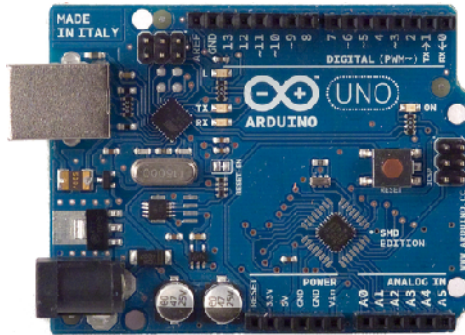


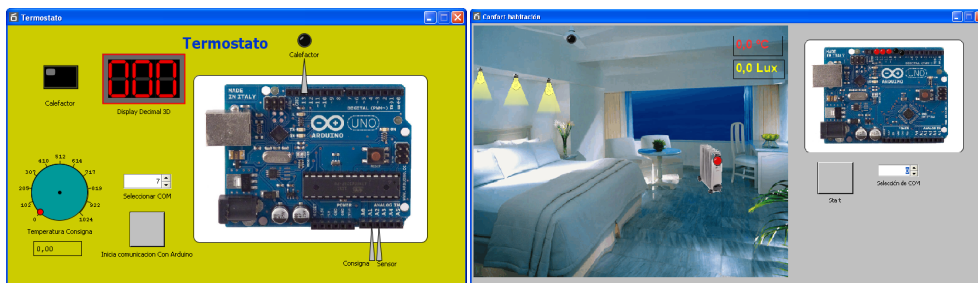
Arduino + MyOpenlab



+



=



Una propuesta de Utilización de Open Hardware y Software Libre GNU
para el Diseño y Simulación de Prototipos en el Laboratorio

José Manuel Ruiz Gutiérrez

Noviembre 2011

Índice

1. Objetivo de este trabajo.
2. Una Introducción general a MyOpenLab
3. Test de Conexión
4. Salida Intermitente
5. Salida Intermitente (otra opción)
6. Comparador de Entrada analógica con una constante.
7. Generador de Impulsos en el PIN 13 de Arduino
8. Termostato
9. Semáforo
10. Test Tarjeta
11. Confort
12. Prensa Hidráulica
13. Contador de impulsos de entrada
14. Parking
15. Puerta de entrada a una finca
16. Máquina de café
17. Datalogger Tipo 1
18. Datalogger Tipo 2

Noviembre de 2011 Versión de Documento: V1.0

José Manuel Ruiz Gutiérrez j.m.r.gutierrez@gmail.com

Blog de referencia: <http://josemanuelruizgutierrez.blogspot.com/>

1. Objetivo de este trabajo.

Con el presente trabajo práctico pretendo dar a conocer las posibilidades de la herramienta MyOpenLab en conjunción con Arduino. Se trata de un software de libre distribuido bajo licencia GNU de código abierto, escrito en Java, multiplataforma y orientado a la realización de aplicaciones de modelado y simulación.

Existe la posibilidad de conexión del entorno con el mundo físico a través de los puertos USB del ordenador de distintos tipos de hardware entre os que se encuentra Arduino.

Las posibilidades graficas de MyOpenLab así como su potencia del cálculo y proceso de datos le hacen adecuado para los fines de experimentación y elaboración de prototipos en el laboratorio y en el aula.

En este trabajo apporto una colección de aplicaciones que permitirán al lector comprender las posibilidades de esta poderosa conjunción Arduino + MyOpenLab y le animaran a continuar desarrollando materiales que permitan expandir la poderosa idea de la las plataformas Open Hardware y el software libre.

Para poder comenzar a trabajar con MyOpenLab recomiendo que el lector interesado se descargue los materiales que yo mismo elaboré sobre la herramienta, tutoriales y manual de usuario así como la versión última del software de la que me honro en ser colaborador.

Los siguientes documentos creados por mi ayudan al manejo de MyOpenLab y especialmente [Diagramas de Flujo V2.4.9.9.pdf](#)

Para conseguir el programa os podéis dirigir a : <http://es.myopenlab.de>

[Guía de usuario Versión 3.010](#)

[Objetos CANVAS.pdf](#)

[Tratamiento de Datos.pdf](#)

[Diagramas deFlujo V2.4.9.9.pdf](#)

[Simulacion de Modelos Matemáticos y Temporales.pdf](#)

[Manejo de Datos en Matrices y Tablas.pdf](#)

[Robotica MyOpenLab.pdf](#)

2. Una Introducción general a MyOpenLab

(Una herramienta para la modelización y simulación orientada a la educación)

Descripción.

[MyOpenlab](#) es un entorno orientado a la simulación y modelado de sistemas físicos, electrónicos y de control con un amplio campo de aplicaciones.

La aplicación esta desarrollada en el lenguaje JAVA y por ello resulta portable a distintas plataformas. En el campo del modelado y simulación es muy interesante contar con una herramienta flexible que a partir de una amplia biblioteca de bloques funcionales permita realizar modelos a base de conectar bloques funcionales.

MyOpenLab es capaz de conectarse al mundo físico mediante una interface de amplia difusión en el mercado K8055 de Valleman y también a la tarjeta Arduino.

La presentación de los resultados y/o el control de las simulaciones se hace mediante un potente conjunto de bloques de función de visualización y/o interacción capaz de manejar todo tipo de datos (analógicos, digitales, matrices, vectores, imágenes, sonidos, etc.). Mediante MyOpenLab es posible diseñar instrumentos virtuales (VI) a través de los cuales se puede realizar una aproximación a los sistemas de medida y control de una manera más realista.

La realización de una simulación se hace mediante dos pantallas o áreas de trabajo: Panel Circuito y Panel Visualización. En el primero se diseña el algoritmo de simulación mediante "bloques" o "elementos de función" y el segundo se muestran los datos o se generan los estímulos cuando se está en el modo de "simulación"

El programa puede funcionar en plataformas Linux y sus requerimientos de sistema son muy poco restrictivos, lo cual lo hace ideal para usar en casi cualquier equipo. bastará que se instale el runtime de JAVA JRE o JDK.

Esta herramienta está recomendada para estudiantes de prácticamente todos los niveles: ESO, Bachillerato, Formación Profesional y Primeros Cursos de las Carreras Técnicas Universitarias.

CARACTERÍSTICAS

- Facilidad de uso
- Amplia biblioteca de funciones tanto para manejo de señales analógicas como digitales.
- Posee una potente biblioteca de objetos gráficos tipo "canvas" mediante la que se puede dotar de movimiento cualquier objeto o imagen asociándola a variables de los modelos a simular.
- Tratamiento de los tipos de datos y operaciones con estos.
- Realización de las aplicaciones mediante el uso de bloques de función con la posibilidad de encapsularlos en "macros".

- Facilidad para crear pantallas de visualización que recojan el estado de las variables y eventos de las simulaciones.
- Posibilidad de ampliación de su librería de componentes, editándolos en código JAVA
- Posibilidad de creación de "submodelos de panel" y "submodelos de circuito" encapsulados.
- Algunas librerías que incorpora MyOpenlab:

Librería de elementos de Visualización y Control (Panel Frontal)



Elementos de Decoración

Elementos de visualización numérica

Elementos de activación digital

Elementos de Entrada y salida de cadenas de caracteres

Elementos de entrada y salida tipo vectores y matrices de datos

Elementos de visualización grafica en ejes coordenados I

Elementos de visualización grafica en ejes coordenados II

Librería de Extras

Elementos de Automatización

Elementos de librería de usuario

Robot 2D

Librerías de Elementos Funcionales (Panel Lógico)



Elementos de Decoración

Operadores Digitales

Operadores Numéricos

Tratamiento de Caracteres

Elementos Analógicos

Utilidades

Ficheros de Entrada/Salida

Comparators

Tratamiento de Imágenes

Tratamiento de Sonidos

Color

Pines de E/S

Vectores y matrices

Agrupación de Elementos

Objetos Gráficos “canvas”

Librería de Física

Librería de Diagramas de Flujo

Librería de Extras

Librería de Conexiones entre aplicaciones

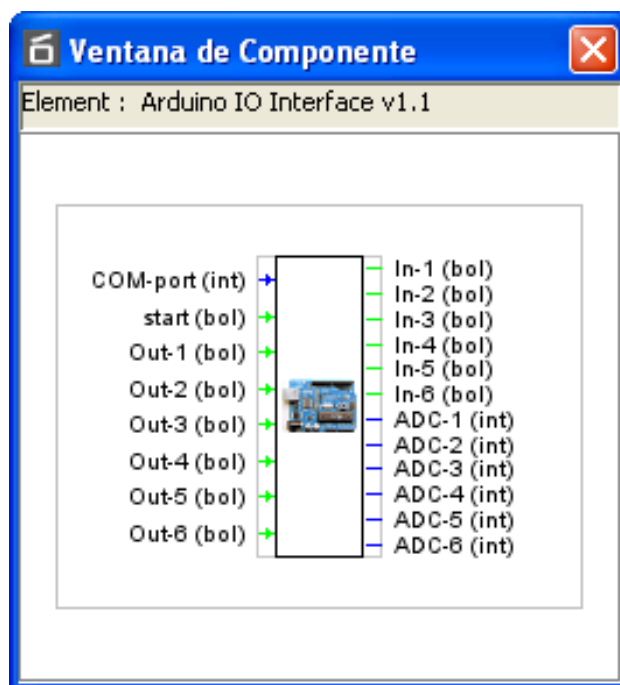
Librería definida por el Usuario

Automation+ibrería de Automatización Interfaces

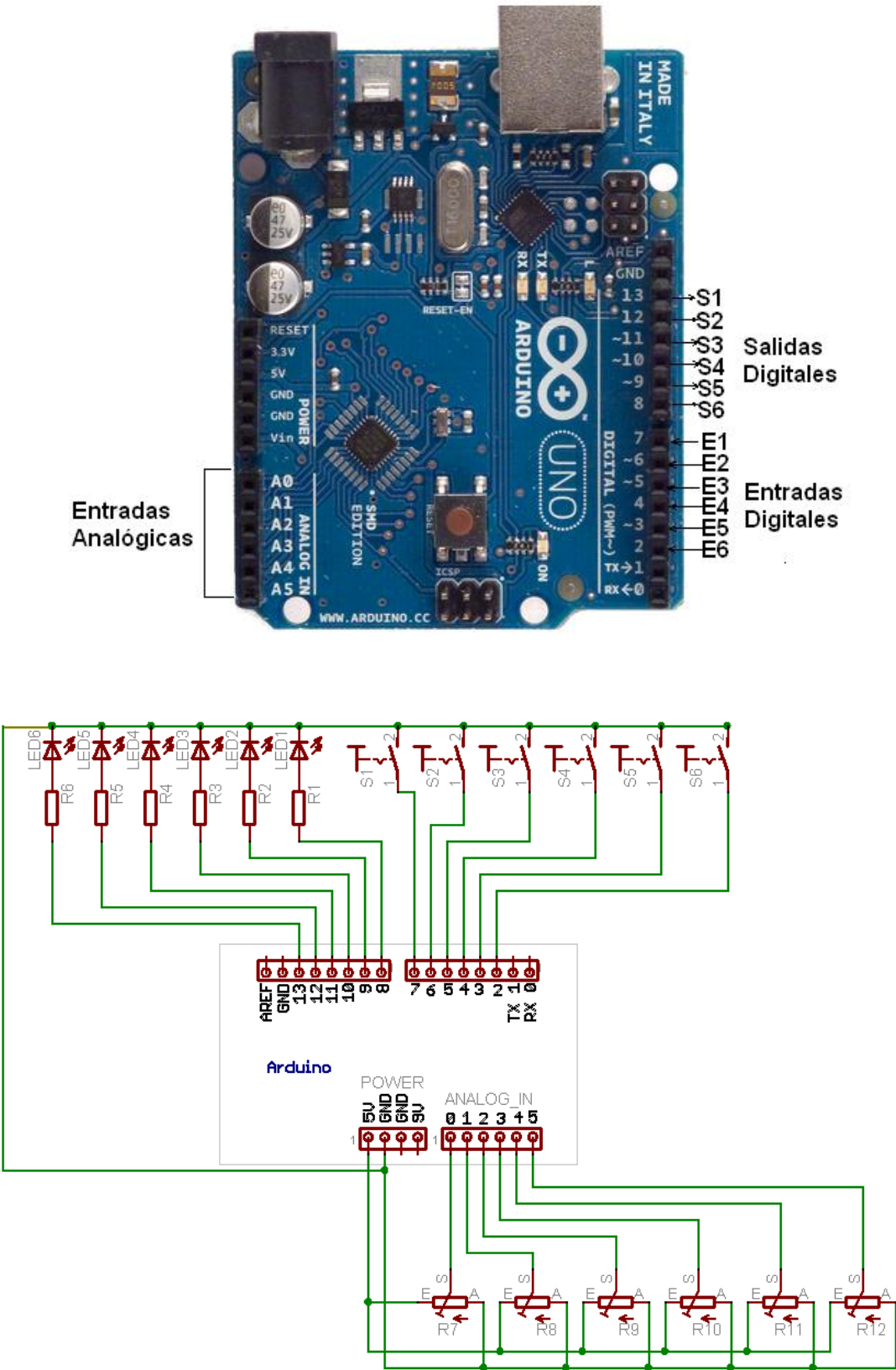
Forma de conexionado con Arduino.

Para conseguir la comunicación entre Arduino y Myopenlab se seguirán los siguientes pasos:

1. Cargar el Firmware en la tarjeta Arduino con la ayuda del IDE Arduino. El fichero se encuentra en la carpeta D:\distribution3032\MyOpenLab_and_Arduino V\arduino y el fichero se llama: arduino.pde
2. Realizar el diseño dentro del entorno de Myopenlab utilizando la librería “Arduino IO Interface 1.1” que es la que se encargara de la comunicación con la tarjeta Arduino.
3. Consignar los elementos de entrada y salida de acuerdo con la configuración que el firmware coloca en la tarjeta y que se muestra en la figura.
4. No olvidar que hay que dejar previsto un elemento de entrada de valor numeroco para consignar el numero del puerto por el que se realizara la comunicación y una entrada digital (tipo pulsador) para que se active el diálogo entte Arduino y Mypenlab
5. En el panel de visualización se podrán colocar elementos graficos para mostrar los valores de entrada y salida que le darán a la aplicación un aspecto visual muy interesante.
6. Finalmente una vez realozado el diseño se activara el modo Run de Myopenlab y veremos la aplicación funcionar.



Componente de librería para conexión con Arduino



Designación de E/S para la conexión de Arduino con Myopenlab

3. Test de Conexión

A continuación se muestra un ejemplo de aplicación genérico de conexión de Arduino con MyOpenLab que permitirá realizar un test de cada una de las entradas salidas configuradas en el protocolo de comunicación.

No olvidemos la asignación de estas entradas salidas que se recuerda en la siguiente tabla.

Tipo de E/S	Pin en la tarjeta Arduino	Pin en la librería MyOpenLab
Entrada Digital 1	D7	In-1
Entrada Digital 2	D6	In-2
Entrada Digital 3	D5	In-3
Entrada Digital 4	D4	In-4
Entrada Digital 5	D3	In-5
Entrada Digital 6	D2	In-6
Salida Digital 1	D8	Out-1
Salida Digital 2	D9	Out-2
Salida Digital 3	D10	Out-3
Salida Digital 4	D11	Out-4
Salida Digital 5	D12	Out-5
Salida Digital 6	D13	Out-6
Entrada Analógica 1	A0	ADC-1
Entrada Analógica 2	A1	ADC-2
Entrada Analógica 3	A2	ADC-3
Entrada Analógica 4	A3	ADC-4
Entrada Analógica 5	A4	ADC-5
Entrada Analógica 6	A5	ADC-6

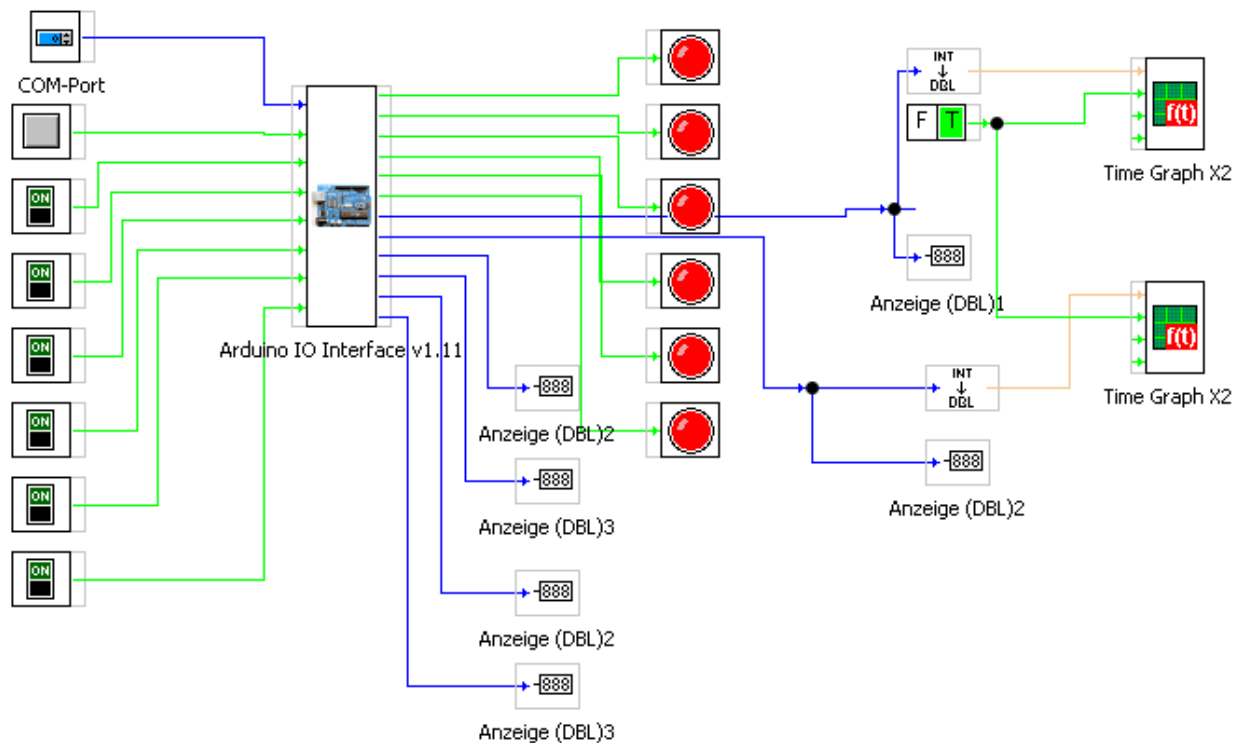
Obsérvese que en esta versión de librería para Arduino de MyOpenLab no se han considerado las salidas analógicas tipo PWM

Ejemplo de conexonado operación manejando todas las E/S de la tarjeta Arduino

El esquema del montaje en la parte de MyOpenLab consta de la librería correspondiente de comunicación con Arduino “Arduino IO Interface V1.11” a la que se han conectado interruptores, diodos leds y medidores analógicos y un trazador registrador grafico para poder representar el estado de la variable así o c el gobierno de las salidas de la tarjeta Arduino.

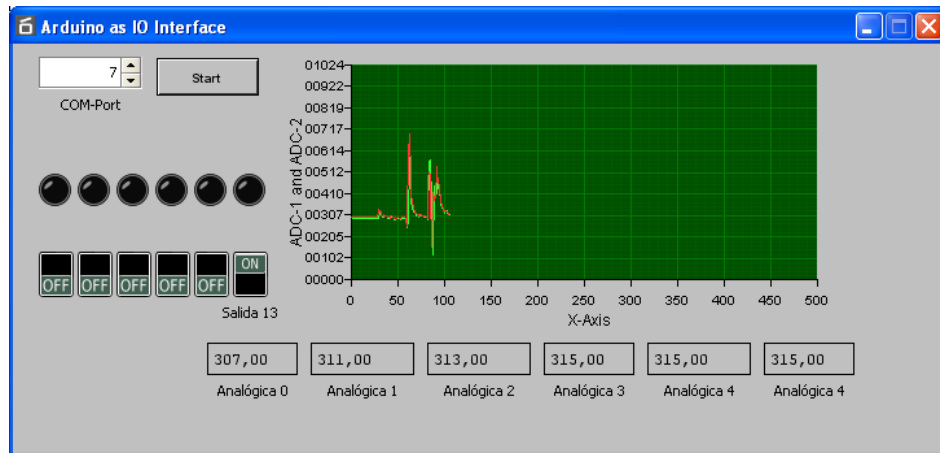
Conexión del MyOpenlab y Arduino

Esta conexión se hace mediante las entradas de la librería: **Com-port** y **Start** que permiten indicara MyOpenLab el puerto por donde recibirá y entregara datos y la orden para estableces la comunicación respectivamente,. La primera es una variable de tipo entero (int) y la segundo es de tipo booleano (bol). En la figura siguiente se ven estos objetos conectados a la librería Arduino



En modo ejecución la pantalla e conexión con Arduino es la de la figura siguiente.

En modo ejecución lo primero que haremos será fija el numero de puerto COM con el se conecta Arduino pulsar el botón **Star** y seguidamente podemos manipular en la pantalla para gobernar las salidas y manipular en las entradas de Arduino para observar los valores que toman en la pantalla.



Ejemplo de salida intermitente en el PIN 13 de Arduino salida 0

4. Salida Intermitente

La siguiente aplicación es la mas sencilla de todas y suele servir para probra que las cosas funcionan. Se trata de encender y apagar la salida digital establecida en el pin **D13**.

Para ello se selcciona el bloque de funcion de la librería correpondiente **Arduino IO Interface v1.11** y se procede a la conexión de un elemento de entrada de valor tipo Integer para la sellcion del COM de comunicaciones y un boton para iniciar la comunicaci3n con la tarjeta Arduino. Esta operaci3n ser3 comun y necesaria en todas las alplicaciones que montemos.

La se3al que hemos de mandar en forma de pulsos se obtiene de un bloque de funcion llamado **Temporizador** de la librería **Utilidades** del **Panel de Circuito**. A este bloque le damos los vaores correpondientes de tiempo encendido (Tiempo activado) y tiempo apagado (Tiempo desactivado).

Colocamos un led para mostrarnos informacion del estado de la se3al en el lado fisico de la trajeta Arduino (Pin Digital 13)

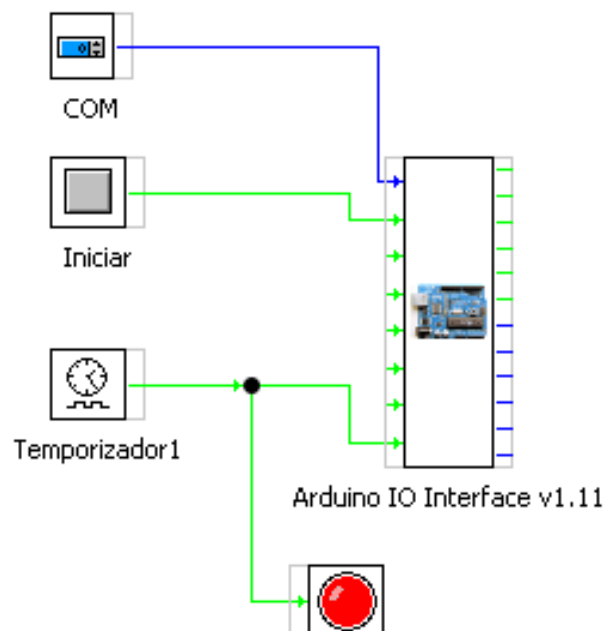
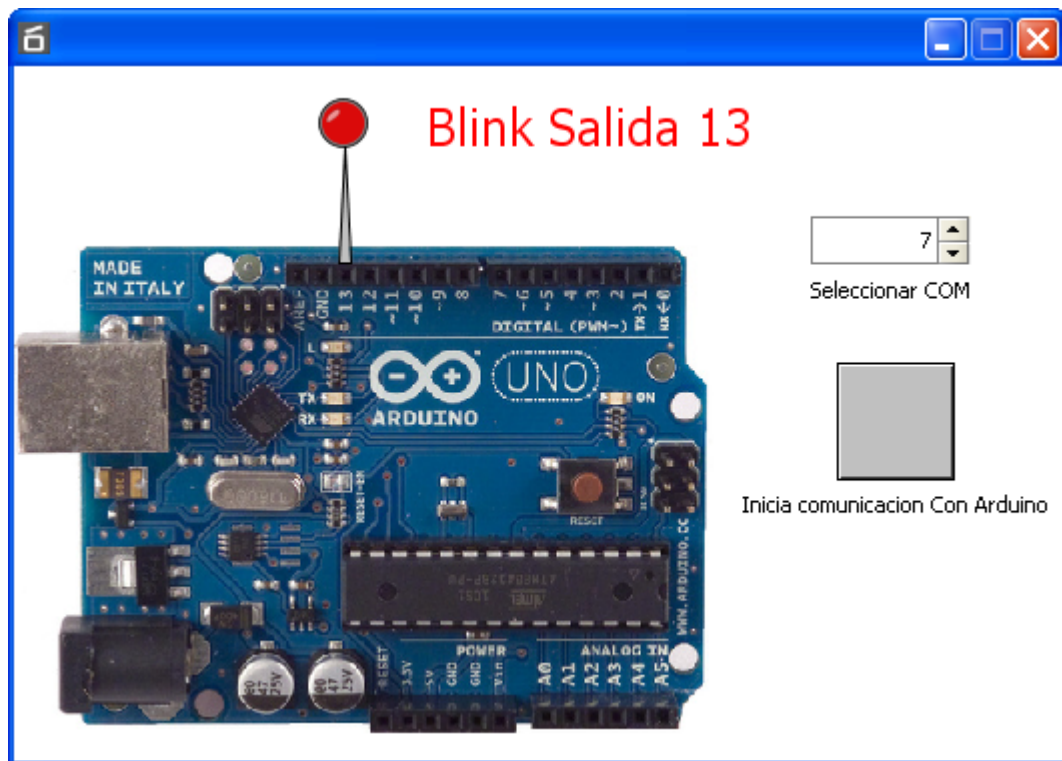


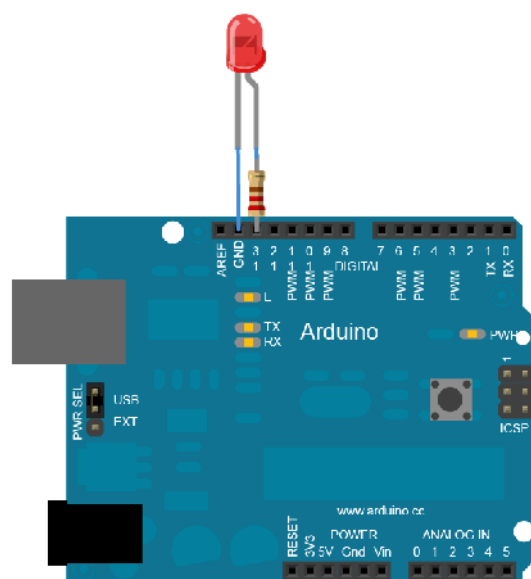
Figura del Panel Circuito

En el Panel Frontal podremos una imagen de Arduino haciendo uso de l objeto **Raster Imagen** d ela librería **Decoraci3n** perteneciente al Panel Frontal. Pondremos tambien un texto mediante el objeto **Label** de la misma librería. El resto de objeytos, el boton y el Spinner (INT) aparecen dado que se pusieron en el Panel Circuito.



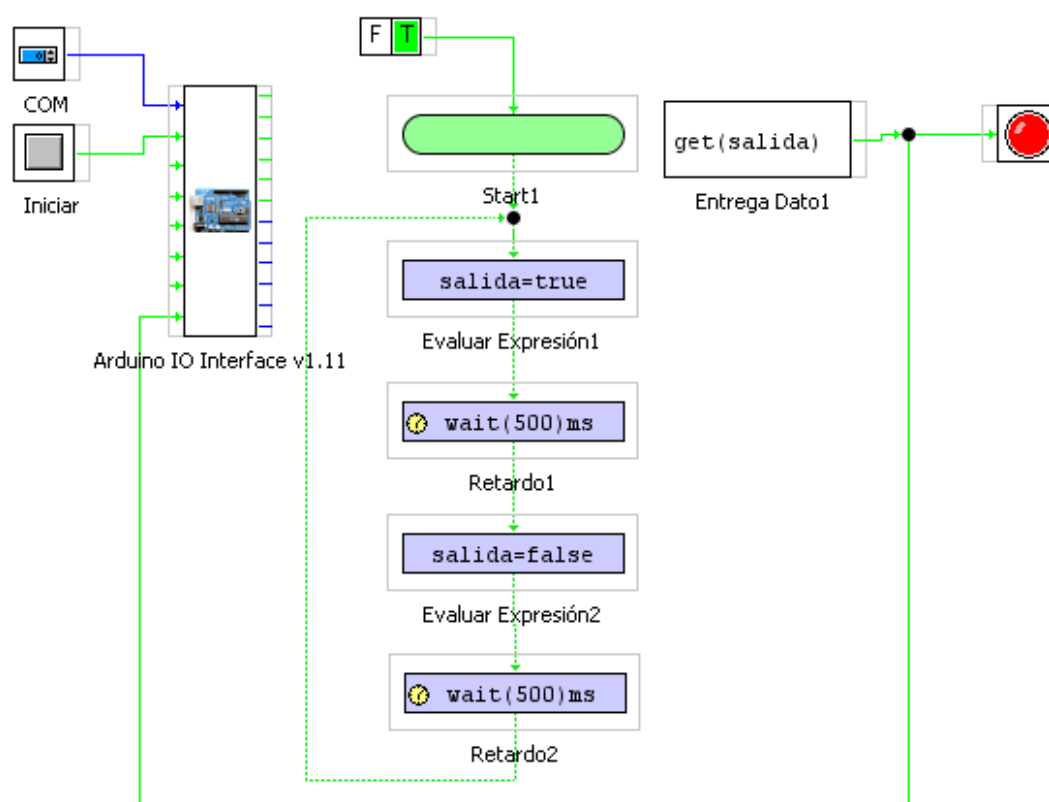
Para iniciar la aplicación ejecutamos mediante el botón Arrancar VM del entorno. Seleccionamos el puerto con el selector y pulsamos el Botón “Inicia comunicación con arduino”

Si todo va bien veremos parpadear la salida 13 de Arduino en la que lógicamente habremos colocado un diodo led de acuerdo a la figura siguiente.



5. Salida Intermitente (otra opción)

Se trata de relizar una aplicación haciendo uso de la librería de programación de Diagramas de Flujo del Panel de Circuito de MyOpenLab. Se crea una variable que se llama “salida” y se activa y desactiva cada 500 ms de acuerdo con el valor que se establece en el correpondiente bloque Retrado de la librería. La variable salida me diente un bloque Recoge dato se lleva a la salida del Pn 13 del bloque d liberioa de Arduino y a un Led para que sirve de indicacro del estado de la señal en la pantalla



6. Comparador de Entrada analógica con una constante.

En esta aplicación se trata de realizar la comparación de una de las señales de entrada analógica “i” de la tarjeta Arduino (procedente de un sensor) con un valor ajustable a través de un Slider “P” en la pantalla del ordenador. En función del resultado de la comparación se activará la salida PIN 13 de Arduino (0 si $i < p$ y 1 en caso contrario)

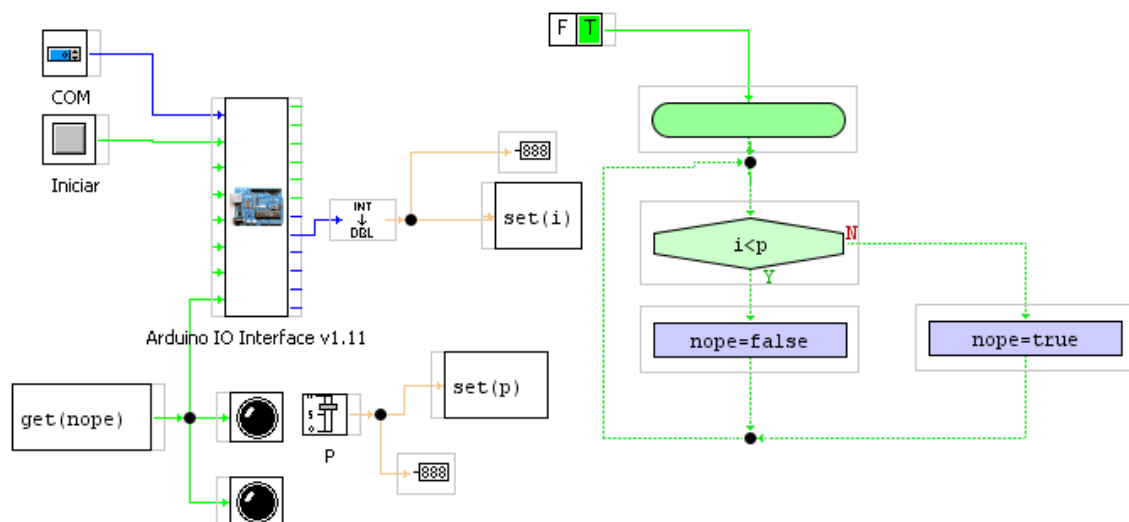
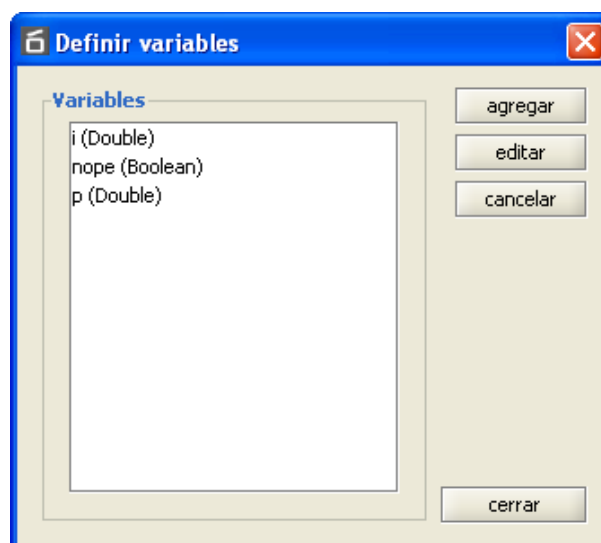


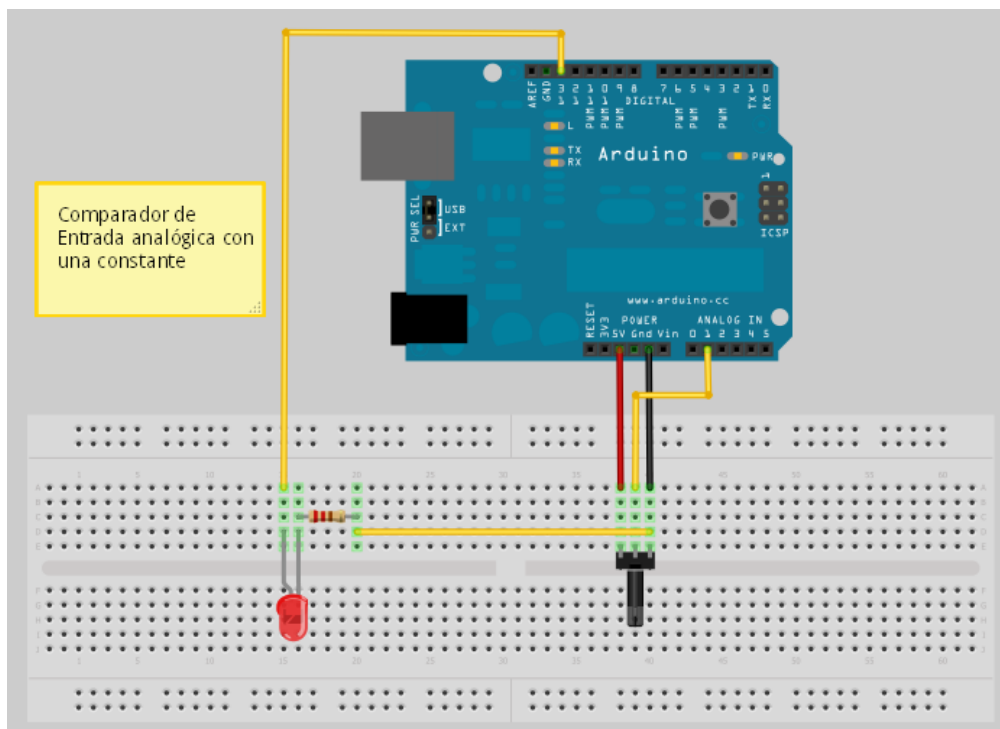
Diagrama funcional de la aplicación contenido en el Panel circuito de MyOpenLab

Las variables que debemos definir en la aplicación son las mostradas en la ventana de definicioon de variables de MyOpenLab





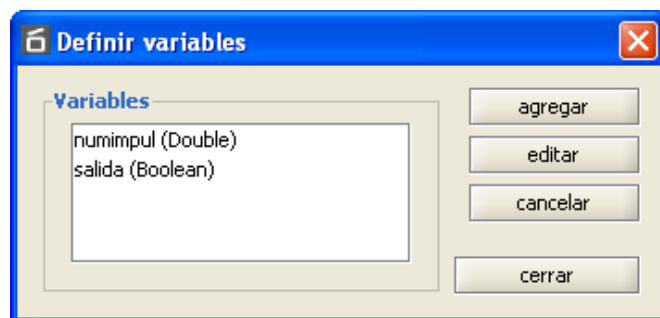
En la figura se muestra el Panel Frontal de la aplicación



7. Generador de Impulsos en el PIN 13 de Arduino

En esta aplicación se trata de generar por una salida de la tarjeta Arduino un número determinado de impulsos que previamente podemos seleccionar mediante la pantalla de interacción con la tarjeta.

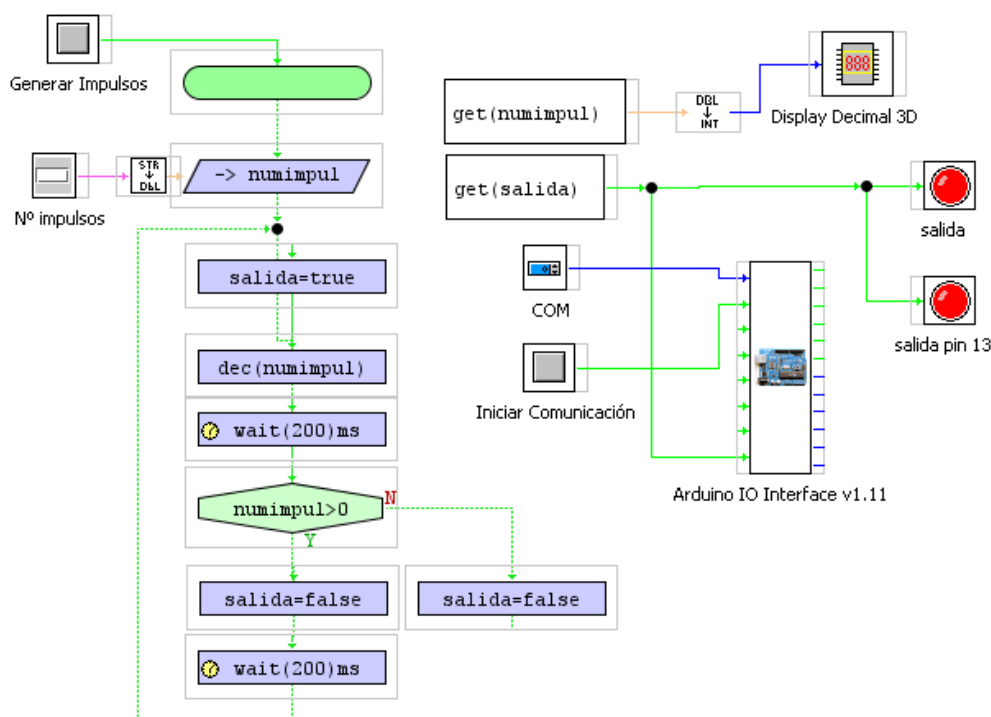
Las variables que tendremos que definir previamente serán las que figuran en la venta de la imagen siguiente.



“**numimpul**” Es el valor del número de impulsos a generar

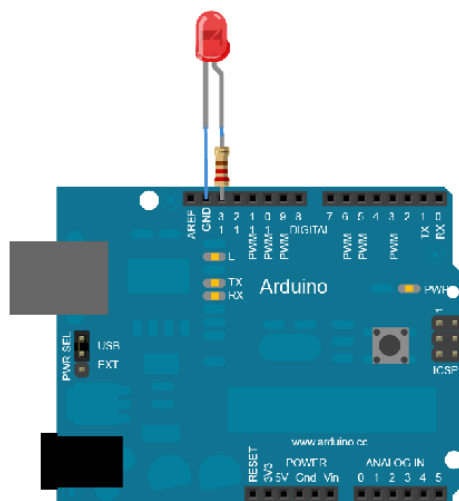
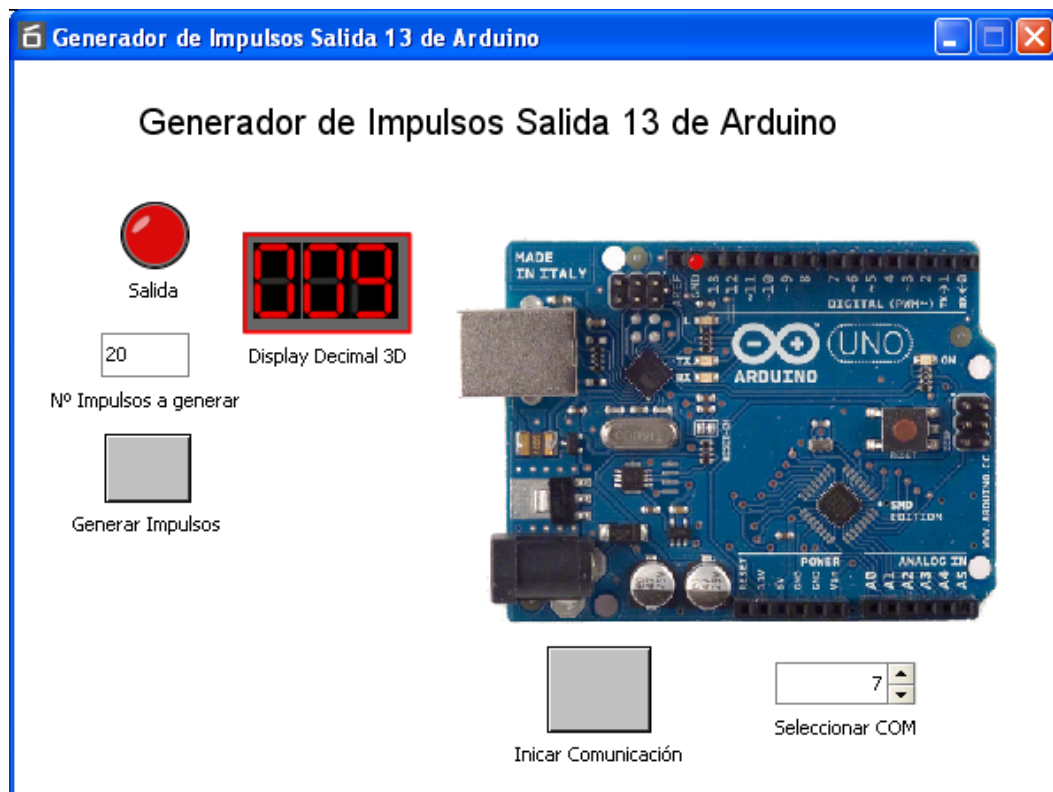
“**salida**” es la variable asociada a la salida que se dirigirá en nuestro caso al PIN 13 digital de Arduino

En el esquema siguiente vemos la implementación del diagrama de flujo que resolverá nuestro problema



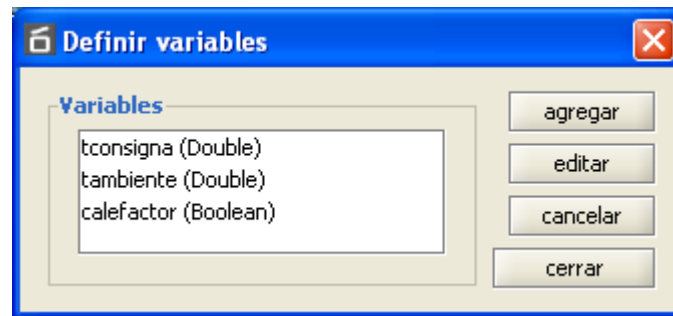
Básicamente se trata de implementar un bucle dentro del cual se genera un impulso activando y desactivando la variable “salida” cada **200 ms**. Hacemos uso de la instrucción **dec** que lo que hace es decrementar el valor de la variable **numimpul** hasta que se hace cero y termina el programa.

Con los bloques de función **get** recogemos las variables y las llevamos a un contador, para leer el estado de la variable **numimpul** y a unos diodos leds indicadores y a la propia salida **PIN 13** del Bloque Arduino que se corresponde con la señal **out-6** del bloque **Arduino IO Interface V1.11**



8. Termostato

Con este ejemplo se pretende controlar el encendido de la calefacción de una casa teniendo en cuenta la temperatura a la que queremos que este la casa (temperatura de consigna) y el valor de la temperatura ambiente.



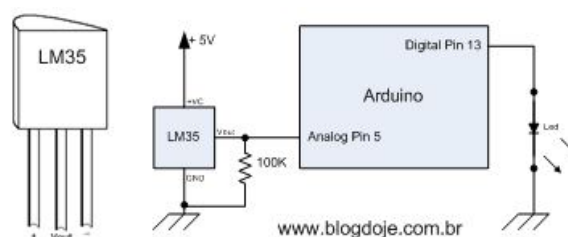
Las variables que debemos considerar son:

- tconsigna** Temperatura a la que deseamos que este la casa (valor de consigna). Tipo double.
- tambiente** Valor de la temperatura ambiente medido mediante una sonda de temperatura.
- calefactor** Señal que activa la calefacción (tipo booleano)

El algoritmo es muy sencillo, basta que incluyamos un bloque condicional en el que se pregunte si el valor de la *tambiente* < *tconsigna* si se cumple la condición se activara el calefactor y en caso contrario se desactivara.

Al ser los valores analógicos que entrega el bloque de comunicación con Arduino de tipo *Integer* no olvidemos que hemos e convertirlos en tipo *Double*.

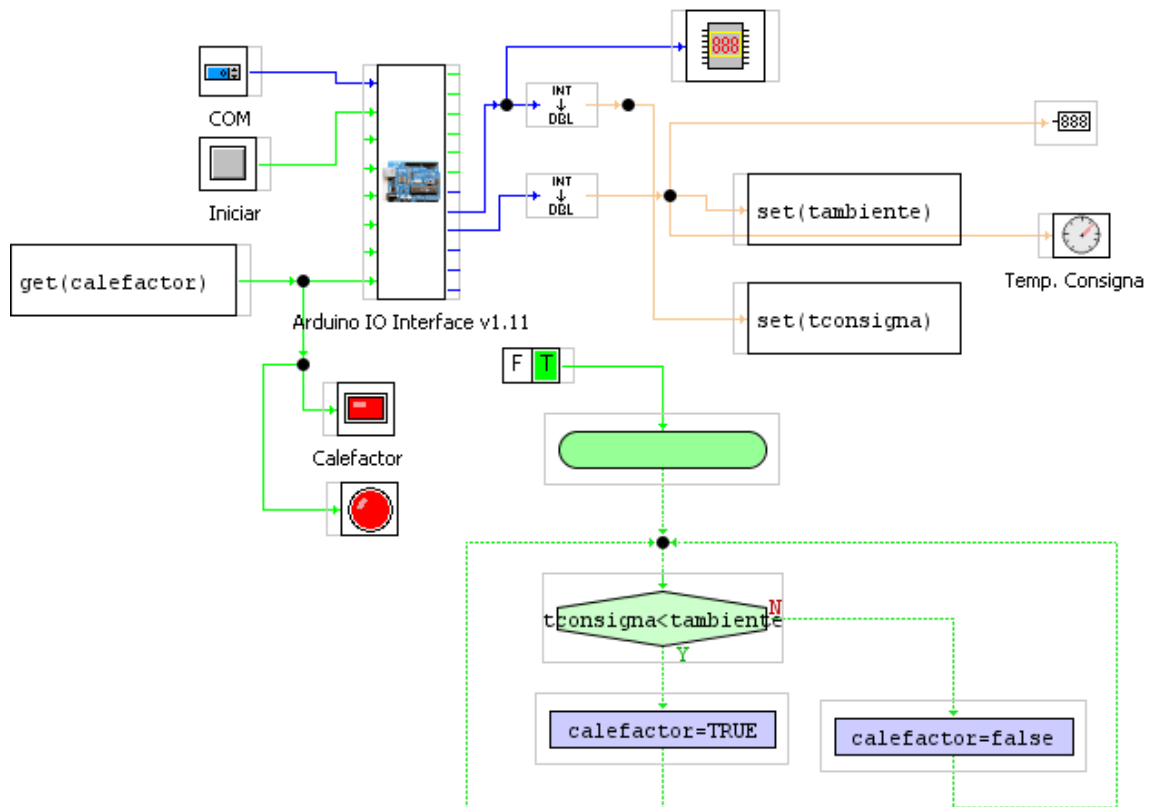
Para medir la temperatura se puede utilizar un sensor como el que se muestra a continuación.



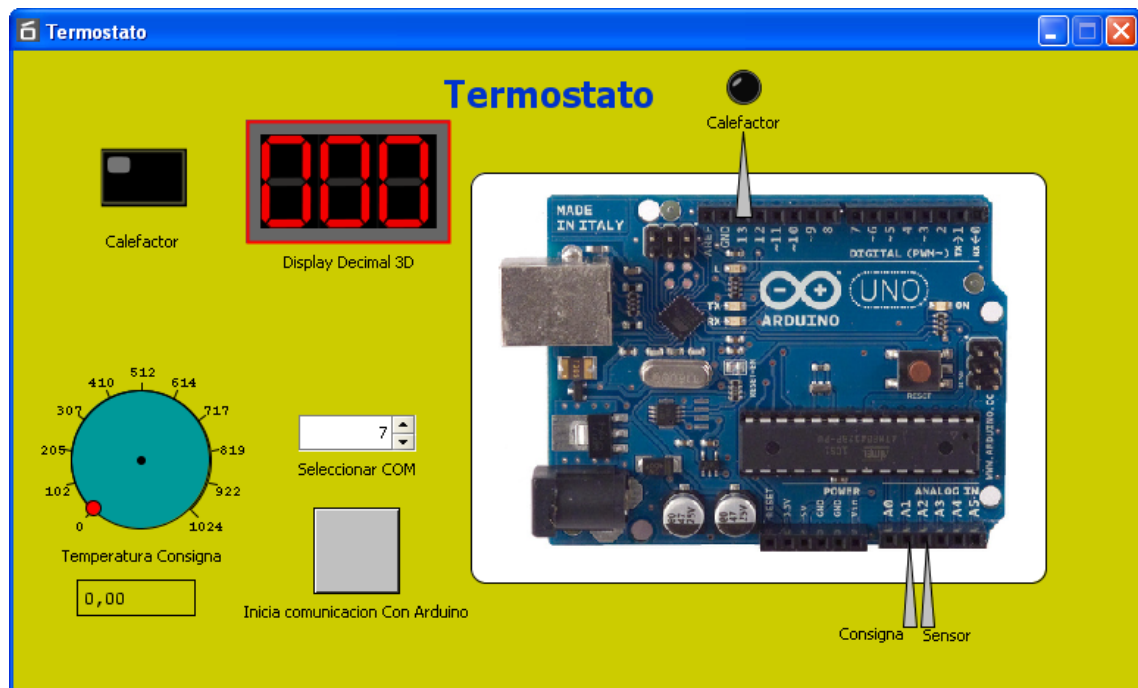
Sensor de temperatura

A continuación se muestra el esquema correspondiente del *Panel Circuito* de

MyOpenlab

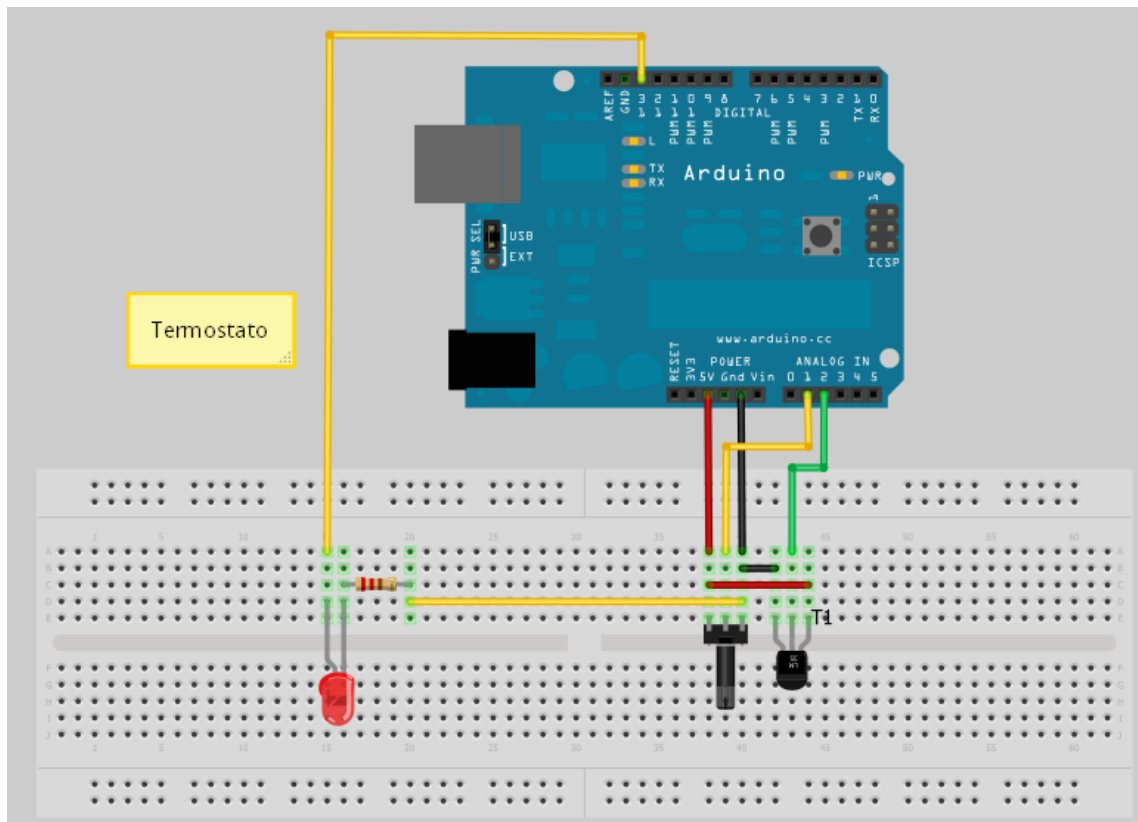


Panel Circuito



Panel Frontal en modo ejecución

La imagen siguiente corresponde al montaje de la aplicación con Fritzing



9. Semáforo

Se trata de realizar un semáforo que gobierne tres salidas en forma de diodos led (rojo, ámbar y verde)

Señales de salida:

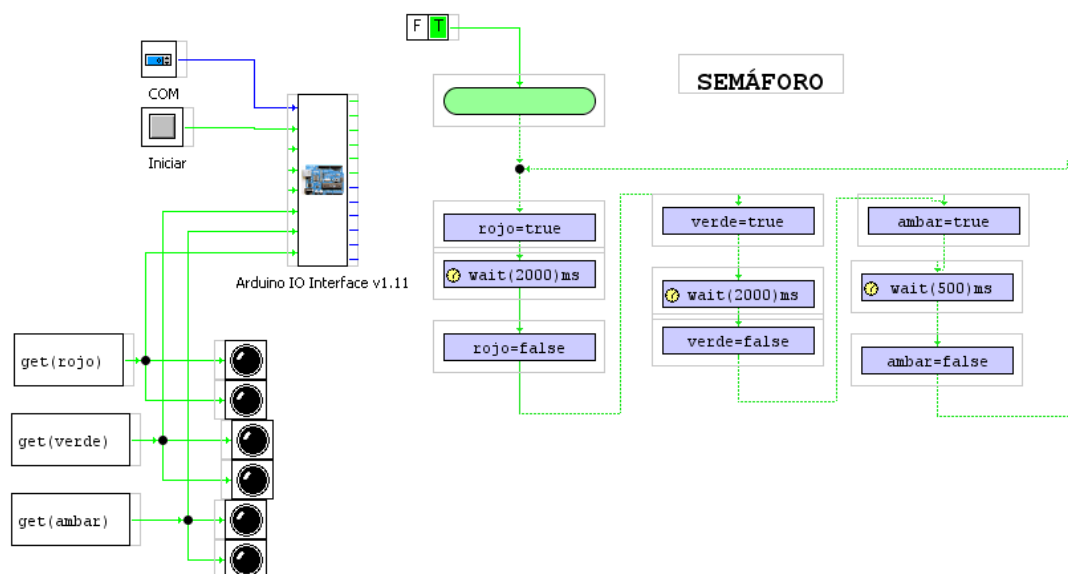
roja, ambar y verde (todas de tipo Boolean)

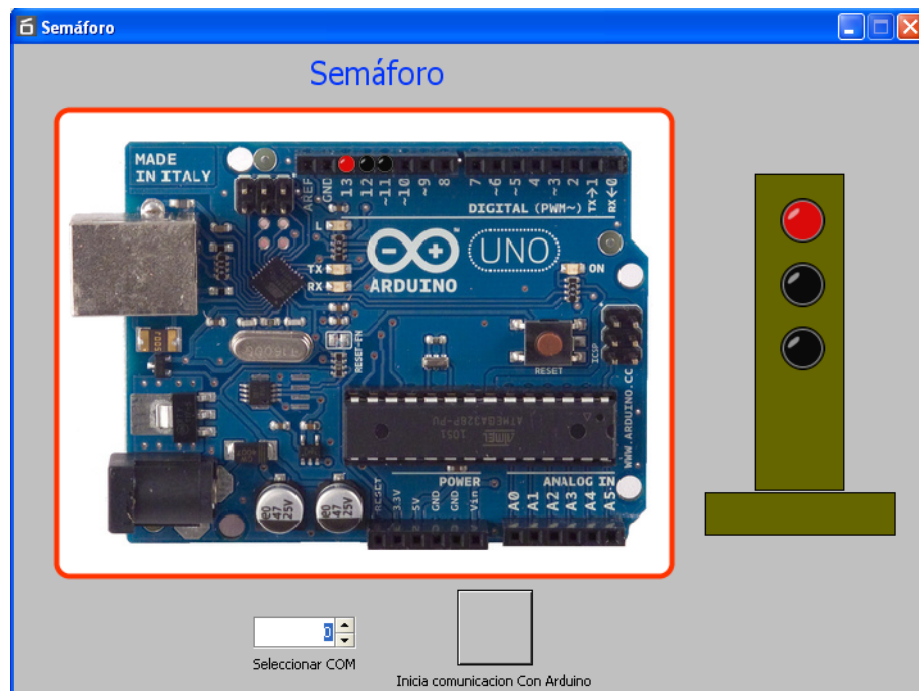
Parámetros:

tiempo_rojo=2 seg. Tiempo_ambar=2 seg. Tiempo_verde=0,5seg.

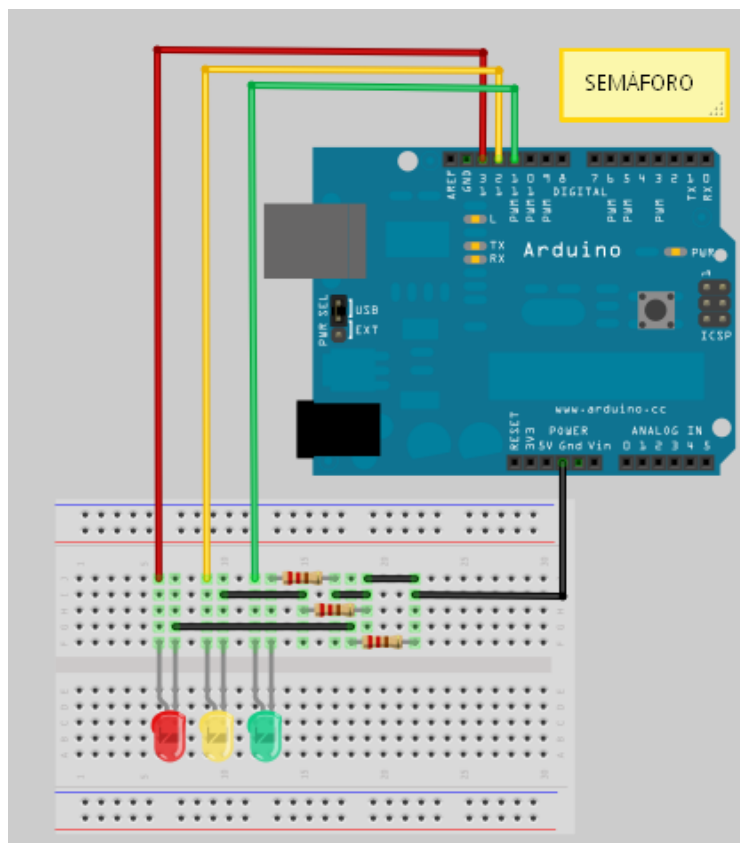
El algoritmo es muy sencillo. Se trata de activar las señales correspondientes a las tres lámparas del semáforo con intervalos d tiempo fijados con el bloque **wait**

Esquema del Panel de Circuito de MyOpenLab





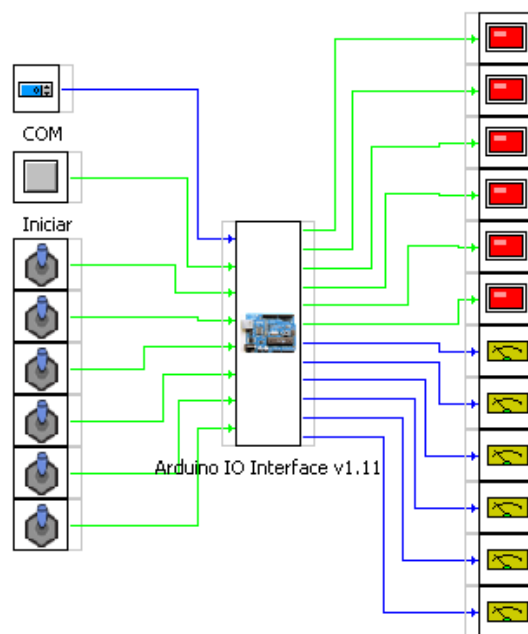
Este es el panel Frontal en modo ejecución del semáforo



Montaje sobre protoboard

10. Test Tarjeta

Con este montaje se pretende realizar el test de la tarjeta Arduino manejando todas las E/S que se programan en el firmware que permite y establece la comunicación con MyOpenLab



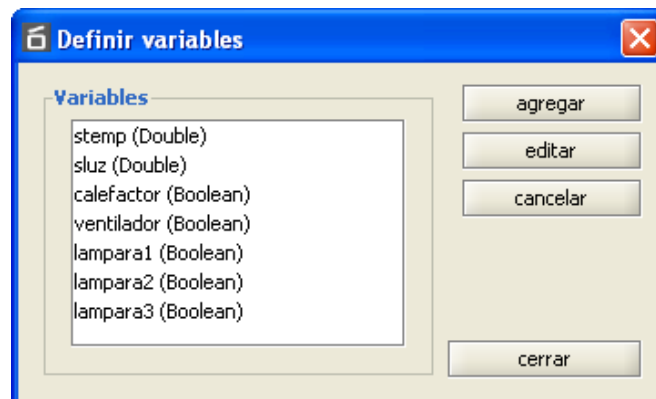
En realidad no hay ningún elemento de control solo interruptores, leds e instrumentos de panel para medir los valores de los canales analógicos.



11. Confort

Con esta práctica nos introducimos en el mundo de la Domótica. Se trata de poder controlar la activación de tres lámparas en un dormitorio en función de la cantidad de luz que midamos mediante un sensor de luz y por otro lado controlar el encendido de un radiador eléctrico también haciendo uso de un sensor, en este caso de temperatura.

Las señales que debemos definir y manejar son las indicadas en la figura siguiente.



Los señales son **stemp** Sensor de temperatura y **sluz** sensor de luz y las salidas de gobierno de las lámparas son **lampara1**, **lampara2** y **lamapara3** y para la climatización las salidas son **ventilador** y **calefactor**

En la figura vemos el aspecto de la pantalla de iteración visual con la aplicación. Se leen los valores de los sensores y sobre la tarjeta Arduino los indicadores leds de su estado así como en el radiador y en el ventilador.



El algoritmo de control:

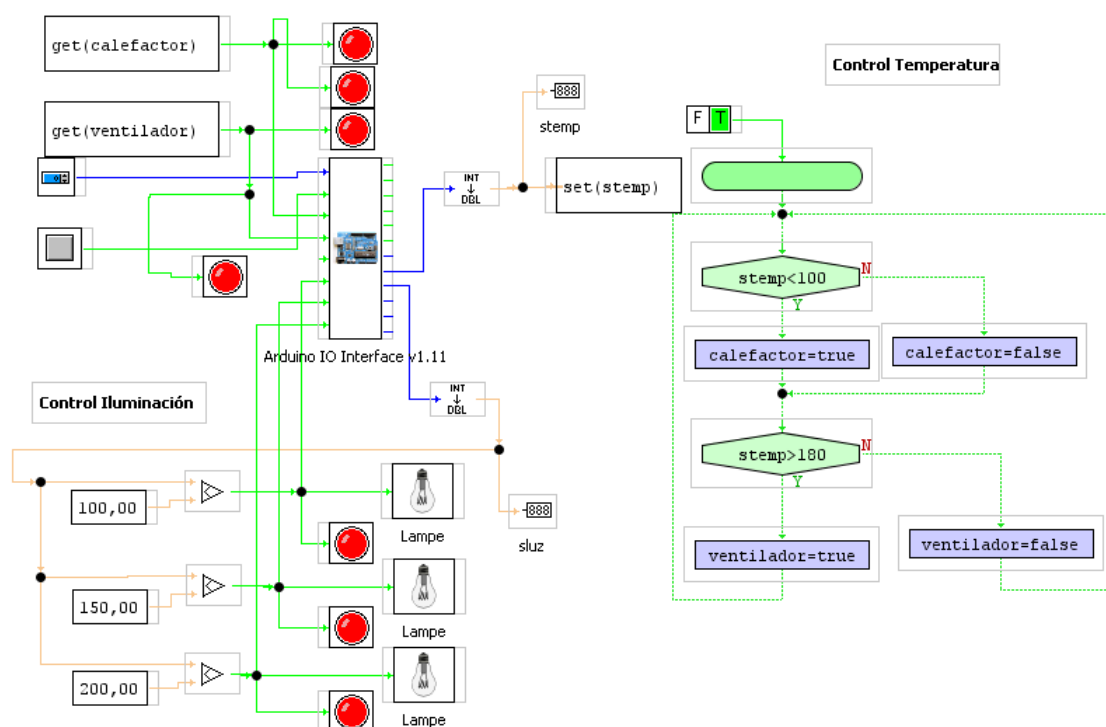
En la figura siguiente se muestra el esquema del algoritmo de control que hay que implementar. Las lámparas se encienden de acuerdo a las siguientes condiciones

Si $sluz < 100$ entonces lampara1 se enciende

Si $sluz < 150$ entonces lampara2 se enciende

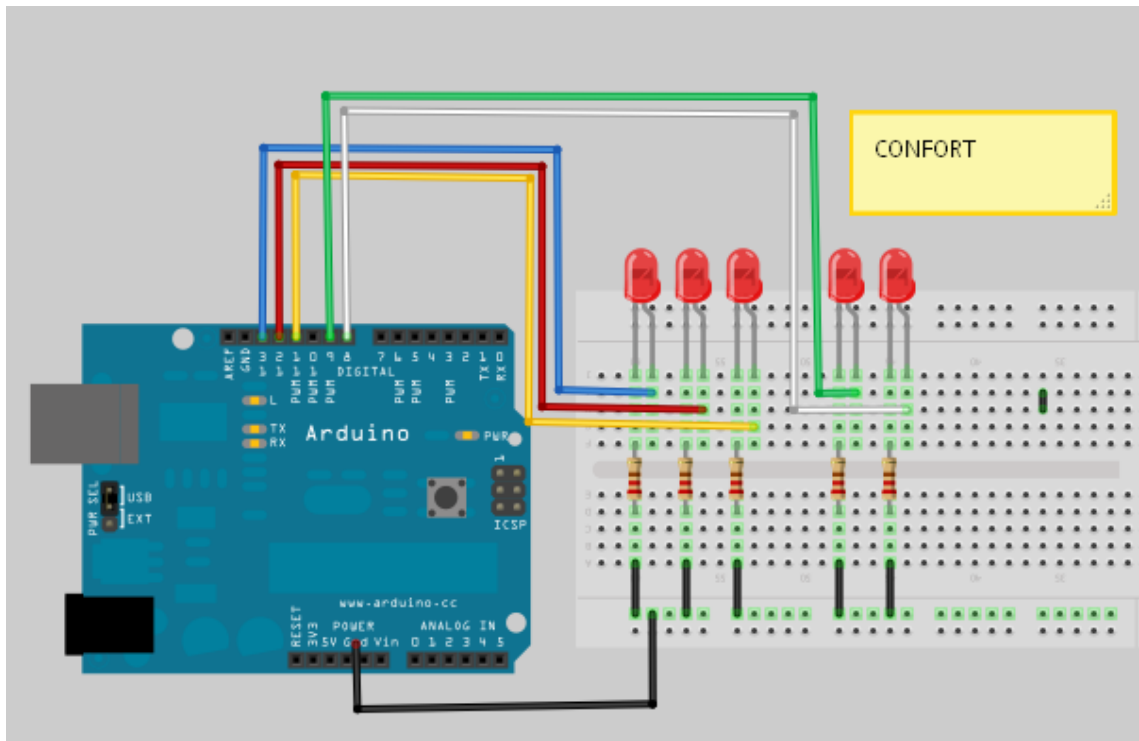
Si $sluz < 200$ entonces lámpara3 se enciende

El sistema se resuelve utilizando bloques de comparación tipo menor que.



Para el control de la calefacción y el ventilador se ha realizado un diagrama de flujo de control que recoge el valor de la variable **stemp** y lo compara con la consigna de valor **>180** para activar el ventilador y **<100** para activar el radiador

Los valores que se toman de los sensores y se muestran en la pantalla de visualización no han sido escalados a °C o Lux dado que se trata de un ejemplo puramente demostrativo

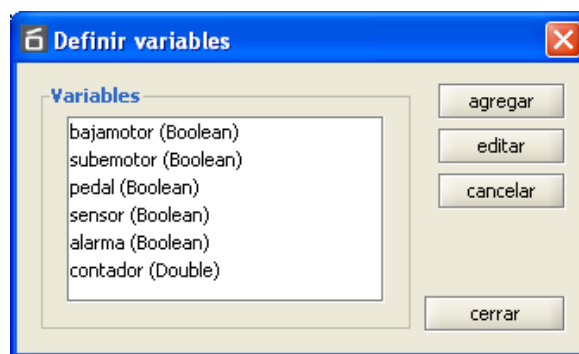


12. Prensa Hidráulica

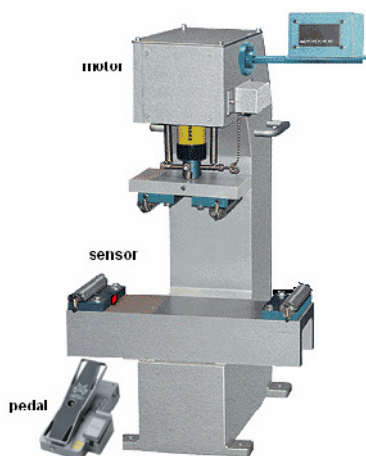
Disponemos de una prensa hidráulica que se acciona mediante un pedal de tal manera que cuando lo accionamos baja el émbolo de la prensa y permanece bajado durante un tiempo de 0,8 seg. Al cabo del cual sube el cilindro y se vuelve a su posición de reposo para quedar en situación de volver a realizar otra operación de prensado.

La prensa dispone de un sensor en la mesa de tal manera que si el operario tiene la mano sobre esta se interrumpe la barrera del sensor y esta señal impide que baje el cilindro. Al activarse el sensor se encenderá una lámpara roja de alarma.

El motor se gobierna mediante dos señales “*bajamotor*” y “*subemotor*”



Señales a tener en cuenta:



bajamotor: Acciona el motor para que baje el cilindro (booleana) RLB (Salida Out 1) **subemotor:** Acciona el motor para que suba el cilindro (booleana) RLS (Salida Out 2)

pedal: Orden de actuación al pulsar el pedal (booleana) (Entrada Inp 5)

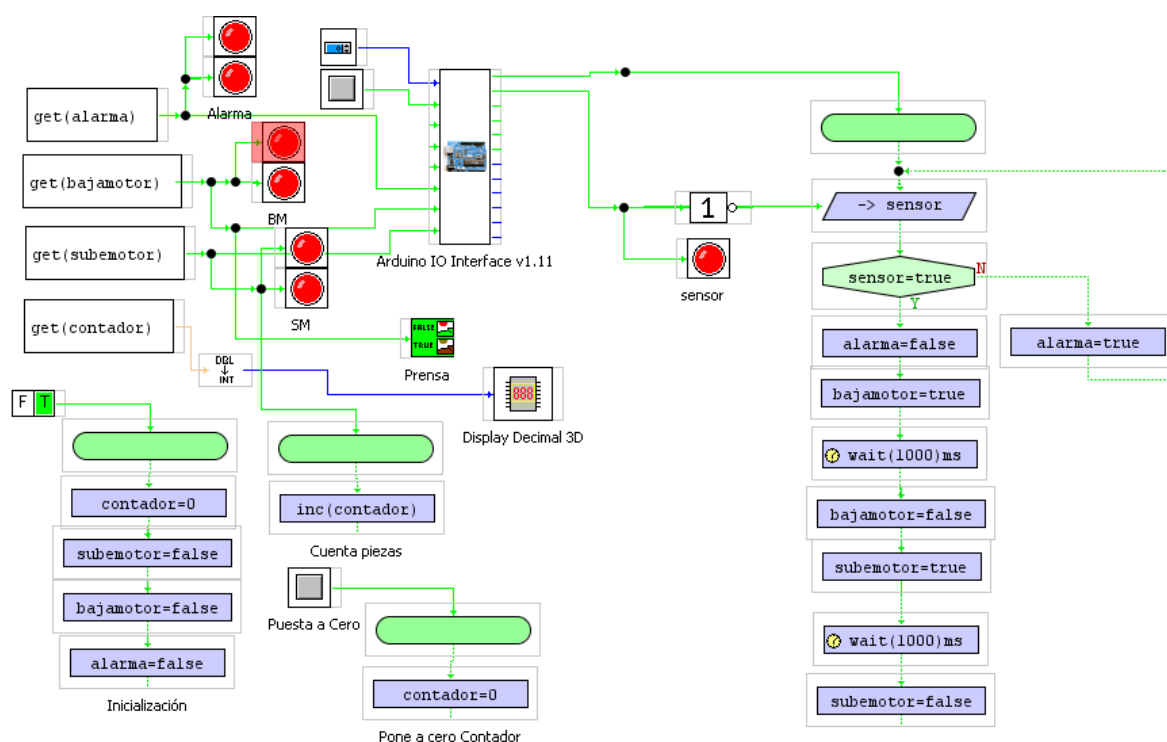
sensor : Sensor de seguridad de la barrera fotoeléctrica (booleana) (Entrada Inp 4) **alarma:** Señalización de alarma para el caso de que el sensor este activado (booleana) (Salida Out 3)

Funcionamiento

- Cuando se active el pedal la prensa (pedal=true) deberá bajar e cabezal (bajamotor=true) siempre y cuando la señal que llegue del sensor de la mesa sea sensor=false, en caso contrario no bajara la prensa. Se dispondrá de un indicador de la señal del sensor que nos pondrá en aviso de que hay una

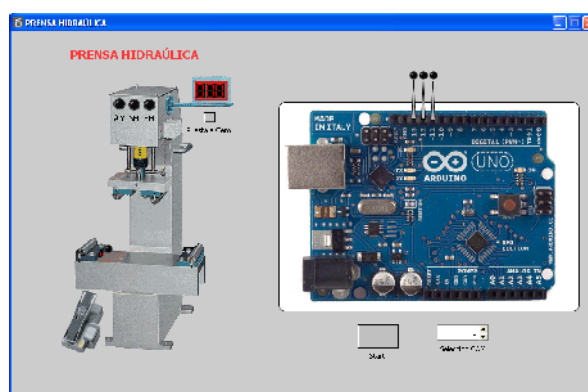
alarma (alarma=true).

- La prensa una vez que llega abajo permanecerá allí 1 seg. Para después retornar (submotor=true y bajamotor=false). Una vez arriba permanecerá la señal activa 1 seg. y de nuevo el sistema vuelve a reposo.
- Se dispondrá un contador de piezas que se activara cada vez que submotor=true (pieza terminada) y también se dispondrá de un pulsador de puesta a cero.



Esquema de funcionamiento diseñado en el Panel de Circuito

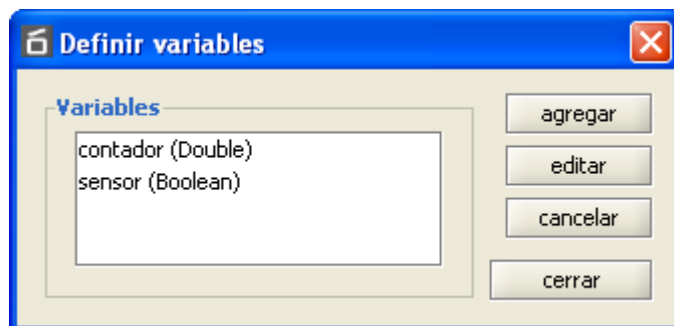
A continuación se muestra el Panel de visualización en modo ejecución de MyOpenLab



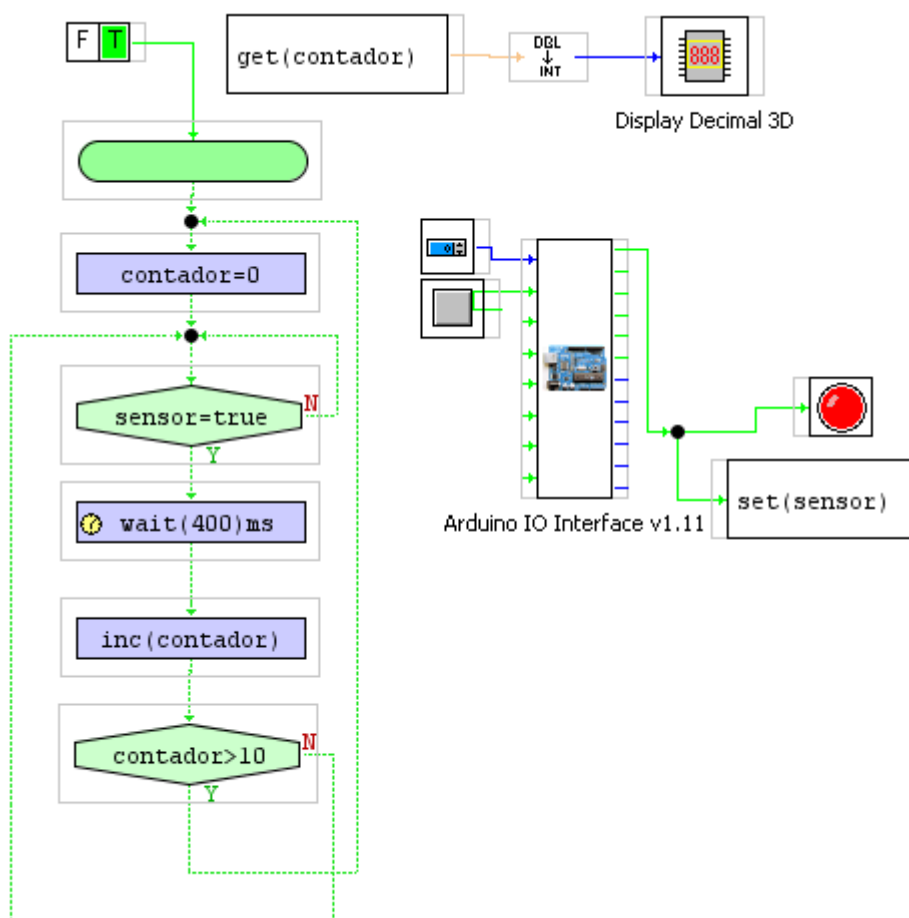
13. Contador de impulsos de entrada

Se trata de contar hasta 10 impulsos que procedan de una de las entradas de Arduino

Se definirán dos variables: **contador** que recoge el valor del número de impulsos, y **sensor**, que es la entrada por donde entraran los impulsos que debemos contar.



Definición de variables



El algoritmo es muy sencillo. Se trata de recoger la variable de entrada (estado de la entrada sensor) y si es uno se incremente la variable contador **inc(contador)**.

La variable contador se recoge y se muestra mediante el bloque **get(contador)** a una display numérico. La variable contador es de tipo *double* y el display deber recoger una variable de tipo *integer* por lo tanto debemos convertir un tipo en el otro con un bloque de conversión **dbl -> int**



14. Parking

Diseñar un Parking de acuerdo a las siguientes características:

Los coches al entrar tienen que recoger un ticket junto a la “barrera de entrada” e inmediatamente que lo recojan se levantará esta dejando pasar el coche. A la entrada habrá un semáforo con dos lámparas una verde (libre) y otra roja (lleno).

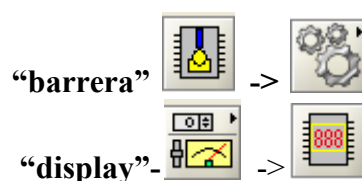
Se dispondrá de un contador de coches que nos indicará en todo momento los coches que hay dentro. Este contador se debe incrementar cada vez que llega un coche y decrementar cada vez que sale. La salida del coche se detectará con un sensor y se subirá la “barrera de salida”. En el parking caben 11 coches

La actuación de las barreras se simplificará de tal manera que cuando se recibe la orden se subir (sensor de entrada o sensor de salida) se suben y transcurrido 1,5 seg. Se bajan.

Señales a tener en cuenta:

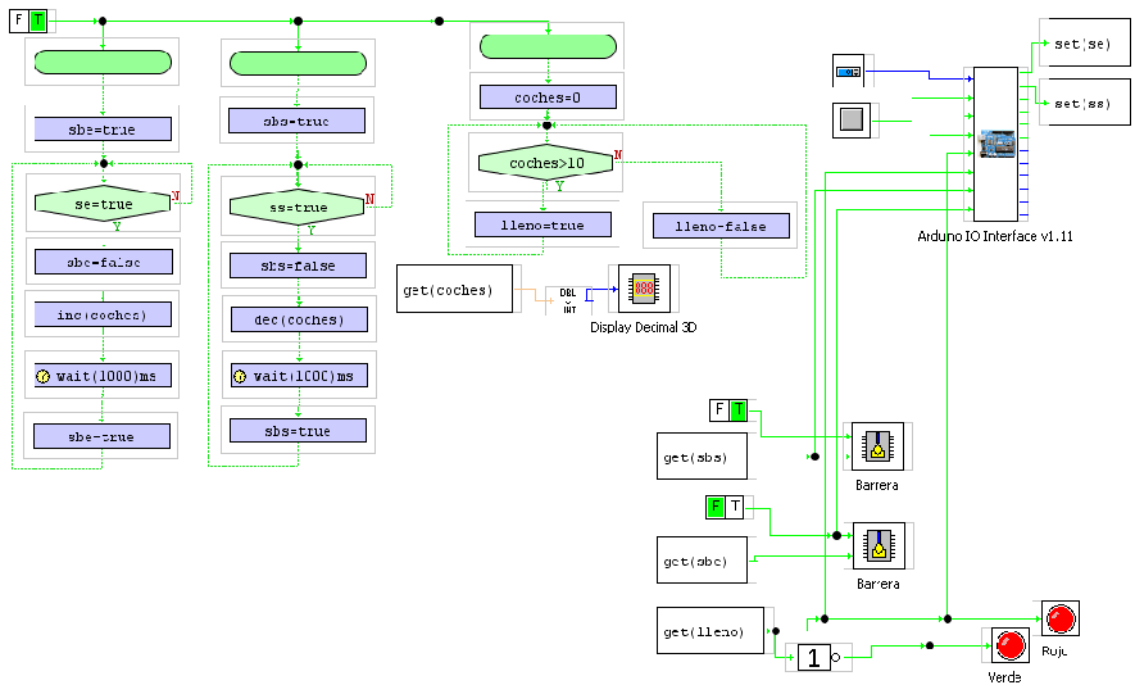
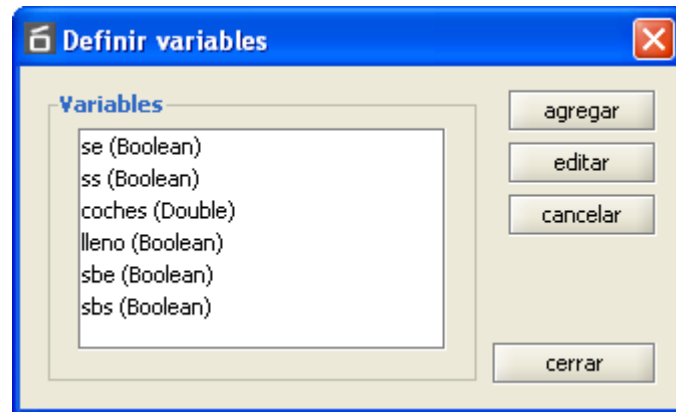
Sensor de entrada de coche **se** (booleana)
 Sensor de salida de coche **ss** (booleana)
 Numero de coches dentro del parking **coches** (double)
 Parking lleno **lleno** (booleana)
 Sube barrera de entrada **sbe** (booleana)
 Sube barrera de salida **sbs** (booleana)

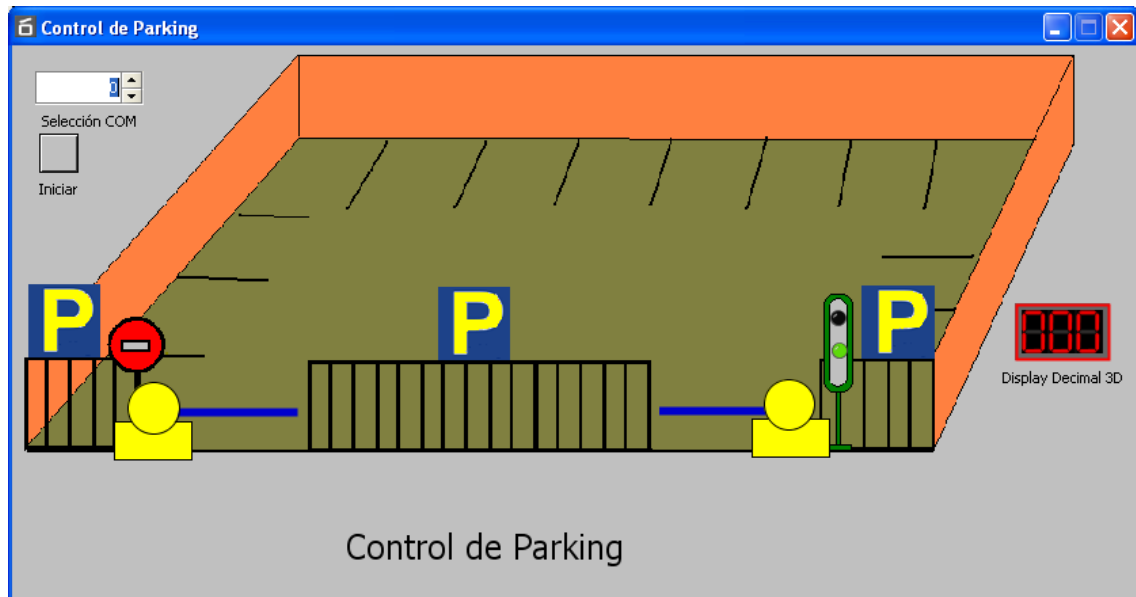
Librerías especiales que se deben utilizar:



Se podrá dibujar el esquema del parking más o menos como el que se pone en la

solución.





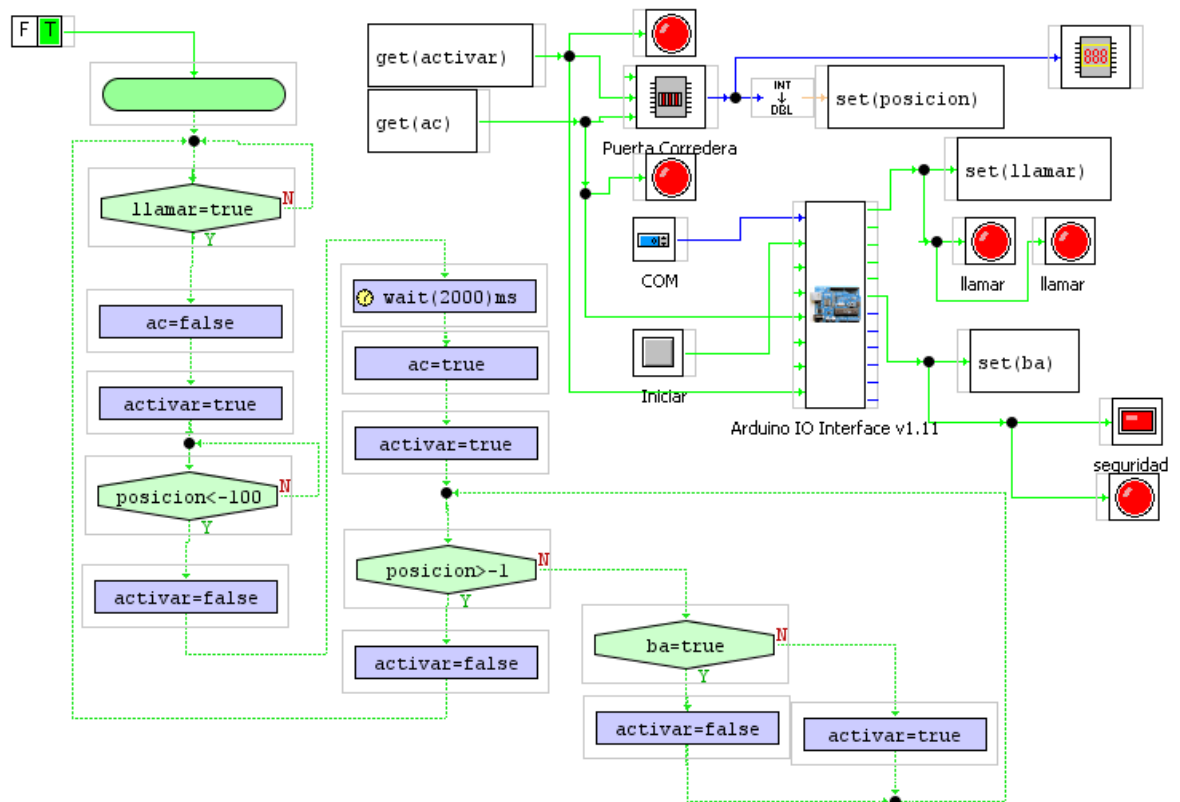
Comentario a la solución.

Obsérvese que se han realizado varios diagramas de flujo con el fin de facilitar la comprensión del funcionamiento. No debemos olvidar que se pueden ejecutar varios diagramas a la vez.

Diagramas realizados:

Tratamiento de entrada de coche (se testea la señal se) Tratamiento de salida de coche (se testea la señal ss) Lectura permanente de los sensores se y ss
Realización del conteo de los coches (entrantes y salientes)

Solución:



15. MÁQUINA DE CAFÉ

Se trata de realizar la simulación del funcionamiento de una máquina expendedora de bebidas (café, te y manzanilla).

Para su funcionamiento se deben cumplir las siguientes condiciones:

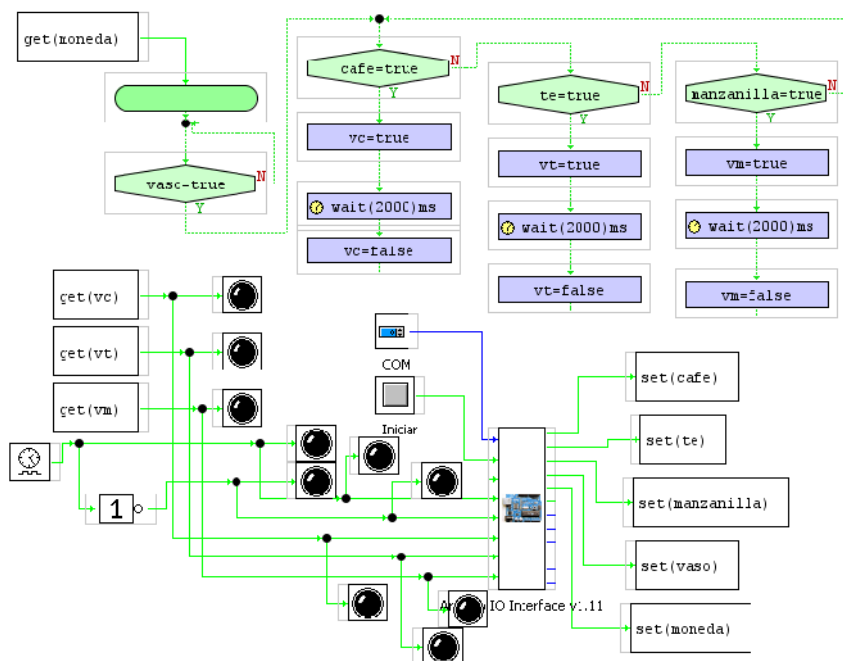
El ciclo de trabajo se iniciará cuando coloquemos una moneda en el lugar correspondiente (pulsador moneda). A continuación se deberá colocar un vaso en el lugar correspondiente lo cual hará que se produzca un impulso en el sensor de vaso (pulsador vaso). A continuación debemos seleccionar una de las tres opciones de bebida a suministrar: té, café o manzanilla. Se pulsará el correspondiente pulsador y se activará la electro válvula de salida de la bebida que se mantendrá activada un tiempo de 2 seg. Transcurrido este tiempo la máquina debe estar dispuesta a realizar otro ciclo de suministro de bebida.



Señales a tener en cuenta:

moneda (boolean) pulsador que simula la entrada de la moneda. **vaso** (boolean) pulsador que simula la existencia de un vaso. **cafe** (boolean) pulsador de petición de café
te (boolean) pulsador de petición de te **manzanilla** (boolean) pulsador de petición de manzanilla.



Solución:

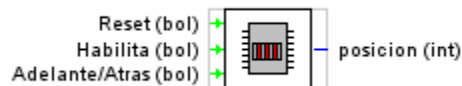




Panel de visualización de la aplicación en modo ejecución

16. PUERTA DE ENTRADA A UNA FINCA

Se trata de diseñar el automatismo de una puerta de una finca haciendo uso de la librería  ->  que representa una puerta que se desplaza sobre un carril y es gobernada de acuerdo con las señales que se indican en la siguiente figura



Se colocara de fondo una imagen que represente una casa o finca para darle más realismo a la simulación

La forma de actuar debe ser la siguiente: Cuando se pulsa en el “Pulsador de llamada” (PIN D7 de Arduino) la puerta comienza a abrirse (desplazamiento a la izquierda) hasta que se abre del todo. Una vez abierta estará un tiempo y comenzará la fase de cierre. Si cuando esta cerrándose la puerta se interfiere el sensor de seguridad (célula infrarroja) automáticamente la puerta se detendrá hasta que desaparezca esta señal de seguridad y pueda continuarse el cierre.

Las señales a tener en cuenta son:

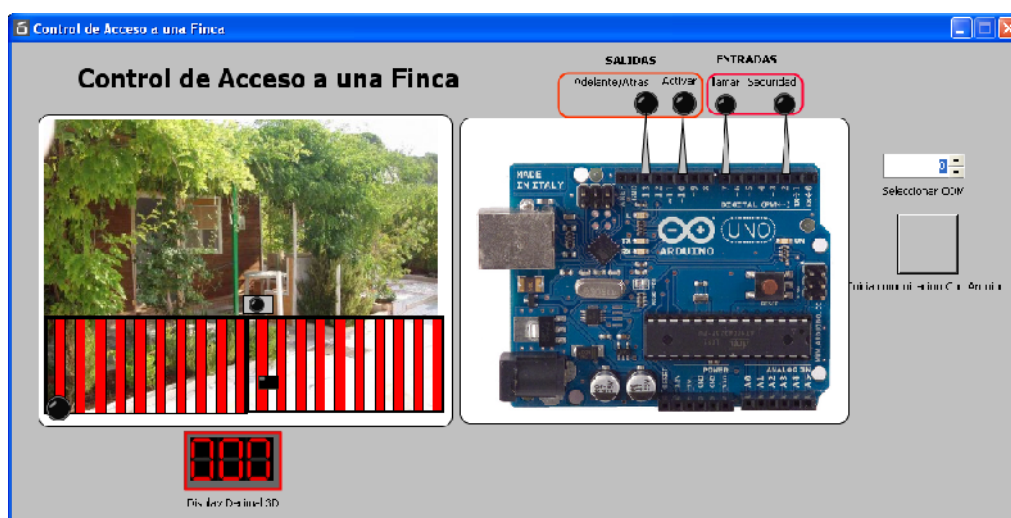
llamar (boolean) inicia el ciclo de apertura de la puerta (PIN D7 de Arduino).

activar (booleana) activa el movimiento de la puerta (PIN D13 de Arduino).

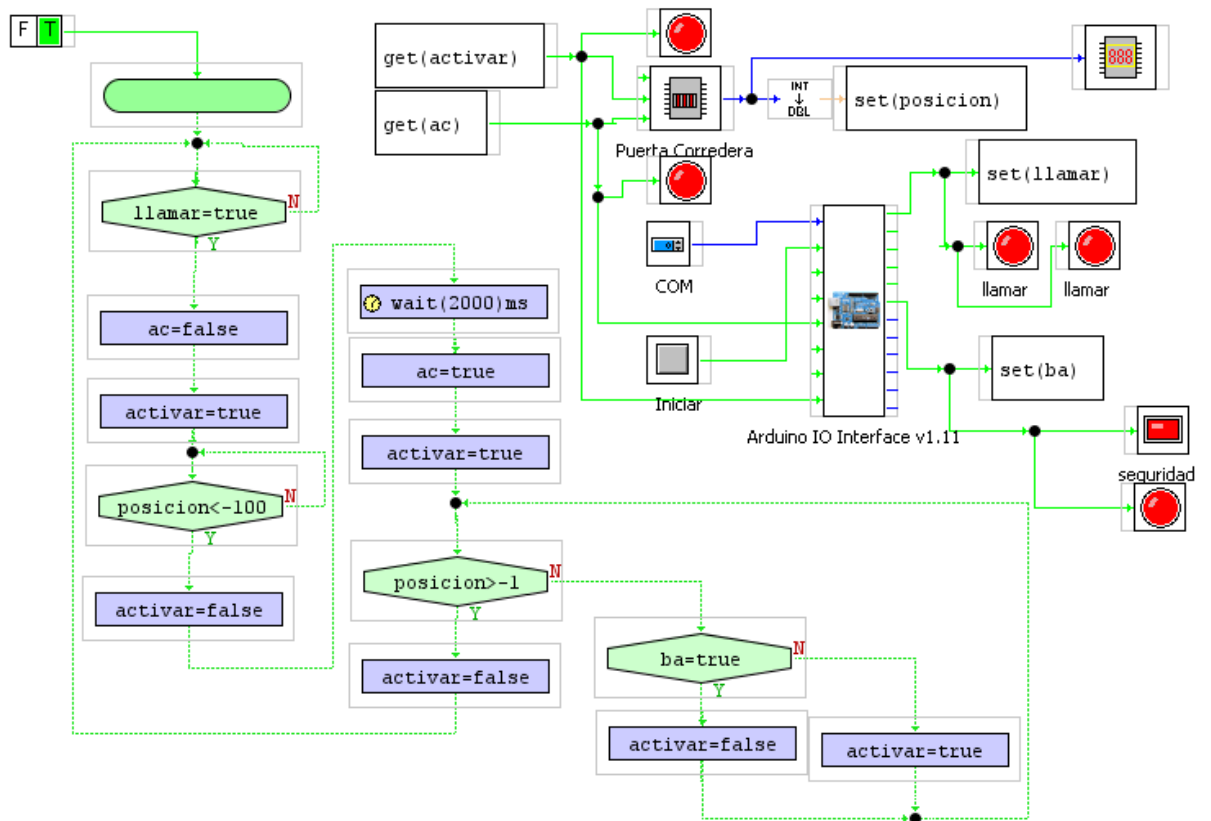
posición (double) nos indica la posición en la que se encuentra la puerta.

ac (boolean) da la orden del sentido de movimiento de la puerta (Adelante/Atrás PIN D10 Arduino)

ba (boolean) sensor de seguridad de la puerta, se activa cuando hay algún obstáculo (PIN D2 de Arduino).



La señal de **posición** debemos considerarla para que en el movimiento de apertura se detenga en un punto (posición).

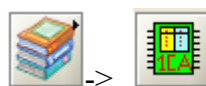


17. Datalogger Tipo 1

Con esta práctica se pretende demostrar la capacidad que tiene MyOpenLab de implementar un sistema de captura de datos procedentes de una variable y su almacenamiento en un fichero para posteriormente poder ser tratados en una hoja de cálculo como EXCEL o simplemente representados gráficamente.

Para este ejemplo vamos a utilizar el canal A2 de entrada analógica de datos de la tarjeta Arduino. A este canal vamos a conectar un sensor de iluminación y los valores leídos los vamos a almacenar en una tabla que después almacenaremos en un fichero y podremos leer cuando lo deseemos.

La captura de los datos la queremos hacer de modo automático con una cadencia de 2 segundos y el número de datos a guardar queremos que sea de 50. Con estos parámetros configuraremos el correspondiente elemento de librería de usuario.

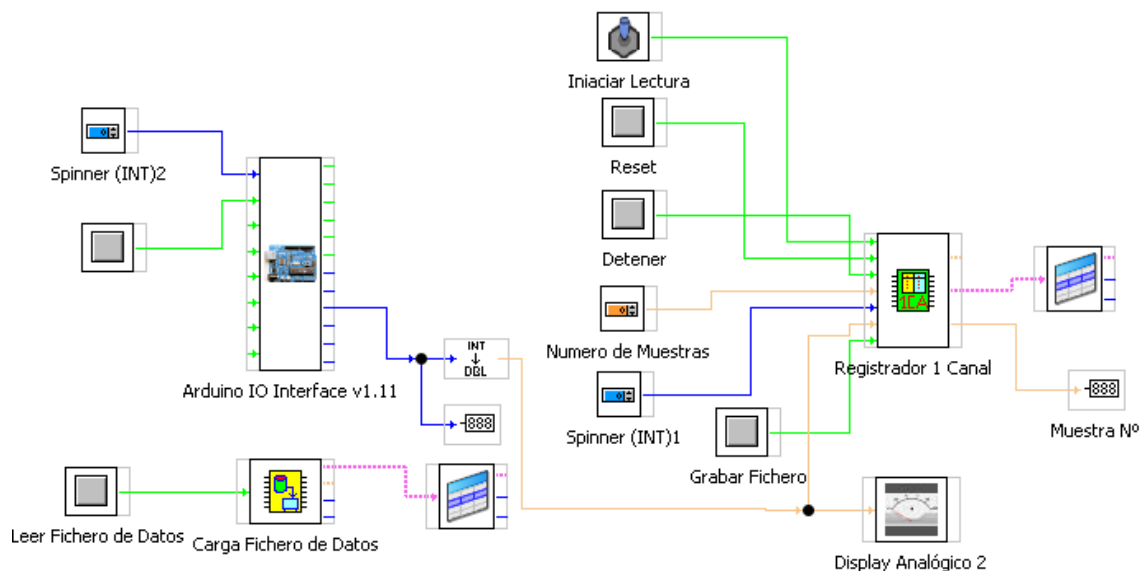


Para poder ver el contenido de ficheros de datos que ya están grabados se ha

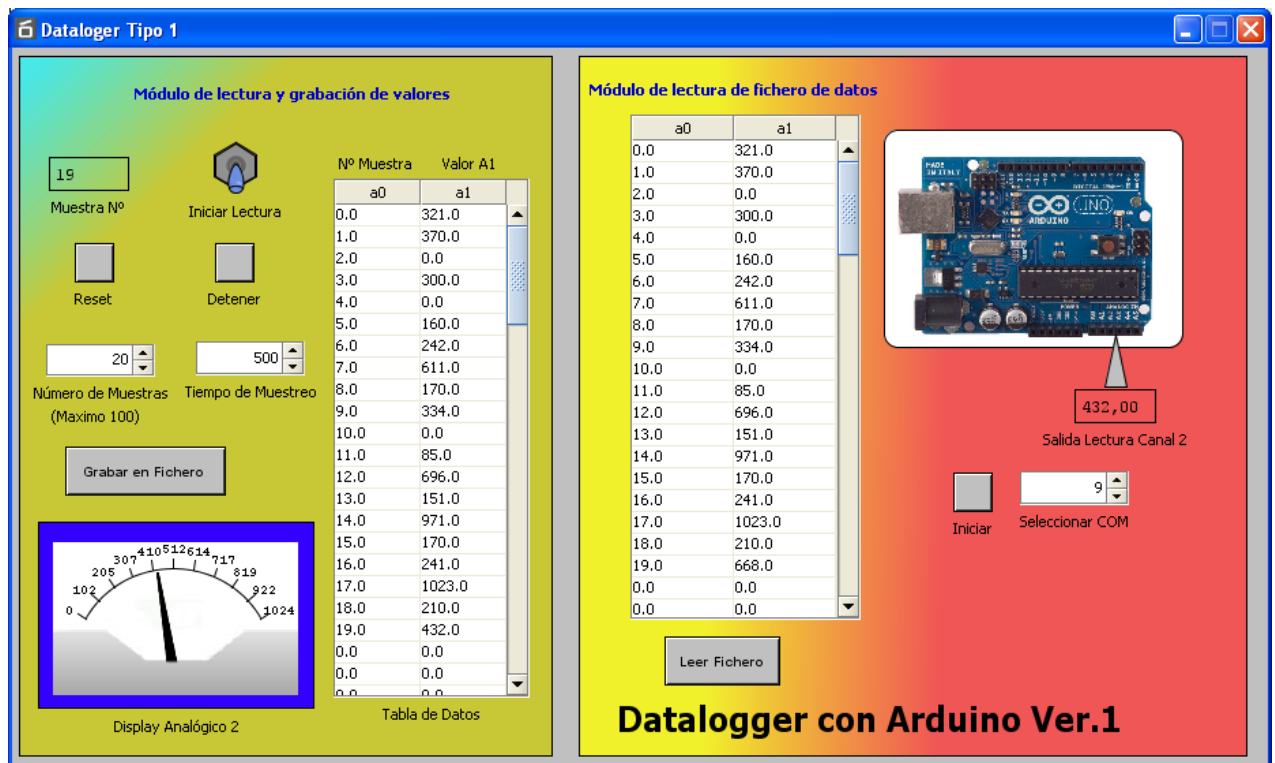
recurrido al bloque de librería de lectura de ficheros.



Este montaje es muy útil para realizar experimentos de laboratorio y guardar los datos en ficheros. Se pueden guardar los seis canales de entrada de la tarjeta Arduino tomando un número máximo de muestras es de 100.



Este sería el esquema del Panel de Circuito de la aplicación



Este sería el Panel Frontal de la aplicación en modo ejecución

18. Datalogger Tipo 2

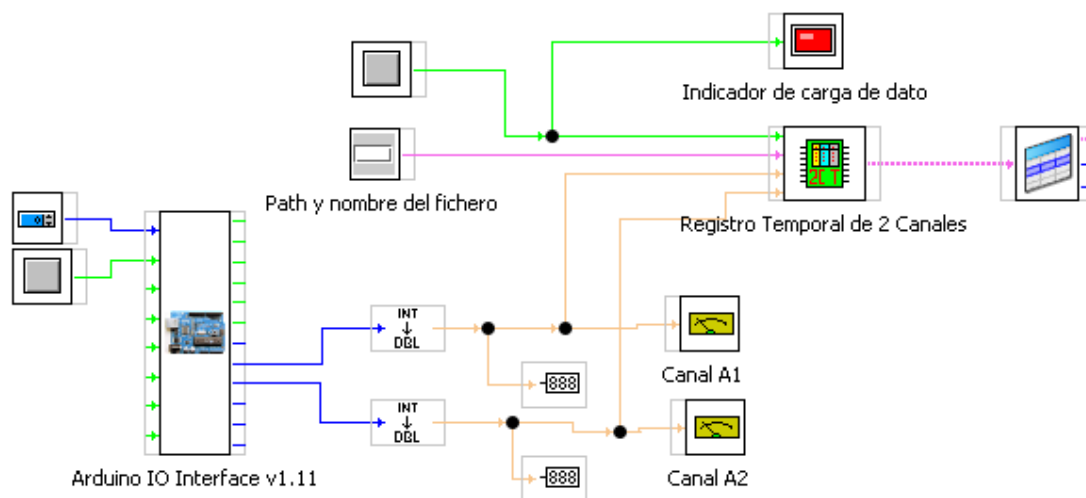
En este segundo ejemplo vamos a realizar una captura de datos de los canales A1 y A2 de la tarjeta Arduino pero esta vez grabaremos los datos acompañando cada muestra con la fecha y la hora en la que se ha tomado.

En esta aplicación las muestras se van a tomar cuando se active el botón “Lee dato”.

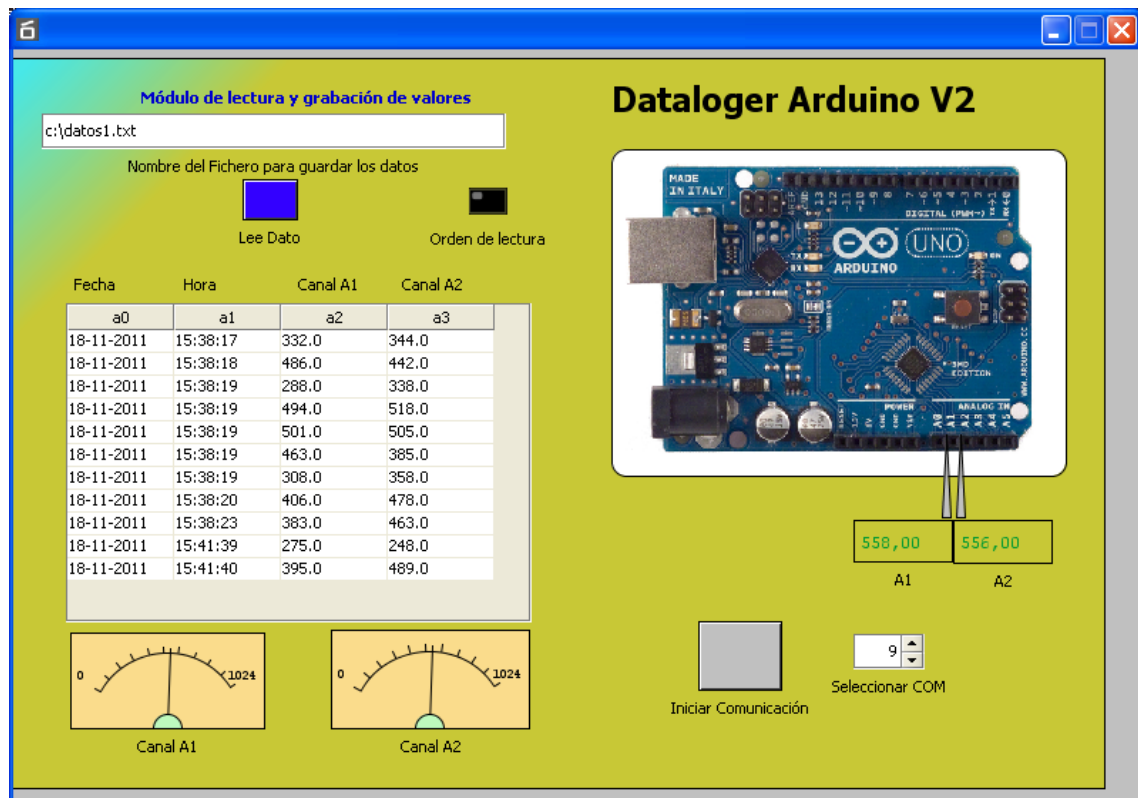
Para esta aplicación vamos a hacer uso del bloque de función de Librerías de Usuario



Es importante que antes de lanzar el programa se haya creado un fichero (ejemplo datos.txt) y guardado vacío con el fin de que cuando arranquemos el programa lo encuentre para abrirlo.



Esquema de la aplicación



Pantalla en modo ejecución

Noviembre de 2011 Versión de Documento: V1.0

José Manuel Ruiz Gutiérrez j.m.r.gutierrez@gmail.com

Blog de referencia: <http://josemanuelruizgutierrez.blogspot.com/>