BLOQUE 3:
MARIETA MERCHÁN VIZCAÍNO

3.1 - Lectura de señales analógicas

Las señales digitales sólo tienen dos estados opuestos: 1 o 0.

Pero el mundo real no es digital, y hay estados intermedios, por lo quelas señales digitales no se pueden utilizar.

En su lugar, se deben utilizar **señales analógicas.**

- Si usamos un **sensor de luz,** podemos obtener muchos valores diferentes que indican cuánta luz hay en la habitación, en lugar de un simple oscuro / brillante.
- Con el panel de control, podemos obtener valores analógicos de los pines analógicos. Cuando no hay voltaje en el pin, la lectura es 0. Para leer estos valores necesitará **analogRead ()** en lugar de la función **digitalRead ()**

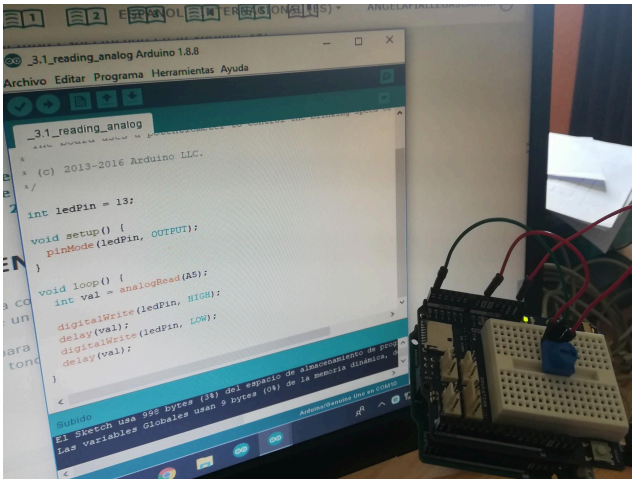
Para entenderlas mejor, necesitamos introducir el potenciómetro. Un **potenciómetro** es un botón que puede girar para controlar algo.
Cuando los dos pines externos del potenciómetro están conectados a GND y Power respectivamente, puede usar la perilla para controlar la cantidad de voltaje que se aplica al pin central.

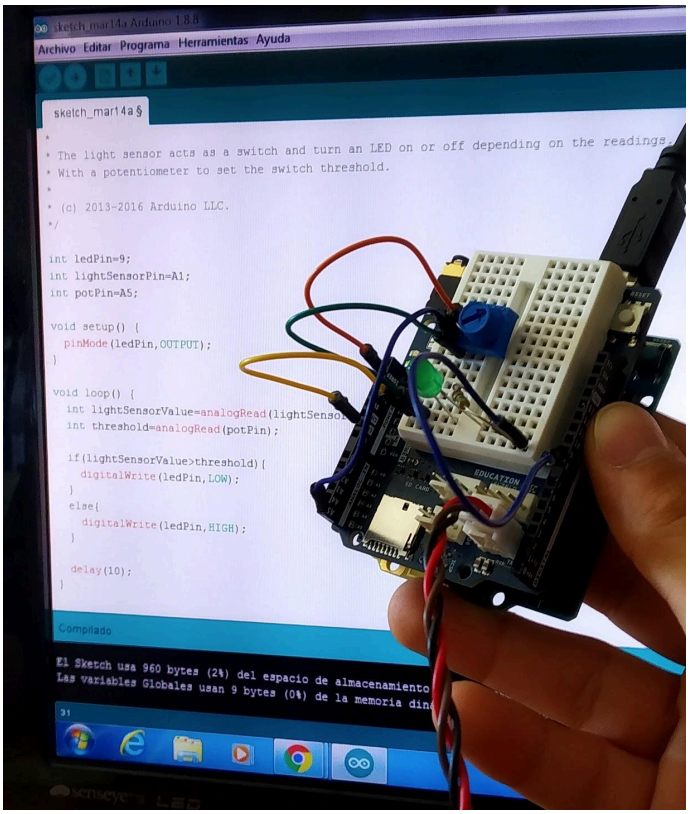
Comandos

AnalogRead (pinNumber) : lee el valor analógico en el pinNumber analógico . Devuelve un valor entre 0 y 1023.
Los pines analógicos solo se pueden usar como entradas, por lo que no es necesario usar pinMode () en la configuración () .

Cómo funciona

Se declara la variable ledPin .
En la configuración () , el pin 13 (o número donde se ha colocado) se configura como una salida.
En el bucle (), la variable val se declara para mantener el valor analógico de lectura en el pin A5 (o donde se ha colocado).
El LED de a bordo está encendido.
El programa se detiene por tantos milisegundos como el valor de val .
El LED de a bordo está apagado.
El programa se detiene de nuevo durante tantos milisegundos como el valor de val .
El bucle () continúa ejecutándose, lo que significa que el valor analógico en el pin A5 se verifica continuamente.





3.2 - Escribiendo señales analógicas

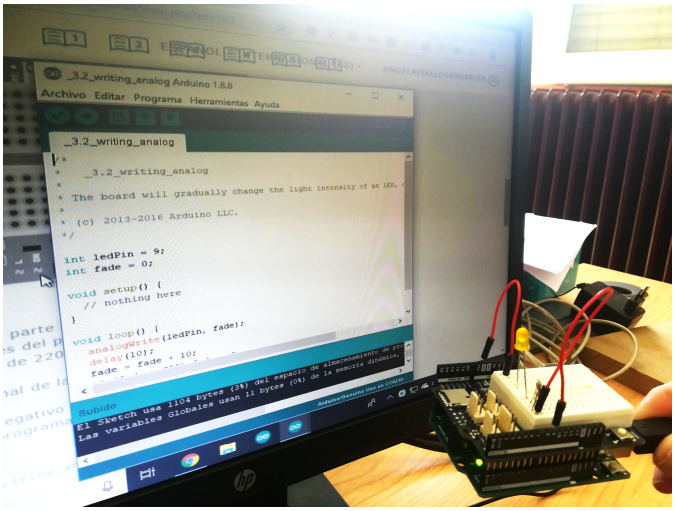
La placa utiliza pines de modulación de ancho de pulso (o **PWM)** para escribir señales analógicas.

Cuando se usa un pin digital normal para escribir ALTO o BAJO , su salida es la tensión máxima o la tensión mínima. Pero los pines PWM tienen una capacidad diferente: puede usarlos para escribir un nivel de voltaje en cualquier lugar entre el máximo y el mínimo.
Con esto, se puede atenuar gradualmente un LED.
Para usar esta habilidad especial de pines PWM, necesitará usar la función de analogWrite () .Se necesitan dos **parámetros**: el número del pin PWM y el nivel de salida. El nivel de salida es un número entre 0 y 255. 0 es equivalente a LOW digital y 255 es equivalente a HIGH digital .

Cómo funciona

Se declara que la variable ledPin tiene el valor n (el número del pin PWM al que está conectado el LED).
Se declara que la variable fade mantiene el valor 0. Esta variable se utilizará para configurar la intensidad de la luz del LED.
Nada se hace en setup () .
En loop () , se escribe un valor analógico en el pin 9. La primera vez que se ejecuta loop () , el valor es 0 (fade = 0), lo que apaga el LED.

El programa se detiene por 10 milisegundos.
El desvanecimiento se incrementa con 10.
Se comprueba que el desvanecimiento es más de 255.
La primera vez que se ejecuta loop () , fade no es más de 255, por lo que el programa salta al principio de loop () nuevamente.
Se escribe un nuevo valor analógico en el pin 9. Esta vez, ese valor es 10 (fade = 10), lo que hace que el LED se ilumine con una intensidad muy baja.
El programa se detiene por 10 milisegundos.
El desvanecimiento se incrementa de nuevo con 10, lo que es igual a 20.
Fade todavía no es más de 255, por lo que el programa salta al principio de loop () .
El 26.o bucle de tiempo () se ejecuta, el desvanecimiento es igual a 250, por lo que el LED se enciende a una intensidad de luz casi total.
Cuando el desvanecimiento se incremente con 10 nuevamente será igual a 260.
Cuando la sentencia if se ejecuta este tiempo, la decoloración es más de 255 por lo que este tiempo de desvanecimiento se le asigna el valor 0 de nuevo.
Bucle () continúa en bucle.



3.3 - Sensor de luz

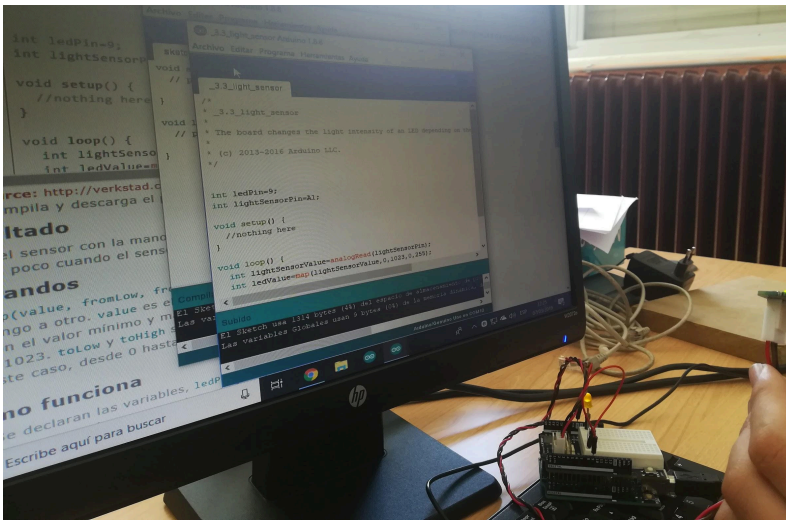
El sensor de luz,es un sensor analógico que detecta la cantidad de luz presente. Dependiendo de la cantidad de luz que brille sobre él, el sensor devolverá un valor analógico específico.

Cómo funciona

Se declaran dos variables, ledPin y lightSensorPin .
No pasa nada en la configuración () .

En loop () , la variable lightSensorValue se declara para mantener el valor analógico de lectura en el pin A1. Este valor puede oscilar entre 0 y 1023, que es un rango demasiado grande para escribir un valor analógico en un pin.
La variable ledValue se declara para mantener el valor reasignado de lightSensorValue . Este valor oscilará entre 0 y 255. Cuando lightSensorValue = 0, ledValue = 0, cuando lightSensorValue = 1023, ledValue = 255, cuando lightSensorValue = 512, ledValue = 127, etc.
Un valor analógico se escribe en el pin 9 usando ledValue .
El programa se detiene por 10 milisegundos.
bucle () continúa en bucle.

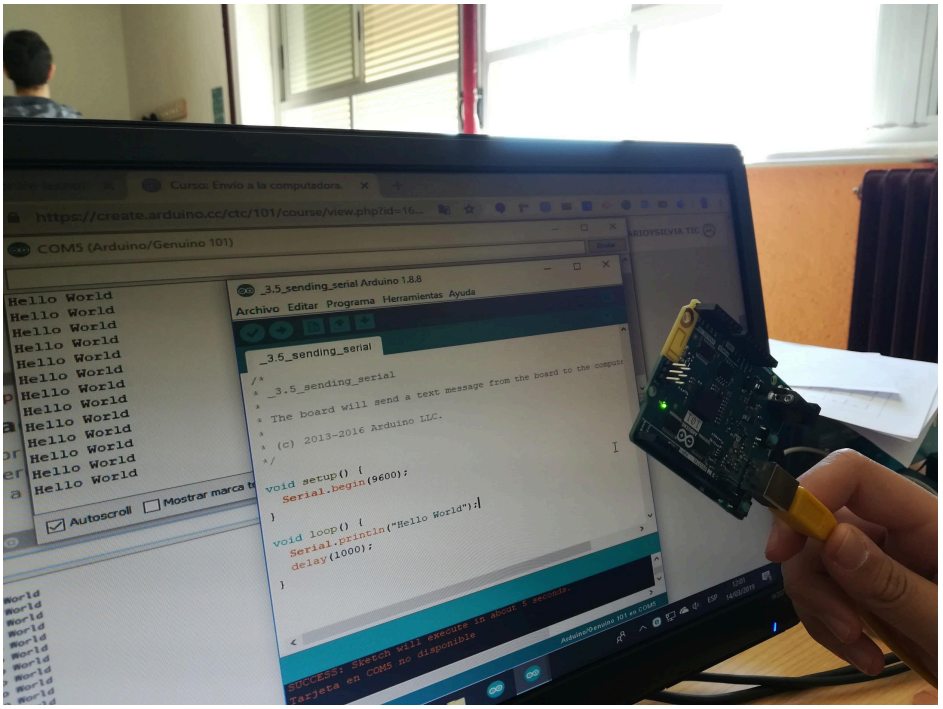
Cómo funciona



Se declaran tres variables, ledPin , lightSensorPin y potPin .potPin es mantener el número del pin analógico al que está conectado el potenciómetro.
Debido a que está utilizando el LED como un actuador digital en este ejemplo, el pin 9 se configura como una salida en la configuración () .
En loop () , se declara que la variable lightSensorValue mantiene el valor analógico de lectura en el pin A1.
Se declara que el umbral variable mantiene el valor analógico leído del pin A5.
Si lightSensorValue es mayor que el umbral , el LED se apaga.
Si lightSensorValue no es mayor que el umbral , el LED se enciende.

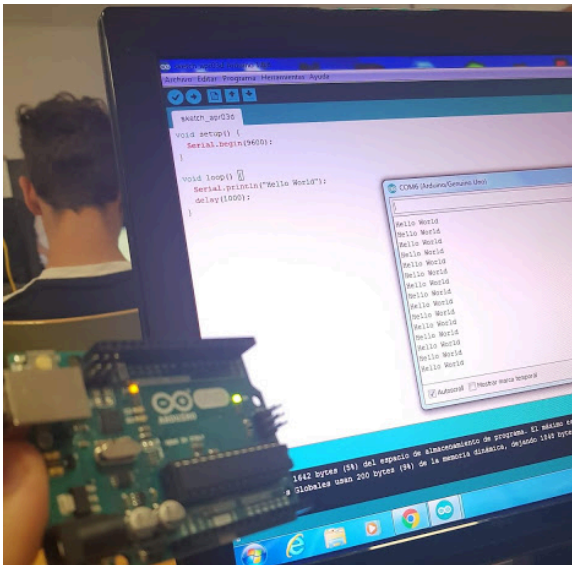
3.4 - Puerto serial

Los tableros de control se conectan al ordenador con un cable USB. La forma en que las tarjetas "hablan" con el ordenador es a través de algo que se llama un **puerto serie.**
Los puertos serie pueden usarse para intercambiar datos bastante complicados con el ordenador. En lugar de las señales digitales o analógicas, puede enviar o recibir texto.
Es importante que el remitente y el receptor se comuniquen a la misma velocidad. De lo contrario, será como dos personas tratando de comunicarse, una que habla en chino y la otra en español. La velocidad se llama velocidad en baudios y mide bits por segundo.



3.5 - Enviando a la computadora

Para enviar un mensaje desde el panel de control a la computadora, necesitará 2 comandos: **Serial.begin () y Serial.println () o Serial.print ()** .
Serial.begin (speed) : Inicializa la comunicación serial. La velocidad en baudios, o bits por segundo, se ajusta con la velocidad .
Serial.println ("mensaje") : imprime el mensaje en el puerto serie. El siguiente mensaje impreso comenzará en una nueva línea.
Serial.print ("mensaje") : imprime el mensaje en el puerto serie. El siguiente mensaje comenzará justo después del anterior, en la misma línea.



Cómo funciona

En la configuración (), la comunicación en serie se inicializa con la velocidad de 9600 bits por segundo.
En loop () , el mensaje "Hello World" se imprime en el puerto serie.
El programa se detiene por 1000 milisegundos.
bucle () continúa en bucle.

Impresión de los valores de la luz

Una cosa importante para la que necesitaremos comunicación en serie es la solución de problemas de nuestros programas.Podemos enviar valores que cambian con el tiempo.
Esto es útil cuando queremos usar un sensor analógico pero no sabe exactamente qué valores nos darán.

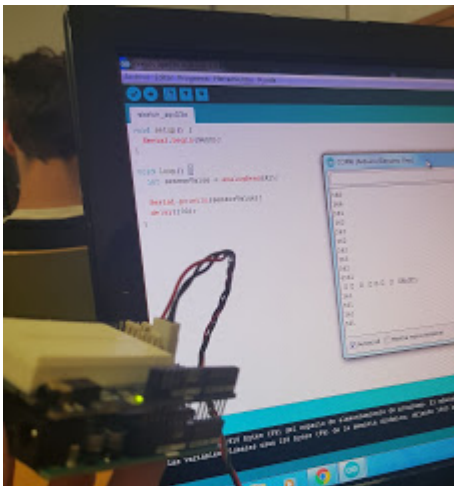
Cómo funciona

En la configuración (), la comunicación en serie se inicializa con la velocidad de 9600 bits por segundo.
En loop (), la variable sensorValue se declara para mantener el valor analógico leído en el pin A1.
El valor de sensorValue se imprime en el puerto serie.
El programa se detiene por 100 milisegundos. Esta pausa se hace solo para facilitar la lectura de los datos impresos en el monitor de serie.
bucle () continúa en bucle.

➔ Nota: recuerde que el conector del módulo analógico, A1, está realmente conectado a A1, por lo que al usar el conector de tres clavijas, no se puede usar ningún otro sensor en ese pin analógico.

3.6 - Recibiendo de la computadora

Para recibir datos a través del puerto serie, necesita dos comandos: Serial.available () y Serial.read () .



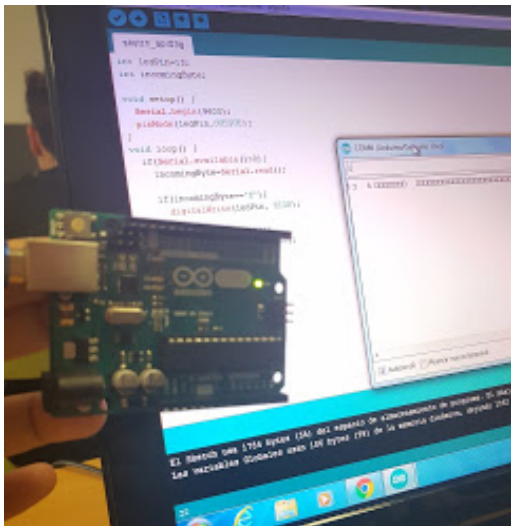
Cómo funciona

Se declaran las variables ledPin y incomingByte . incomingByte se utilizará para mantener los datos en serie entrantes.
En la configuración () , la comunicación en serie se inicializa con la velocidad de 9600 bits por segundo.
Pin 13 se configura como una salida.
Solo desea leer desde el puerto serie si hay datos que llegan. Por lo tanto, lo primero que sucede en loop () es verificar si el número de bytes en espera de lectura es mayor que 0.
Si el número de bytes en espera no es mayor que 0, el programa omite el código dentro de los corchetes. Como no hay otro código que ejecutar después de esta instrucción if, el programa salta al principio de loop () .
Si el número de bytes es mayor que 0, los datos en serie se leen y se almacenan en los bytes entrantes .
Si entranteByte es igual a 'H', el LED de a bordo se enciende.
Si entranteByte es igual a 'L', el LED de a bordo se apaga.
bucle () continúa en bucle.

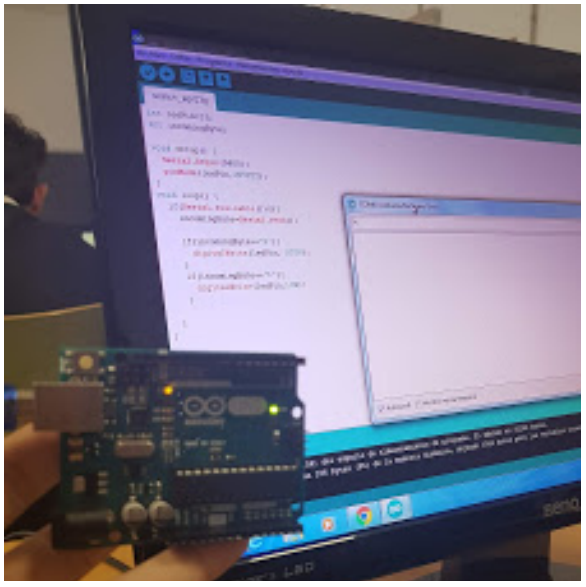
3.7 RECIBIENDO DE LA COMPUTADORA

En la parte superior del monitor de serie hay un cuadro de entrada de texto con un botón de envío al lado. Debes escribir 'H' en el cuadro de entrada y hacer clic en el botón enviar. El LED de a bordo ahora debería encenderse. Del mismo modo, debe enviar 'L' y el LED debe apagarse. Tenga en cuenta que solo funcionará con letras mayúsculas .

CON L:



CON H:



En la parte superior del monitor de serie hay un cuadro de entrada de texto con un botón de envío al lado. Debes escribir 'H' en el cuadro de entrada y hacer clic en el botón enviar. El LED de a bordo ahora debería encenderse. Del mismo modo, debe enviar 'L' y el LED debe apagarse. Tenga en cuenta que solo funcionará con letras mayúsculas.

PROYECTO (P.V.O)