

## Laboratorio de Sensores y Actuadores

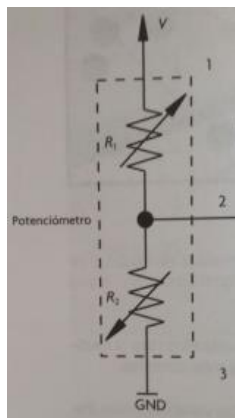
### Practica 2

#### Potenciómetro

#### Objetivo

#### Marco Teórico

Un Potenciometro es un transductor entre la posicion de un objeto, ya sea lineal o angular, y un cambio de resistencia. Este tipo de elementos resistivos se utilizan normalmente con un voltaje de CD. Constan de 3 terminales, una en cada extremo del material, y una tercera terminal que recorre el cuerpo del elemento resistivo, de tal manera que la resistencia entre la terminal movil y cada una de las terminales fijas varia cuando el elemento movil cambia de posicion; asi, cuando una resistencia disminuye, la otra nescersariamente aumenta.



Para lograr una correlación entre voltaje de salida y la posición de un determinado objeto, suele ser común acoplar mecánicamente el elemento móvil de potenciometro al objeto; así, cuando el elemento móvil del potenciometro s encuentre en uno de los extremos, el voltaje a la salida será máximo, de otra manera cuando el elemento móvil este en la parte proximal, el voltaje a la salida será mínimo. Esta correlación se puede calcular usando la siguiente formula:

$$V_{out} = \frac{R_b}{R_a + R_b} V_T$$

Donde la resistencia  $R_a$  estará dada entre el nodo de referencia a tierra y el nodo de, mientras que la resistencia  $R_b$  estará dada entre el nodo de  $V$  y el nodo de  $V_T$ . Al momento de utilizar este tipo de instrumentación para determinar la posición de un objeto, se debe tener en cuenta cuales son las características del potenciometro, ya que para estas aplicaciones lo más recomendable es utilizar un potenciometro cuyo cambio a la salida sea lineal. Cabe hacer la distinción aquí que cuando se habla de linealidad en el comportamiento del sensor se hace referencia a que la entrada sea linealmente proporcional a la salida y no al tipo de movimiento que presenta el elemento móvil. Que tan lineal será el comportamiento de cada potenciometro dependerá, en mayor medida, del tipo de material con el que esté constituido.

## Tipos de potenciómetros

A continuación se muestran en la figura 1 y 2 algunos de los diferentes tipos de potenciómetros.



Figure **¡Error! Secuencia no especificada.** Potenciómetro lineal dual



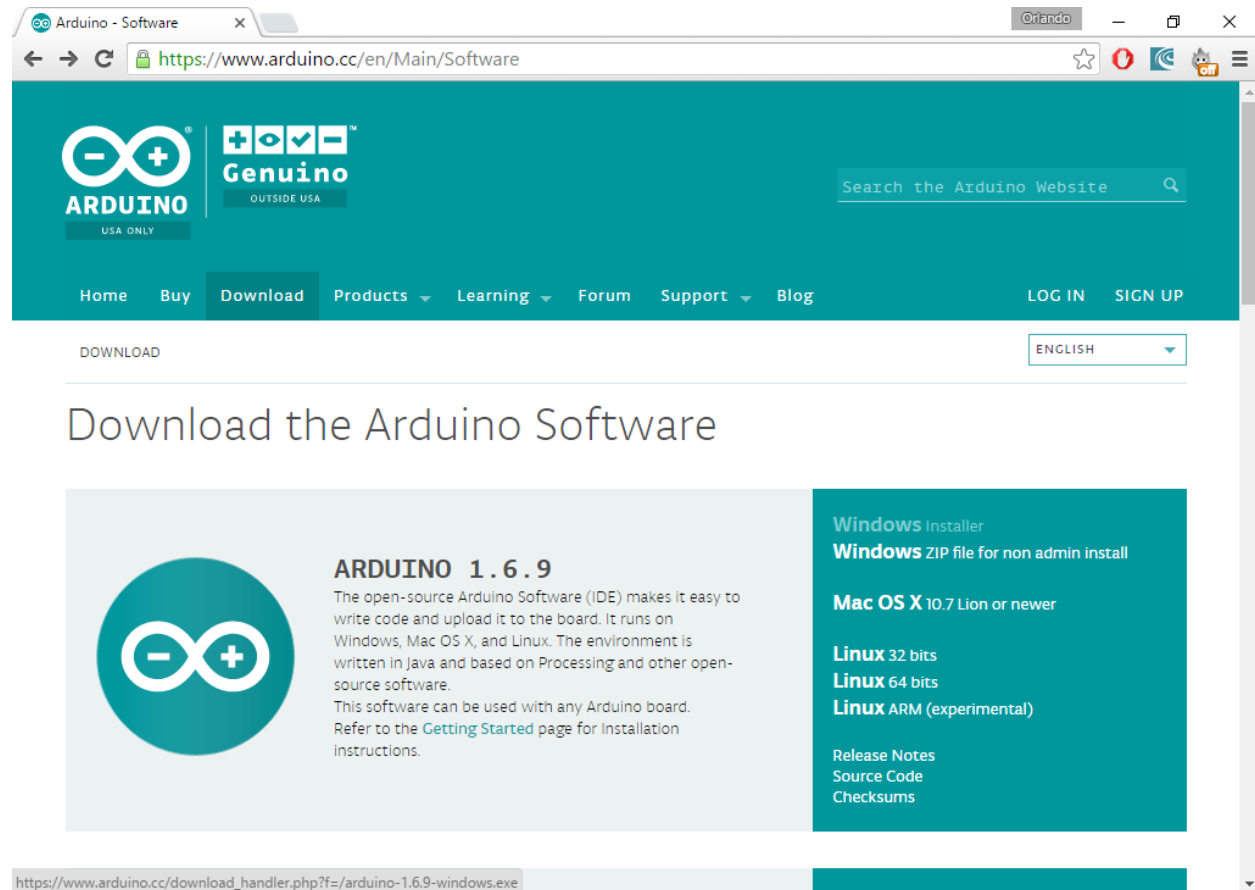
Figure 2 Potenciómetro rotatorio dual

## Desarrollo

### Arduino

Para esta y las demás practicas usaremos un Arduino Uno por ser el mas comercial, sin embargo todas las practicas pueden ser realizadas con cualquier otra versión de arduino.

Para descargar el IDE de Arduino nos dirigimos a su pagina oficial y nos vamos a Download. Una vez ahí seleccionamos Windows Installer para instalar la versión mas nueva(1.6.9 cuando se realizo esta practica).



The screenshot shows the Arduino Software download page in a web browser. The browser's address bar displays the URL <https://www.arduino.cc/en/Main/Software>. The page features the Arduino logo and the text "Genuino OUTSIDE USA". A search bar is present with the placeholder text "Search the Arduino Website". The navigation menu includes links for Home, Buy, Download, Products, Learning, Forum, Support, and Blog. The "Download" link is highlighted. Below the navigation menu, there is a "DOWNLOAD" section with a language selector set to "ENGLISH". The main heading is "Download the Arduino Software". The page is divided into two columns. The left column features the Arduino logo and the text "ARDUINO 1.6.9". The right column lists the available download options: "Windows Installer", "Windows ZIP file for non admin install", "Mac OS X 10.7 Lion or newer", "Linux 32 bits", "Linux 64 bits", and "Linux ARM (experimental)". Below these options are links for "Release Notes", "Source Code", and "Checksums". At the bottom of the page, the URL [https://www.arduino.cc/download\\_handler.php?f=/arduino-1.6.9-windows.exe](https://www.arduino.cc/download_handler.php?f=/arduino-1.6.9-windows.exe) is displayed.

Arduino - Software

Orlando

<https://www.arduino.cc/en/Main/Software>

Search the Arduino Website


Home Buy Download Products Learning Forum Support Blog

LOG IN SIGN UP

DOWNLOAD

ENGLISH

## Download the Arduino Software



**ARDUINO 1.6.9**

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.

This software can be used with any Arduino board. Refer to the [Getting Started](#) page for installation instructions.

Windows Installer

Windows ZIP file for non admin install

Mac OS X 10.7 Lion or newer

Linux 32 bits

Linux 64 bits

Linux ARM (experimental)

Release Notes

Source Code

Checksums

[https://www.arduino.cc/download\\_handler.php?f=/arduino-1.6.9-windows.exe](https://www.arduino.cc/download_handler.php?f=/arduino-1.6.9-windows.exe)

## Ejemplo 1.

Una vez instalado arduino abrimos el ejemplo 1(Se encuentra en la carpeta arduino > ejemplo 1 > ejemplo1.ino).

```
ejemplo1 Arduino 1.6.9
Archivo Editar Programa Herramientas Ayuda

/*
 * Ejemplo1 de la practical de lab. de Sensores y actuadores.
 */

// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  float value = (float)sensorValue*5/1023;
  // print out the value you read:
  Serial.print("Voltaje: ");
  Serial.print(value);
  Serial.println("V");
  delay(10);      // delay in between reads for stability
}
```

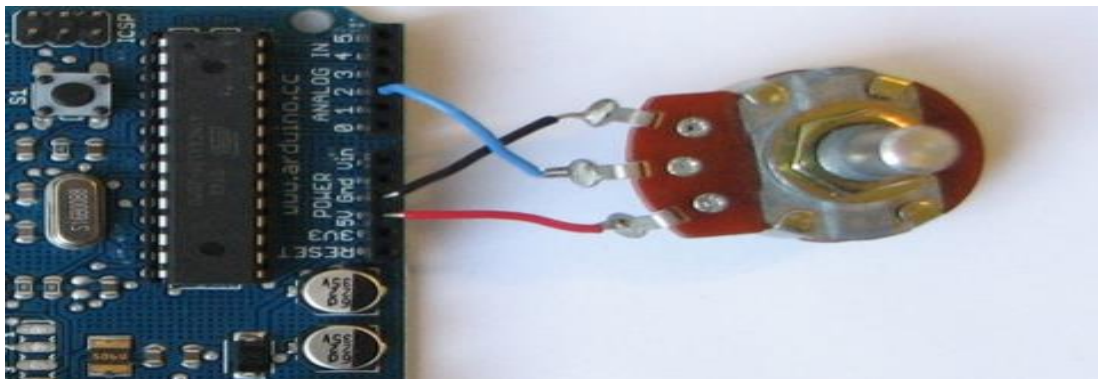
Compilado

El Sketch usa 4,020 bytes (12%) del espacio de almacenamiento de programa. El máximo es 32,256 bytes.  
Las variables Globales usan 212 bytes (10%) de la memoria dinámica, dejando 1,836 bytes para las variables locales. El máximo es 2,048 bytes.

16 Arduino/Genuino Uno en COM3

En este ejemplo vamos a leer los valores arrojados por el potenciómetro en el arduino y los imprimiremos en pantalla para ver la variación en tiempo real de la diferencia de potencial generada por el potenciómetro.

Conectamos las 3 terminales del potenciómetro directamente al arduino. La primera va a tierra(GND), la segunda terminal va conectada a cualquier entrada analógica del arduino para este caso utilizaremos el pin 2 de entradas analógicas(En código se define como A2), y la tercer terminal del potenciómetro va 5V.



El código del ejemplo es el siguiente:

```
// La funcion void se ejecuta una sola vez al correr el codigo:
void setup() {
  // Inicializa la comunicacion Serial a una velocidad de 9600 bits por segundo:
  Serial.begin(9600);
}

// La funcion loop se ejecuta infinitamente despues de que se ejecuto la funcion setup:
void loop() {
  // Lee el valor en la entrada analogica :
  int sensorValue = analogRead(A2);
  // Convertimos los valores leidos de 0-1023 a 0-5
  float value = (float)sensorValue*5/1023;
  // Imprimimos los valores leido como voltaje:
  Serial.print("Voltaje: ");
  Serial.print(value);
  Serial.println("V");
  delay(10);      // esperamos 10 ms antes de volver a ejecutar el loop
}
```

Arduino no es un lenguaje, si no que usa lenguaje C con librerías específicas para la plataforma arduino.

Los programas en arduino se llaman sketch y al momento compilarlos la computadora carga ese sketch al arduino, por lo que si cargas un sketch en el arduino y lo conectas con un adaptador directamente a la corriente eléctrica o con cualquier otra fuente de CD de entre 5V y 9V seguirá funcionando.

Un sketch en arduino debe tener estas dos funciones para poder compilar correctamente:

-void setup: Se ejecuta solo una vez cada que se enciende o reinicia el arduino. En esta función se deben de configurar e inilizar todas las variables que vayamos a usar

```
void setup() {
  // Inicializa la comunicacion Serial a una velocidad de 9600 bits por segundo:
  Serial.begin(9600);
}
```

-void loop: Es el análogo al método main de un programa en C, es un ciclo(loop) infinito que se ejecuta una vez que termine de ejecutarse la función setup, en esta función escribiremos toda la lógica del programa.

```
// La funcion loop se ejecuta infinitamente despues de que se ejecuto la funcion setup:
void loop() {
  // Lee el valor en la entrada analogica :
  int sensorValue = analogRead(A2);
  // Convertimos los valores leidos de 0-1023 a 0-5
  float value = (float)sensorValue*5/1023;
  // Imprimimos los valores leido como voltaje:
  Serial.print("Voltaje: ");
  Serial.print(value);
  Serial.println("V");
  delay(10);      // esperamos 10 ms antes de volver a ejecutar el loop
}
```

Primero definimos una variable de tipo int para guardar los valores leídos por el sensor.

```
// Lee el valor en la entrada analogica
int sensorValue = analogRead(A2);
```

En esta línea de código definimos una variable de nombre sensorValue de tipo int(numero entero) y le asignamos

lo que el arduino este leyendo en la entrada analógica 2 que es donde esta conectada la segunda terminal del potenciómetro utilizando la función analogRead, básicamente lo que le estaremos diciendo al arduino es “Crea una variable de tipo entero que se llame sensorValue, lee el pin analogico 2 del arduino y guarda este valor en la variable sensorValue”.

```
// Convertimos los valores leídos de 0-1023 a 0-5
float value = (float)sensorValue*5/1023;
```

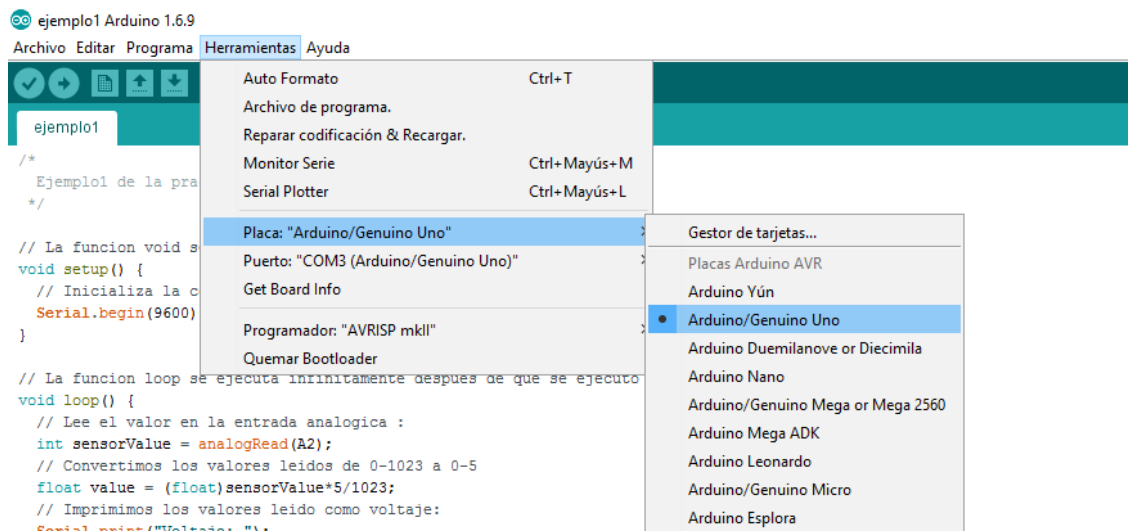
Los valores leídos varían de 0 a 1023, para una mejor lectura se convierten a valores entre 0 y 5, para esto el valor leído por el sensor se divide entre 1023 y se multiplica por 5 y se guarda en una variable de tipo float(de punto flotante o decimal) a la que llamamos value, como los valores que arroja el pot son de tipo int se usa un cast que es una conversión de un tipo a otro “(float)sensorValue\*5/1023”.

```
Serial.print("Voltaje: ");
Serial.print(value);
Serial.println("V");
```

Ahora todo lo que falta es imprimir los valores leídos por el potenciómetro, el comando Serial.print se encarga de imprimir en el monitor serial lo que le pasemos como parámetro, el comando Serial.println hace lo mismo pero termina con un salto de línea.

Al final se encuentra la línea de código “delay(10);”, el comando delay hace una pausa durante el tiempo en milisegundos que le pasemos como parámetro para este caso 10, se recomienda siempre dejar un pequeño delay al final del código para no saturar al arduino.

Para cargar el sketch al arduino primero hay que asegurarnos de que esta seleccionado el arduino que estamos usando en este caso el UNO y el puerto correspondiente.





## Ejemplo 2.

Abrimos el ejemplo 2, se encuentra en la carpeta de arduino de la practica 2.

```
ejemplo2 Arduino 1.6.9
Archivo Editar Programa Herramientas Ayuda

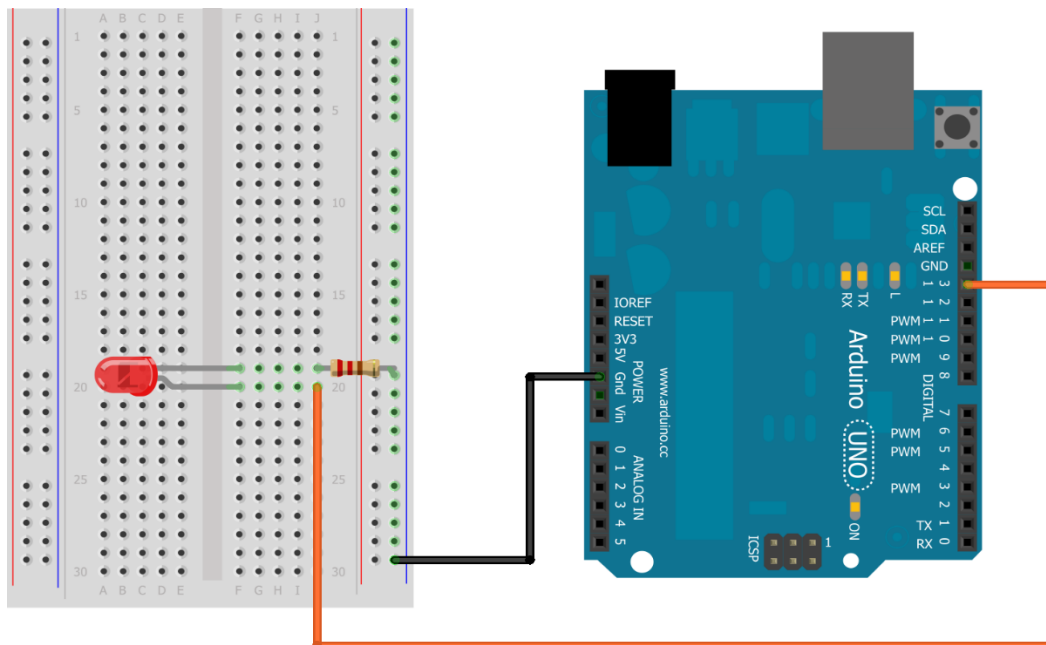
ejemplo2

int ledRojo = 13;
int ledVerde = 12;
void setup() {
  Serial.begin(9600);
  pinMode(ledRojo, OUTPUT);
  pinMode(ledVerde, OUTPUT);
}

void loop() {
  // read the input on analog pin 0:
  int pot = analogRead(A0);
  if(pot <= 512){
    digitalWrite(ledRojo, HIGH);
    digitalWrite(ledVerde, LOW);
  }else{
    digitalWrite(ledRojo, LOW);
    digitalWrite(ledVerde, HIGH);
  }
  delay(10);
}
```

En este ejemplo usamos dos leds uno rojo y uno verde y tomamos decisiones a partir de los datos leídos por el potenciómetro. El pin analógico de lectura tiene una resolución de 10 bits por lo que sus posibles valores varían de 0 a 1023, si el valor del pot es menor o igual a 512 encendemos el led rojo y si su valor es mayor a 512 encendemos el led verde.

La conexión de para un led en arduino es la siguiente.



El negativo del led se conecta a una resistencia de 220 ohms y la resistencia va a tierra y el positivo del led va al pin digital del arduino en este ejemplo el led rojo va al 13 y el verde al 12. El potenciómetro se queda conectado igual que en el ejemplo anterior.



Primero definimos los pines digitales a los que estarán conectados los leds como enteros.

```
int ledRojo = 13;
int ledVerde = 12;
```

Y los definimos como salida dentro de la función setup().

```
void setup() {
  Serial.begin(9600);
  pinMode(ledRojo, OUTPUT);
  pinMode(ledVerde, OUTPUT);
}
```

La función pinMode() define pines digitales ya sea como entrada o como salida, necesita dos parámetros, el primero es el pin al que se quiere definir y el segundo es la forma en la que trabajara el pin ya sea como entrada o

salida(OUTPUT o INPUT).

Después dentro de la función loop es donde tomamos decisiones a partir de los valores leídos

```
void loop() {
  // read the input on analog pin 0:
  int pot = analogRead(A0);
  if(pot <= 512){
    digitalWrite(ledRojo, HIGH);
    digitalWrite(ledVerde, LOW);
  }else{
    digitalWrite(ledRojo, LOW);
    digitalWrite(ledVerde, HIGH);
  }
  delay(10);
}
```

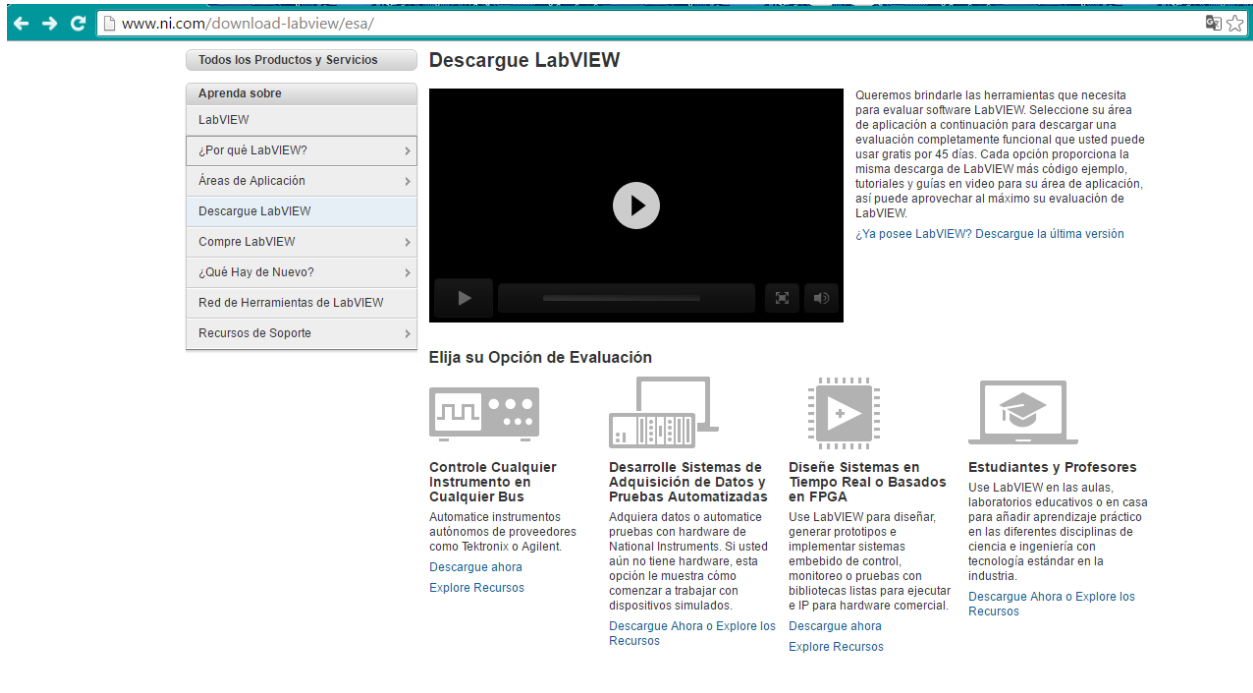
Primero obtenemos el valor leído por el pin A0 del arduino y lo guardamos en una variable de tipo int que llamamos pot.

Después con una sentencia if preguntamos si el valor leído es menor o igual a 512. Si la sentencia es verdadera entonces encendemos el led rojo y apagamos el verde en caso de que este encendido, para esto usamos la función digitalWrite() que pide dos parámetros, el primero es el pin al que quieres hacer referencia y el segundo es el nivel lógico al que quieres cambiar el pin(HIGH para encender el pin y LOW para apagar el pin). Los pines digitales trabajan con 5v así que mientras estén encendidos tendrán una diferencia de potencial de 5v con respecto de la tierra del arduino. En caso de que el valor del pot sea mayor a 512 la sentencia pasa a ser falsa por lo que entra al bloque "else" ignorando las líneas de código del bloque if y usando otra vez la función digitalWrite encendemos el led verde y apagamos el rojo.

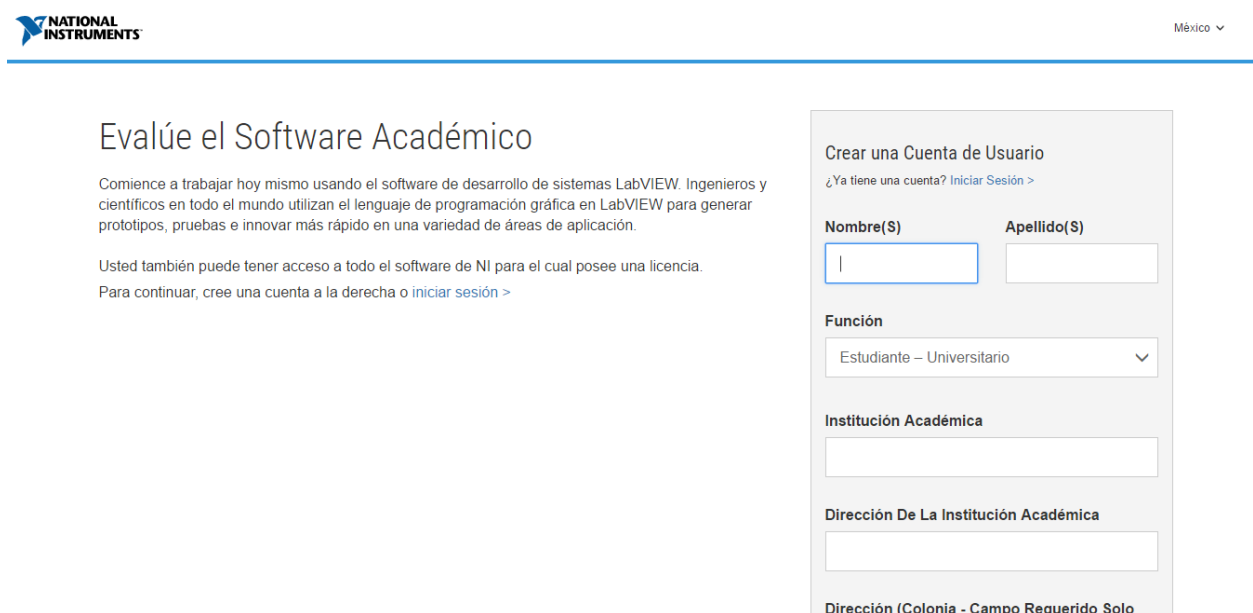
## LabView

Las practicas que hemos realizado con Arduino también se pueden realizar en LabView y en esta parte de la practica la explicaremos.

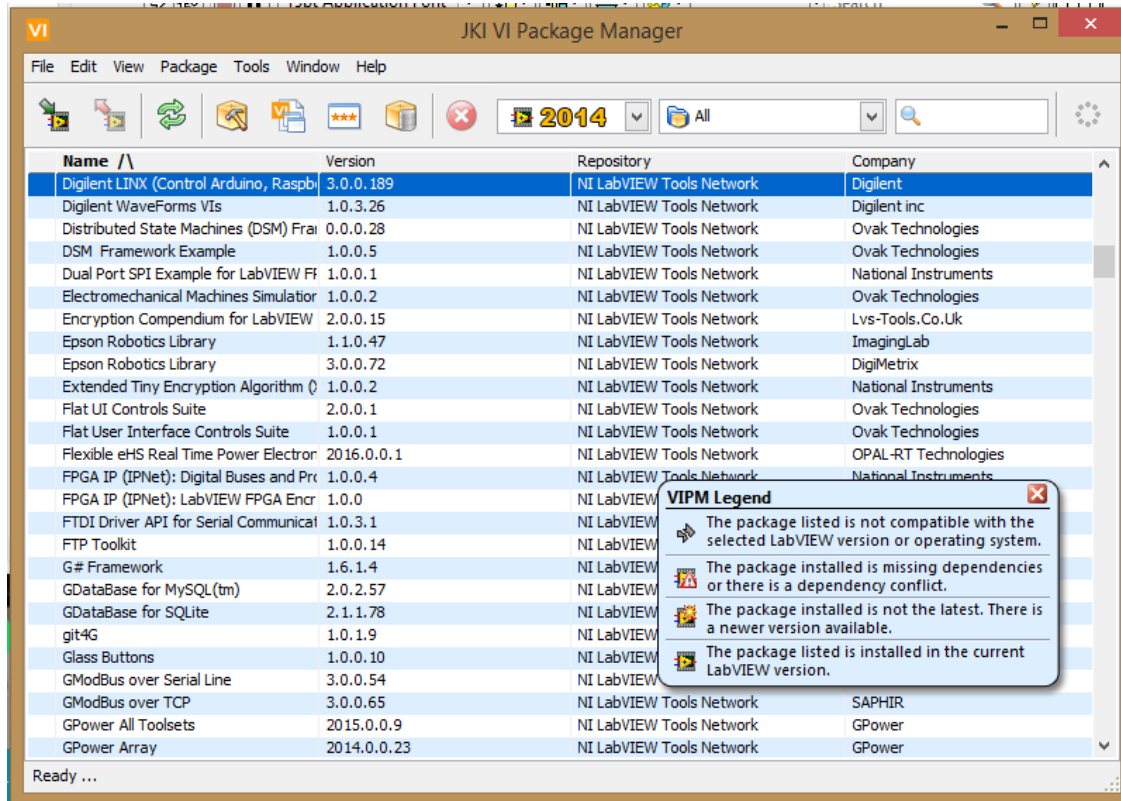
Primero requerimos descargar labview, lo podemos hacer directamente desde la pagina, utilizando la opción de Estudiantes y Profesores.



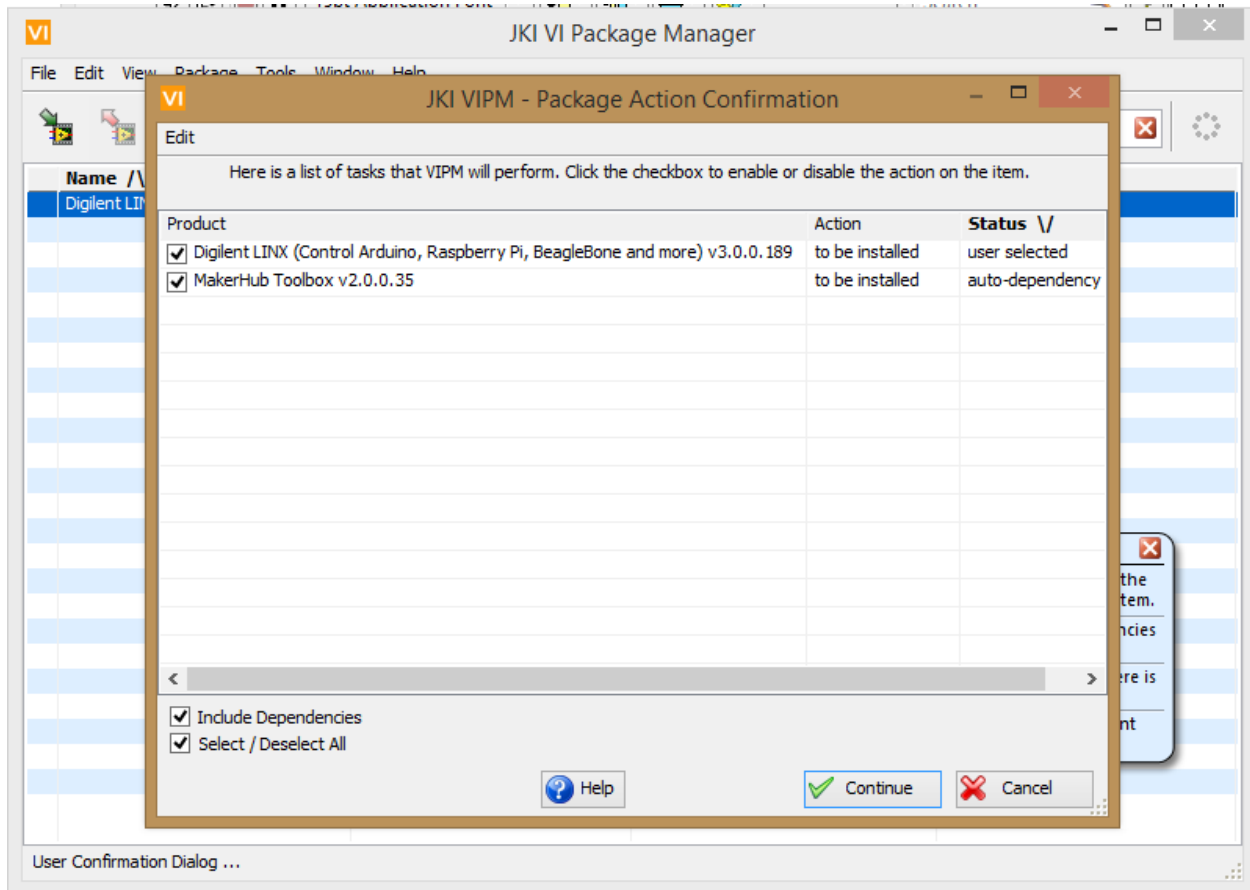
Al dar click aparece una página donde nos permite registrarnos y obtener una versión para estudiantes:



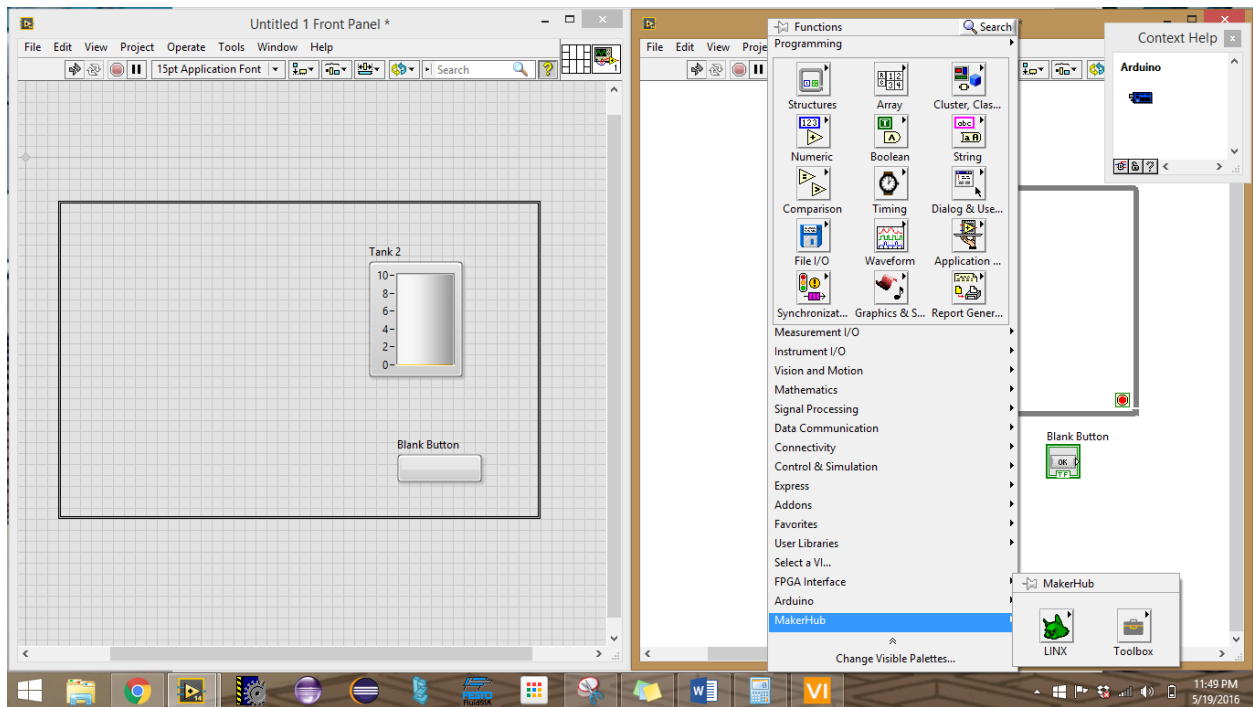
Después de que descargamos el Software, necesitamos habilitar una librería para poder usar LabView junto con el hardware de Arduino Uno, por lo que buscaremos en nuestra computadora JKI VI Package Manager que se descargó junto con el software de labview. Ya en el Package Manager buscaremos Digilent LINX



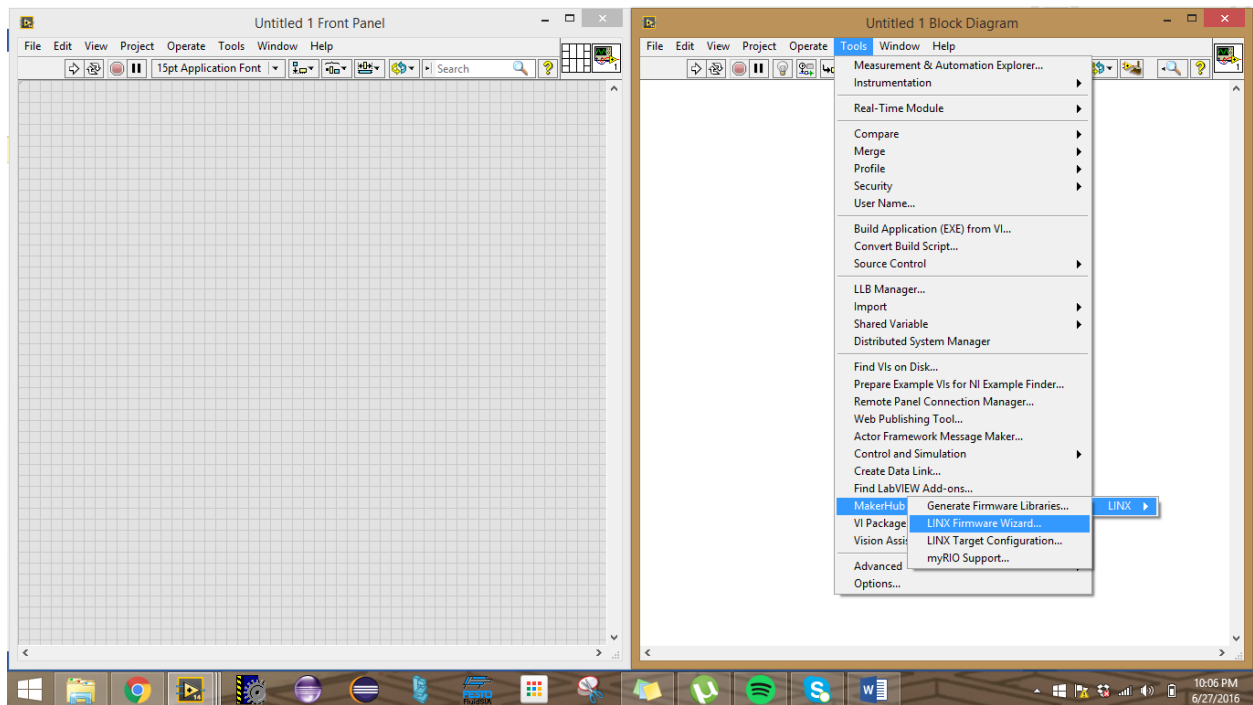
Damos click en la opción y aparecerá una ventana de Package Action Confirmation para instalar la librería y el toolbox y se da click en continuar.



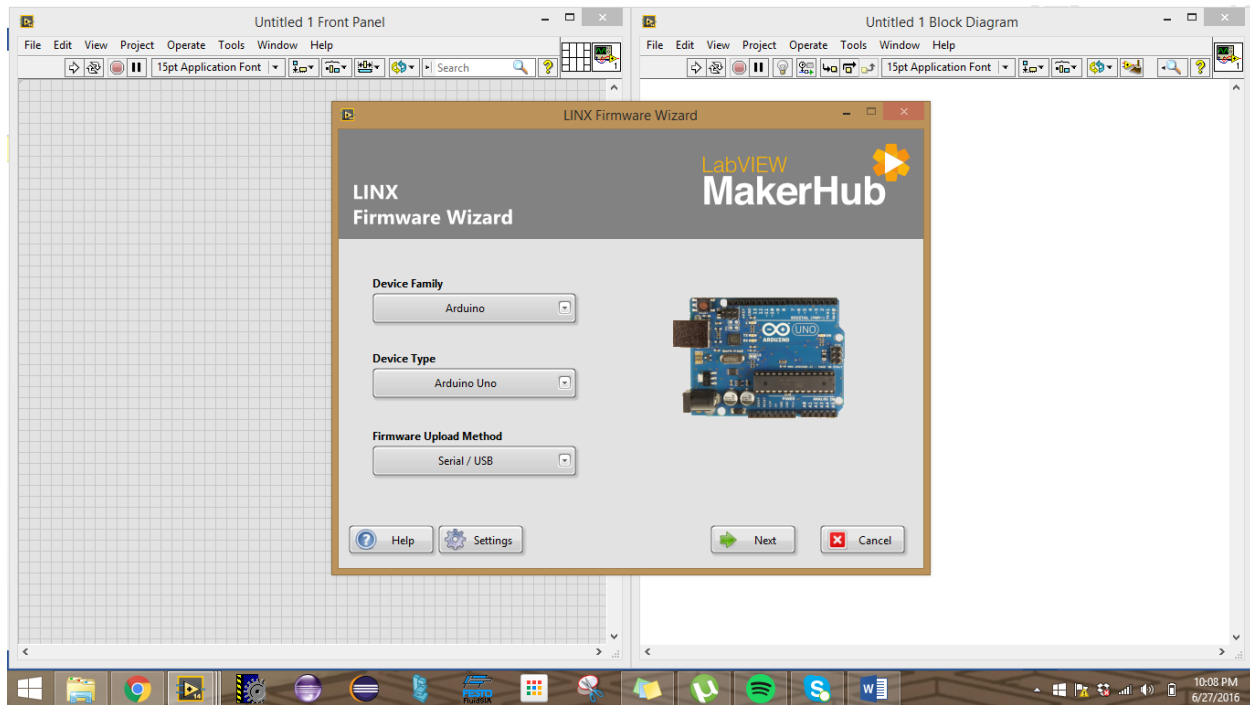
A continuación abrimos el software labview y al dar click derecho sobre la ventana de Block Diagram nos aparecerá la librería llamada LINUX como se ve en la imagen anterior.



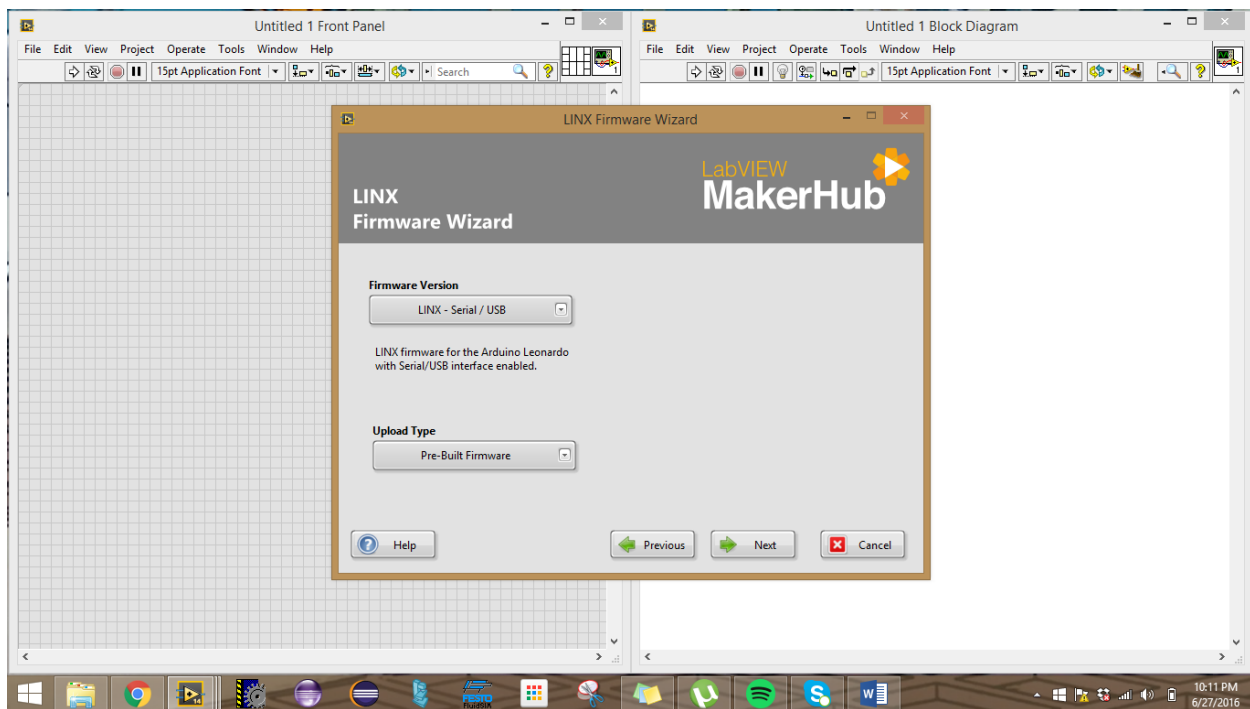
Despues se corre un programa para asegurar que hay conexion entre LabView y Arduino. Para hacer esto es necesario navegar en la barra de operaciones Tools>>MakerHub>>LINX>>LINX firmware Wizard



En Familia se pone Arduino, en Device Type el tipo de arduino que se esta utilizando, en esta caso Arduino UNO y por ultimo el Firmware Upload Method que será Serial/USB



Ahora se da click en Next y cargara el programa para la conexión.

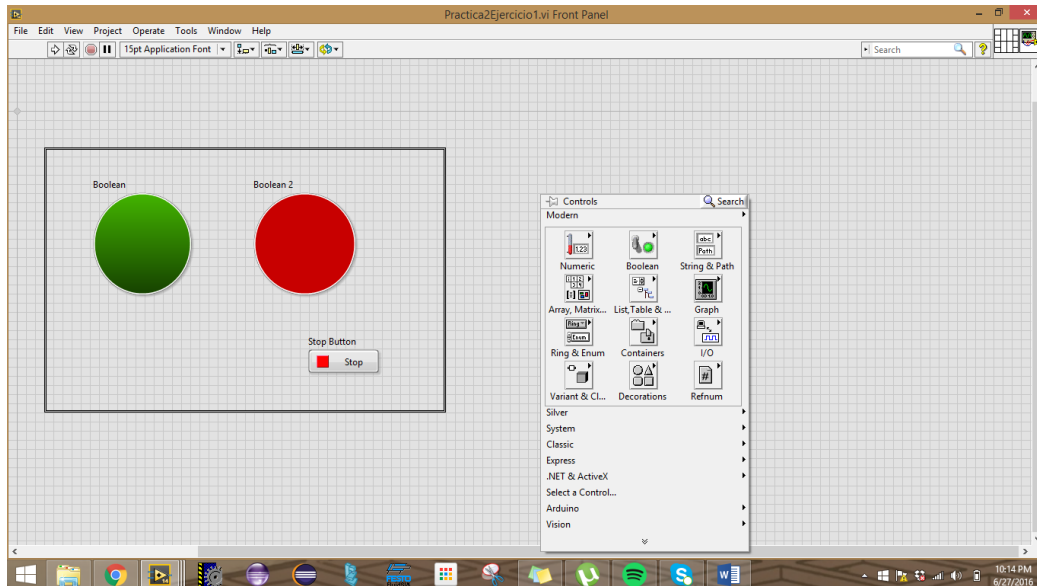


## Ejemplo 1

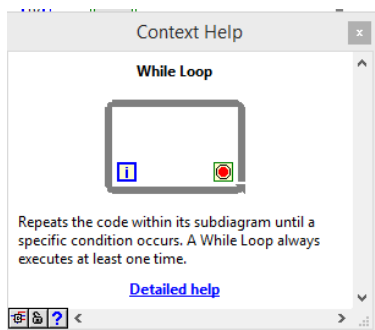
Después de haber realizado este procedimiento, se puede comenzar con el ejemplo 1.

El Front Panel sirve como la parte de la interfaz, en esta parte solo se puede visualizar los controladores y los indicadores. Los controladores pueden ser botones, perillas, cuadros de texto, etc; mientras que los indicadores pueden ser LED's, cuadro de texto, gráficos, etc.

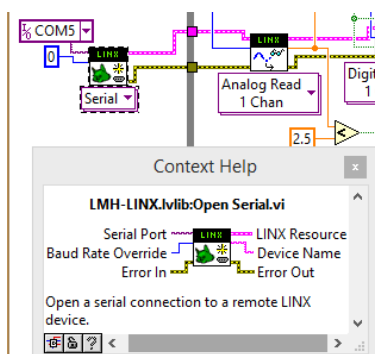
Empezando con el Front Panel se posicionan dos indicadores en forma de LED y un botón de stop, estos controles es posible encontrarlos dando click derecho en la pantalla de trabajo.



## Programacion

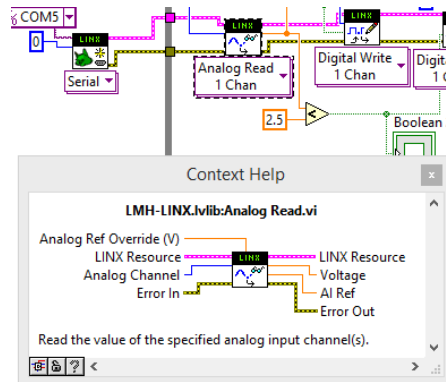


Para empezar la programación se necesita conocer que hay diferentes tipos de estructuras para programar, en este caso usaremos un While loop, que repite el código (lo que tiene adentro) hasta que una condición específica se cumple, en este ejemplo la condición específica va a ser cuando el botón de stop que posicionamos en el Panel Control sea verdadero, es decir, este presionado.

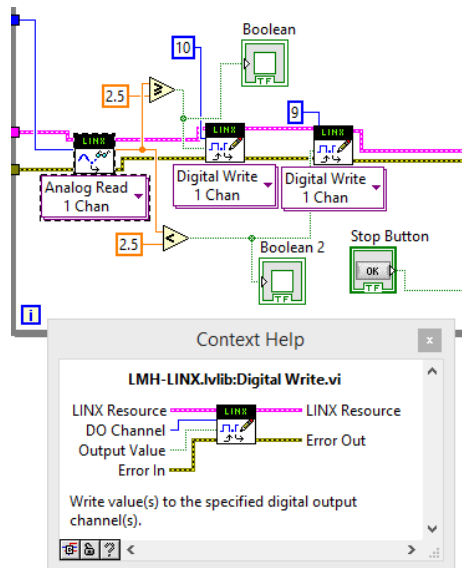


Para inicializar el serial elegimos este bloque de función de Linx llamado Open, los bloques de funciones normalmente tienen sus entradas del lado izquierdo y sus salidas del lado derecho, en este caso podemos ver que la primera conexión se refiere al puerto serial y creamos una constante y elegimos en COM que se está usando. A continuación se crea una constante para la conexión Baud Rate

Override y se le pone un 0, y las salida LINX Resource y Error Out son las que llegan al siguiente bloque de función entrando al While Loop.



Para poder leer los datos que estan en el potenciometro, se utiliza el block de funciones llamado Analog Read, que como su nombre lo indica lee una senal analogica. Del lado de las entrada se conecta la linea rosa y verde con su inmediato anterior, ya que estas le dan continuidad y orden al codigo, es decir en el orden en que se vayan conectando los bloques de funciones es el orden en el que se ejecutara el programa, en Analog Channel en el Pin al que esta conectado fisicamente el potenciometro en este caso el Pin 0 y creamos una constante para indicarlo. En la parte de las salidas podemos ver como nos da el voltaje que esta leyendo al arduino y que utilizaremos para hacer la logica y saber que Led prendera si el rojo o el verde.



Asi como encenderemos dos Leds en la interfaz, lo haremos en fisico, por lo que usaremos dos bloques de funciones llamados Digital Write, uno se usara para el led verde y otro para el led rojo. Para las entradas de este bloque de funciones conectamos el cable rosa y verde como se muestra en la imagen, despues elegimos en canal de entrada que sera conectado a los pines fisicos del Arduino Uno 10 y 9 respectivamente, la siguiente entrada que debemos considerar es el valor de salida que le vamos a dar, como este es un pin digital un 0 significaria apagado o falso y un valor 255 significaria prendido o verdadero. Por lo que, con el block Analog Read, sacamos el valor del voltaje que no variara a mas de 0 a 5V, despues hacemos una comparacion con los bloques menor que y mayor que, poniendo el valor del voltaje que se esta leyendo en la parte superior y el valor a comparar en la parte inferior, el resultado de esta comparacion se traducira en un falso o verdadero que cablearemos a los bloques de funciones Digital Write respectivamente. Y por ultimo cableamos los Boolean que son los Leds o indicadores que se muestran en el front panel, tambien una a cada Digital Write respectivamente.

Por ultimo cableamos el Stop Button a la condicion y corremos el programa dando click en la flecha Run de la barra de tareas.



