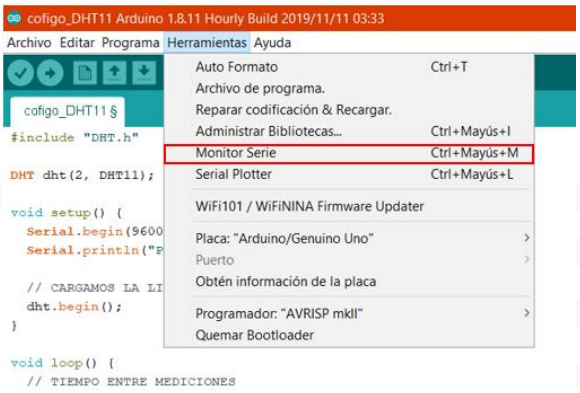
(e)


Archivo Editar Programa Herramientas Ayuda
Verificar/Compilar Ctrl+R
Subir Ctrl+U
Subir Usando Programador Ctrl+Mayús+U
Exportar Binarios compilados Ctrl+Alt+S
Mostrar Carpeta de Programa Ctrl+K
Incluir Librería
Añadir fichero...
Administrar Bibliotecas... Ctrl+Mayús+I
Añadir biblioteca .ZIP...
Arduino bibliotecas
Bridge
EEPROM
Esplora
Ethernet
Firmata
GSM
DHT dht(2,
void setup
Serial.b
Serial.println("Prueba del sensor DHT11");
// PREPARAR LA LIBRERIA PARA COMUNICARSE CO
dht.begin();
}
void loop() {
// ESPERAR ENTRE MEDICIONES
delay(2000);

(f)


Gestor de Liberías
Tipo Todos
Tema Todos
DHT sensor library
by Rob Tillaart
Arduino library for AM2320 AM2321 and AM2323 I2C temperature and humidity sensor. Supports AM2320, AM3231, AM2322. These sensors are similar to DHT12 with I2C interface.
More info
DHT sensor library
by Adafruit Versión 1.3.8 INSTALLED
Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors
More info
DHT sensor library for ESPx
by beegee_tokyo
Arduino ESP library for DHT11, DHT22, etc Temp & Humidity Sensors Optimized libray to match ESP32 requirements. Last changes: Back to working version by removing the last commit
More info
Versión 1.17.0
Instalar
DHT12
by Rob Tillaart
Arduino library for I2C DHT12 temperature and humidity sensor. DHT12
More info
Cerrar

(g)


cofigo_DHT11 Arduino 1.8.11 Hourly Build 2019/11/11 03:33
Archivo Editar Programa Herramientas Ayuda
cofigo_DHT11 §
#include "DHT.h"
DHT dht(2, DHT11);
void setup() {
Serial.begin(9600
Serial.println("P
// CARGAMOS LA LI
dht.begin();
}
void loop() {
// TIEMPO ENTRE MEDICIONES
Auto Formato Ctrl+T
Archivo de programa.
Reparar codificación & Recargar.
Administrar Bibliotecas... Ctrl+Mayús+I
Monitor Serie Ctrl+Mayús+M
Serial Plotter Ctrl+Mayús+L
WiFi101 / WiFiNINA Firmware Updater
Placa: "Arduino/Genuino Uno"
Puerto
Obtén información de la placa
Programador: "AVRISP mkII"
Quemar Bootloader

(h)

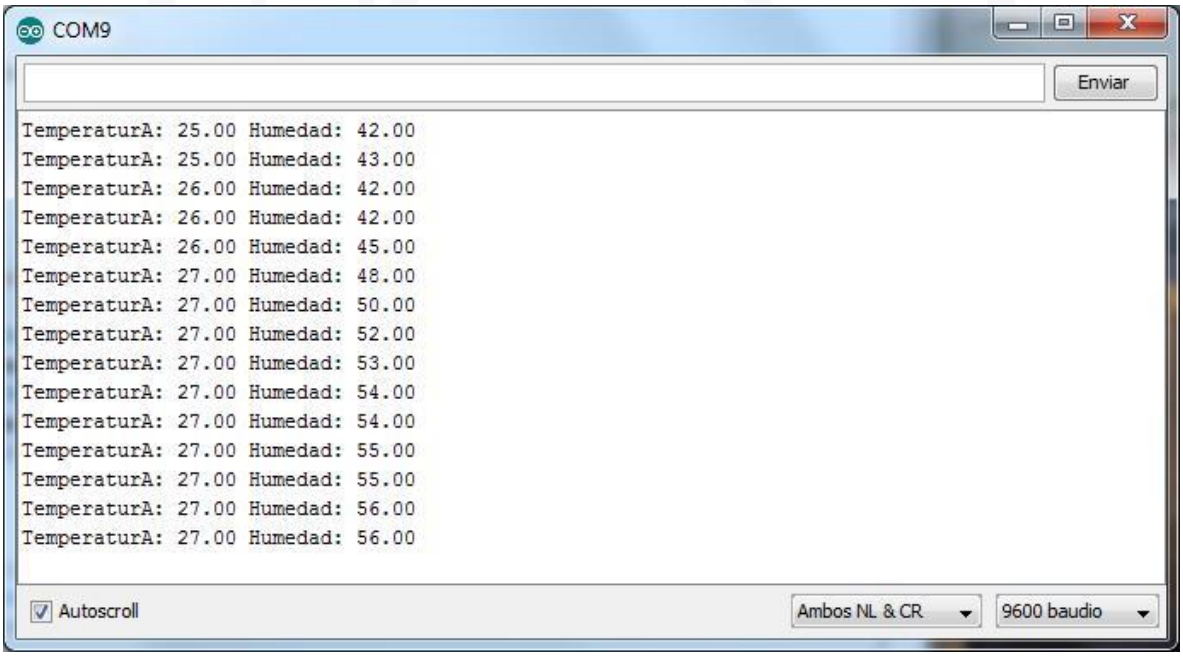

COM9
Enviar
TemperaturA: 25.00 Humedad: 42.00
TemperaturA: 25.00 Humedad: 43.00
TemperaturA: 26.00 Humedad: 42.00
TemperaturA: 26.00 Humedad: 42.00
TemperaturA: 26.00 Humedad: 45.00
TemperaturA: 27.00 Humedad: 48.00
TemperaturA: 27.00 Humedad: 50.00
TemperaturA: 27.00 Humedad: 52.00
TemperaturA: 27.00 Humedad: 53.00
TemperaturA: 27.00 Humedad: 54.00
TemperaturA: 27.00 Humedad: 54.00
TemperaturA: 27.00 Humedad: 55.00
TemperaturA: 27.00 Humedad: 55.00
TemperaturA: 27.00 Humedad: 56.00
TemperaturA: 27.00 Humedad: 56.00
Autoscroll
Ambos NL & CR
9600 baudio

Figura 23: (a) Incorporación de conexión de la carga negativa (GND) de la barra color azul de la protoboard hacia el sensor. (b) Incorporación de conexión de la carga positiva (5V) de la barra color roja de la protoboard hacia el sensor. (c) Incorporación de conexión del PIN digital N° 13 (cable color celeste). (d) Sensor DHT11 incrustado en la protoboard,.(e) y (f) Instalación de librería y (g) Ingreso a monitor serie y (h) visualización de resultados.

4.3. Sensor de temperatura y encendido de luces LED.

Pues bien, una vez que sabemos cómo encender y apagar luces LED, y cómo medir la temperatura con el sensor DHT11 es que es posible elaborar un sistema sencillo de alarma que nos permita saber cuándo las temperaturas registradas representan algún tipo de riesgo (Temperaturas extremas).

Para ello, es importante conocer algunos conceptos importantes, tales como:

- **Alerta:** Es una posibilidad muy alta de que algo ocurra, esta se emite de forma preventiva, para de esta manera tomar las acciones correspondientes y así mitigar o prevenir daños de cualquier tipo.
- **Alarma:** Esta significa una amenaza inminente de que algo ocurra. Al emitir una alarma se pueden ejecutar acciones para reducir los daños, sin embargo, estos de todas formas serán considerables.

Figura 24: Señal de Alarma y señal de Alerta respectivamente

4.4 ¡Desarrolla un sistema preventivo!

El desafío actual es elaborar un sistema de **Alerta** para temperaturas extremas que en caso de ser necesario permita tomar medidas de control a tiempo.

Este sencillo sistema se basa a partir de la utilización de dos luces LED, de diferente color.

• El encendido de la luz verde indica que las temperaturas registradas por el sensor son iguales o están sobre los 5°C	• El encendido de la luz roja indica que las temperaturas registradas por el sensor están bajo los 5°C.

Para realizar este sistema de alerta preventiva se necesitan los siguientes materiales:

- 1 Placa arduino mega 2560
- 1 Protoboard
- 1 Sensor DHT11
- 1 Cable largo de cada color (Negros, Rojo y color a elección)
- 1 Cable corto de cada color (rojo y negro)
- 1 Led color rojo
- 1 Led color verde
- 2 Cables de distintos colores
- 2 resistencias 220Ω.

A continuación, seguiremos los siguientes pasos para realizar la conexión del sensor DHT11 con las luces led:

Paso 1: Utilizaremos las mismas conexiones vistas en la actividad anterior. (ver Fig. 25(a)).

Paso 2: Incorporaremos el Led de color verde con su respectiva resistencia, y conectamos al pin 6. Ver Fig. 25(b).

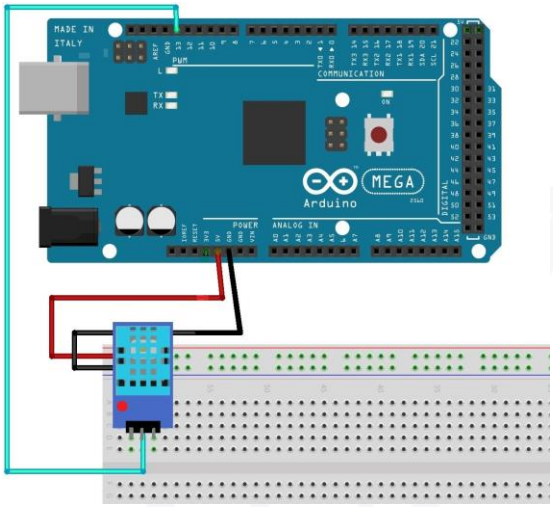
Paso 3: Incorporamos Led de color rojo de la misma manera que el verde, este va conectado al pin 5 del arduino. Ver Fig. 25(b).

Paso 4: Una vez que conectamos todo, cargamos el siguiente código:

sketch_may06a §

```
#include "DHT.h"

DHT dht(2, DHT11);
const int ledrojo = 5;  // pin rojo
const int ledverde = 6; // pin verde
int brillo;


void setup() {
  Serial.begin(9600);
  Serial.println("Prueba del sensor DHT11");
  pinMode(ledrojo,OUTPUT);
  pinMode(ledverde, OUTPUT);

  // CARGAMOS LA LIBRERIA PAA NUESTRO SENSOR
  dht.begin();
}

void loop() {
  // TIEMPO ENTRE MEDICIONES
  delay(2000);

  // REALIZAR LECTURA DE HUMEDAD
  float h = dht.readHumidity();
  // REALIZAR LECTURA DE TEMPERATURA
  float t = dht.readTemperature();
   brillo = map(t, 5, 45, 0, 255) ;           // se ajusta escala de temperatura
brillo = constrain(brillo, 0, 255);                    //7 s elimita el rango



analogWrite(ledrojo, brillo);
analogWrite(ledazul, 255 - brillo);         // ajuste color

  // REVISAR QUE LOS RESULTADOS SEAN VALORES NUMERICOS VALIDOS, INDICANDO QUE LA COMUNICACION ES CORRECTA
  if (isnan(h) || isnan(t)) {
    Serial.println("Falla al leer el sensor DHT11!");
    return;
  }

  // VER RESULTADOS EN MONITOR SERIE
  Serial.print("Humedad: ");
  Serial.print(h);
  Serial.print(" % ");
  Serial.print("Temperatura: ");
  Serial.print(t);
  Serial.println(" *C");
}
```

(a)



(b)

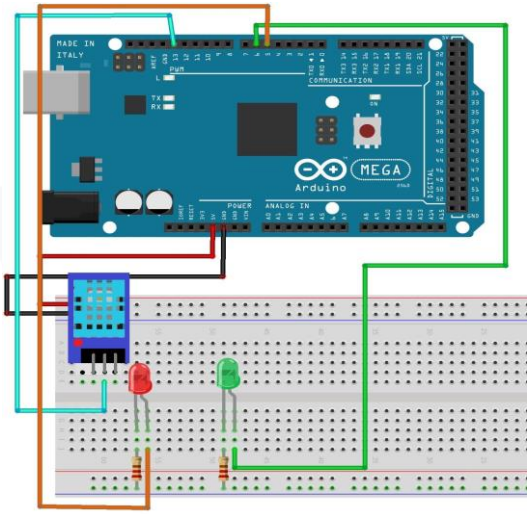


Figura 25: (a) conexión sensor de temperatura DHT11. (b) Sensor DHT11 más incorporación de leds para el sistema de alerta.

4.2.4. Buzzer y sensor de temperatura DHT11.

Ya sabiendo usar y configurar el Buzzer, es posible utilizarlo como una componente de los proyectos futuros que se deseen desarrollar. Es así como se puede desarrollar un sistema de **alarma** a partir del sensor de temperatura DHT11 y el Buzzer, el cual nos permitirá tomar medida para tratar de minimizar cualquier tipo de daño por temperaturas extremas.

¡Desarrolla un sistema de alarma!

Pues bien, el sistema de alarma que se desarrollará consiste en que a temperaturas por sobre los 30°C el buzzer se activará emitiendo un sonido como señal para tomar las medidas pertinentes al caso.

Para los siguiente, se requieren los siguientes materiales:

Materiales

- 1 Placa arduino mega 2560
- 1 Protoboard
- 1 Buzzer
- 1 sensor DHT11
- 1 Cable largo de cualquier color
- 2 Cables cortos de cada color (rojo, Negro)
- 1 Cable largo de cada color (Negros, Rojo, color a elección)

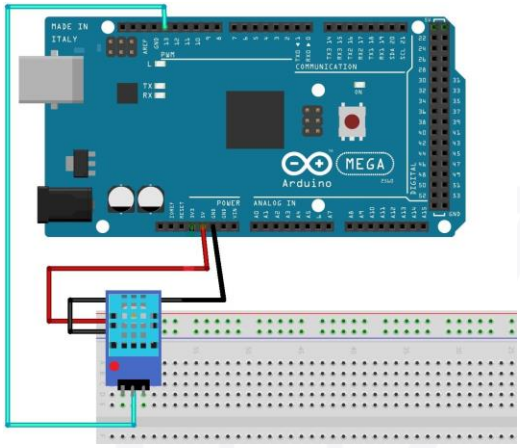
Paso 1: Repetiremos los pasos indicados para la conexión del sensor DHT11 que utilizamos en las actividades anteriores. (ver Fig. 27(a))

Paso 2: Conectaremos el buzzer de la misma manera que en la actividad número 4.2.2.(ver Fig. 27(b)).

Paso 3: Carga el siguiente código:

```
#include "DHT.h"
DHT dht(13, DHT11);
const int ZumTemp =  9;

void setup() {
  Serial.begin(9600);
  Serial.println("Prueba del sensor DHT11");
   pinMode(ZumTemp, OUTPUT);
  // CARGAMOS LA LIBRERIA PAA NUESTRO SENSOR
  dht.begin();
}

void loop() {
  // TIEMPO ENTRE MEDICIONES
  delay(2000);

  // REALIZAR LECTURA DE HUMEDAD
  float h = dht.readHumidity();
  // REALIZAR LECTURA DE TEMPERATURA
  float t = dht.readTemperature();

  if (isnan(h) || isnan(t)) {
    Serial.println("Falla al leer el sensor DHT11!");
    return;
  }

  // VER RESULTADOS EN MONITOR SERIE
  Serial.print("Humedad: ");
  Serial.print(h);
  Serial.print(" % ");
  Serial.print("Temperatura: ");
  Serial.print(t);
  Serial.println(" *C");
  // si la temperatura es mayor a 22 grados celsius se activará la alarma
    if(t>22){
      digitalWrite(ZumTemp, HIGH);
      delay(300);
      digitalWrite(ZumTemp, LOW);
    }
    else{
      digitalWrite(ZumTemp, LOW);
      }

}
```

(a)



(b)

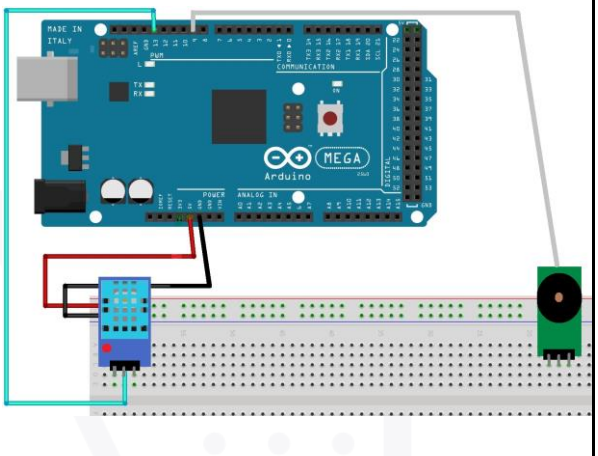


Figura 27: (a) Conexión de dht11 y (b) Incorporación de Buzzer

4.5. Configuración de pantalla LCD.

Algo que resulta importante para el análisis de los datos, resulta ver la visualización de estos, para ello es necesario emplear una pantalla LCD, la cual permitirá poder ver en tiempo real las mediciones hechas por los sensores.

4.5.1. ¿Qué es un LCD?

Su nombre LCD viene de las siglas en inglés Liquid Crystal Display, que traducido al español es pantalla de cristal líquido. Este es un dispositivo visualizador de diferentes tipos de informaciones y contenidos. Estos poseen dos capas de material polarizante. Entre las capas se introduce una solución de cristal líquido, luego una señal eléctrica hace que los cristales se alinean de tal manera que impidan o no el paso de la luz. Cuando la pantalla se pone negra, todos sus cristales están alineados para que no pase luz.

Figura 28: Pantalla LCD convencional.

Sin embargo, el motivo por el cual logran funcionar los LCD, es a partir del **principio fotoeléctrico**, el cual corresponde al fenómeno mediante el cual, los **fotones** provenientes de la radiación electromagnética, inciden sobre una superficie de metal.

Según cierta **frecuencia e intensidad** tienen la capacidad de desprender los electrones de dicha superficie, los cuales salen con un determinado valor de energía cinética.

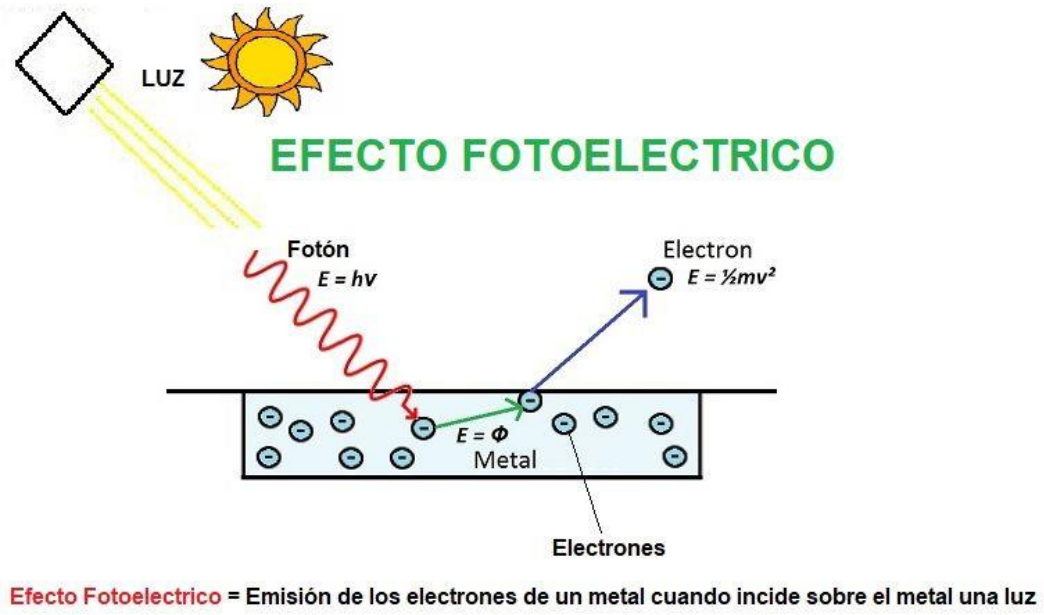


Figura 29: Imagen explicativa del principio fotoeléctrico

4.5.2. Conexión del Potenciómetro de variación lineal.

El potenciómetro es un controlador de luminosidad en la pantalla LCD, ya que este maximiza y minimiza el voltaje dirigido hacia la pantalla.

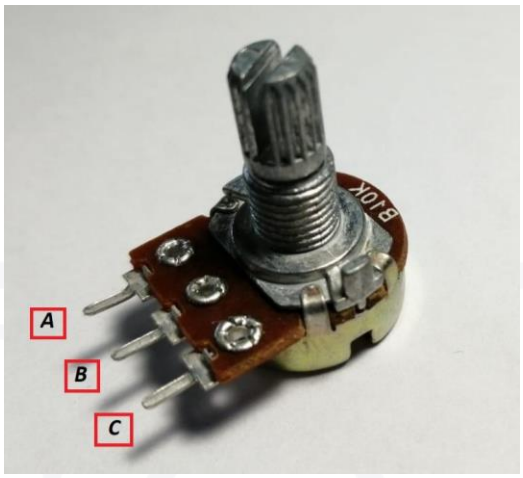


Figura 30: Potenciómetro con sus pines identificados Donde A = Voltaje, B = Salida y C = Conexión tierra.

- Para realizar la conexión de este artefacto, se debe tomar como referencia la **figura 30,** en donde cada patilla está representada con una letra, las cuales se definen como:

4.5.3. ¡Visualiza tu propia información!

Básicamente, un LCD permite la visualización de cualquier tipo de información, para ello es necesario establecer líneas de código, las cuales, al momento de hacer correr el código general en el software de Arduino, permitirán poder ver en la pantalla lo mismo que fue escrito anteriormente, además los LCD también pueden configurarse en conjunto con los sensores a modo de poder ver en tiempo real los datos que estos obtienen.

Para poder escribir un texto y poder verlo en nuestra pantalla LCD son necesarios los siguientes componentes.

Materiales

- 1 Placa arduino mega 2560

- 1 Protoboard
- 16 Cables largos de cualquier color
- Pantalla LCD
- Potenciómetro

A continuación, seguiremos los siguientes pasos para la instalación del LCD:

Paso 1: Realizamos la conexión del GND y los 5V entre el Arduino y la Protoboard. (Ver Fig 31(b)).

Paso 2: Para la conexión de la pantalla LCD con Arduino seguiremos el esquema de conexión. (Ver Fig 31(a)). Conectamos la pantalla LCD a la protoboard de manera que queden conectados todos sus pines en línea. (Ver Fig. 31(b)).

Paso 3: Empezamos a conectar los pines de la pantalla LCD con el Arduino. Primero conectaremos todos los negativos de la pantalla LCD con la barra azul de la protoboard. (ver Fig. 31(b).

Paso 4: Conectaremos los positivos de la pantalla LCD con la barra azul de la protoboard. (Ver Fig. 31(c)).

Paso 5: Conectaremos el pin RS de la pantalla LCD con el pin Digital 5 del Arduino. (ver Fig. 31(d)

Paso 6: Conectaremos el pin D4 de la pantalla LCD con el pin Digital 5 del Arduino. (ver Fig. 31(e)

Paso 7: Conectaremos el pin D5 de la pantalla LCD con el pin Digital 4 del Arduino. (ver Fig. 31(f)

Paso 8: Conectaremos el pin D6 de la pantalla LCD con el pin Digital 3 del Arduino. (ver Fig. 31(g)

Paso 9: Conectaremos el pin D7 de la pantalla LCD con el pin Digital 2 del Arduino. (ver Fig. 31(h)

Paso 10: Conectaremos un cable desde el pin VO de la pantalla LCD hacia el pin central del potenciómetro en un costado de la pantalla LCD. (Ver Fig. 31(i)).

Paso 11: El pin "C" del potenciómetro lo conectaremos con un cable al negativo de la protoboard. (Ver (i)).

Paso 12: El pin "A" del potenciómetro lo conectaremos a la barra positiva de la protoboard (Ver Fig. 31(k).

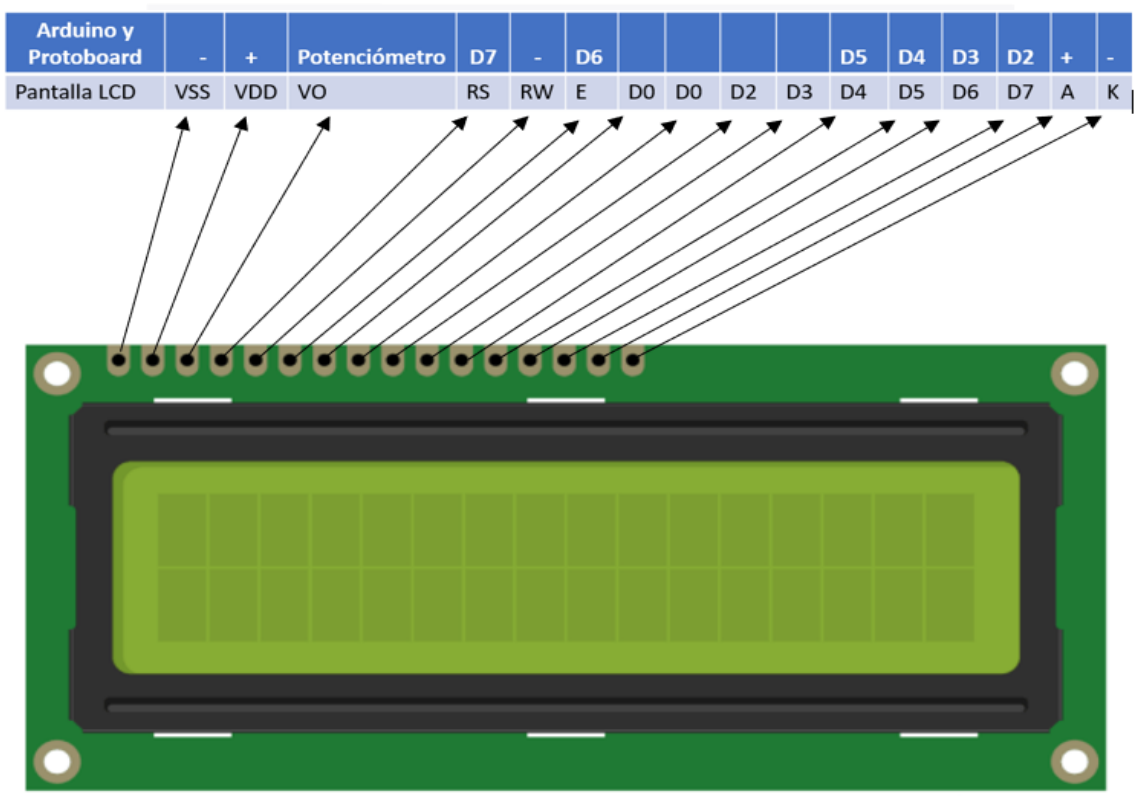
Paso 13: Carga el siguiente código

sketch_may20a §

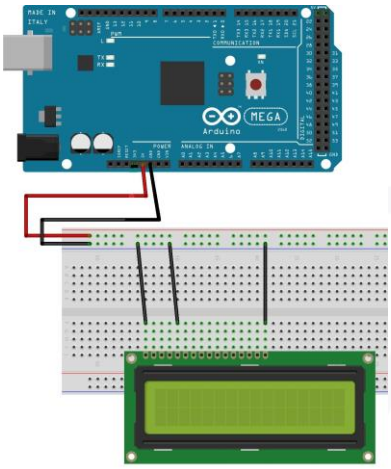
```
#include <LiquidCrystal.h> // Incluye la Libreria LiquidCrystal
LiquidCrystal lcd(1, 2, 4, 5, 6, 7); // Crea un Objeto LC. Parametros: (rs, enable, d4, d5, d6, d7)
void setup() {
  lcd.begin(16,2); // Inicializa la interface para el LCD screen, and determina sus dimensiones (ancho y alto) del display
}
void loop() {
  lcd.print("  Muestra "); // Imprime "LCD Tutorial" sobre el LCD
  delay(3000);   // 3 segundos de espera
  lcd.setCursor(0,1); // Seteamos la ubicacion texto 0 linea 1 que sera escrita sobre el LCD
  lcd.print("de como");
  delay(4000);
  lcd.clear(); // Limpia la pantalla
  delay(1500);
  lcd.blink(); // Displayamos el Blinking del Cursor sobre el LCD
  delay(3000);
  lcd.noBlink(); // Apagamos el Blinking del Cursor sobre el LCD
  lcd.print("poder escribir");
  lcd.print("");
    delay(3000);
  lcd.clear();
  lcd.setCursor(0,1);
  lcd.print("un texto");
  delay(5000);
  lcd.clear(); //
  delay(2000);
  }
```

Conexión de pantalla LCD 16x2 cm

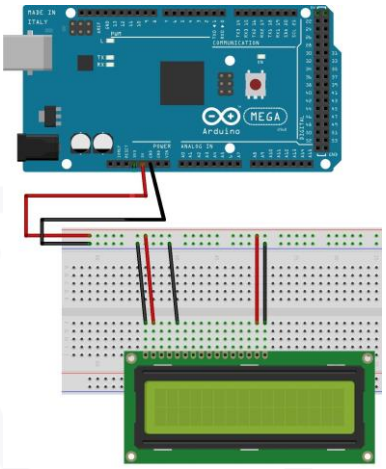
(a)



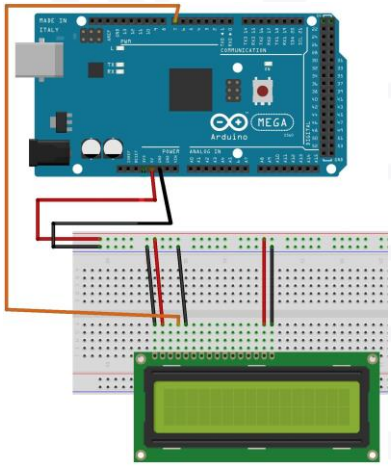
(b)



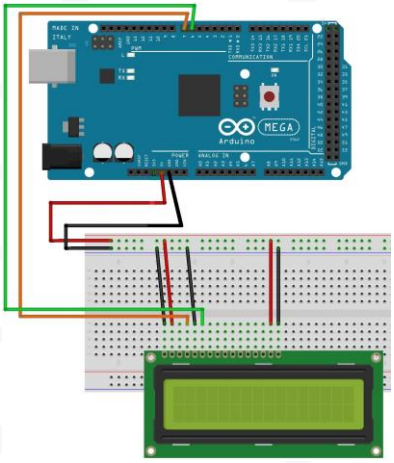
(c)



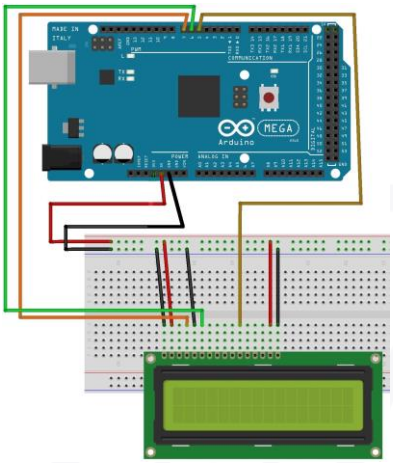
(d)



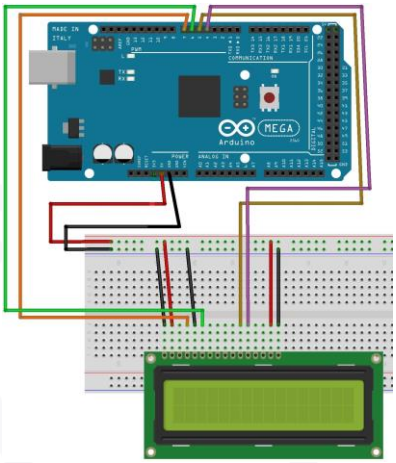
(e)



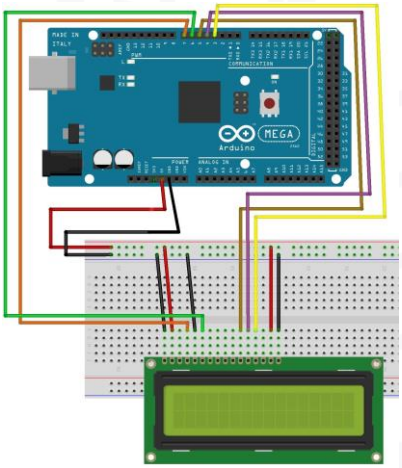
(f)



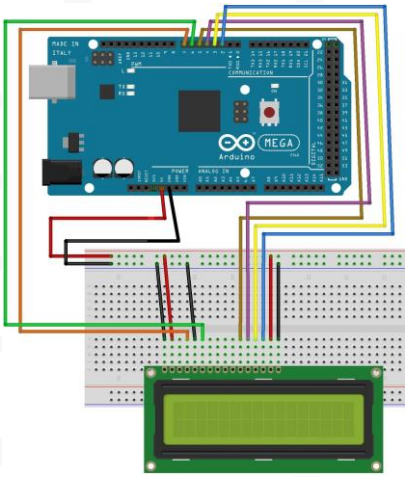
(g)



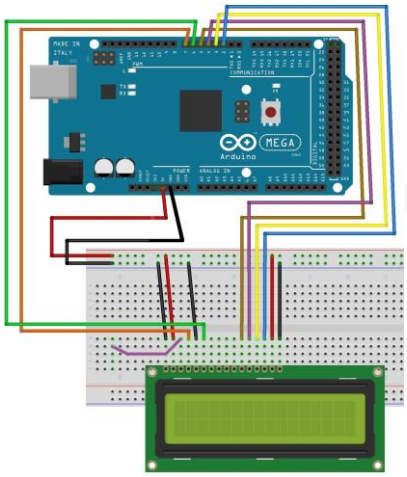
(h)



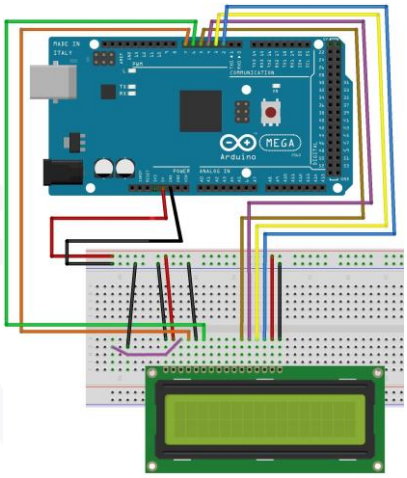
(i)



(j)



(k)



(l)

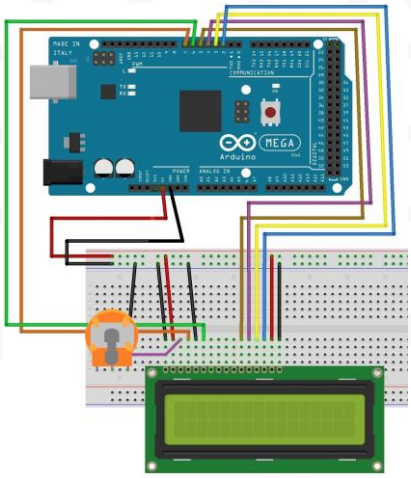


Figura 31: (a)Esquema de conexión LCD(b) Incorporación de conexión de las cargas negativa (GND) de la barra color azul de la protoboard hacia la pantalla LCD. (c) Incorporación de conexión de las cargas positivas(5V) de la barra color rojo de la protoboard hacia la pantalla LCD. (d) Incorporación de conexión del PIN N° 7. (e) Incorporación de conexión del PIN N° 6. (f) Incorporación de conexión

del PIN N° 5. (g) Incorporación de conexión del PIN N° 4. (h) Incorporación de conexión del PIN N° 3. (i) Incorporación de conexión del PIN N° 2. (j) Conexión de VO a potenciómetro. (k) Conexión de 5V a potenciómetro y (l) Conexión de tierra al potenciómetro

TALLER 4
"Estación Meteorológica"
Parte 2

4.5.4. Pantalla LCD y sensor de temperatura DHT11 (visualización de datos) con sistema de alerta y alarma.

Una vez realizados los prácticos anteriores es momento de unificar todas las piezas y componentes básicas que el kit arduino incluye.

Pues bien, el siguiente paso será realizar un prototipo cual se capaz de medir la temperatura y nos permita visualizar los datos en tiempo real, además se incluirá un sistema de alerta compuesto por dos Leds (Rojo <= 10°C, Verde > 10°C), acompañado de un sistema de alarma compuesto por un Buzzer, el cual se activará con Temperaturas menores a 5°C.

¡Manos a la obra!

Para realizar el prototipo anteriormente mencionado serán necesarios los siguientes materiales:

Materiales

- 1 Placa arduino mega 2560
- 1 Protoboard
- 1 Buzzer
- 1 sensor DHT11
- 1 LCD 16x2
- 2 Led (Rojo y Verde)
- 1 Potenciómetro
- 2 cables cortos y 1 largo para el Buzzer
- 3 cables largos para el sensor DHT11
- 2 cables medianos y 2 resistencias de 220Ω.
- 11 cables largos para el LCD
- 3 cables medianos para el potenciómetro
- 2 cables (De preferencia rojo y negro para las conexiones positivas y negativas desde la protoboard a la placa arduino)

A continuación, seguiremos los siguientes pasos para la instalación del sistema de alerta y alarma con medición y visualización de datos de temperatura.

Paso 1: Realizamos la conexión del sensor DHT11 en el PIN 13 (ver Fig. 32(a))

Paso 2: A lo realizado anteriormente agregamos la conexión de ambos Leds. Luz roja debe ir conectada al PIN 10, y la luz verde al PIN 12(ver Fig 32(b))

Paso 3: A continuación, es necesario realizar la conexión del Buzzer, el cual debe ir conectado al PIN 9(Ver figura 32(c))

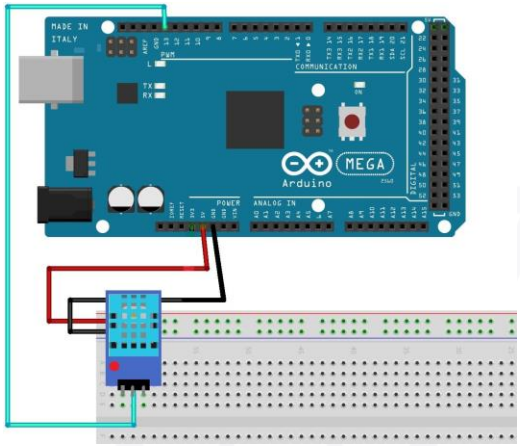
Paso 4: Finalmente se debe realizar la conexión del LCD incluyendo el potenciómetro, en donde las conexiones de estos son las mismas realizadas anteriormente. (Ver figura 32(d))

Paso 5: Una vez realizados todos los pasos anteriormente mencionados se debe cargar el siguiente código

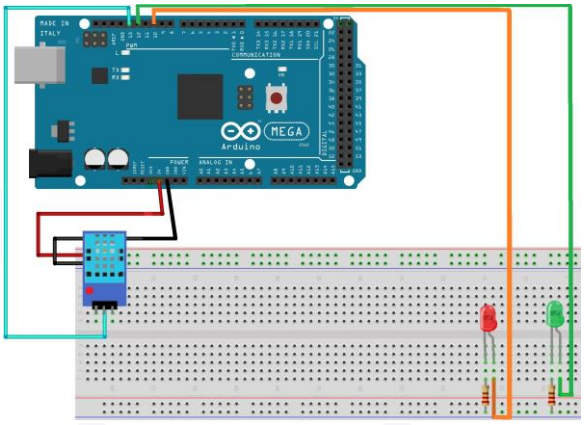
```
#include <LiquidCrystal.h>
LiquidCrystal lcd(7, 6, 5, 4, 3, 2);
#include <DHT.h>

int redLed = 10;
int greenLed = 12;
int buzzer = 9;
// Your threshold value

int SENSOR = 13;


int temp;
int humedad;

DHT dht (SENSOR, DHT11);


void setup() {
  dht.begin();
  pinMode(redLed, OUTPUT);
  pinMode(greenLed, OUTPUT);
  pinMode(buzzer, OUTPUT);
  Serial.begin(9600);
  lcd.begin(16,2);
}
```

```
void loop() {
   humedad = dht.readHumidity();
    temp = dht.readTemperature();


    lcd.setCursor(0,0);
    lcd.print("Temperatura: ");
    lcd.print(temp);
    lcd.print("C");
    lcd.setCursor(0,1);
    lcd.print("Humedad: ");
    lcd.print(humedad);
    lcd.print("%");
    delay(5000);
    lcd.clear();

  if (temp <= 10)
  {
    digitalWrite(redLed, HIGH);
    lcd.setCursor(0, 2);
    lcd.print("---ALERTA---");
    digitalWrite(12, LOW);
  }
     if (temp < 5)
  {
    digitalWrite(redLed, HIGH);
    lcd.setCursor(0, 2);
    lcd.print("---ALERTA---");
    digitalWrite(9, HIGH);
  }
  if(temp > 10)
  {
    digitalWrite(redLed, LOW);
    lcd.setCursor(0, 2);
    lcd.print("---SIN ALERTA---");
    digitalWrite(12, HIGH);
  }
  delay(5000);
  lcd.clear();
  }
```

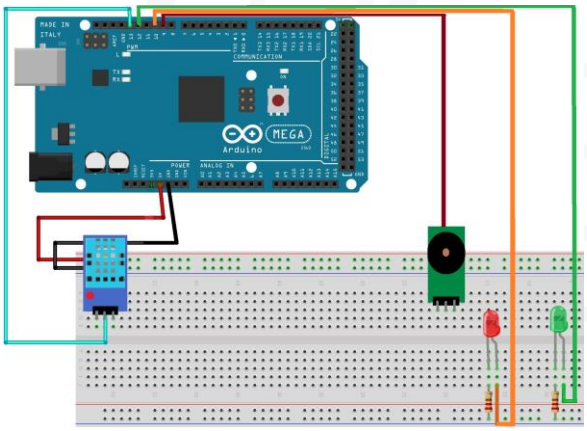
(a)



(b)



(c)



(d)

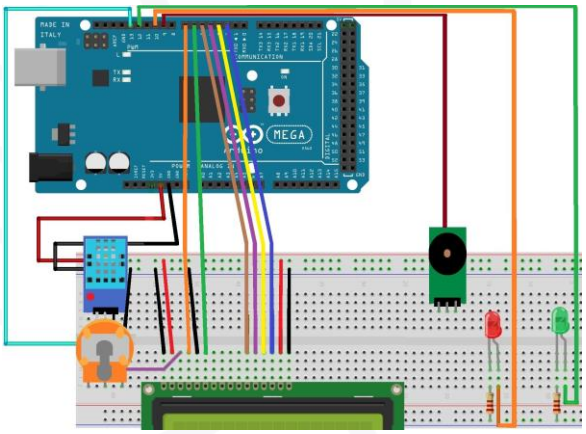


Figura 32: (a) Incorporación del sensor DHT 11, (b) Incorporación de luces LCD´s, (c) Incorporación de alarma buzzer, (d) Incorporación de pantalla LCD.

TALLER 5
“Estación Meteorológica”
Parte 3

4.6. Actividad sensor de gases.

Hoy en día uno de los mayores problemas a nivel mundial resulta ser la contaminación, esta se produce por un sin fin de causas, sin embargo, una de las más importantes corresponde a la emisión de todo tipo de gases.

Sin embargo, el problema no solo resulta ser la contaminación ambiental, esto también trasciende de forma que afecta la salud y calidad de vida de las personas.

Para poder detectar estos niveles letales para la salud, existen algunos sensores los cuales permiten medir la cantidad de gas nocivo en la atmósfera.

Es así como en el Kit Arduino podemos encontrar los siguientes:

4.6.1. Sensor de calidad del aire MQ135.

Este es un sensor de diversos gases peligrosos como el amoniaco, sulfuro, benceno y humo, por ende nos indicará la calidad del aire, este presenta sus unidades de medidas en partículas por millón (ppm).

Figura 33: Sensor MQ135

BANDERA	PUNTOS IMECA	CALIDAD DEL AIRE
	0-50	Buena
	51-100	Regular
	101-150	Mala
	151-200	Muy mala
	➢ 201	Extremadamente mala

Figura 34: Unidades de medida para calidad de aire (ppm)

AO = ANALOG 0

GND = -

VCC = +

Materiales

- 1 Placa arduino mega 2560
- 1 Protoboard
- Sensor MQ135
- 5 Cables largos.

Para la instalación del sensor MQ135 es necesario seguir los siguientes pasos:

Paso 1: Conecta un cable desde la parte indicada como positiva en la protoboard hacia el pin de los 5V, a su vez conecta un cable desde la parte negativa a algún pin de arduino que se indica como GND (Ver Fig. 36(a))

Paso 2: Conecte dos cables, uno a la parte positiva y otro a la parte negativa de la protoboard, estos servirán para establecer la conexión desde la protoboard al sensor. (Ver Fig. 36(b))

Paso 3: Conecte un cable desde la protoboard hacia el pin análogo de arduino indicado como A0. (Ver Fig. 36(c))

Paso 4: Conectar el sensor MQ135 a la protoboard (Ver Fig. 36(d)).

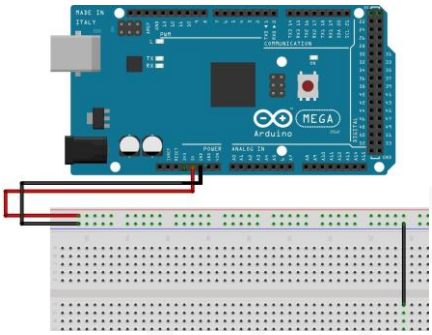
Paso 5: Cargue el siguiente código:

MQ135solo

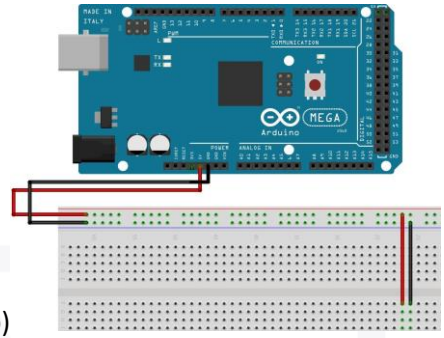
```
void setup()
{
Serial.begin(9600); //Set serial baud rate to 9600 bps
}

void loop()
{
int val;
val=analogRead(0); // Read Gas value from analog 1
Serial.println(val,DEC);//Print the value to serial port
delay(3000); //Registrar datos cada 3 segundos
}
```

Paso 6: Revisa los datos en monitor serie. (Fig. 36(e))

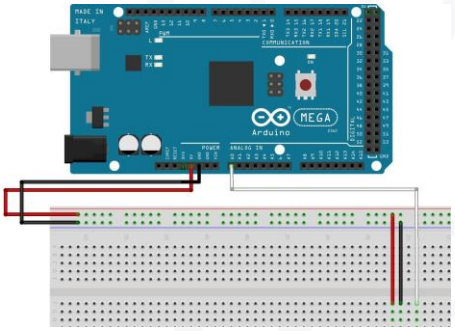
(a)



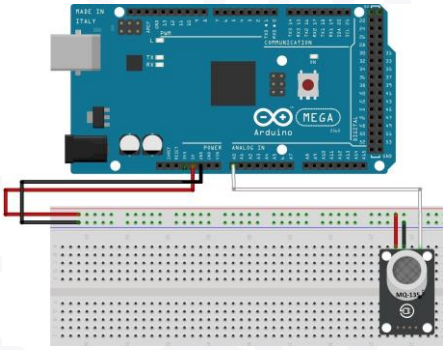
(b)



(c)



(d)



(e)

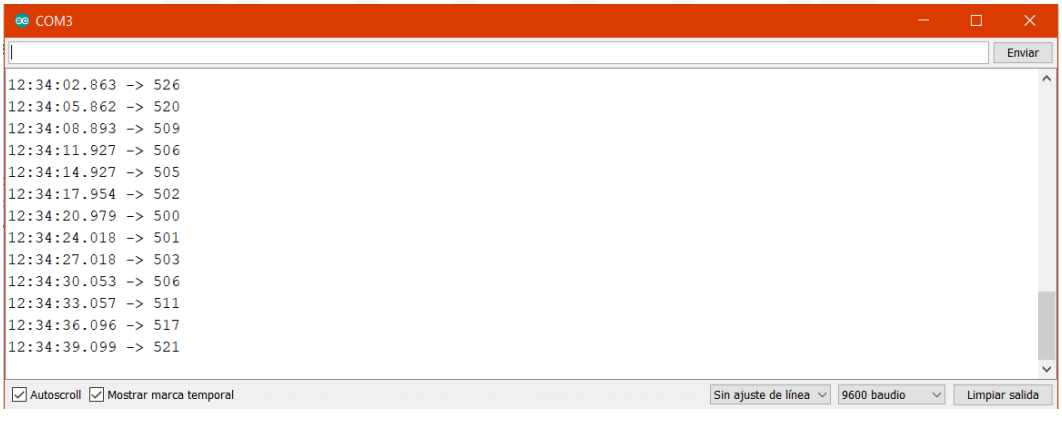


Figura 36: (a) Incorporación de conexión de la **carga negativa (GND)** de la barra color azul de la protoboard hacia el sensor. (b) Incorporación de conexión de la **carga positiva (5V)** de la barra color rojo de la protoboard hacia el sensor. (c) Incorporación de conexión del PIN analogico A0. (d) MQ-135 incrustado en la protoboard y (e)Monitor serie.

4.6.2. Sensor de monóxido de carbono MQ7.

Este sensor nos indica el nivel de monóxido de carbono presente en el aire. La medición sobre la presencia de este gas es sumamente importante, ya que es liberado principalmente por una combustión incompleta, esta se puede producir en braseros, estufas a parafinas, cocinas y calefones, por lo que es común encontrarlo en la mayoría de los hogares. este gas presenta una mayor afinidad con los glóbulos que el oxígeno, por lo que la presencia de CO produce una intoxicación también llamada asfixia química o hipoxia, para conocer de mejor manera las concentraciones en que se mide este gas en partículas por millón (ppm) podemos ver la figura 37. El Threshold Limit Value - Time Weighted Average (TLV-TWA) o Valor Umbral Límite - Media Ponderada en el Tiempo son indicadores de la concentración máxima a la que puede estar expuesta una persona durante una jornada de trabajo (8 horas), también existe la concentración inmediatamente peligrosa para la vida y salud(IPVS), la que indica el límite sobre los cuales las concentraciones de CO son inmediatamente perjudiciales a la salud.

Figura 37: Sensor de gases MQ7

CONCENTRACIÓN EN AIRE	EFECTO
55 mg/m^3 (50 ppm)	TLV-TWA
0,01 %	Exposición de varias horas sin efecto
0,04 - 0,05 %	Exposición una hora sin efectos
0,06 - 0,07 %	Efectos apreciables a la hora
0,12 - 0,15 %	Efectos peligrosos a la hora
165 mg/m^3 (1500 ppm)	IPVS
0,4 %	Mortal a la hora

Figura 38: Tabla de concentración de CO y sus efectos

AO = ANALOG 2

GND = -

VCC = +

Materiales

- 1 Placa arduino mega 2560
- 1 Protoboard
- Sensor MQ135
- 5 Cables largos.

Para la instalación del sensor MQ135 es necesario seguir los siguientes pasos:

Paso 1: Conecta un cable desde la parte indicada como positiva en la protoboard hacia el pin de los 5V, a su vez conecta un cable desde la parte negativa a algún pin de arduino que se indica como GND (Ver Fig. 39(a))

Paso 2: Conecte dos cables, uno a la parte positiva y otro a la parte negativa de la protoboard, estos servirán para establecer la conexión desde la protoboard al sensor. (Ver Fig. 39(b))

Paso 3: Conecte un cable desde la protoboard hacia el pin análogo de arduino indicado como A2. (Ver Fig. 39(c))

Paso 4: Conectar el sensor MQ7 a la protoboard (Ver Fig. 39(d)).

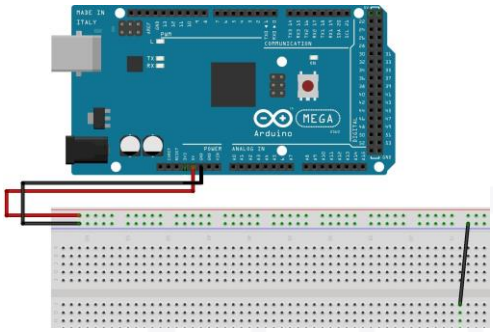
Paso 5: Cargue el siguiente código:

MQ7solo §

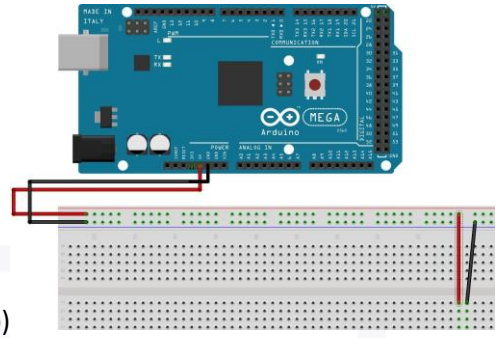
```
void setup()
{
Serial.begin(9600); //Set serial baud rate to 9600 bps
}

void loop()
{
int val;
val=analogRead(2); // Read Gas value from analog 2
Serial.println(val,DEC);//Print the value to serial port
delay(3000); //Registrar datos cada 3 segundos
}
```

Paso 6: Revisa los datos obtenidos en el monitor serie(Fig 39(e)).

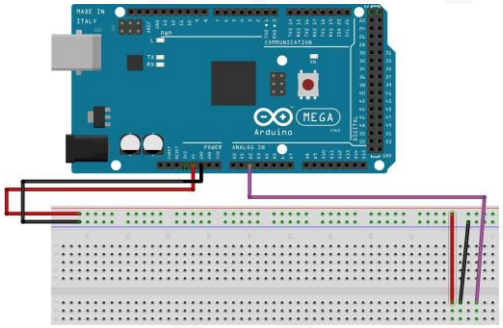
(a)



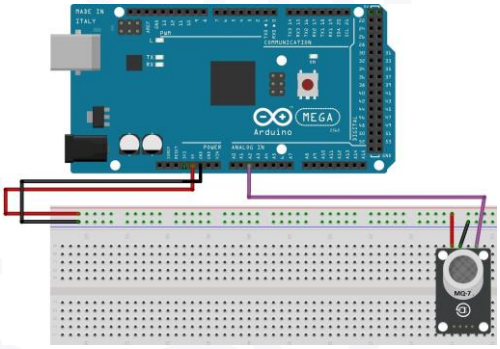
(b)



(c)



(d)



(e)


COM3
Enviar
21:05:40.344 -> 100
21:05:43.380 -> 100
21:05:46.402 -> 99
21:05:49.401 -> 98
21:05:52.431 -> 98
21:05:55.451 -> 98
21:05:58.478 -> 97
21:06:01.507 -> 97
21:06:04.534 -> 96
21:06:07.554 -> 96
21:06:10.548 -> 96
21:06:13.580 -> 96
21:06:16.602 -> 95
Autoscroll
Mostrar marca temporal
Sin ajuste de línea
9600 baudio
Limpiar salida

Figura 40: (a) Incorporación de conexión de la **carga negativa (GND)** de la barra color azul de la protoboard hacia el sensor. (b) Incorporación de conexión de la **carga positiva (5V)** de la barra color rojo de la protoboard hacia el sensor. (c) Incorporación de conexión del PIN analógico A2. (d) MQ-7 incrustado en la protoboard y (e)Monitor serie.

4.6.3. Sensor de gas natural MQ5.

Este sensor nos permite realizar mediciones sobre gas natural (GNC) y Gas Licuado de Petróleo (GLP). La medición sobre la presencia de este gas es sumamente importante, ya que es utilizado principalmente como combustible para calefacción o para las cocinas a gas, por lo que es común encontrarlo en la mayoría de los hogares. este gas presenta una densidad mayor que el oxígeno, por lo que la presencia de GLP o GNC produce un desplazamiento del oxígeno si las concentraciones de GLP o GNC son muy altas el nivel de oxígeno disminuye y si alcanza un nivel menor al 19,5% en la atmósfera empieza a ser riesgoso para el ser humano, generando asfixia en las personas cuando su concentración sobrepasa ciertos límites (figura 41).

Figura 41: Sensor de gas MQ5

AO = ANALOG 1

GND = -

VCC = +

Materiales

- 1 Placa arduino mega 2560
- 1 Protoboard
- Sensor MQ135
- 5 Cables largos.

Para la instalación del sensor MQ135 es necesario seguir los siguientes pasos:

Paso 1: Conecta un cable desde la parte indicada como positiva en la protoboard hacia el pin de los 5V, a su vez conecta un cable desde la parte negativa a algún pin de arduino que se indica como GND (Ver Fig. 43(a))

Paso 2: Conecte dos cables, uno a la parte positiva y otro a la parte negativa de la protoboard, estos servirán para establecer la conexión desde la protoboard al sensor. (Ver Fig. 43(b))

Paso 3: Conecte un cable desde la protoboard hacia el pin análogo de arduino indicado como A1. (Ver Fig. 44(c))

Paso 4: Conectar el sensor MQ5 a la protoboard (Ver Fig. 45(d)).

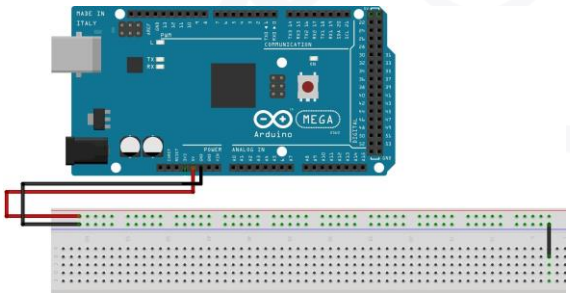
Paso 5: Cargue el siguiente código:

Paso 6: Revisa los datos obtenidos en el monitor serie (Fig. 45(e)).

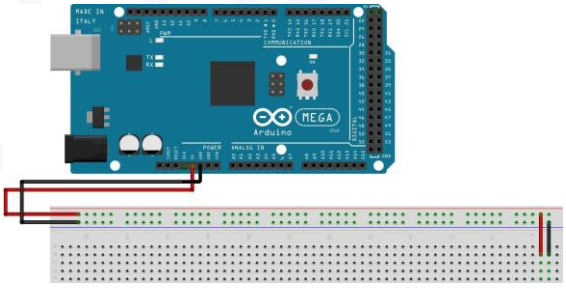
MQ5solo §

```
void setup()
{
Serial.begin(9600); //Set serial baud rate to 9600 bps
}

void loop()
{
int val;
val=analogRead(1); // Read Gas value from analog 1
Serial.println(val,DEC);//Print the value to serial port
delay(3000); //Registrar datos cada 3 segundos
}
```

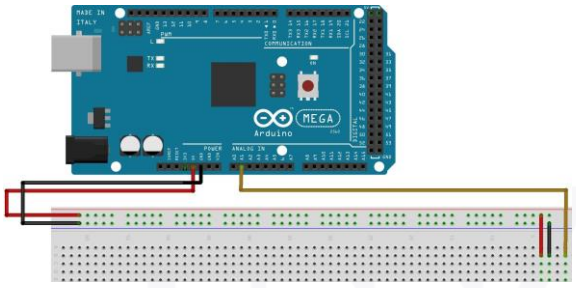
(a)



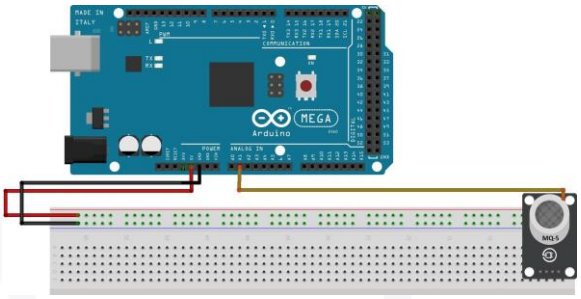
(b)



(c)



(d)



(e)

Figura 45: (a) Incorporación de conexión de la **carga negativa (GND)** de la barra color azul de la protoboard hacia el sensor. (b) Incorporación de conexión de la **carga positiva (5V)** de la barra color rojo de la protoboard hacia el sensor. (c) Incorporación de conexión del PIN analógico A1. (d) MQ-5 incrustado en la protoboard y (e)Monitor serie.

4.6.4. Sistema integrado kit Arduino

Es así como después de haber desarrollado múltiples proyectos con nuestro kit Arduino hemos llegado al proyecto final, ya hemos encendido luces led, medido temperatura del aire, desarrollado un sistema de alerta y alarma, hemos visualizado la información en tiempo real a partir del LCD y hemos medido la concentración de diferentes gases nocivos para la salud.

Sin embargo, ya es momento de integrar todas las componentes del kit Arduino para generar un sensor capaz de hacer todas las funciones anteriormente mencionadas, para lo siguiente necesitaremos los siguientes materiales:

Materiales

- 1 Placa arduino mega 2560
- 1 Protoboard
- 1 Buzzer
- 1 Sensor DHT11
- 1 LCD 16x2
- 2 Led (Rojo y Verde)
- 1 Potenciómetro
- 2 Cables cortos y 1 largo para el Buzzer
- 3 Cables largos para el sensor DHT11
- 2 Cables medianos y 2 resistencias de 220Ω.
- 11 Cables largos para el LCD
- 3 Cables medianos para el potenciómetro
- 2 Cables (De preferencia rojo y negro para las conexiones positivas y negativas desde la protoboard a la placa arduino)
- 1 Sensor de gases MQ135
- 1 Sensor de gases MQ7
- 1 Sensor de gases MQ5
- 3 Cables para las conexiones del MQ135
- 3 Cables para las conexiones del MQ7
- 3 Cables para las conexiones del MQ5

Para las conexiones del sistema integrado del kit Arduino es necesario seguir los pasos que se muestran a continuación:

Paso 1: Realizamos la conexión del sensor DHT11 en el PIN 13 (ver Fig. 47(a))

Paso 2: A lo realizado anteriormente agregamos la conexión de los 3 Leds. Luz roja debe ir conectada al PIN 10, la luz amarilla va conectada al pin 11 y la luz verde al PIN 12(ver Fig. 47(b))

Paso 3: A continuación, es necesario realizar la conexión del Buzzer, el cual debe ir conectado al PIN 9(Ver Fig. 47(c))

Paso 4: Realizar la conexión del LCD incluyendo el potenciómetro, en donde las conexiones de estos son las mismas realizadas anteriormente. (Ver Fig. 47(d))

Paso 5: Conectar el sensor MQ135, MQ7 y MQ5 desde la protoboard al pin análogo A0, A2 y A1 respectivamente (Ver Fig. 47(e)).

Paso 6: Una vez realizados todos los pasos anteriormente mencionados (ver Fig. 47(f)), se debe cargar el siguiente código:

Parte 1

```
#include <LiquidCrystal.h>
LiquidCrystal lcd(7, 6, 5, 4, 3, 2);
#include <DHT.h>

int redLed = 10;
int yellowLed = 11;
int greenLed = 12;
int smokeA0 = A0;
int MQ_5 = A1;
int MQ_7 =A2;
int buzzer = 9;
// Your threshold value

int sensorMQ135;
int sensorMQ7;
int sensorMQ5;
int SENSOR = 13;


int temp;
int humedad;

DHT dht (SENSOR, DHT11);


void setup() {
  dht.begin();
  pinMode(redLed, OUTPUT);
  pinMode(greenLed, OUTPUT);
  pinMode(yellowLed, OUTPUT);
  pinMode(smokeA0, INPUT);
  pinMode(MQ_5, INPUT);
  pinMode(MQ_7, INPUT);
  Serial.begin(9600);
  lcd.begin(16,2);
}
```

Parte 2

```
void loop() {
   humedad = dht.readHumidity();
    temp = dht.readTemperature();


    lcd.setCursor(0,0);
    lcd.print("Temperatura: ");
    lcd.print(temp);
    lcd.print("C");
    lcd.setCursor(0,1);
    lcd.print("Humedad: ");
    lcd.print(humedad);
    lcd.print("%");
    delay(5000);
    lcd.clear();

 int sensorMQ7 = analogRead(MQ_7);
  Serial.print("Pin A2: ");
  Serial.println(sensorMQ7);
  lcd.print("Cantidad CO:");
  lcd.print(sensorMQ7 -50);
  // Checks if it has reached the threshold value


 if(sensorMQ7 -50 >= 330)
  {
    digitalWrite(redLed, HIGH);
    lcd.setCursor(0, 2);
    lcd.print("---RIESGO---");
    digitalWrite(12, LOW);
    digitalWrite(11, LOW);
    tone(buzzer,200);

  }
  if(sensorMQ7 -50 >= 300)
  {
    digitalWrite(redLed, HIGH);
    lcd.setCursor(0, 2);
    lcd.print("---MALA---");
    digitalWrite(12, LOW);
    digitalWrite(11, LOW);
    noTone(buzzer);
  }
```

Parte 3

```
 if(sensorMQ7 -50 >= 55 && sensorMQ7 -50 < 300)
  {
    digitalWrite(yellowLed, HIGH);
    lcd.setCursor(0, 2);
    lcd.print("---REGULAR---");
    digitalWrite(10, LOW);
    digitalWrite(12, LOW);
    noTone(buzzer);

  }
  if(sensorMQ7 -50 < 55)

  {
    digitalWrite(12, HIGH);
    lcd.setCursor(0, 2);
    lcd.print("---BUENA---");
    digitalWrite(10, LOW);
    digitalWrite(11, LOW);
    noTone(buzzer);

    }
  delay(5000);
  lcd.clear();



int sensorMQ135 = analogRead(smokeA0);


   Serial.print("Pin A0: ");
  Serial.println(sensorMQ135);
  lcd.print("Calidad del aire:");
  lcd.print(sensorMQ135);
  // Checks if it has reached the threshold value
  if (sensorMQ135 > 200)
  {

    lcd.setCursor(0, 2);
    lcd.print("---MUY MALA---");
```

Parte 4

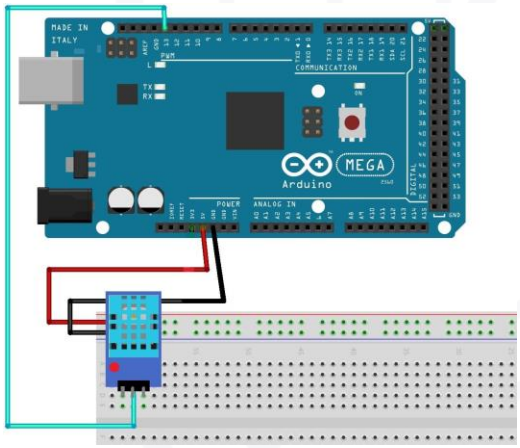
```
if(sensorMQ135 -50 >= 100 && sensorMQ135 -50 <=200)
{
  lcd.setCursor(0, 2);
  lcd.print("---MALA---");

}
if(sensorMQ135 -50 >= 50 && sensorMQ135 -50 <= 100)
{
  lcd.setCursor(0, 2);
  lcd.print("---REGULAR---");


}
if(sensorMQ135 -50 < 50)

{
  lcd.setCursor(0, 2);
  lcd.print("---BUENA---");

 }
delay(5000);
lcd.clear();

int sensorMQ5 = analogRead(MQ_5);

Serial.print("Pin A1: ");
Serial.println(sensorMQ5);
lcd.print("Cantidad GLP:");
lcd.print(sensorMQ5 -50);
// Checks if it has reached the threshold value
if (sensorMQ5 -50 > 400)
{
  lcd.setCursor(0, 2);
  lcd.print("---MUY ALTO---");
}
if(sensorMQ5 -50 >= 150 && sensorMQ5 -50 <= 400)
{
  lcd.setCursor(0, 2);
  lcd.print("---ALTO---");
}
```

Parte 5

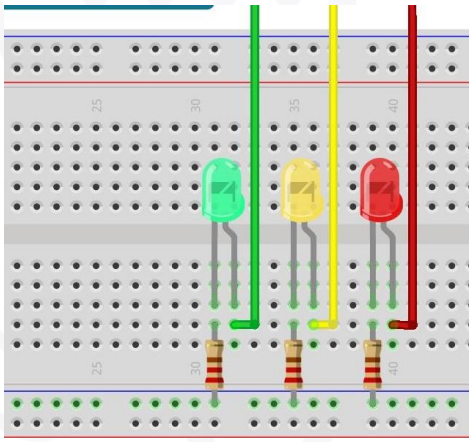
```
if(sensorMQ5 -50 >= 50 && sensorMQ5 -50 <= 150)
 {
   lcd.setCursor(0, 2);
   lcd.print("---REGULAR---");

 }
 if(sensorMQ5 -50 < 50)

 {
   lcd.setCursor(0, 2);
   lcd.print("---BAJO---");
  }
 delay(5000);
 lcd.clear();
 }
```

Figura 46: Código Sensores, LED`s, pantalla LCD y Buzzer

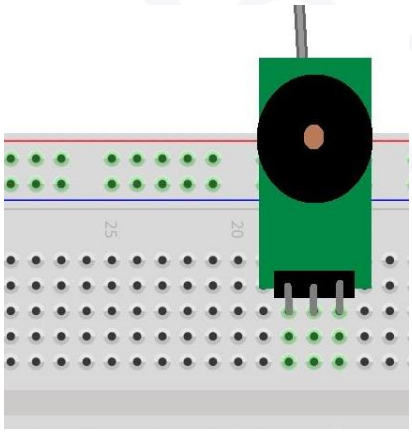
(a)



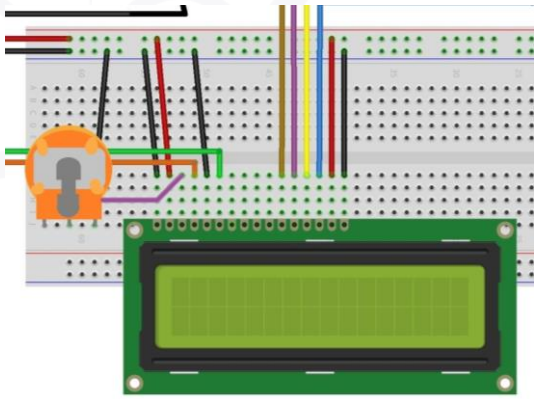
(b)



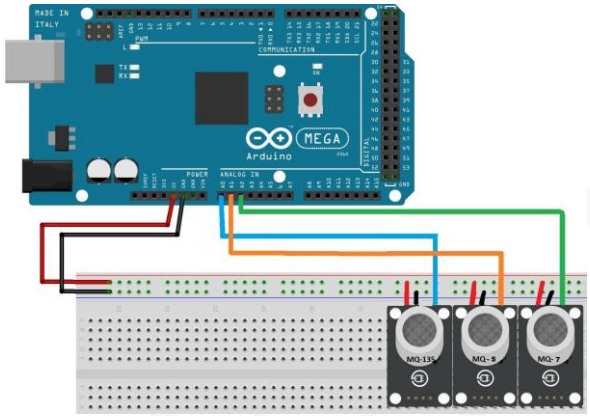
(c)



(d)



(e)



(f)

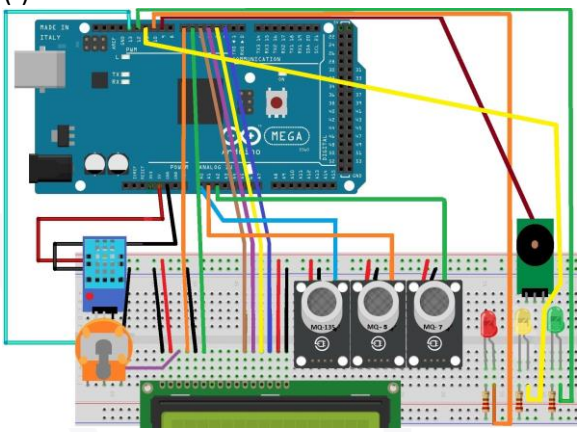


Figura 47: (a)Conexión DHT11, (b)conexión de luces LED, (c) conexión Buzzer, (d)Conexión pantalla LCD y Potenciómetro, (e)Conexión MQ5, MQ7 y MQ135 y (f)Conexión final.

TALLER 6
“Controlador WeMoS y Sensor de Polvo”

5. Placa WeMos

Para los siguientes ejercicios, utilizaremos la placa WeMos que viene dentro del Kit. Esta placa microcontroladora integra internamente un módulo Arduino UNO convencional y un módulo ESP8266 que permite al Arduino conectarse a una red de internet mediante Wi-Fi.

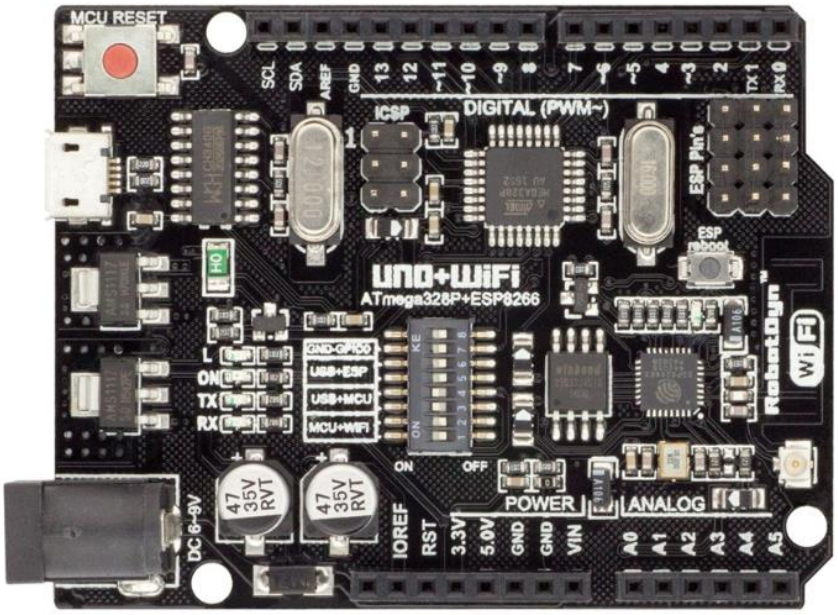


Figura 48. Placa WeMos (UNO+ESP8266).

La funcionalidad de la placa WeMos es similar a la de una placa Arduino convencional, con la diferencia de tener unos pequeños Pines (Figura 49) que permiten cambiar el modo de uso de la placa y alternar los distintos módulos que la placa contiene.

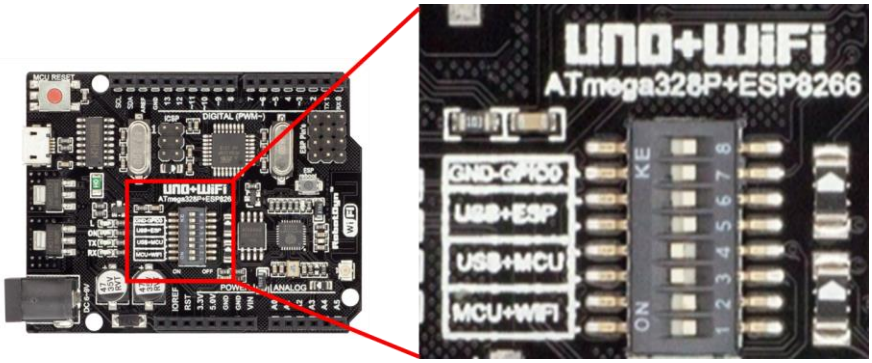


Figura 49. Pines y combinaciones de modos de la placa.

Las distintas combinaciones de Pines en encendido (ON) y apagado (OFF) están resumidas en la siguiente tabla.

TIPOS DE CONEXIÓN	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
A. USB Conecta a ESP8266 (subir código)	OFF	OFF	OFF	OFF	ON	ON	ON	No Usar
B. USB Conecta a UNO (subir código)	OFF	OFF	ON	ON	OFF	OFF	OFF	No Usar
C. UNO+ESP8266	ON	ON	OFF	OFF	OFF	OFF	OFF	No Usar
D. UNO+ESP8266 (Serial UNO)	ON	ON	ON	ON	OFF	OFF	OFF	No Usar
E. UNO+ESP8266 (Serial ESP8266)	ON	ON	OFF	OFF	ON	ON	OFF	No Usar
F. Modulos independientes	OFF	OFF	OFF	OFF	OFF	OFF	OFF	No Usar

Combinación de pines y funciones del módulo.

5.1 Instalación de paquetes en Arduino IDE

Para poder utilizar y enviar códigos hacia el módulo de ESP8266 es necesario instalar la librería de ESP8266 mediante los siguientes pasos:

1) Entrar a Archivo > Preferencias

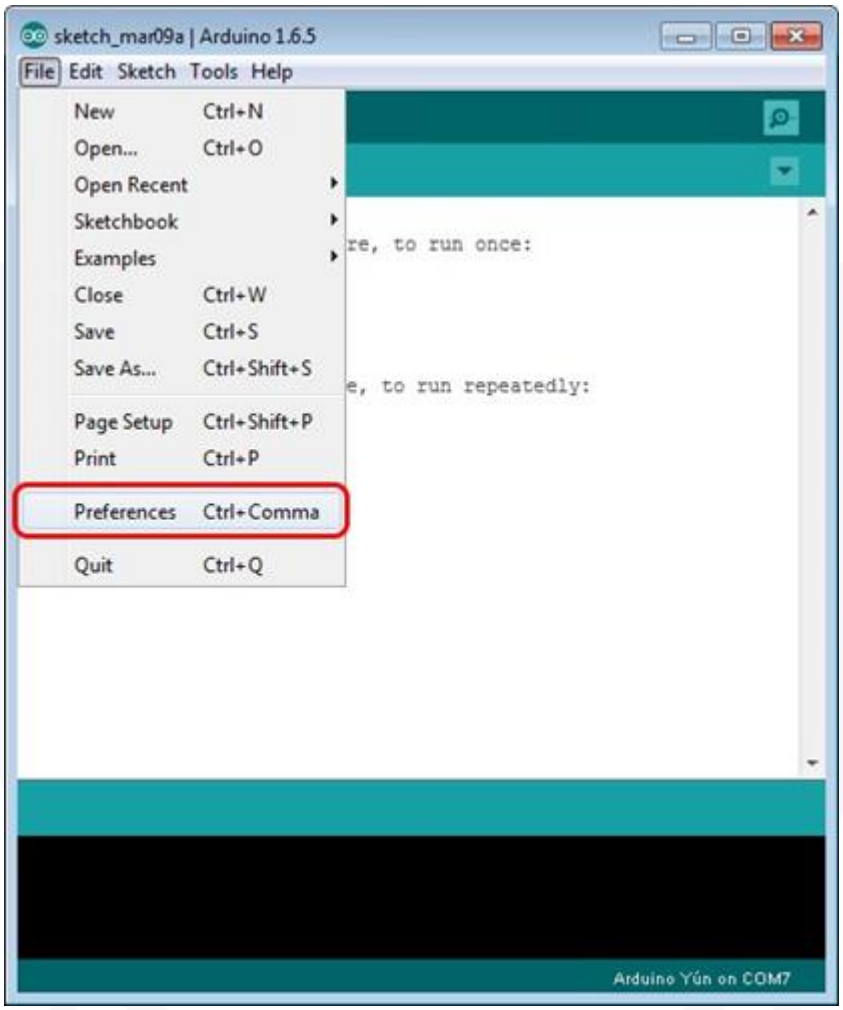


Figura 50. Interfaz de Arduino, Archivos -> Preferencias.

2) Dentro de las preferencias en la pequeña ventana de "URL de Gestión de Tableros adicionales" (Marcado en rojo) colocar el siguiente link: "http://arduino.esp8266.com/stable/package_esp8266com_index.json". Arduino IDE instalará configuraciones adicionales para compilar correctamente los códigos respectivos para el módulo ESP8266. Luego presionar OK.

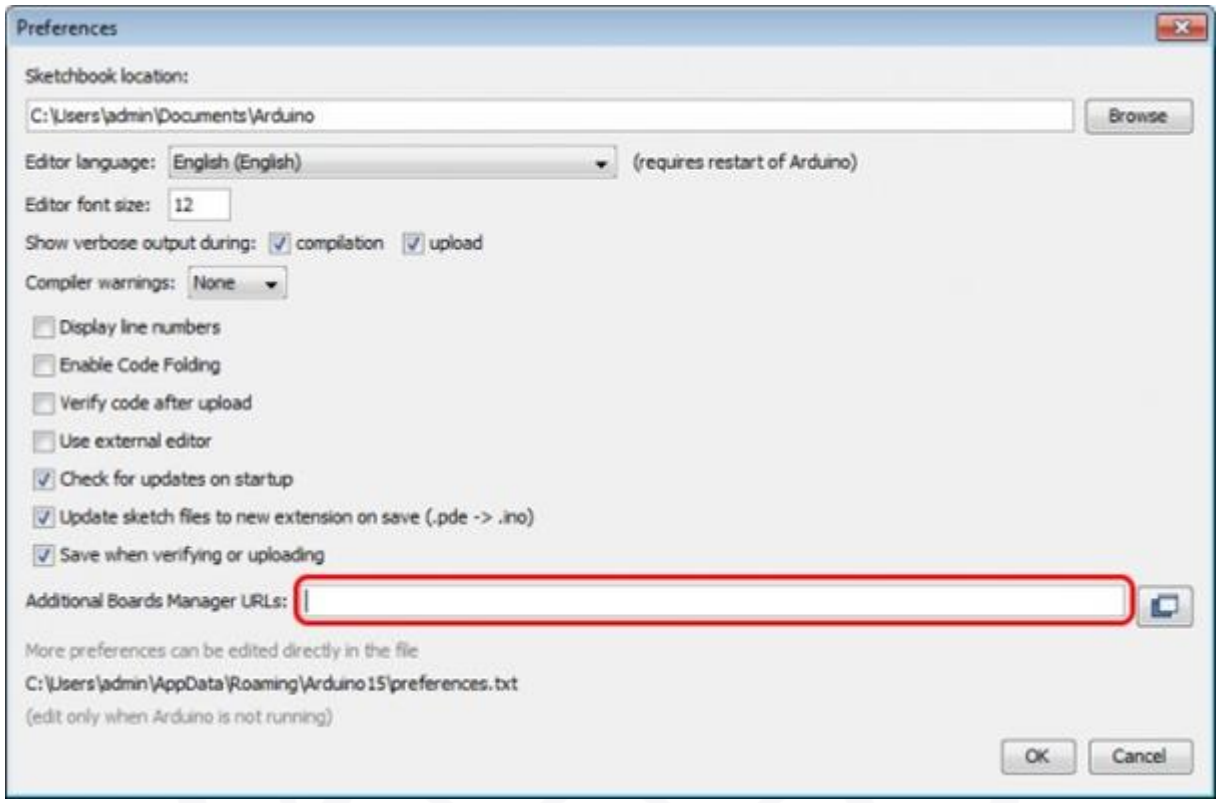


Figura 51. Ventana de Preferencias donde colocar el link para descarga de librería.

3) Ahora seleccionamos en la pestaña Herramientas > Tableros > Gestión de tableros.

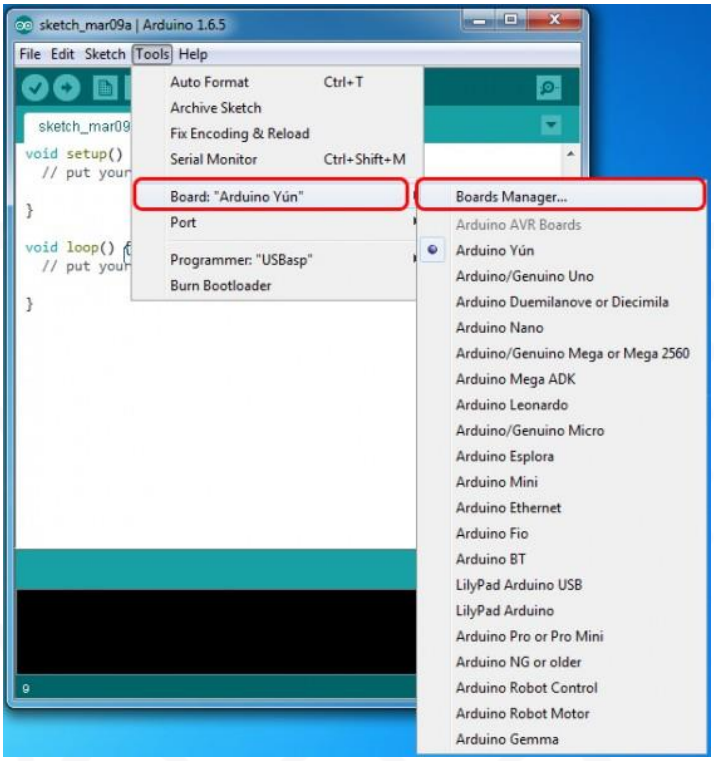


Figura 52. Pestaña de herramientas en configuración de Placas.

4) En la ventana emergente, buscamos y seleccionamos el script ESP8266 hecho por la comunidad de ESP8266. Y seleccionamos la versión 2.1.0 y presionamos instalar

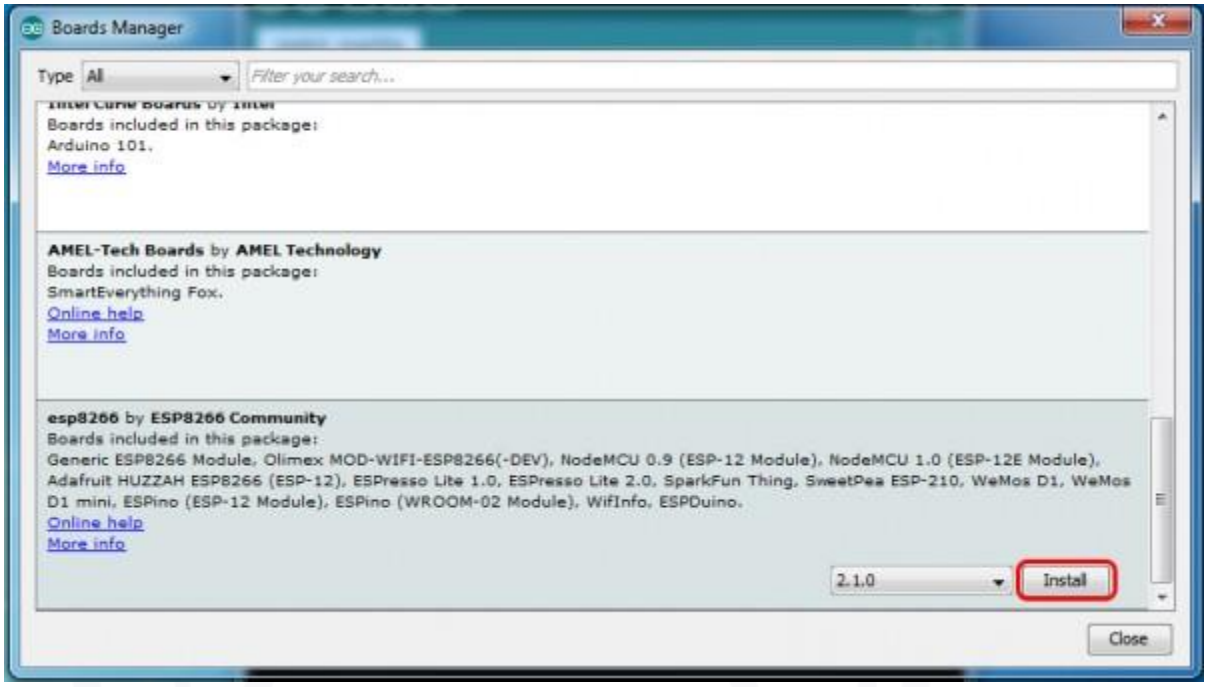


Figura 53. Ventana de Administrador de placas.

5) Luego de la instalación, volvemos a la pestaña de Herramientas>Tableros para encontrar y seleccionar “Generic ESP8266 Module”

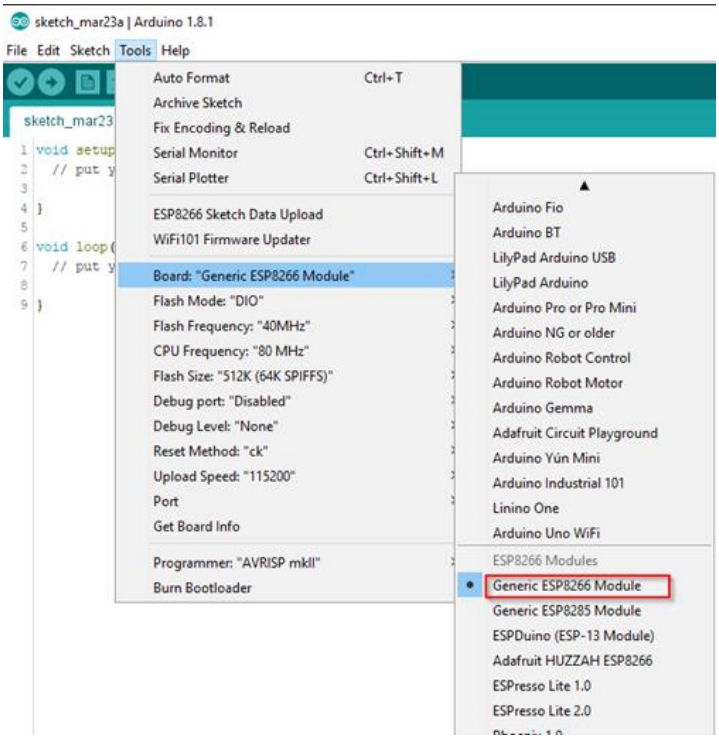


Figura 54. Nuevas opciones de placas para elegir

6) Con estos pasos listos Arduino IDE ya es posible enviar un script o código hacia el módulo ESP8266. Por medio de la opción Sketch Data Upload

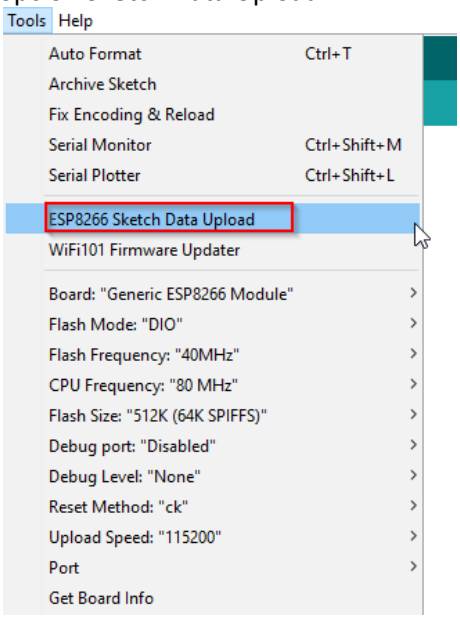


Figura 55. Nueva opción dentro de herramientas para importar un código.

5.2 Sensor de partículas del Aire Grove (HM3101)

El sensor Grove - Laser PM2.5 (HM3301) es una nueva generación de sensores de detección de polvo por láser, que se utiliza para la detección continua y en tiempo real de polvo en el aire.

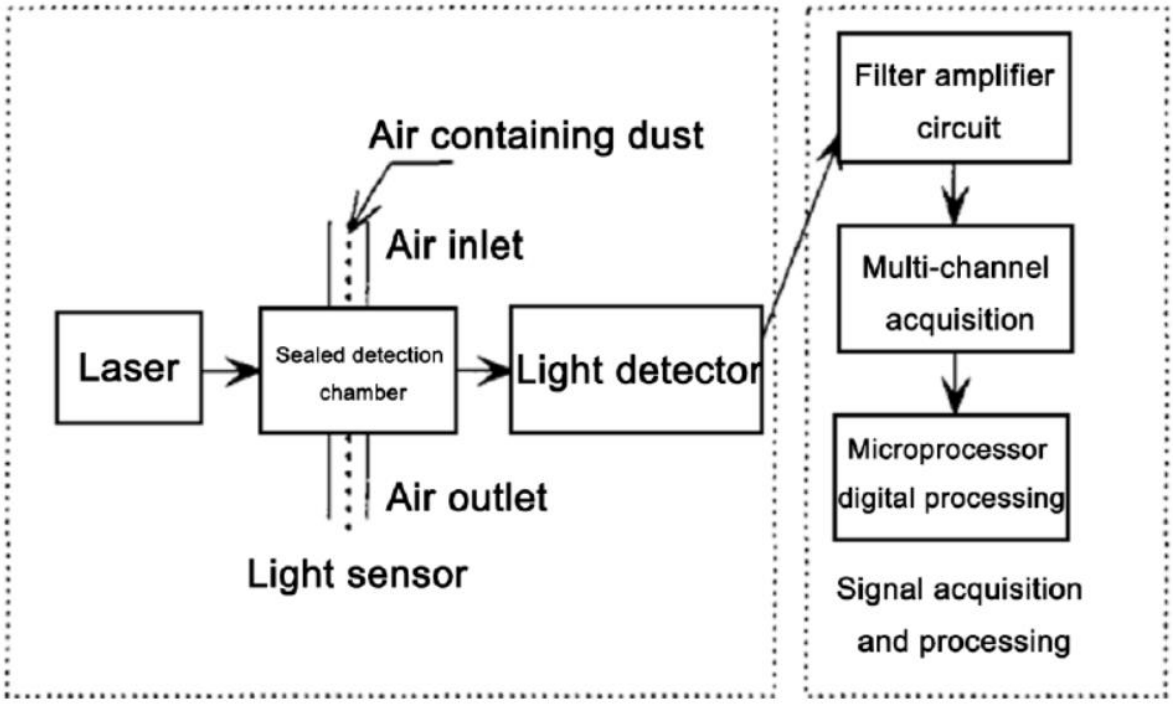
A diferencia del sensor de detección de polvo de bombeo, el HM-3301 utiliza de manera innovadora aspas de ventilador para impulsar el aire, y el aire que fluye a través de la cámara de detección se utiliza como muestra de prueba para realizar pruebas continuas y en tiempo real en polvo de diferentes tamaños de partículas en el aire.

Este módulo es adecuado para detectores de polvo, purificadores de aire inteligentes, acondicionadores de aire inteligentes, ventiladores de ventilación inteligentes, pruebas de calidad del aire, medidores de neblina, monitoreo ambiental y productos y aplicaciones relacionados.

¿Cómo Funciona?

El sensor de polvo HM-3301 se basa en la avanzada teoría de dispersión de Mie. Cuando la luz atraviesa partículas con una cantidad igual o mayor que la longitud de onda de la luz, producirá una dispersión de luz. La luz dispersa se concentra en un fotodiodo altamente sensible, que luego es amplificado y analizado por un circuito. Con un modelo y algoritmo matemático específico, se obtiene la concentración de recuento y la concentración de masa de las partículas de polvo.

El sensor de polvo HM-3301 está compuesto por componentes principales como un ventilador, una fuente de láser infrarrojo, un espejo de condensación, un tubo fotosensible, un circuito amplificador de señal y un circuito de clasificación de señal.



5.3 Estación Grove con transmisión de datos WiFi

Una de las maneras de ampliar nuestro proyecto de medición de Calidad del Aire, es agregándole módulos más complejos y avanzados a nuestro proyecto, esta vez, por medio de transmisión de datos en tiempo real y de manera remota. Existe una variedad de alternativas para incluir un sistema de transmisión de datos en los Arduino que se basan en distintas tecnologías de comunicación, como el Bluethoot y el WiFi, para el siguiente ejercicio, utilizaremos un componente WiFi que está implementado de manera interna en nuestra placa Arduino WeMos. Pero primero, realizaremos unos ejercicios más sencillos, para poder comprender el funcionamiento del sensor de polvo Grove.

5.4 Sensor de Polvo Grove HM-3301

Materiales:

- Placa Arduino Mega 2560
- Protoboard
- Sensor HM-3301
- Cables Pin

Paso 1: Conecta un cable desde la parte indicada como positiva en la protoboard hacia el pin de los 5V, a su vez conecta un cable desde la parte negativa a algún pin de Arduino que se indica como GND.

Paso 2: Conecta el sensor Grove en sus respectivos pines SDA y SCL a la placa Arduino, esta tiene una entrada SDA y SCL a la izquierda de las entradas de los pines digitales (Sección superior de la placa Arduino).

Paso 3: Conecta los pines positivos y negativos del sensor Grove a la fila positiva y negativa, respectivamente, del protoboard. Con esto ya tendremos energizado el sensor y podemos trabajar con el código del sensor.

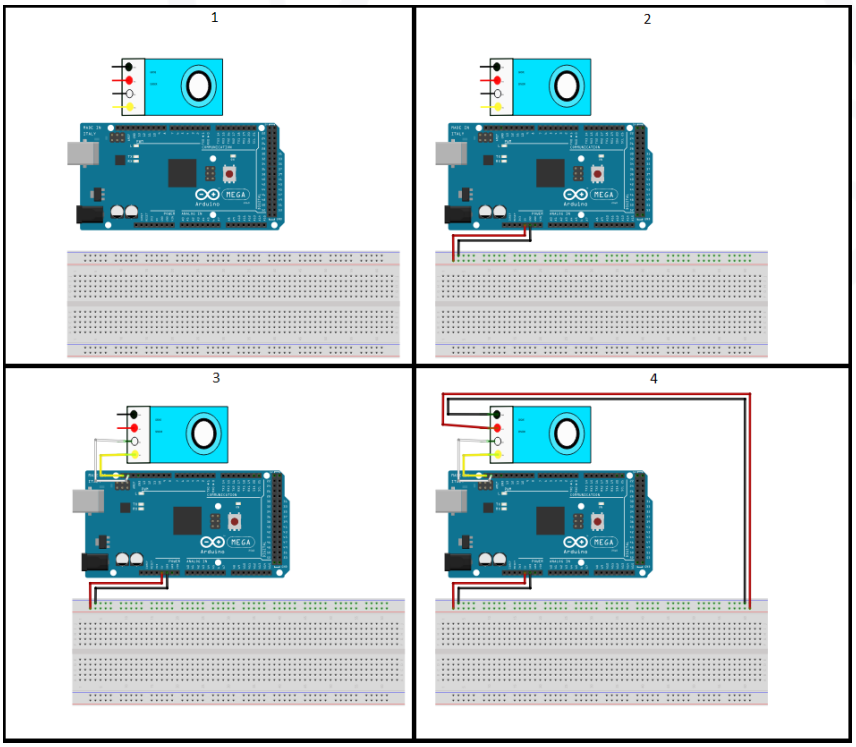


Figura 56. Diagrama de conexión Sensor de polvo Grove al Arduino.

Parte 4: Cargar el siguiente código que se encuentra en Archivos > Ejemplo > Grove – Laser PM2.5 Sensor HM3301 > Basic_demo.

basic_demo

```
#include "Seeed_HM330X.h"

#ifdef  ARDUINO_SAMD_VARIANT_COMPLIANCE
#define SERIAL_OUTPUT SerialUSB
#else
#define SERIAL_OUTPUT Serial
#endif


HM330X sensor;
u8 buf[30];


const char *str[]={"sensor num: ","PM1.0 concentration(CF=1,Standard particulate matter,unit:ug/m3): ",
                   "PM2.5 concentration(CF=1,Standard particulate matter,unit:ug/m3): ",
                   "PM10 concentration(CF=1,Standard particulate matter,unit:ug/m3): ",
                   "PM1.0 concentration(Atmospheric environment,unit:ug/m3): ",
                   "PM2.5 concentration(Atmospheric environment,unit:ug/m3): ",
                   "PM10 concentration(Atmospheric environment,unit:ug/m3): ",
};

HM330XErrorCode print_result(const char* str,u16 value)
{
    if(NULL==str)
        return ERROR_PARAM;
    SERIAL_OUTPUT.print(str);
    SERIAL_OUTPUT.println(value);
    return NO_ERROR;
}

/*parse buf with 29 u8-data*/
HM330XErrorCode parse_result(u8 *data)
{
    u16 value=0;
    if(NULL==data)
        return ERROR_PARAM;
    for(int i=1;i<8;i++)
    {
        value = (u16)data[i*2]<<8|data[i*2+1];
        print_result(str[i-1],value);

```

Paso 5: Ahora dentro del Serial Monitor, vamos a poder observar las medidas que el sensor detecta.

```
sensor num: 0
PM1.0 concentration(CF=1,Standard particulate matter,unit:ug/m3): 45
PM2.5 concentration(CF=1,Standard particulate matter,unit:ug/m3): 63
PM10 concentration(CF=1,Standard particulate matter,unit:ug/m3): 69
PM1.0 concentration(Atmospheric environment,unit:ug/m3): 34
PM2.5 concentration(Atmospheric environment,unit:ug/m3): 50
PM10 concentration(Atmospheric environment,unit:ug/m3): 59
```

5.5 Sensor de Polvo Grove HM-3301 Y pantalla LCD

En el siguiente ejercicio, usaremos las conexiones del ejercicio pasado, pero ahora añadiremos una pantalla LCD para poder observar los valores sin la necesidad de recurrir al Monitor Serial.

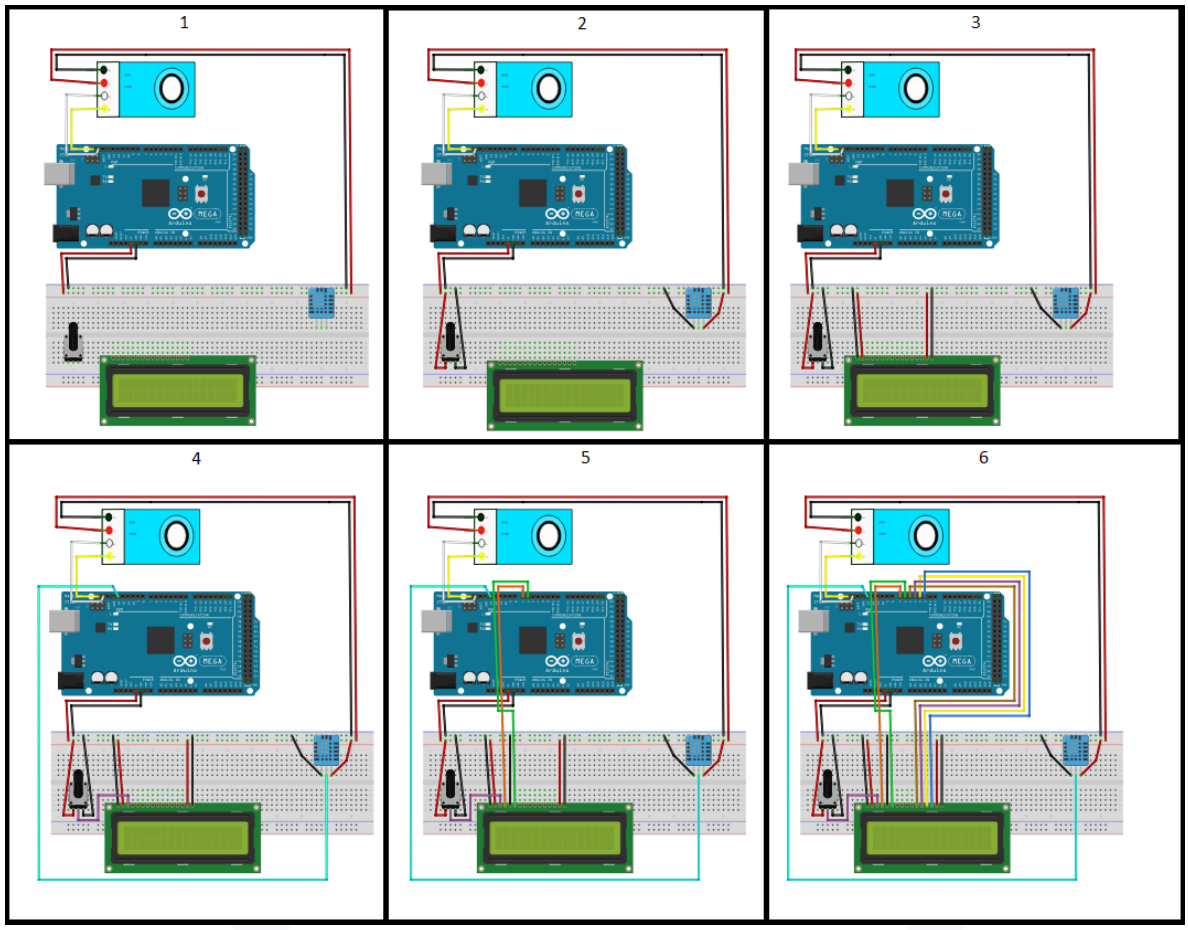
Materiales:

- Arduino Mega 2560
- Protoboard
- Sensor Polvo Grove - HM3301
- Pantalla LCD20x4
- Potenciometro
- DHT11
- Cables pin M-M

Paso 1: Realizar las conexiones del ejercicio pasado (Ejercicio 5.4).

Paso 2: Añadir los cables LCD en el protoboard siguiendo las mismas indicaciones que en el ejercicio 4.2.2 (página 55).

Paso 3: Añadir el potenciómetro al protoboard siguiendo las mismas indicaciones que el ejercicio 4.2.2. (página 56).


1
2
3
4
5
6

Paso 4: Cargar el nuevo código del sensor Grove HM 3301, adjuntado en los materiales digitales del kit "*Clase_6.ino*". Con esto podremos ahora observar los resultados de nuestro kit en la placa Arduino

```
#include "LiquidCrystal.h"
LiquidCrystal lcd(7, 6, 5, 4, 3, 2);

#include "DHT.h"

#define DHTPIN_1 9 // Establecemos la entrada digital en la que tenemos
 // conectada la salida data del sensor.
#define DHTTYPE DHT22 // Definimos el sensor que vamos a usar DHT 11
int buzzer = 8;


DHT dht_1(DHTPIN_1, DHTTYPE); // Configuramos la entrada y el sensor

float tmax=0, tmin=50, hmax=0, hmin=95; // Creamos las variables para los

int delays=5000;
 // valores criticos.

// ----
#include <Seeed_HM330X.h>

#ifdef  ARDUINO_SAMD_VARIANT_COMPLIANCE
    #define SERIAL_OUTPUT SerialUSB
#else
    #define SERIAL_OUTPUT Serial
#endif

HM330X sensor;
uint8_t buf[30];


const char* str[] = {"sensor num: ", "PM1.0 concentration(CF=1,Standard particulate matter,unit:ug/m3): ",
                     "PM2.5 concentration(CF=1,Standard particulate matter,unit:ug/m3): ",
                     "PM10 concentration(CF=1,Standard particulate matter,unit:ug/m3): ",
                     "PM1.0 concentration(Atmospheric environment,unit:ug/m3): ",
                     "PM2.5 concentration(Atmospheric environment,unit:ug/m3): "
```

Paso 5: Verificar en su pantalla los datos medidos por los sensores conectados a su monitor Arduino.

TALLER 7
“Estación de calidad del aire remota”

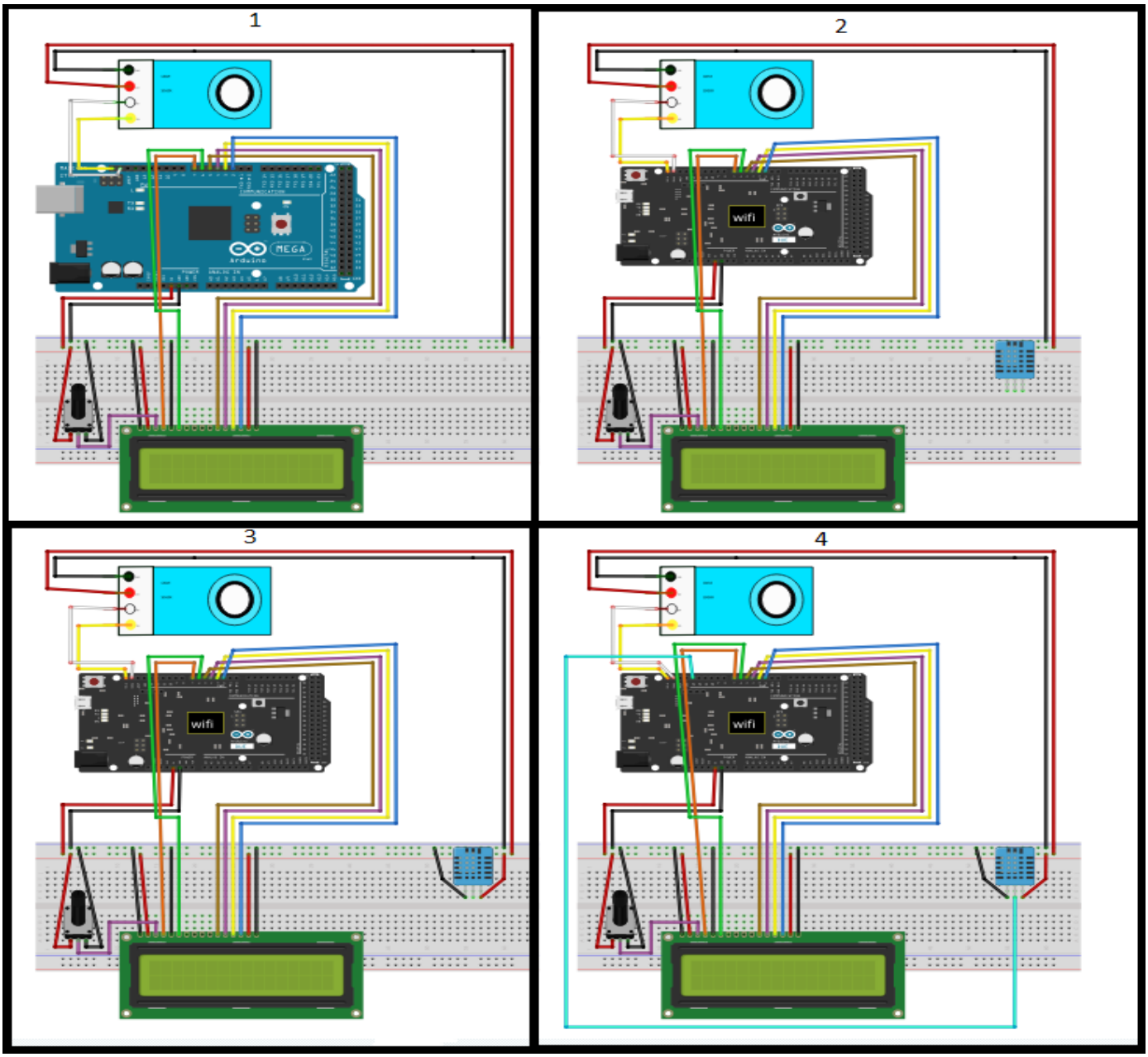
5.6 Transmisión de datos por WiFi WeMos + Grove HM3301

Para el último ejercicio, utilizaremos la placa WeMos y las conexiones del último ejercicio. La placa WeMos tiene la misma distribución de entradas que una placa Arduino común y corriente, por lo que no hay incompatibilidad entre las conexiones que realizamos en el ejercicio anterior. Sin embargo y como vimos al comienzo de este capítulo 5, la placa cuenta con una configuración interna de comunicación mediante pines o interruptores que se encuentran al centro de la placa, para encenderlas o apagarlas es posible utilizar un cable macho para usar de guía.

Materiales

- Placa WeMos
- Protoboard
- Pantalla LCD
- Potenciometro
- Sensor Grove HM3301
- DHT11
- Cables Pin

Paso 1: Realizar las conexiones del ejercicio anterior (Ejercicio 5.5). Esta vez utilizando la placa WeMos.

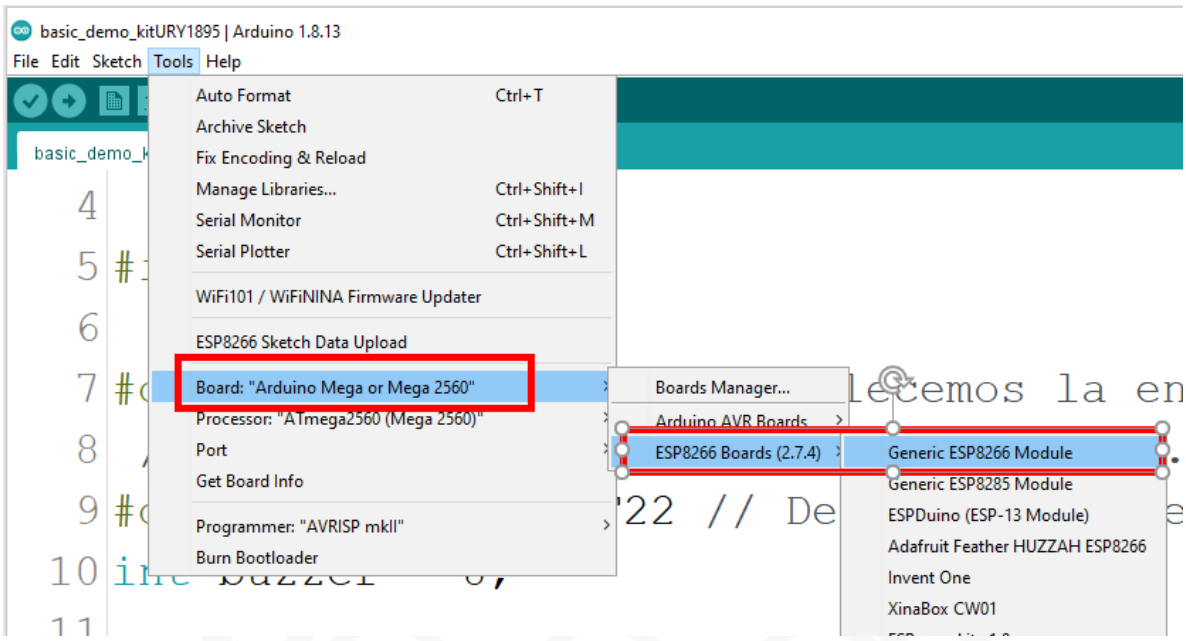


Paso 2: Primero debemos configurar las configuraciones del controlador Arduino UNO. Utilizando la configuración **B.** ubicada en la tabla de combinaciones de pines (Pag. 88) conectamos el WeMos al computador y cargamos el código "*Clase_7_UNO*". Con esto, se podrá medir y observar en la pantalla LCD: Temperatura, Humedad relativa y Material Particulado MP 10 y MP 2.5.

```
#include <Wire.h>
#include <Seeed_HM330X.h>
#ifdef  ARDUINO_SAMD_VARIANT_COMPLIANCE
    #define SERIAL_OUTPUT SerialUSB
#else
    #define SERIAL_OUTPUT Serial
#endif
/*#define I2C_SCL 12                    //Connect SCL pin of BMP180 to GPIO12(D6) of Nodemcu
#define I2C_SDA 13                    //Connect SDA pin of BMP180 to GPIO13(D7) of Nodemcu */
HM330X sensor;
uint8_t buf[30];
const char* str[] = {"sensor num: ", "PM1.0 concentration(CF=1,Standard particulate matter,unit:ug/m3): ",
                     "PM2.5 concentration(CF=1,Standard particulate matter,unit:ug/m3): ",
                     "PM10 concentration(CF=1,Standard particulate matter,unit:ug/m3): ",
                     "PM1.0 concentration(Atmospheric environment,unit:ug/m3): ",
                     "PM2.5 concentration(Atmospheric environment,unit:ug/m3): ",
                     "PM10 concentration(Atmospheric environment,unit:ug/m3): ",
                    };
int PM1_0 = 0, PM2_5 = 0, PM10 = 0;

/*#include <SoftwareSerial.h>
SoftwareSerial mySerial(10, 11); // RX, TX */

String PMs = "";
void setup() {
Serial.begin(115200);
//mySerial.begin(115200);
delay(100);
  Wire.begin();
  SERIAL_OUTPUT.println("Serial start");
    if (sensor.init()) {
        SERIAL_OUTPUT.println("HM330X init failed!!!");
        while (1);
    }
}
void loop() {
PMs = "";
if (sensor.read_sensor_value(buf, 29)) {
        SERIAL_OUTPUT.println("HM330X read result failed!!!");
    }
//parse_result_value(buf);
```

Paso 3: Con la primera parte completa, procedemos a configurar las configuraciones del controlador para poder conectarla a una red WiFi. Para esto, desconectamos el WeMos del computador y procedemos a mover los interruptores según la configuración **C.** de la tabla de combinaciones de pines (Pag. 88). Ahora debemos abrir el archivo "*Clase_7_WiFi.ino*" que contiene el código que modificaremos.

Paso 4: Con el código abierto, debemos cambiar el tipo de controlador con el que trabajaremos. Debemos modificar la placa para trabajar con un módulo ESP8266, para eso seleccionaremos las siguientes opciones del software: Pestaña Herramientas > Controladores/Placas > ESP8266 > **"Generic ESP8266 Module"** .



Paso 5: Ahora se debe configurar líneas del código de trabajo para que la placa pueda conectarse correctamente a una red WiFi. La línea 13 “ssid” debe contener el nombre exacto de su red WiFi, respetando mayúsculas y caracteres especiales, a su vez, la línea 14 “Password”, debe contener la contraseña de la red. La placa WeMos buscará automáticamente una red que cumpla con estos parámetros y se conectará automáticamente. Estos parámetros deben ser cambiados cada vez que se mueva el Arduino de una red WiFi a otra red.

write_data §

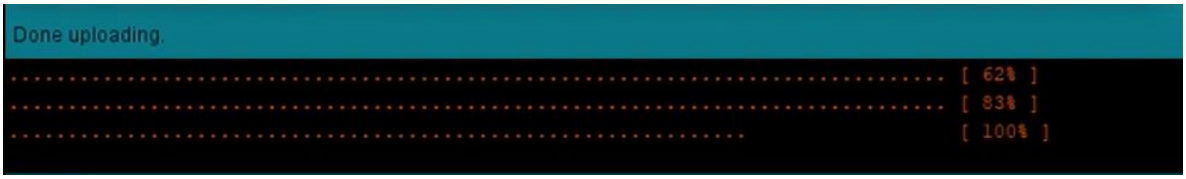
```
1  //-----------------------------------------------
2  // Author: Geolab
3  // Email: geolab@uaysen.cl
4  // Publish date: 02-Mayo-2020
5  // Description: This code for demonstration send data from ESP8266 into Google Spreads
6  // update ssid, password and GAS_ID
7  //-----------------------------------------------
8  #include <ESP8266WiFi.h>
9  #include <WiFiClientSecure.h>
10 #include <SoftwareSerial.h>
11
12 //SoftwareSerial mySerial(D5, D6); // RX, TX
13 const char* ssid = "Comunidad_5G";     // name of your wifi network!!!!!!!!!!!!!!!!!!!
14 const char* password = "micontraseña123";
15 const char* host = "script.google.com";
16 const int httpsPort = 443;
17 // Use WiFiClientSecure class to create TLS connection
18 WiFiClientSecure client;
19 // SHA1 fingerprint of the certificate, don't care with your GAS service
```

Paso 6: Para enviar los datos a la red, debemos indicarle el lugar de destino de los datos a la placa WeMos, para esto, nosotros trabajaremos utilizando una planilla de datos (tipo Excel) de Google Drive. Para efectos de este ejercicio, utilizaremos una planilla pre-creada, por lo que debemos modificar una línea del código, que será la "llave" de acceso a la planilla. Esta línea debe ser reemplazada por su correspondiente llave que pueden encontrar en el material digital.

write_data §

```
// Publish date: 02-Mayo-2020
// Description: This code for demonstration send data from ESP8266 into Google Spreadsheet
// update ssid, password and GAS_ID
//-----------------------------------------------
#include <ESP8266WiFi.h>
#include <WiFiClientSecure.h>
#include <SoftwareSerial.h>

//SoftwareSerial mySerial(D5, D6); // RX, TX
const char* ssid = "Comunidad_5G";     // name of your wifi network!!!!!!!!!!!!!!!!!!!!!!!!!
const char* password = "micontraseña123";
const char* host = "script.google.com";
const int httpsPort = 443;
// Use WiFiClientSecure class to create TLS connection
WiFiClientSecure client;
// SHA1 fingerprint of the certificate, don't care with your GAS service
const char* fingerprint = "5d 7b 1c 8b 6f 73 e4 d6 c5 1d a5 dc 8b 0f 4d 01 2f b3 1b bf";

String GAS_ID =   "Llave para su planilla";
float concentration1;
float concentration2;
float concentration2_ugm3;
float concentration1_ugm3;
```

Paso 7: Con los pasos 5 y 6 realizados, conectamos el Arduino y cargamos el código al WeMos. Comenzaremos a observar el progreso de la subida del código y veremos un mensaje en porcentajes que nos indicará si ya está listo.

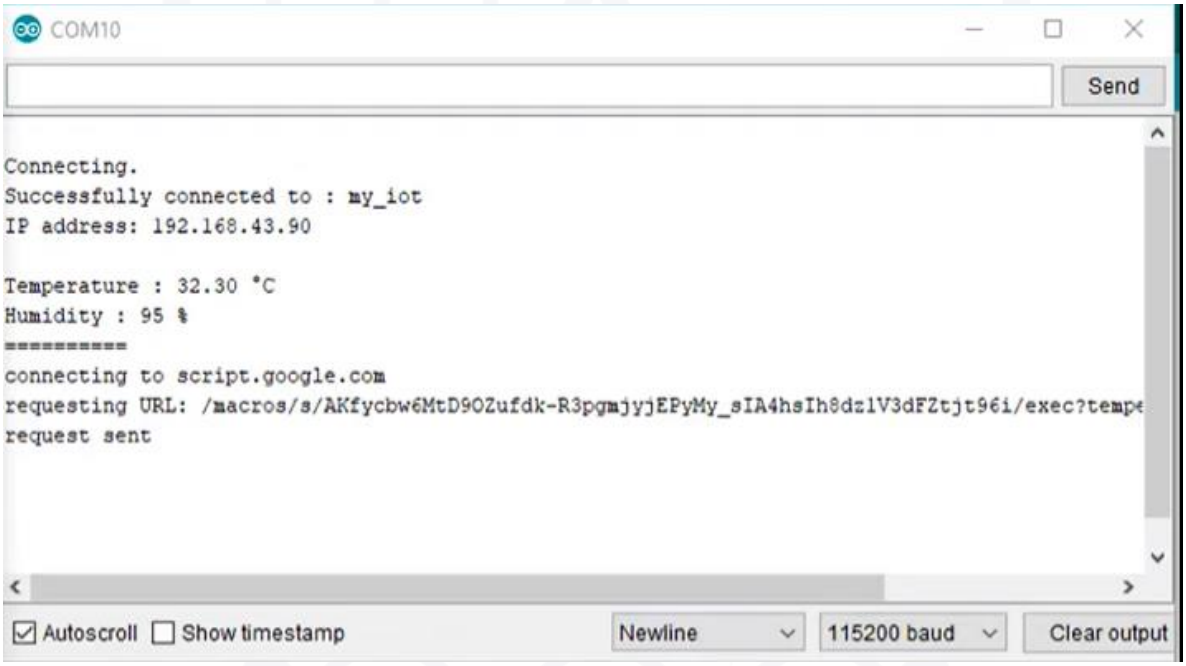


Paso 8: Una ves que el código se haya cargado exitosamente en la placa, desconectamos el Arduino del computador y procederemos a mover los interruptores/pines de la placa según la configuración

E. (pag 88). Esta configuración permitirá comenzar a medir datos, conectarse a la red WiFi y enviar los datos a la planilla.

Paso 9: Con los pines configurados correctamente, conectamos nuevamente el Arduino y seleccionamos el monitor serie. Configuramos la ventana a 115200 baudios y deberíamos observar que la conexión al WiFi haya sido exitosa y veremos como se envían los datos a través de la llave.

En caso de que la conexión haya fallado, repetir nuevamente los pasos 5 y 6.



Paso 10: Con esto logrado, periódicamente nuestra estación, comenzará a registrar datos en línea que podrán ser observados desde cualquier parte del mundo. Solo necesitamos que el Arduino cuente con conexión a la red WiFi y energía.

Date	Time	MP2.5	MP10
18/06/2021	3:23:35	10	10
18/06/2021	3:45:21	0.00	0.00
18/06/2021	3:45:28	0.00	0.00
18/06/2021	3:45:37	0.00	0.00
18/06/2021	3:45:42	0.00	0.00
18/06/2021	3:46:10	0.00	0.00
18/06/2021	3:46:30	0.00	0.00
18/06/2021	3:47:01	0.00	0.00
18/06/2021	3:47:08	0.00	0.00
18/06/2021	3:47:18	0.00	0.00
18/06/2021	3:47:21	0.00	0.00
18/06/2021	3:47:27	0.00	0.00
18/06/2021	3:47:37	0.00	0.00
18/06/2021	3:47:47	0.00	0.00
18/06/2021	3:47:57	0.00	0.00
18/06/2021	3:48:07	0.00	0.00
18/06/2021	3:48:17	0.00	0.00
18/06/2021	3:48:27	0.00	0.00
18/06/2021	3:48:37	0.00	0.00