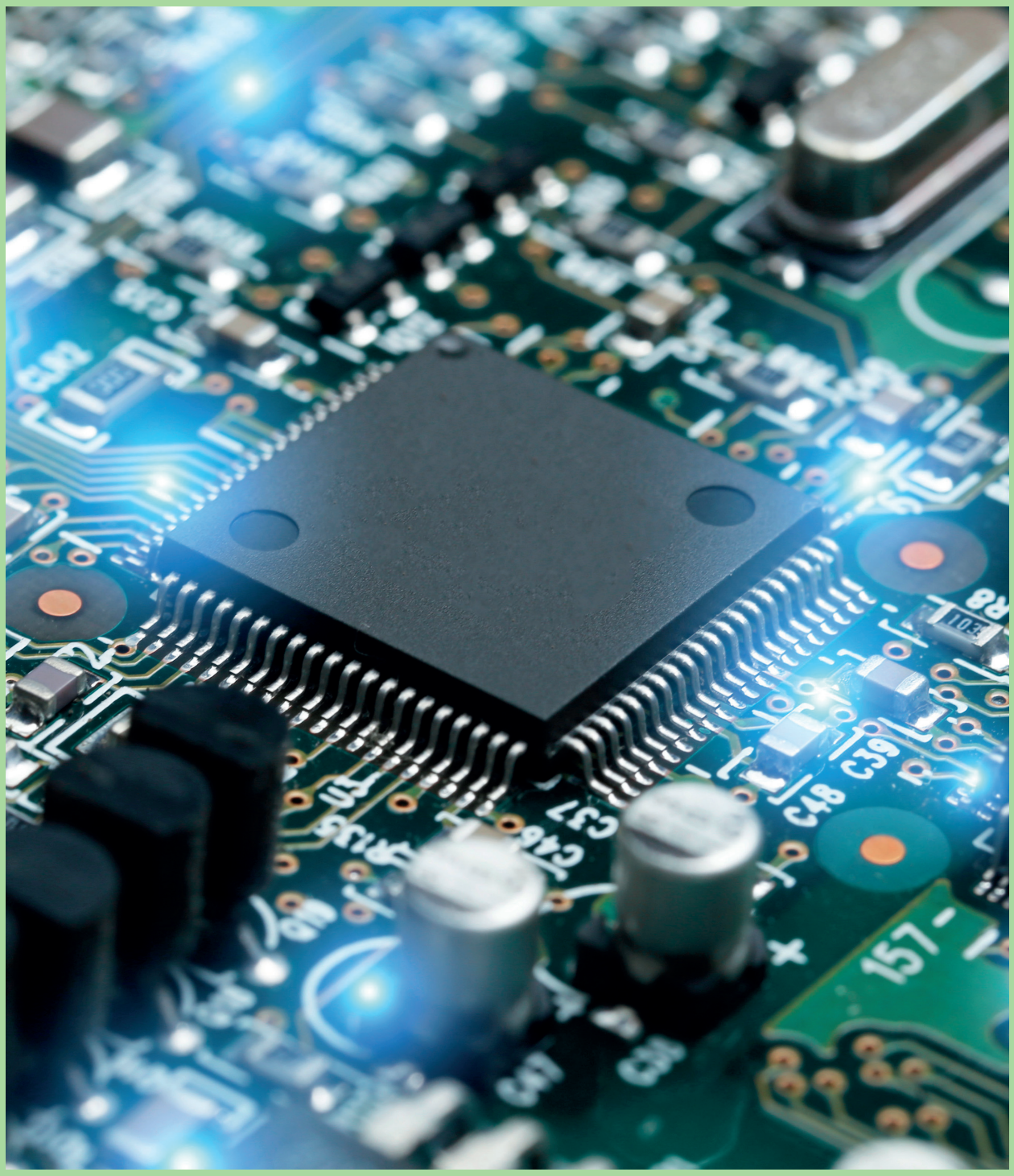
ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA



Programando Arduino como PLC

Sixto Rafael Reinoso Villamarín y Marco Antonio Pilatasig Panchi

Programando Arduino como PLC

Sixto Rafael Reinoso Villamarín, Mgtr. y Marco Antonio Pilatasig Panchi, Mgtr.

Primera edición electrónica: abril, 2023
ISBN: 978-9942-765-83-3
Revisión científica:
Ing. Fátima Adriana Avilés Castillo, Mgtr.
Ing. Jorge Luis Buele León, Mgtr.

Universidad de las Fuerzas Armadas-ESPE
Crnl. de C.S.M. Víctor Villavicencio A., Ph. D.
Rector

Publicación autorizada por:
Comisión Editorial de la Universidad de las Fuerzas Armadas-ESPE
Cpcb. Rolando Patricio Reyes Chicango, Ph.D.
Presidente

Corrección de estilo y diseño
Lcdo. Xavier Chinga

Imagen de cubierta: https://acortar.link/ahR0T8

Universidad de las Fuerzas Armadas-ESPE
Av. General Rumiñahui s/n, Sangolquí, Ecuador
www.espe.edu.ec

Sede Latacunga

Programando Arduino como PLC

Ing. Sixto Rafael Reinoso Villamarín, Mgtr.

Ing. Marco Antonio Pilatasig Panchi, Mgtr.

Dedicatoria

A Dios nuestro Creador: ¡Pero gracias a Dios, que nos da la victoria por medio de nuestro Señor Jesucristo! 1 Corintios 15:57.

A la memoria de Lucila Villamarín, mi adorada Madre.

Sixto Rafael Reinoso Villamarín
srreinoso@espe.edu.ec

Posee una formación académica de pregrado en la Escuela Politécnica del Ejercito como Ingeniero en Electrónica e Instrumentación, en la Universidad Indoamérica como Ingeniero Industrial, posgrado en la Universidad Técnica de Ambato, donde obtuvo el título de Magíster en Tecnologías de la Información y Multimedia Educativa. Tiene una amplia experiencia en educación técnica, siendo actualmente docente en la Universidad de las Fuerzas Armadas ESPE sede Latacunga, donde también ha publicado otros libros de Programación de Microcontroladores.

Marco Antonio Pilatasig Panchi
mapilatagsig@espe.edu.ec

Posee una formación de Ingeniero en Electrónica e Instrumentación en la Universidad de las Fuerzas Armadas – ESPE, Sede Latacunga, posgrado en Gestión de Energías en la Universidad Técnica de Cotopaxi, actualmente es Director de Carrera de Electrónica y Automatización de la Universidad de las Fuerzas Armadas – ESPE, Sede Latacunga, ha publicado 32 artículos indexados a revistas científicas, participó en 3 proyectos de investigación y 2 proyectos de vinculación con la sociedad.

Índice

Índice de tablas

Índice de figuras

Introducción

Arduino es una plataforma de software y hardware libre muy versátil, que es de gran utilidad para realizar diversos tipos de aplicaciones, desde el nivel de proyectos caseros hasta el industrial.

La información difundida para el desarrollo de aplicaciones con Arduino es muy amplia. En la web se encuentran sitios y páginas especializados, así como videos tutoriales, tutoriales, manuales, apuntes, entre otros recursos, lo cual ha permitido el desarrollo de aplicaciones complejas y que personas con un mínimo de conocimiento en programación y electrónica las puedan desarrollar.

Otra de las ventajas de usar Arduino es la gran cantidad de accesorios compatibles con la tarjeta que se disponen para el desarrollo de todo tipo de aplicaciones electrónicas. Es así que se tiene una gran variedad de sensores como: sensores de sonido, láser, magnético, humedad, acelerómetros, módulos de relés, Leds de todo tipo, potenciómetro encoder, pantallas LCD, TFT, tarjetas de Ethernet, drivers para manejo de motores, etcétera, etcétera. Todos los accesorios de hardware incluyen las librerías para manejar los componentes mediante el IDE de Arduino.

Por todas estas razones, la tarjeta Arduino en sus diferentes versiones se ha convertido en la herramienta más apetecida por programadores profesionales o simplemente aficionados que desarrollan proyectos electrónicos y con costos relativamente bajos comparados con otros dispositivos similares.

Los lenguajes de programación de Arduino en su mayoría son de alto nivel, pero muy pocos utilizan los diagramas Ladder, que son el puntal de la programación de PLCs para la automatización de procesos industriales. El presente texto emplea la programación de Arduino mediante lenguaje Ladder utilizando software libre, lo que lo hace una obra original y única en su categoría. Los ejercicios desarrollados y propuestos son el resultado de la experiencia profesional y docente del autor en la rama de automatización industrial.

Todos los circuitos tratados en el presente texto han sido probados en los simuladores LDmicro, ISIS de Proteus, CADe-SIMU, PC-SIMU, e implementando en la misma tarjeta Arduino, lo que garantiza la funcionalidad de los mismos.

CAPÍTULO I

Fundamentos

Hardware

Arduino es una plataforma de open source, y en consecuencia de esto, se han desarrollado una diversidad de tarjetas, entre las cuales se encuentra Arduino Uno, siendo una de las tarjetas más exitosas en la implementación de proyectos electrónicos básicos y avanzados. Por esta razón, el texto se basa en el desarrollo de aplicaciones utilizando esta tarjeta. Además, la mayoría de softwares, para la programación, conexión de periféricos y otras aplicaciones utilizan como referencia a la tarjeta Arduino Uno. Para el lector interesado en conocer más detalles sobre las características de las demás tarjetas en sus diferentes versiones, recomendamos ingresar a los sitios webs:

- http://arduprojects.blogspot.com/2012/09/tipos-de-arduino.html
- https://www.bolanosdj.com.ar/MOVIL/ARDUINO2/tiposarduino.pdf
- https://hacedores.com/cuantos-tipos-diferentes-de-arduino-hay/

Arduino Uno

Todas las tarjetas de Arduino Uno en sus diferentes versiones utilizan el microcontrolador ATmega328 y son completamente compatibles entre sí.

Figura 1

Tarjetas Arduino Uno R3

Nota. Tomado de: http://proyectosconarduino.com/curso/arduino-uno-r3/

Características

La tarjeta Arduino Uno R3, presenta las siguientes características:

- Procesador: microcontrolador ATmega328P.

- Voltaje de alimentación: 5VDC.
- Entradas/Salidas: 14 pines digitales, de estos 6 proveen salida PWM.
- Entradas analógicas: 6 pines.
- Intensidad de corriente CC en los pines de entrada/salida: 40mA.
- Intensidad de corriente CC en el pin de 3.3VDC: 50 mA.
- Memoria Flash: microcontrolador ATmega328P, 32 KB; usa 0.5KB en el bootloader y el resto está disponible para el usuario.
- Memoria EEPROM: 1 KB.
- Memoria estática SRAM: 2KB.
- Frecuencia de reloj: 16MHz (Asin, 2021).

Software

Muchos técnicos y desarrolladores de programas para automatización están más acostumbrados al uso de los diagramas de contactos eléctricos, denominados Ladder o de escalera, un "lenguaje" nativo de programación para PLCs. El uso de los microcontroladores en muchas ocasiones resulta mucho más conveniente por factores económicos para el desarrollo de aplicaciones de automatización para proyectos donde los PLCs resultan muy caros. Considerando que no existen textos escritos con el contenido que vamos a abarcar, pensamos que el texto será de una obra de mucha utilidad para las personas que les gusta la programación en diagramas Ladder.

Por esta razón, se han desarrollado softwares que permitan programar a los microcontroladores con diagramas Ladder como si se tratara de un PLC. LDmicro es un software open source de distribución libre. LDmicro permite editar, simular y compilar diagramas en lenguaje Ladder para microcontroladores de 8 bits: los AVR-Atmel y los PIC16 de Microchip. Al compilar genera varios archivos para que el usuario pueda utilizarlo según su conveniencia, el archivo con el código binario en formato hexadecimal es útil para cargar en la tarjeta Arduino y para usar en simuladores como Proteus. También en este mismo programa —usando el diagrama Ladder— se puede comprobar el funcionamiento mediante la simulación. Estas características hacen de LDmicro un programa completo, de fácil uso y acceso para el desarrollo de programas basados en diagramas Ladder.

LDmicro, entre otras las instrucciones incluyen:

- Entradas y salidas digitales.

- Temporizadores (TON, TOF, RTO, osciladores).
- Contadores (CTU, CTD, 'contadores circulares' para usar como un secuenciador).
- Entradas analógicas y salidas analógicas basadas en señales PWM.
- Variables tipo enteras e instrucciones para cálculos aritméticos, de movimiento y desplazamiento.
- Comunicaciones seriales fáciles de usar, a un LCD, PC, displays de 7 segmentos u otro dispositivo.
- Tablas de consulta y registros de desplazamiento.
- Variables de memorias EEPROM.
- Compilador, para probar su programa antes de generar código PIC/AVR.
- Genera Sketch en código de alto nivel (lenguaje C++) para usar con el IDE de Arduino.
- Para microcontroladores PIC genera códigos para programación CCS, AVR-GCC, CodeVisionAVR C, Pascal, entre varios más.

LDmicro es un software, que se puede descargar el código fuente y el programa ejecutable en las diferentes versiones que existen, del sitio web:

https://github.com/LDmicro/LDmicro/releases

En el libro el desarrollo y comprobación de los programas se utilizó la versión 4.3.8.3. A la fecha de edición final del libro está disponible la versión 5.4.1.1 que trae nuevas instrucciones. Se realizó algunos ejercicios con esta versión y los resultados fueron satisfactorios. Todas las versiones de LDmicro son mantienen la compatibilidad, por lo que garantiza la migración de una versión inferior a una superior.

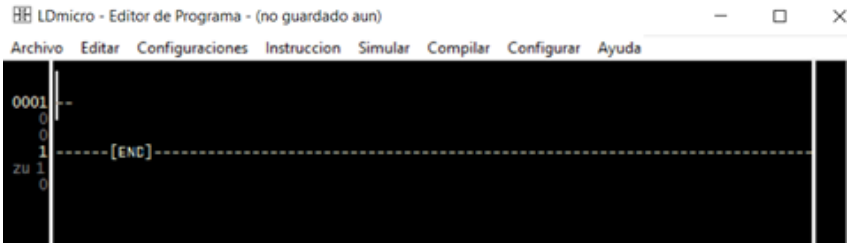
Referencia rápida de LDmicro

Luego de descargar y descomprimir el programa, se tiene varios archivos ejecutables en diferentes idiomas, el lector puede utilizar el que más se adapte a sus necesidades.

Al ejecutar el programa en español (ldmicro-es.exe), abre la ventana de edición del programa, como indica la figura de la página siguiente.

Figura 2

Pantalla editora del LDmicro



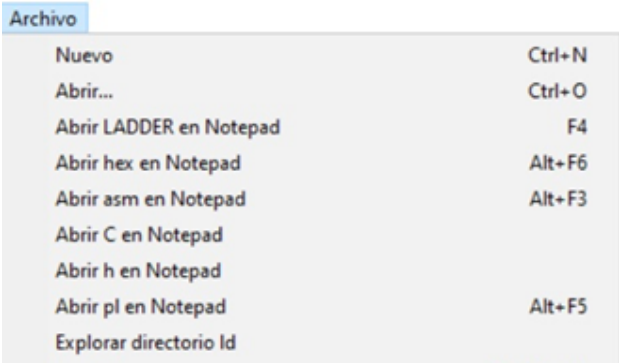
En los menús desplegables se encuentran todos los comandos para la edición, modificación, simulación y compilación del programa. Revisemos brevemente el contenido de cada menú.

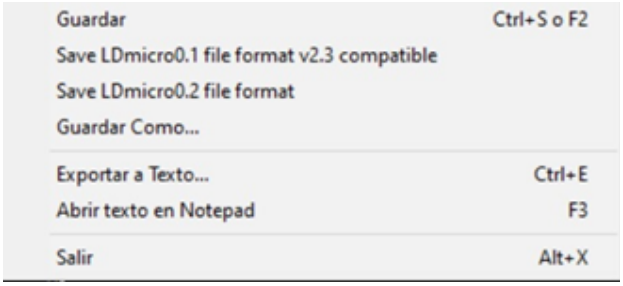
Menú Archivo

Contiene los comandos tradicionales para crear un nuevo programa: *Nuevo*, abrir un programa existente: *Abrir, Guardar como* y *Guardar* que trabajan de forma similar a los programas del sistema operativo Windows, y otros comandos que permiten exportar en formato de archivo de texto: Exportar a Texto. LDmicro genera tres archivos con extensiones .ld, .pl, .hex. El archivo .ld, es el diagrama *Ladder*, archivo .pl. El archivo .pl, es un archivo de listado de instrucciones en lenguaje de alto nivel. Se puede abrir en archivo de texto utilizando la opción *Open* pl in notedpad. El archivo .hex es lenguaje de máquina usado para cargar en microcontrolador o en el simulador ISIS. Este archivo también puede ser abierto con Open hex in notedpad. Mediante la opción *Open* ld in notedpad, se abre en el bloc de notas un listado de la descripción con los elementos que forman el diagrama *Ladder*.

Figura 3

Opciones del menú Archivo





Menú Editar

En la pestaña *editar*, se puede insertar, copiar, pegar, eliminar, duplicar o mover los Rung del diagrama Ladder según las necesidades de diseño.

Figura 4

Comandos del menú editar

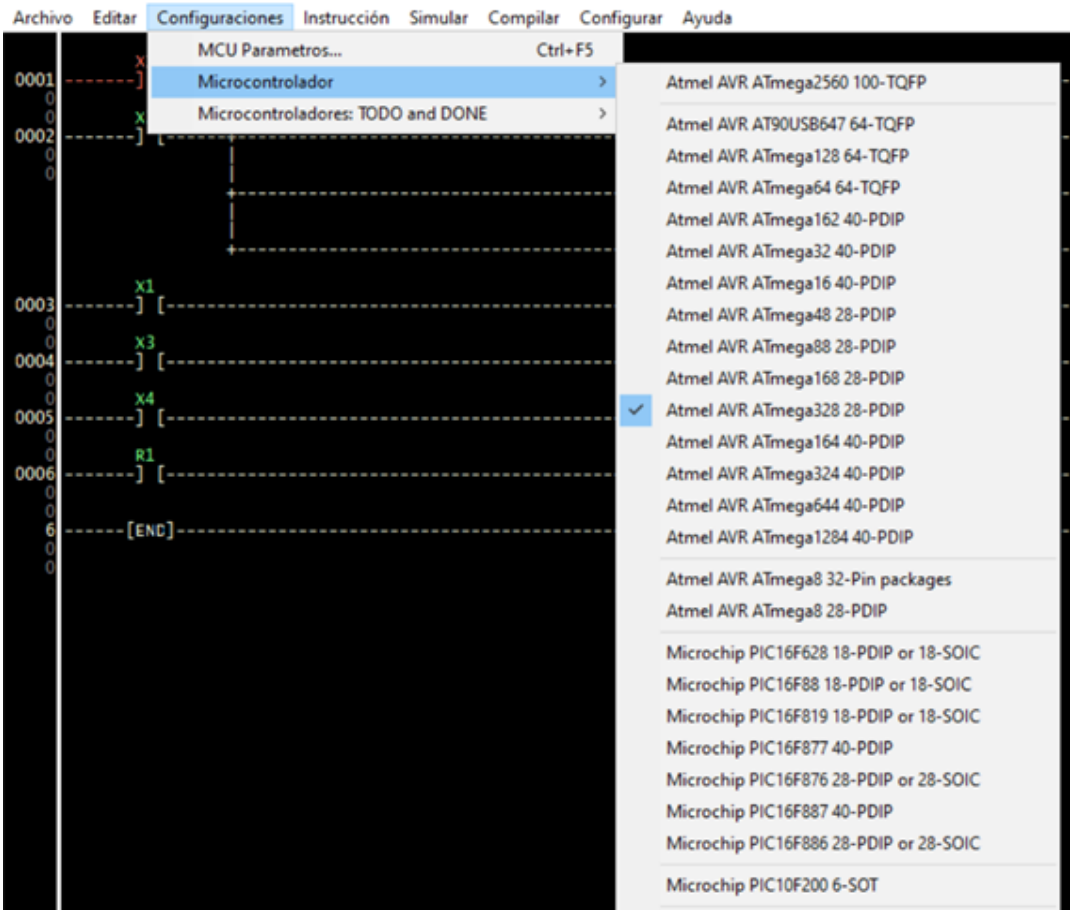
Menú Configuraciones

En este menú, se puede configurar los parámetros y tipo de microcontrolador que se utilizará en el proyecto.

Para realizar un programa el primer paso será seleccionar el tipo de microcontrolador. Vamos utilizar la tarjeta Arduino Uno, que tiene el microcontrolador ATmega328. Esta acción lo realizamos en la pestaña Configuraciones/Microcontrolador/Atmel AVR Atmega328 28-PDIP.

Figura 5

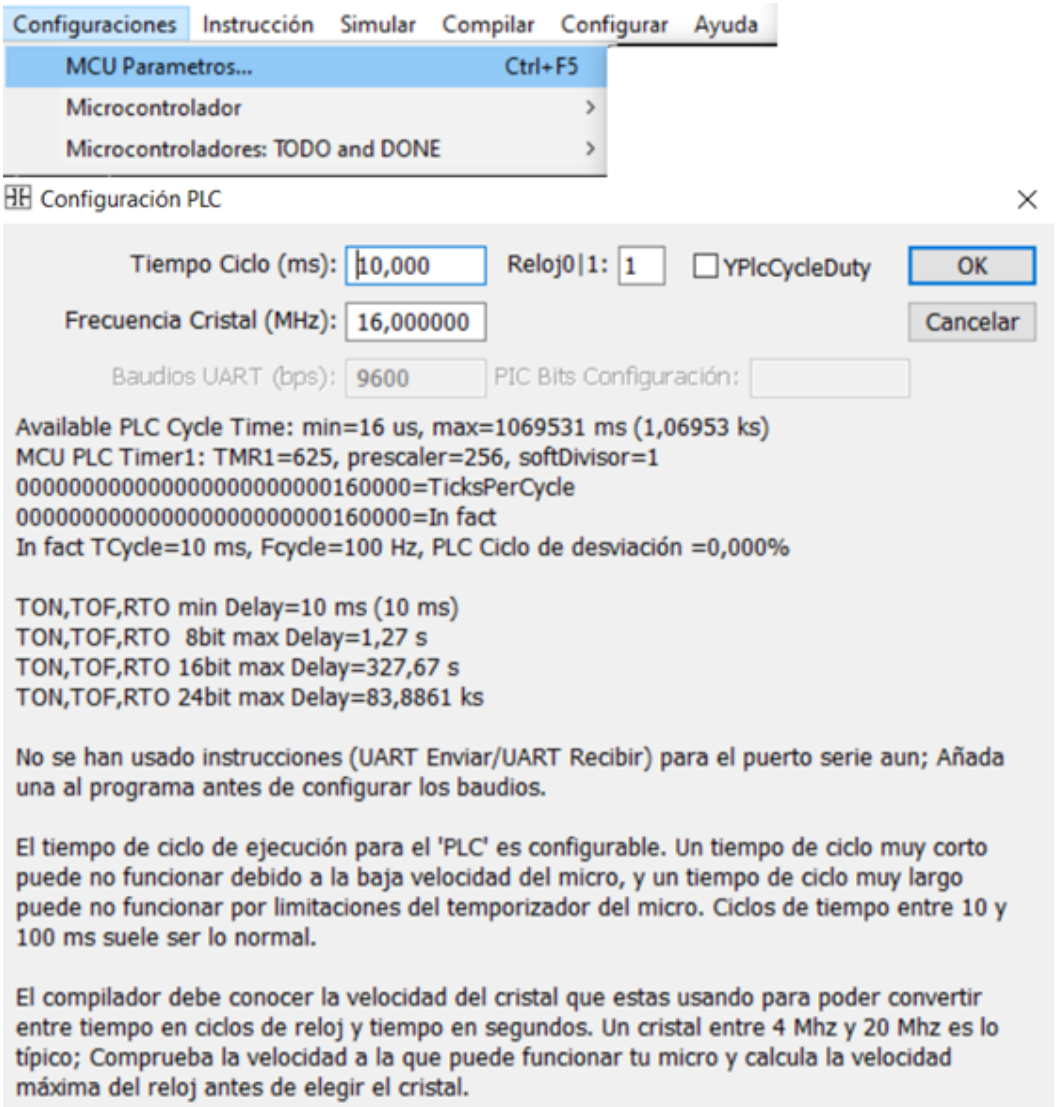
Selección del microcontrolador



En la pestaña Configuraciones/Parámetros del Micro…, podemos verificar los parámetros básicos de trabajo del microcontrolador, como son el ciclo de tiempo, la frecuencia del cristal y la velocidad de transmisión. Estos corresponden a la configuración básica y no deben ser alterados.

Figura 6

Parámetros básicos del microcontrolador



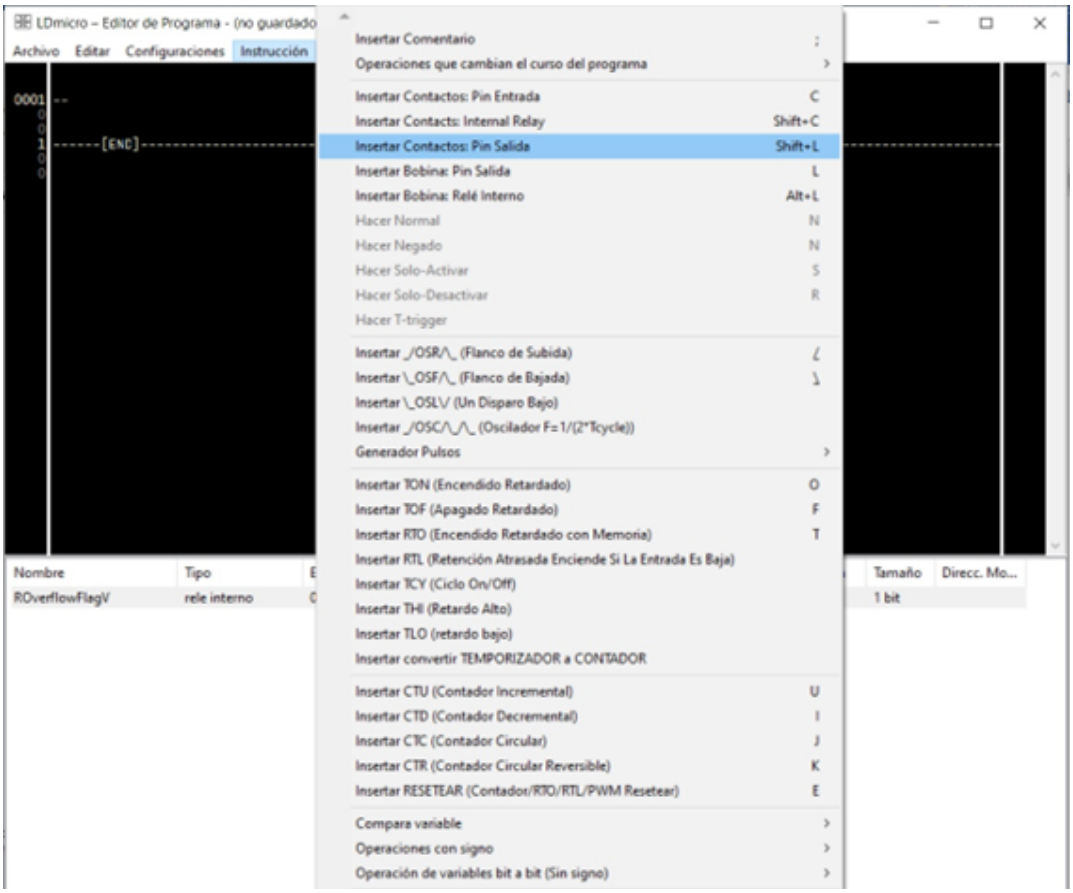
Menú Instrucción

Este menú contiene todos los comandos o instrucciones para elaborar el diagrama Ladder.

En la pestaña Instrucción podemos encontrar todos elementos como: contactos, bobinas, osciladores, contadores, temporizadores, comparadores, operaciones con signo, etc., para elaborar el diagrama Ladder.

Figura 7

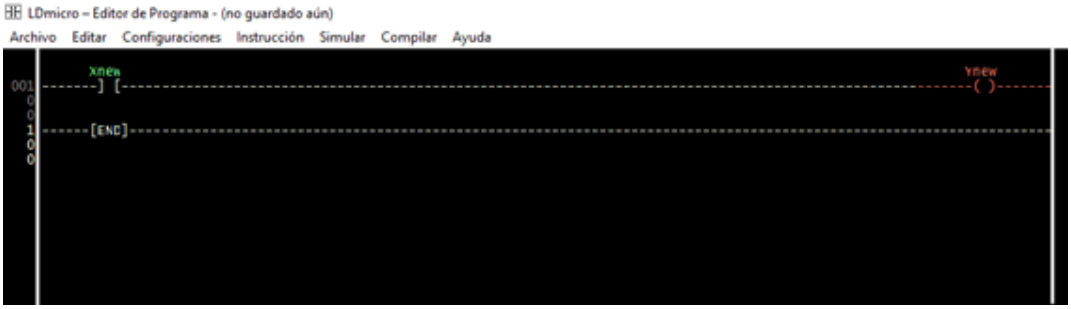
Extracto de operaciones del menú Instrucción del LDmicro



Para insertar un elemento basta con ubicarse en cualquier parte del rung del Ladder y colocar la instrucción deseada.

Figura 8

Insertando componentes en el diagrama Ladder

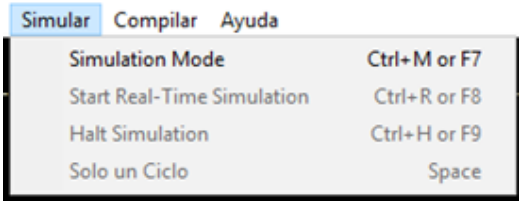


Menú Simular

LDmicro es también un simulador. Una vez que se tiene realizado el diagrama Ladder y para comprobar el funcionamiento ponemos al programa en modo de simulación, *Simulation Mode*. Se activa el comando *Start Real – Simulation*, con lo cual empieza la simulación. Para detener la simulación y poder seguir editando el programa usamos *Halt Simulation*. Cuando se utiliza *Solo un Ciclo*, la simulación se ejecuta una sola vez.

Figura 9

Opciones del Menú Simular



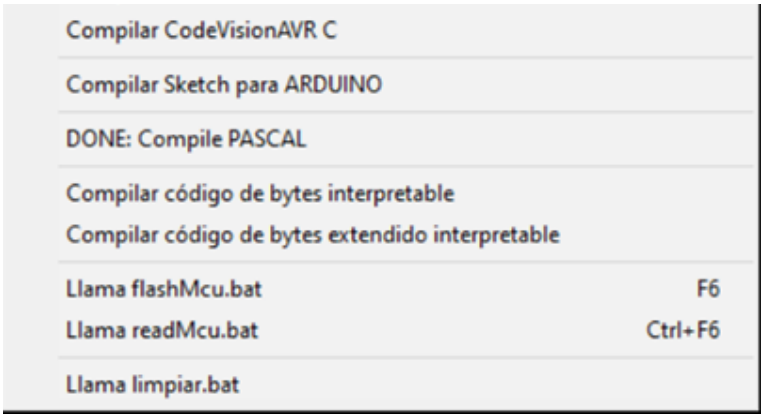
Menú Compilar

Este menú presenta varias opciones de compilación. *Compilar y Compile HEX*, generan el archivo *.HEX* con el nombre del diagrama Ladder grabado. Compilar Como, genera también el archivo *.HEX*, teniendo la opción de cambiar el nombre del archivo. Estas son las opciones que nos interesan para generar el archivo *.HEX*, el cual podemos cargar a la tarjeta Arduino o utilizar en el simulador Proteus.

Las demás opciones generan diversos formatos de códigos para que el usuario pueda utilizarlo según sus necesidades.

Figura 10

Comandos del menú compilar



Menú Configurar

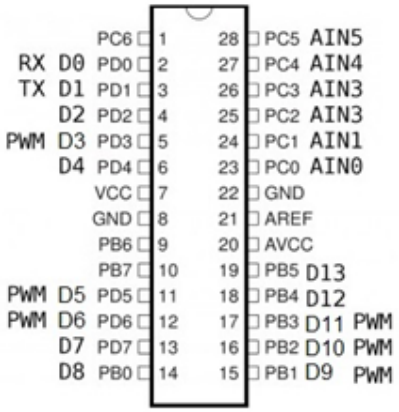
A partir de la versión 4, viene incluido el menú Configurar, que sirve exclusivamente para cambiar el color de la interfaz del programa. El color negro que viene por defecto puede ser cambiado a otros colores como azul, blanco o colores personalizados.

Asignación de pines

Para compilar el programa los pines de entrada y salida del microcontrolador. Recuerde que LDmicro hace referencia a los pines del microcontrolador y no a los que usa la tarjeta Arduino.

Figura 11

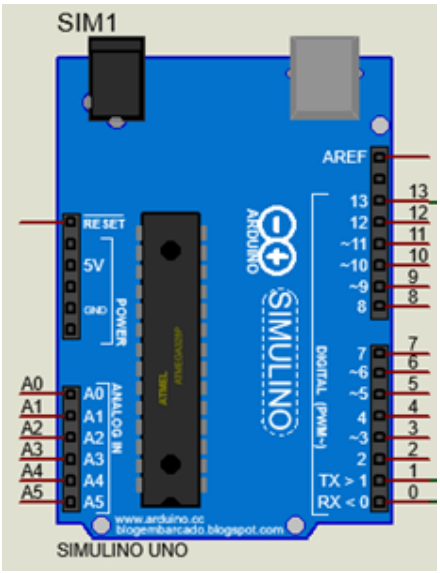
Distribución de pines del microcontrolador ATmega 328 utilizado por Arduino Uno



Así, por ejemplo, el pin2 es el puerto PD0 del microcontrolador y en la tarjeta Arduino corresponde al terminal 0, D0.

Figura 12

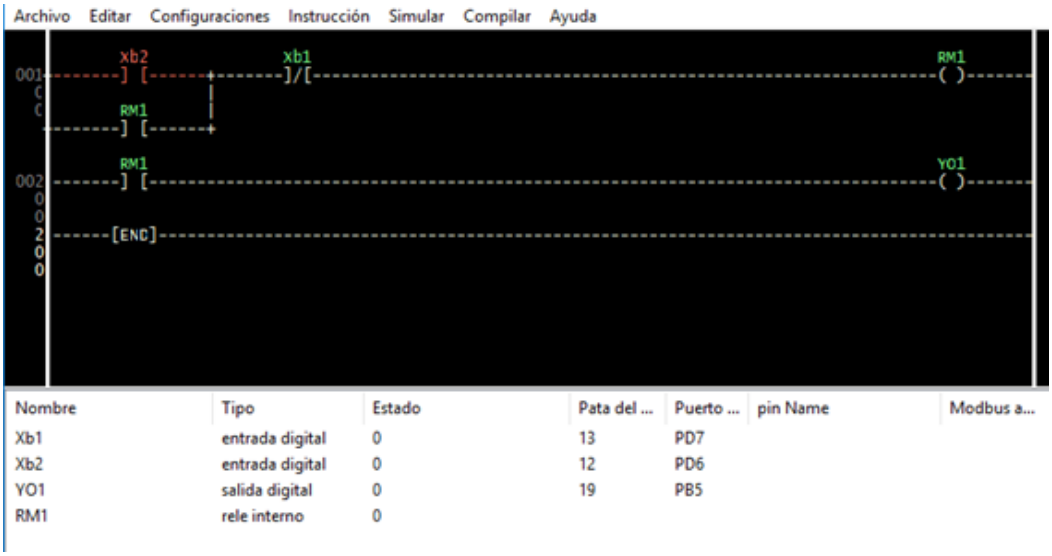
Distribución de pines de la tarjeta Arduino Uno



Ejemplo. Según el diagrama Ladder, asignar los pines al microcontrolador.

Figura 13

Ventana del diagrama Ladder y lista de los pines usados del microcontrolador

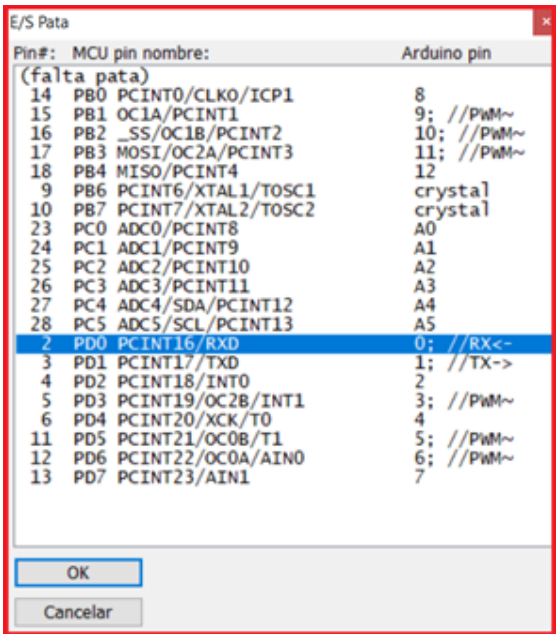


Para obtener el listado de las entradas y salidas y asignar los pines del microcontrolador, hay que ubicarse en el borde inferior de la ventana de edición y desplazar esta ventana hacia arriba apareciendo de inmediato otra ventana con la información de los elementos usados en el diagrama Ladder.

Seguidamente, se selecciona la entrada o salida a la cual se va a asignar un pin del microcontrolador y en el cuadro de diálogo que se despliega procedemos a establecer el pin respectivo, según las necesidades del diseño.

Figura 14

Pines disponibles para asignación de las entradas y salidas

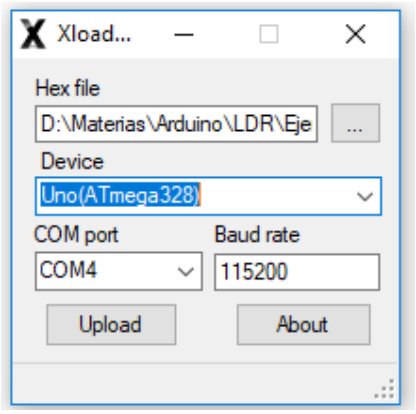


Transferir el archivo HEX a la tarjeta Arduino

Para transferir el archivo HEX al microcontrolador de la tarjeta Arduino, utilizamos el programa Xloader, en el cual debemos direccionar el archivo Hex file, el modelo del microcontrolador de la tarjeta, el puerto de comunicación y la velocidad de transmisión. Al conectar la tarjeta al PC se genera un puerto virtual serial, que Xloader lo detecta automáticamente, o en caso contrario se debe verificar en el panel de control del PC el puerto creado. La velocidad de transmisión utiliza valores estándares de hasta 19600 baudios.

Figura 15

Opciones del programa XLoader



Palabras reservadas

Los nombres reservados que están vinculados a LDmicro han sido tomado del manual de ayuda que incluye el programa y de otras páginas como:

- Electgpl:

http://electgpl.blogspot.com/2016/05/manual-de-uso-para-plc-con-ld-micro.html

- Github:

https://github.com/LDmicro/LDmicro/blob/master/ldmicro/manual-es.txt

En LDmicro, por defecto la primera letra de un nombre especifica la clase de elemento que es.

Las principales palabras reservadas que usa el programa y que se utilizan en los diagramas Ladder para identificar a los componentes son:

- Xname: pin físico de entrada del microcontrolador.
- Yname: pin físico de salida del microcontrolador.
- Rname: relé interno: un bit en la memoria. En el PLC se denotan como MARCAS o FLAGS.
- Tname: un temporizador; retardo al encendido, retardo al apagado o retentivo.
- Cname: un contador, cuenta ascendente o cuenta regresiva.
- Aname: un número entero leído por un conversor A/D.
- Pname: afín a un pin de salida para señal PWM.
- Mname: salida de bobina MODBUS.
- Hname: registro MODBUS Holding.
- Iname: entrada digital MODBUS.
- Name: una variable de propósito general (entero)

Los nombres de las variables para el programa pueden ser letras, números y guiones bajos (_). El nombre de variable no debe empezar con un número o carácter especial. Los nombres de las variables distinguen mayúsculas y minúsculas (CASTAÑEDA, 2015).

Las instrucciones de las variables generales (ADD, SUB, MUL, DIV, MOV, EQU, GRT, GEQ, etc.) pueden funcionar con variables de cualquier nombre. Esto significa que pueden acceder al temporizador y acumuladores de contador (Westhues, 2009).

Las variables son siempre enteros con signo de 16 bits. Por tanto, pueden realizar operaciones con números comprendidos entre -32768 a 32767. Para

asignar los valores se utilizan formatos con números decimales, se hace en forma normal, en hexadecimal el número se precede con 0x, en octal con 0o, en binario con Ob y caracteres ASCII colocando en el carácter con comillas simples. Ejemplo:

- Decimales: 10, 4096, -150, -85
- Hexadecimales: 0xb, 0x3FA, 0xE7C
- Octales: 0o7, 0o14, 0o354
- Caracteres especiales ASCII: 'T', 's', 'M'

Contactos y bobinas

Los componentes básicos de un diagrama lógico de escalera son el contacto y la bobina. El contacto es el nombre dado a un dispositivo de entrada, como un interruptor o pulsador. La bobina es el nombre dado a un dispositivo de salida y es usada para manejar motores, relés, contactores, solenoides y otros procesos de activación.

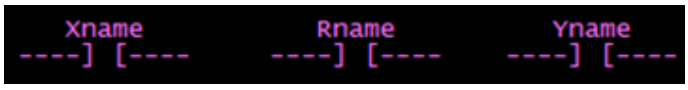
Contacto normalmente abierto (CNA)

Esta la instrucción detecta el estado de los pines de entrada, de salida, o de un relé interno (Westhues, 2009).

Si la señal de la salida o relé es verdadera el contacto se cierra, caso contrario el contacto permanece abierto.

Figura 16

Contactos normalmente abiertos (CNA)

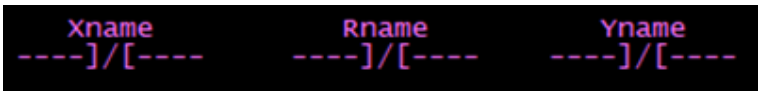


Contacto normalmente cerrado (CNC)

Realiza la misma función de un CNA Esta la instrucción detecta el estado de los pines de entrada, de salida, o de un relé interno. La operación es lo contrario a un contacto normalmente abierto (Westhues, 2009).

Figura 17

Contactos normalmente cerrados (CNC)

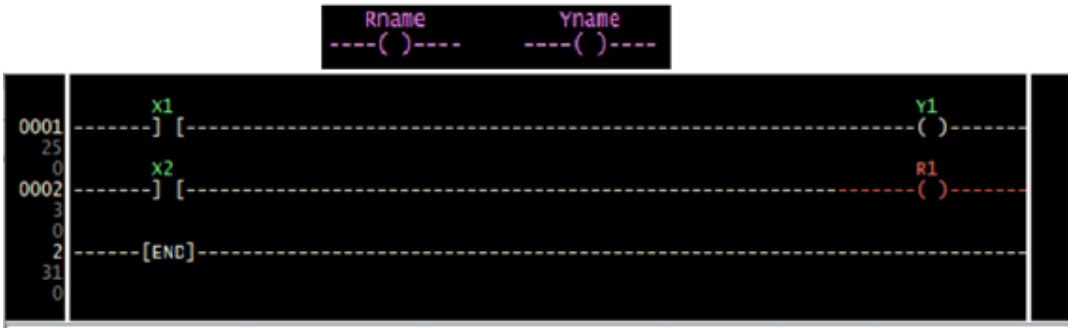


Bobina normal

Esta instrucción es la final en una línea o peldaño en el diagrama Ladder. Cuando la bobina se energiza sus contactos asociados cambian de estado. Un contacto abierto se cierra y cerrado se abre (Castañeda, 2015).

Figura 18

Bobina normal. Salida física o lógica



Nombre	Tipo	Estado	Pata del ...	Puerto ...	Nombre Pin
X1	entrada digital	0	2	PD0	PD0 (PCINT16 / RXD)
X2	entrada digital	0	3	PD1	PD1 (PCINT17 / TXD)
Y1	salida digital	0	19	PB5	PB5 (SCK / PCINT5)
R1	rele interno	0			
ROverflowFlagV	rele interno	0			

Bobina invertida

Opera de forma inversa a una normal. Sin energía en la bobina la salida está activa y los contactos asociados a la bobina han cambiado su estado; con energía en la bobina, la salida está desactivada y los contactos asociados están en estado de reposo.

Figura 19

Bobina invertida

Nombre	Tipo	Estado	Pata del ...	Puerto ...	Nombre Pin
X1	entrada digital	0	2	PD0	PD0 (PCINT16 / RXD)
X2	entrada digital	0	3	PD1	PD1 (PCINT17 / TXD)
Y1	salida digital	0	19	PB5	PB5 (SCK / PCINT5)
R1	rele interno	0			
ROverflowFlagV	rele interno	0			

SET bobina

Si la señal que ingresa en la línea de instrucción es verdadera o 1 lógico, entonces el estado del relé interno o el pin de salida pasa a ser uno, caso contrario, los pines de salida o los relés internos no cambia de estado. Para que la instrucción cambie a estado alto o 1 lógico, la bobina debe estar en cero y con esto se tiene un cambio de cero a uno en la bobina que está asociada a la instrucción. Necesariamente tiene que ser utilizada con una instrucción de RESET para ponerla en 0 lógico (Elecplc, 2016).

La entrada SET funciona de manera similar a una entrada SET de un Flip Flop SR.

Figura 20

Bobina SET

RESET bobina

La instrucción *Reset bobina* sirve para "borrar" o desactivar una bobina SET o que se fijó previamente en 1. Cuando recibe un pulso en la señal de entrada, el pin de salida o el relé interno que esté asociado con la bobina RESET se pone en cero. Por lo general, la acción de RESET ocurre durante el flanco positivo de la señal de entrada.

Para que la instrucción RESET pueda modificar el estado de la bobina de alto a bajo o verdadero a falso, se debe utilizar obligatoriamente la instrucción SET, para fijar previamente en la salida o relé interno (Westhues, 2009).

Esta instrucción es similar a la entrada de RESET de un Flip Flop SR. Es muy útil en los circuitos de automatización porque reemplaza a los enclavamientos eléctricos que normalmente se realizan con contactos.

Figura 21

Bobina RESET

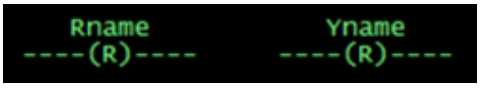
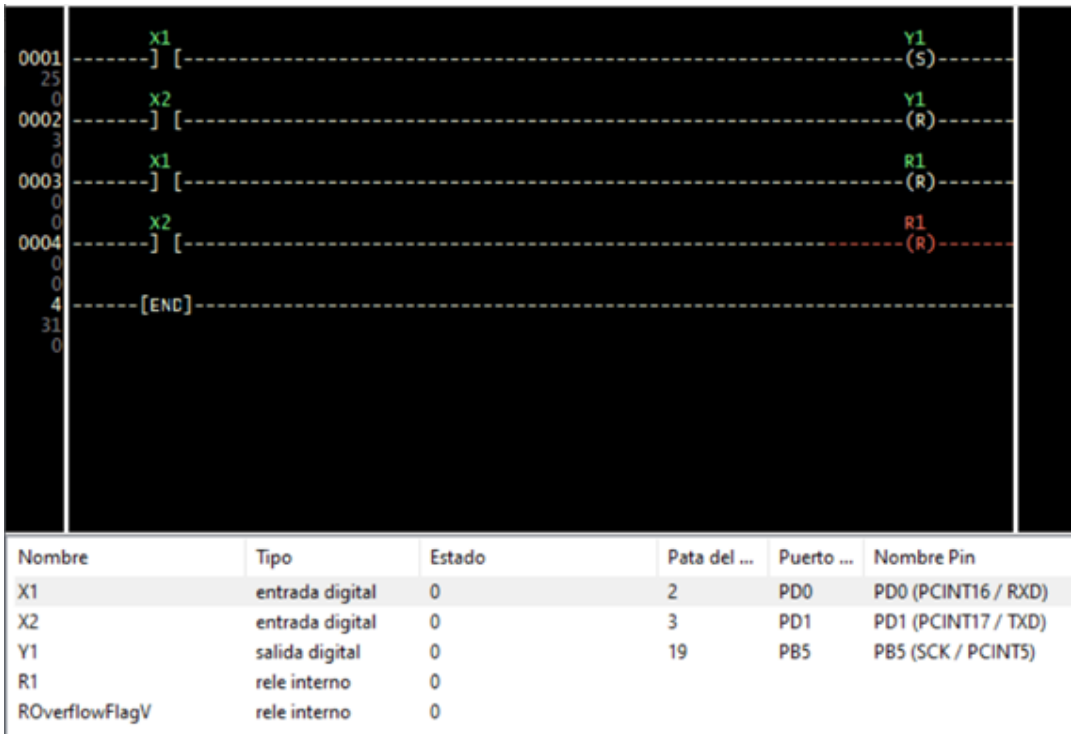


Figura 22

Bobina RESET



Nombre	Tipo	Estado	Pata del ...	Puerto ...	Nombre Pin
X1	entrada digital	0	2	PD0	PD0 (PCINT16 / RXD)
X2	entrada digital	0	3	PD1	PD1 (PCINT17 / TXD)
Y1	salida digital	0	19	PB5	PB5 (SCK / PCINT5)
R1	rele interno	0			
ROverflowFlagV	rele interno	0			

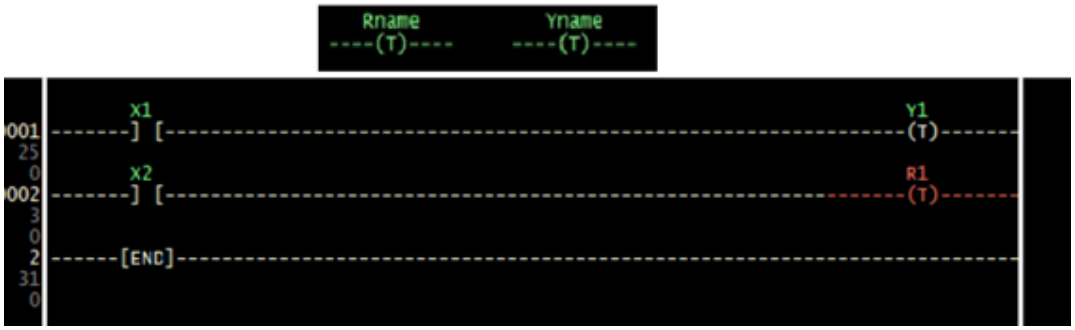
Bobina T-trigger

Una bobina T funciona como un flip-flop tipo T con entrada de reloj, la conmutación se produce en flancos de la señal de entrada. En un flip-flop de bobina, cada flanco ascendente de la condición de entrada cambia de estado la bobina T. Una bobina T debe ser una salida física (YName) o un relé interno (RName) (Westhues, 2009).

Si hay un flanco ascendente, el estado de la bobina 0 cambia a 1. Si hay un nuevo flanco ascendente, la bobina que está en alto pasará a bajo. En flanco descendente no tiene ningún efecto en el estado de la bobina T. Como toda bobina, esta instrucción se ubica al final de la línea.

Figura 23

Salidas de la bobina T – TIGGER

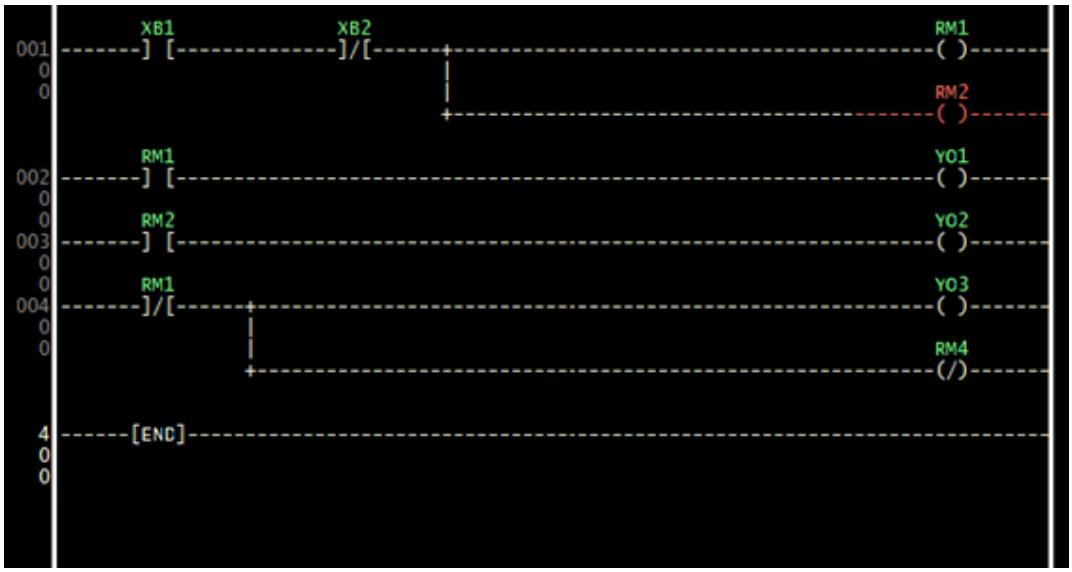


Ejercicio:

Elaborar el siguiente diagrama Ladder y comprobar el funcionamiento.

Figura 24

Diagrama de contactos



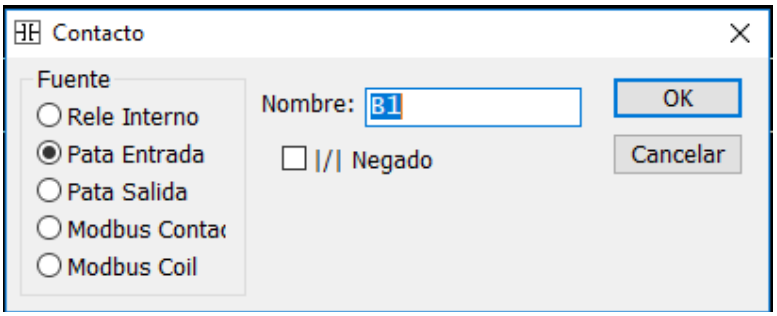
Algunas sugerencias para la elaboración del diagrama Ladder.

- Comience colocando los contactos y luego complete las bobinas.
- Los nombres se cambian fácilmente haciendo doble clic sobre el elemento del Ladder y en la caja de datos seleccione la opción, según el caso.

- Por defecto siempre en el nombre aparecerá la primera letra del elemento ingresado (revise las palabras reservadas, punto 1.4).

Figura 25

Opciones de atributos para un contacto



Simulador Proteus ISIS

Para simular el programa en Proteus – Isis de LabCenter Electronics, se tiene que incluir la librería BLOGEMBARCADO.LIB, en la librería de Proteus. Con esto en los componentes de ISIS dispondremos de la tarjeta Arduino Uno, que está indicado como SIMULINO UNO. Para esto debemos copiar el archivo BLOGEMBARCADO.LIB (viene junto XLoader) en la carpeta de la instalación de Proteus, que por lo general está en:

C:\Program Files (x86)\Labcenter Electronics\Proteus 8 Professional\LIBRARY

Con otras librerías de Arduino para-Proteus también funcionan sin problemas, solo debemos tomar en cuenta las configuraciones que se debe realizar especialmente cuando se trabaje con el conversor AD y otras instrucciones que vienen incluidos en la versión 5 de LDmicro.

Simuladores CADe-SIMU y PC-SIMU

CADe-SIMU es un programa de CAD electrotécnico muy versátil que permite insertar los distintos símbolos eléctricos, electrónicos, electroneumáticos, electromecánicos, dispositivos de control, que están organizados en librerías y dibujar esquemas eléctricos de control y de fuerza, utilizando diagramas de contactos, Ladder, Grafcet, símbolos en 2D y 3D, entre otras funciones, en de una forma fácil y rápida para posteriormente realizar la simulación.

La versión 4.0 de CADe-SIMU incluye el módulo de Arduino UNO, por lo que resulta muy útil trabajar con este simulador conectado con la tarjeta para probar el funcionamiento de circuitos industriales, ya que su entorno de trabajo utiliza componentes y simbología común de control industrial.

Para trabajar con CADe-SIMU en forma online con la tarjeta Arduino UNO, debemos tener instalado el programa IDE de Arduino, que como es un producto Open Source podemos descargarlo en forma gratuita desde cualquier sitio WEB de Arduino o desde su página oficial:

https://www.arduino.cc/en/software

Para poder IDE Arduino con CADe-SIMU se debe agregar como la librería el programa PC-SIMU. PC-SIMU y CADe-SIMU descargamos en formato .zip desde los siguientes links:

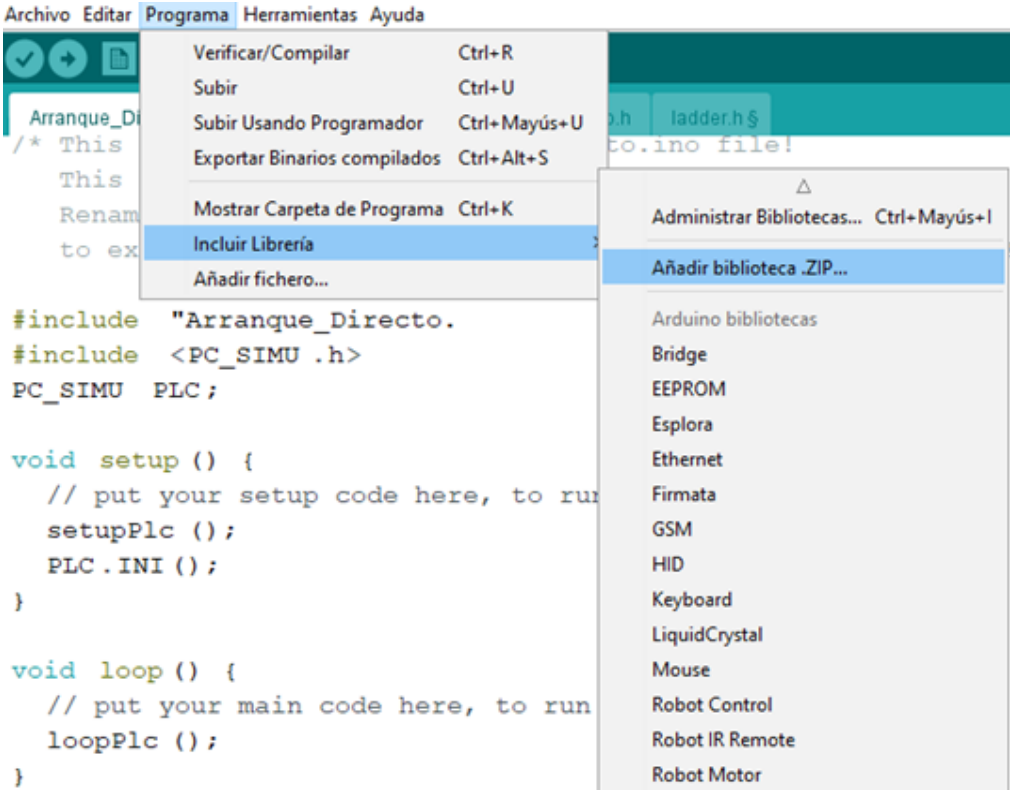
https://drive.google.com/file/d/19_TGsuiOpcalFIhyNdMIHHlN-mKO4y--C/view

https://mega.nz/folder/MaghGYYR#Ad5Y_lC9kxUbDFJpYJN1EA

Abrimos el programa del IDE de Arduino y en la pestaña Programa/Incluir Librería/Añadir biblioteca Zip… damos la ubicación del archivo PC-SIMU. zip y queda listo para ser utilizado con CADe-SIMU.

Figura 26

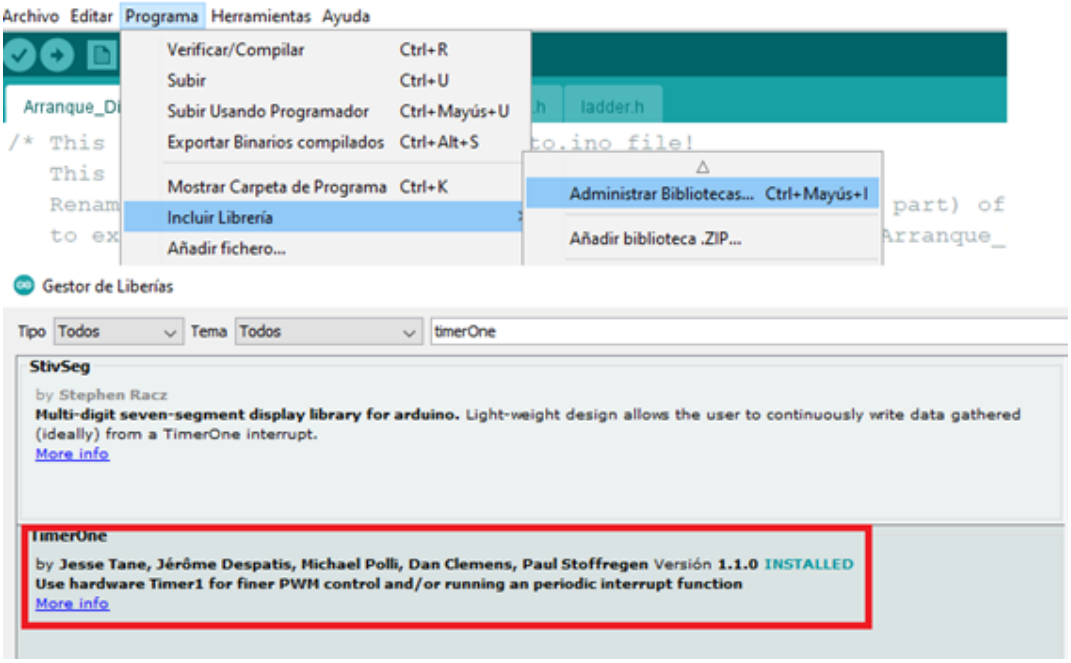
Incluir librería PC-SIMU en el IDE de Arduino



La otra librería necesaria que debemos agregar es la del TimerOne.h, Esta librería podemos gestionarla directamente desde la pestaña Programa/Incluir Librería/Administrar Biblioteca, y en el cuadro de texto del Gestor de librerías, buscar la librería TimerOne e instalar.

Figura 27

Incluir librería TimerOne.h



En los programas que se desarrollen utilizamos generadores de tiempo, servomotores, también debemos incluir estas librerías y las que sean necesarias para el buen funcionamiento de los programas.

PC-SIMU, con Arduino actúa como librería para interactuar con CADe-SIMU con la tarjeta Arduino en forma física, pero podemos utilizar PC-SIMU como programa libre, para trabajar en conjunto con CADe-SIMU o comunicarse con Arduino directamente. Permite crear HMI- SCADA, de una forma rápida y fácil. Para configurar las entradas y salidas del PC-SIMU con el Arduino, abrimos el programa PC-SIMU y en el menú Archivo procedemos a marcar las opciones indicadas:

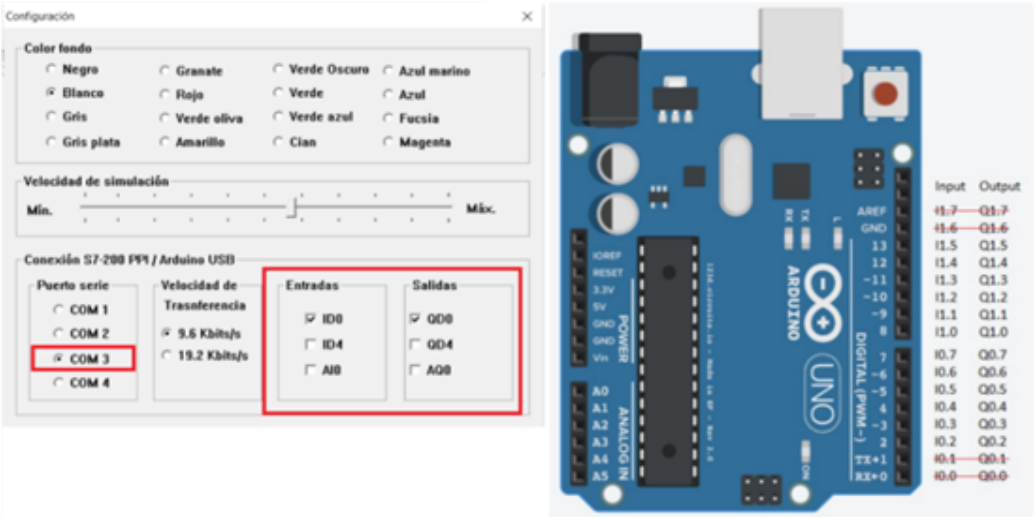
Archivo/Configuración/ConexiónS7-200 PPI/Arduino USB

- Velocidad de transferencia 9.6 Kbits/s
- Marcar ID0 y QD0

Y cuando realicemos la simulación directamente en PC-SIMU, hay que considerar los pines que usa este programa:

Figura 28

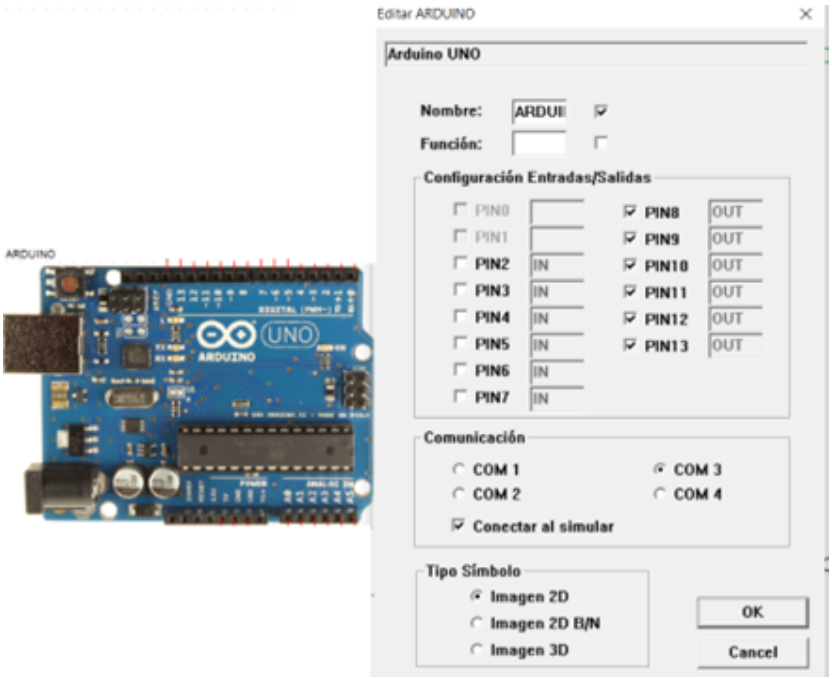
Entradas y salidas ARDUINO-PC_SIMU



De la misma forma procedemos cuando se comunique Arduino con CADe-SIMU. Abrir el programa CADe-SIMU y colocar en el área de dibujo la tarjeta Arduino y mediante un doble clic con el Mous,e en la ventana de edición se aprecia los pines que utiliza CADe-SIMU.

Figura 29

Entradas y salidas Arduino con CADe-SIMU

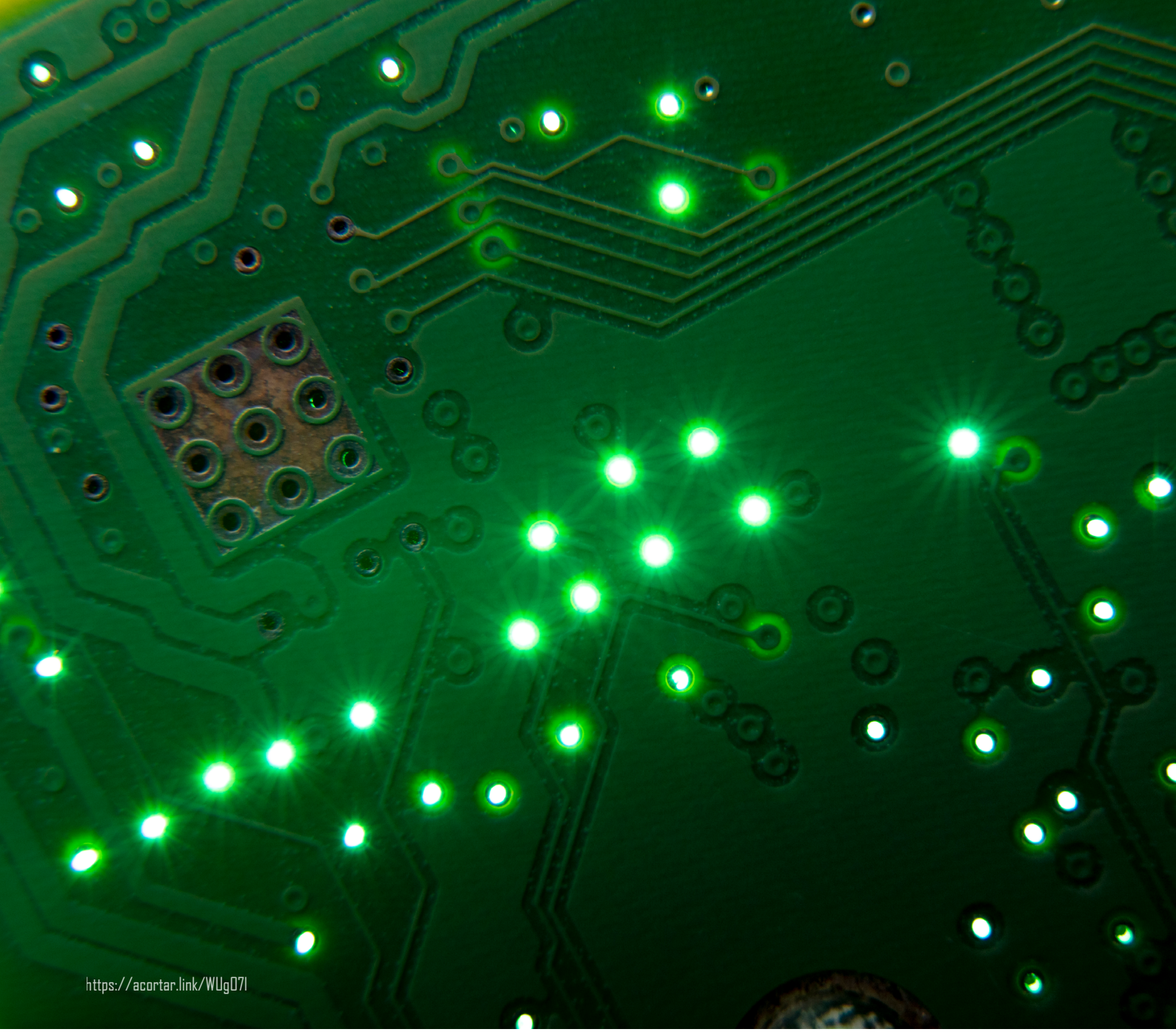


La siguiente tabla resume los pines de Arduino utilizado por CADe-SIMU y PC-SIMU. Los pines 0 y 1 de Arduino no están habilitados para estos programas.

Tabla 1

Distribución de pines utilizados en la tarjeta Arduino por los programas CADe-SIMU y PC-SIMU

PIN ARDUINO	PC-SIMU		CADE SIMU
13	I1.5	Q1.5	I/O (1.6)
12	I1.4	Q1.4	I/O (1.5)
11	I1.3	Q1.3	I/O (1.4)
10	I1.2	Q1.2	I/O (1.3)
9	I1.1	Q1.1	I/O (1.1)
8	I1.0	Q1.0	I/O (1.0)
7	I0.7	Q0.7	I/O (0.7)
6	I0.6	Q0.6	I/O (0.6)
5	I0.5	Q0.5	I/O (0.5)
4	I0.4	Q0.4	I/O (0.4)
3	I0.3	Q0.3	I/O (0.3)
2	I0.2	Q0.2	I/O (0.2)



https://acortar.link/WUg07I

CAPÍTULO II

Circuitos con contactos, memorias y enclavamientos

Los circuitos de control industrial trabajan en base a los enclavamientos eléctricos y memorias realizados con contactos. Por tanto, en el presente capitulo se abordan los circuitos más comunes usando diagramas Ladder.

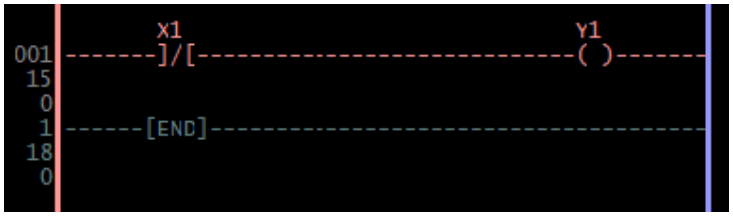
Funciones lógicas con contactos

Función NOT

Un contacto normalmente cerrado (NC) accionando a una salida representa la función lógica NOT. Así si se conecta un pulsador NA (normalmente abierto) en una entrada X1, como el contacto es un NC, entonces la corriente está fluyendo hacia la bobina, la misma que está activada.

Figura 30

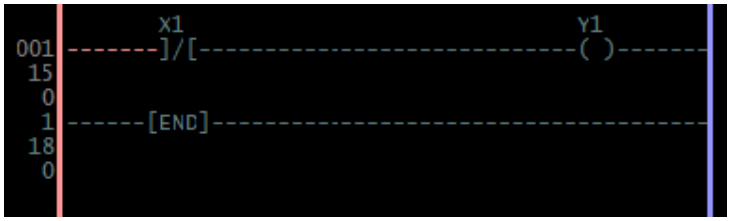
Función lógica NOT. Salida activada



Al accionar el pulsador el contacto NC, X1, del diagrama Ladder se abre, interrumpe la corriente de la bobina y desactiva la salida Y1.

Figura 31

Función NOT. Salida desactivada

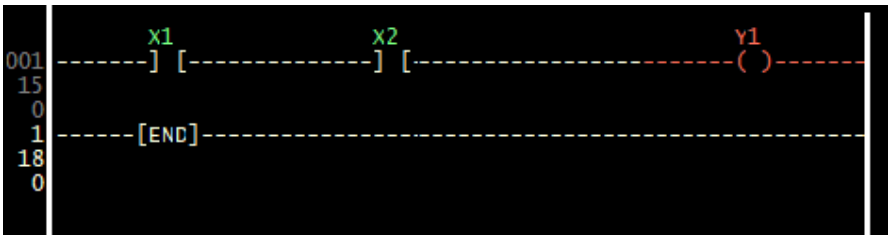


Función lógica AND

Dos contactos en serie representan la función lógica AND. Los contactos abiertos del diagrama Ladder (X1, X2), bloquean la corriente sin permitir que se accione la salida Y1.

Figura 32

Función lógica Y



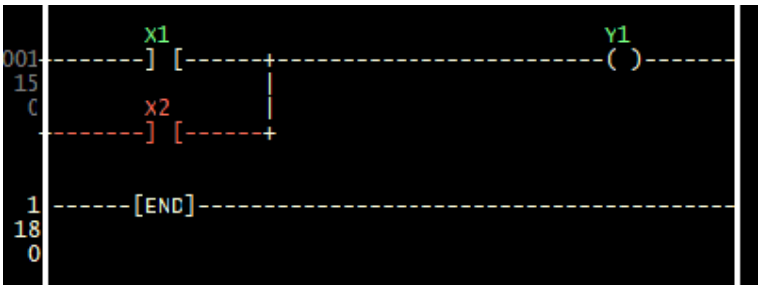
La salida Y1 se activa solo cuando sean accionados los dos pulsadores que se encuentran conectados en las entradas X1 e X2. Si una de estas no es accionada el contacto abierto bloqueará la corriente que se dirige a la bobina Y1.

Función lógica OR

En el circuito se trata de prender una salida Y1 accionar uno de los interruptores o los dos que están conectados en las entradas X1 e X2 respectivamente.

Figura 33

Función lógica OR



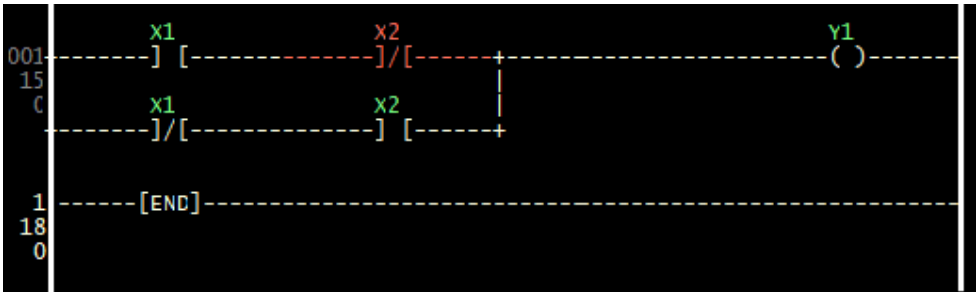
Dos contactos en paralelo representan la función lógica OR. La salida Y1 se activa cuando se accionan cualquiera de los dos pulsadores que se encuentran conectados en las entradas X1 e X2. Si los dos se accionan simultáneamente se tendrá el mismo resultado, es decir, Y1 se activa.

Función lógica XOR

Considerando que la ecuación que describe la función lógica XOR es: X= AB' + A'B. Para la implementación del circuito, la variable negada representa un contacto normalmente cerrado (NC) y la variable normal un contacto normalmente abierto (NA).

Figura 34

Circuito función XOR con contactos



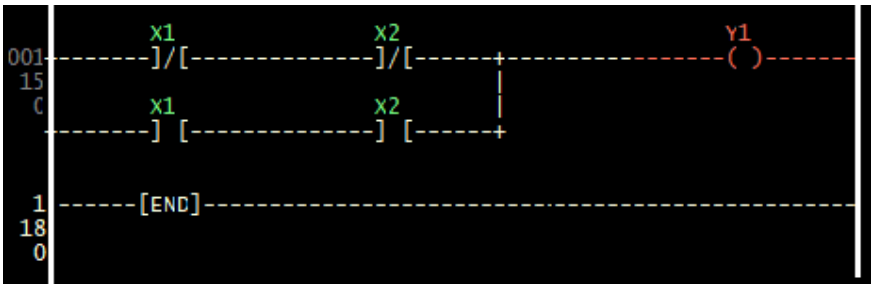
Para la simulación, se consideran como interruptores a entradas X1 e X2. Al accionar uno de los interruptores, por ejemplo, X2, el contacto abierto se cierra pudiendo circular la corriente por el contacto cerrado de X1, energizando la bobina Y1. Al accionar los dos interruptores, todos los contactos se invierten, por tanto, la salida Y1 no puede energizarse.

Función lógica XNOR

Recordando que la ecuación que describe la función lógica XNOR es: X= A'B' + AB. Para la implementación del circuito, la variable negada representa un contacto normalmente cerrado (NC) y la variable normal un contacto normalmente abierto (NA).

Figura 35

Función XNOR



Como se sabe, cuando las entradas son iguales la salida es alta y a entradas diferentes la salida baja. Si X1 y X2 están en el estado, la salida Y1 es alta, si uno de ellos tiene un estado diferente, la salida es baja.

Implementación de ecuaciones Booleanas

Mediante el diagrama Ladder, se puede implementar ecuaciones lógicas dado que mediante los contactos se puede representar cualquier función lógica booleana.

En resumen, de la sección anterior podemos indicar que:

- Un contacto cerrado, representa la negación de una variable booleana.
- Un contacto abierto, es igual a una variable booleana.
- Contactos en paralelo representa a la función lógica OR.
- Contactos en serie la función lógica AND.

Ejemplo:

Dada la ecuación booleana F= A'BC' + AB'C + AB, implementar el circuito con contactos.

Como primer paso se debe realizar la tabla de asignaciones para las variables de entrada A, B y C, y la variable de salida F.

Tabla 2

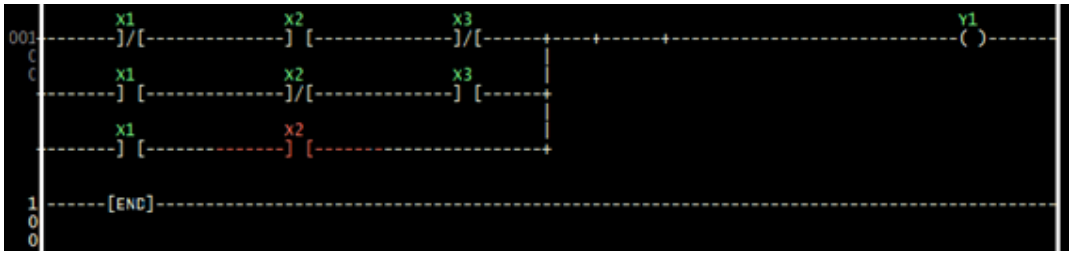
Asignación de pines según las variables

F	Y1
A	X1
B	X2
C	X3

El circuito equivalente con contactos para la ecuación dada, se indica en la figura.

Figura 36

Diagrama Ladder implementado mediante ecuación lógica



Circuitos de enclavamiento y memorias

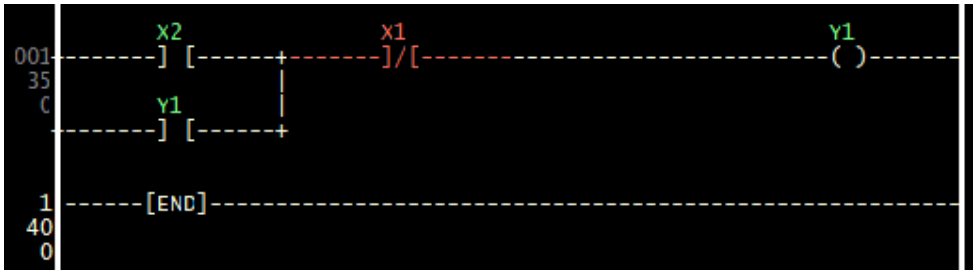
Circuito de enclavamiento

Una de los circuitos básicos de Control Industrial es el enclavamiento de una salida. Este circuito es muy común en el arranque directo de un motor, por ejemplo.

En este circuito la corriente es bloqueada por los contactos abiertos de X2 y Y1, por tanto, la bobina (salida) Y1 no puede energizarse.

Figura 37

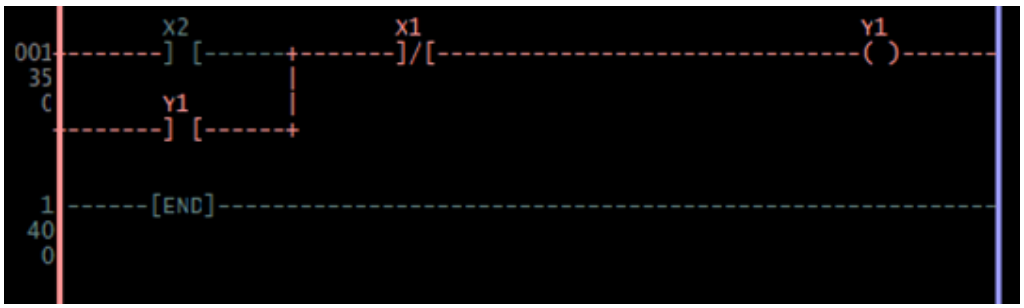
Circuito de enclavamiento



Al accionar el pulsador que se conecta en la entrada X2, la corriente fluye a través de este contacto y del contacto cerrado de X1, estableciendo un camino cerrado hasta la bobina Y1, por tanto, activando la carga conectada en esta salida. Al dejar de accionar el pulsador de la entrada X2, el contacto se abre, pero la corriente sigue circulando por el contacto de Y1, que se cierra al momento que se energiza la bobina, quedando auto enclavado la misma. Al accionar el pulsador de la entrada X1, se bloquea la corriente hacia la bobina, abre el contacto de Y1 y, por consiguiente, la bobina se desactiva.

Figura 38

Funcionamiento del circuito de autoenclamiento

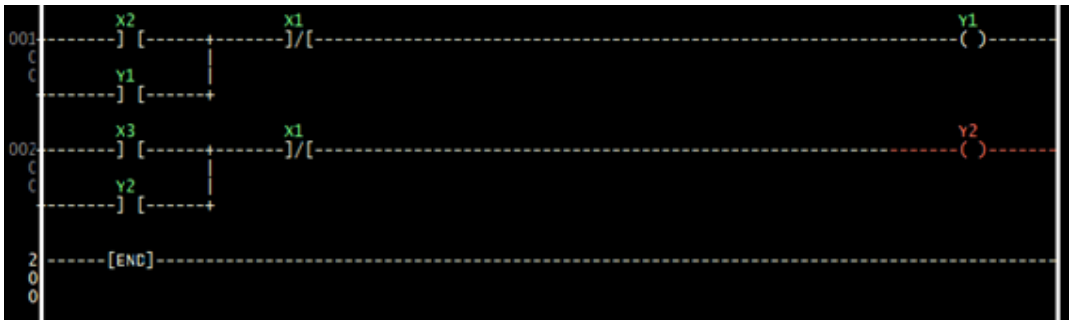


Enclavamiento de dos salidas independientes

En este ejercicio se auto enclavan dos salidas independientes. Mediante un pulsador conectado a la entrada X2 se acciona la salida Y1 y por medio de un pulsador conectado en la entrada X3 se acciona Y2. Con un pulsador conectado en la entrada X1 se desactivan las salidas en cualquier instante.

Figura 39

Enclamiento eléctrico de 2 salidas independientes



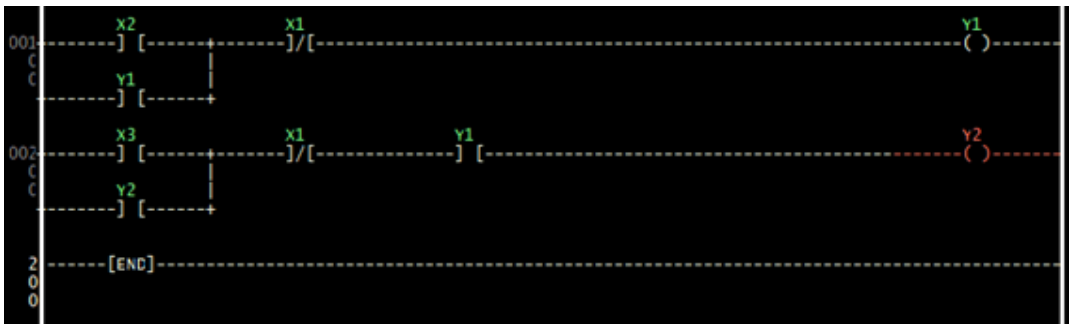
Encendido secuencial de dos salidas

El circuito consiste en activar una salida Y1 mediante la entrada X2 y luego activar la salida Y2 por medio de la entrada X3. Se dispone en la entrada X1 de un pulsador para apagar el circuito.

Para este caso el enclavamiento de la salida Y1, se realiza normalmente por ser la primera en activarse. Como es condición que la salida Y2 se puede activar cuando Y1 está prendida, entonces se coloca en serie un contacto abierto de Y1, para bloquear la corriente de Y2, para que esté no puede prenderse si Y1 no está activa. Con Y1 prendido el contacto abierto de la rama de Y2 se cierra, por tanto, permitirá el funcionamiento de Y2 al accionar la entrada X3.

Figura 40

Encendido secuencial de dos salidas

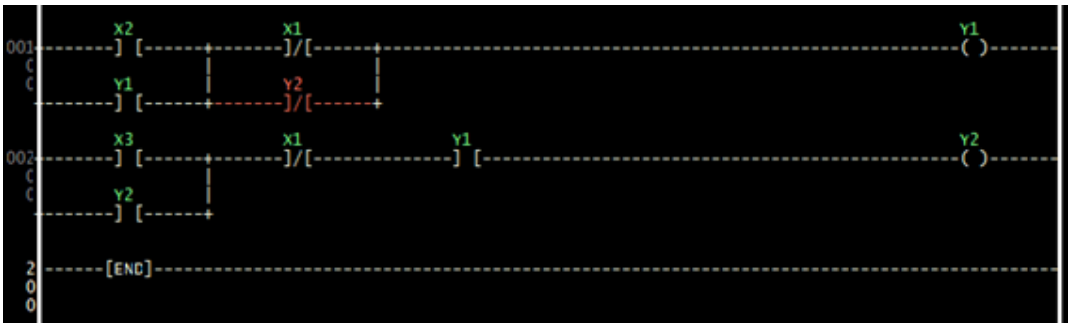


Encendido secuencial de dos salidas y apagado condicionado

Prender secuencialmente dos salidas y apagar si las dos están activadas.

Figura 41

Encendido secuencial de dos salidas y apagado si las dos están funcionando

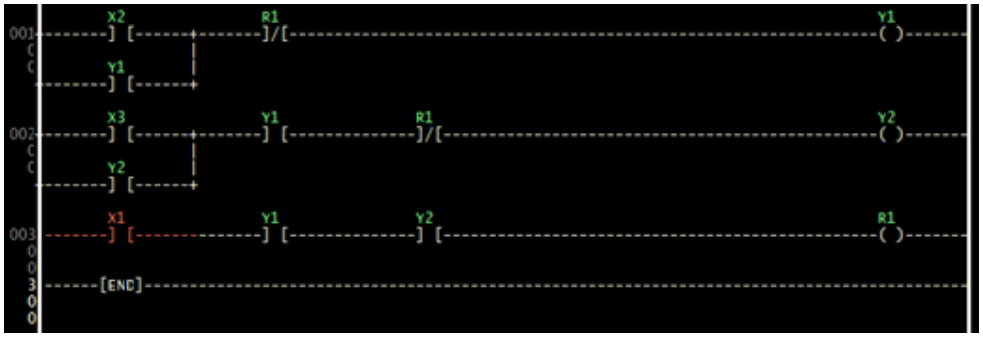


Para este caso, supongamos que Y1 está funcionando y Y2 todavía no está activada. Al accionar el pulsador de la entrada X1, el contacto se abre, pero la corriente sigue fluyendo por el contacto cerrado Y2, que está en paralelo con el contacto cerrado X1 y, por tanto, Y1 no se apaga. Al entrar a funcionar Y2, el contacto cerrado Y2 se abre, con lo cual ahora sí se puede apagar las dos bobinas, cumpliendo con la condición establecida en el ejercicio.

Los PLCs, tienen marcas o flags, que son salidas lógicas internas, y que podemos utilizar para apagar el circuito. LDmicro, dispone de los relés internos que hacen la función de una marca de un PLC. El relé interno R1 se puede activar mediante X1 cuando Y1 y Y2 están prendidas. Al cumplirse esta condición el contacto cerrado M1 se abre desactivando a las salidas Y1 y Y2.

Figura 42

Encendido secuencial de dos salidas y apagado si las dos bobinas están funcionando mediante un contacto cerrado de una marca o relé interno



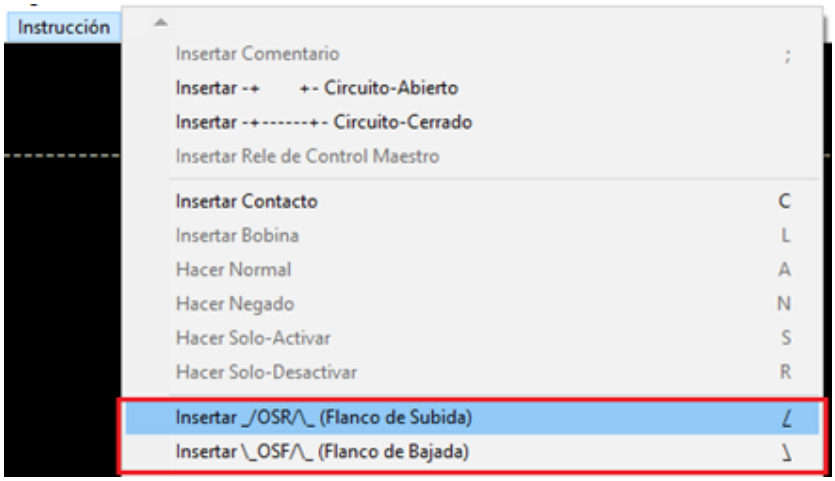
Detector de flancos

Muchas veces es necesario detectar el cambio de flanco, es decir, si una entrada que está en alto cambia a bajo (descendente), o si está en bajo cambia a alto (ascendente). LDMicro dispone de estas funciones, que nos permiten detectar el cambio de estado y realizar la acción respectiva en el flanco, según las necesidades.

En el menú Instrucción podemos encontrar las funciones de flanco de subida y flanco de bajada.

Figura 43

Comando para insertar un flanco de subida y un flanco de bajada

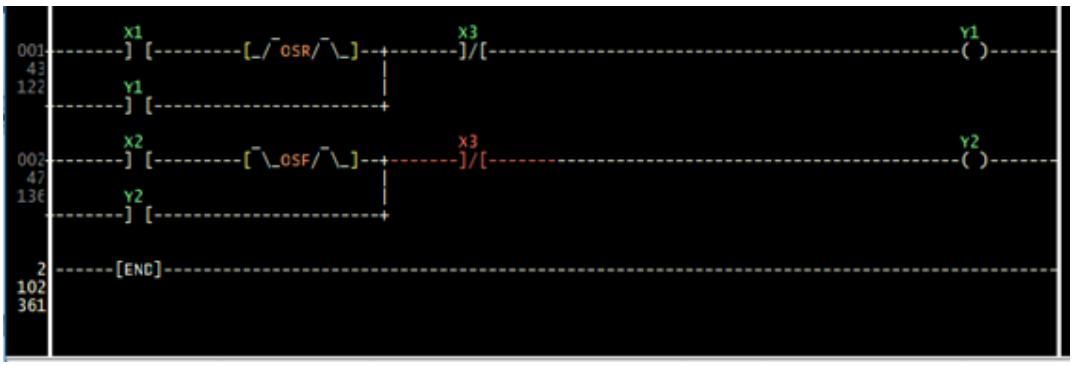


Ejemplo:

Circuito de enclavamiento de dos salidas. La primera salida se enclava en el cambio de estado de la entrada de bajo a alto y la segunda salida en el cambio de estado de la entrada de alto a bajo.

Figura 44

Detector de flanco descendente y ascendente

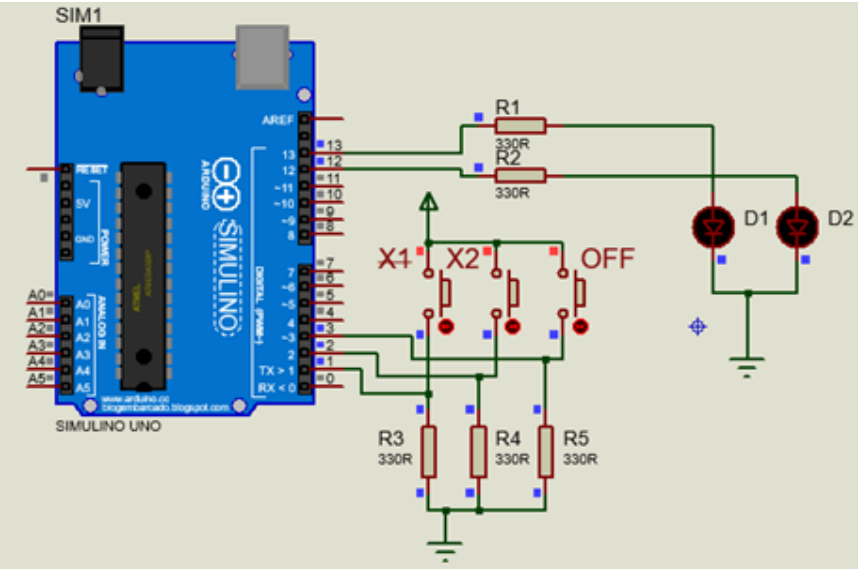


Nombre	Tipo	Estado	Pata del ...	Puerto ...	pin Name	Modbus a...
X1	entrada digital	0	3	PD1		
X2	entrada digital	0	4	PD2		
X3	entrada digital	0	5	PD3		
Y1	salida digital	0	19	PB5		
Y2	salida digital	0	18	PB4		

Para poder observar de una mejor forma el efecto del detector de flanco, para la simulación se ha realizado el circuito en ISIS.

Figura 45

Detector de flanco descendente y ascendente. Simulación en ISIS



Cuando se acciona el pulsador X1 la entrada del pin1, pasa de bajo a alto, lo que genera un flanco ascendente, activando la salida Y1, LED D1. Para el otro caso, al accionar X2, al accionar el pulsador y mantener la posición el estado del pin 2 se pone en alto y al liberarlo pasa a bajo, con lo que se genera un flanco de bajada, y en este momento, se activa la salida Y2, LED D2.

Prender y apagar una salida con un mismo pulsador

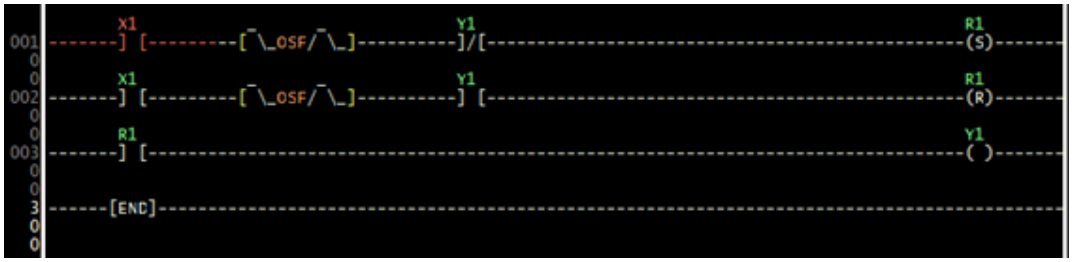
Este circuito es muy útil cuando se desea tener múltiple función en un pulsador. Con el mismo pulsador vamos a prender y apagar una salida.

Para implementar el diagrama Ladder es necesario usar el detector de flanco analizado en el punto anterior. Cuando la bobina Y1 está desactivada, el contacto Y1 está cerrado y el relé interno R1 está listo para activarse. Al accio-

nar X1 el pulsador cambia a alto y al liberarlo pasa a bajo en este instante se acciona R1 activando a Y1 y los contactos internos de Y1 cambia de posición, en este caso, queda habilitado la función RESET del relé interno R1. Al presionar y soltar el pulsador de la entrada X1, se el detector de flanco enviará la señal a RESET de R1 desactivando R1 y en consecuencia a Y1.

Figura 46

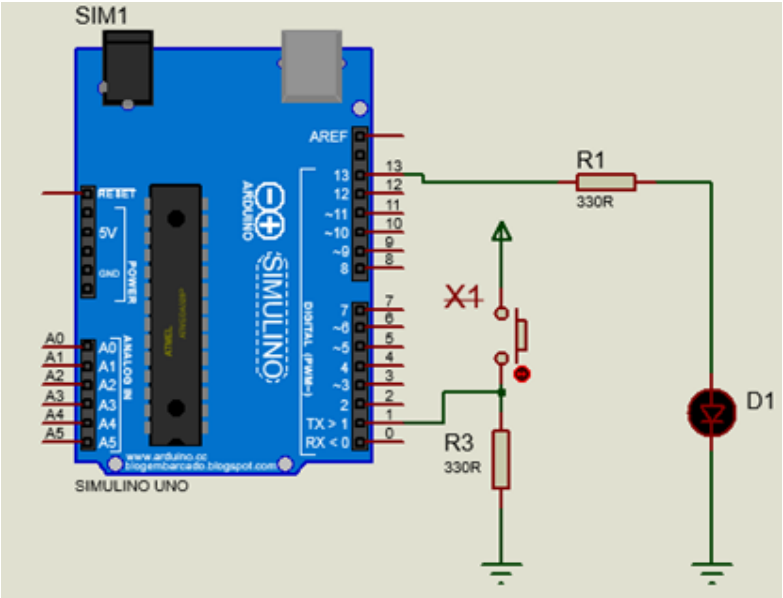
Circuito para activar y desactivar una salida con el mismo pulsador



Para demostrar el funcionamiento, se ha utilizado el simulador ISIS como muestra el gráfico siguiente:

Figura 47

Circuito para prender y apagar un LED con el mismo pulsador



Ejercicios propuestos

1. Implemente la ecuación F=ABC' + A'B'C' + AB'C + A'C

Realice el diagrama Ladder para controlar 3 salidas con sus respectivos pulsadores, que funcionará de la siguiente forma:

- Para activar cualquier salida todas deben estar apagadas.
- Solo se puede activar una salida a la vez
- Al activarse una salida las otras dos no podrán activarse, mientras no se apague la salida que está activada. Para esto se dispone de otro pulsador que puede apagar la salida que esté activa.

2. Diseñe el diagrama Ladder para prender dos lámparas con un mismo pulsador en secuencia. Al pulsar P1 se prende L1, al volver a pulsar P1 se prende la lámpara L2 y con un pulsador P2 se pueden apagar las dos lámparas si las dos están activadas.

3. Diseñe el diagrama Ladder para prender una lámpara L2, si una lámpara L1 ha sido previamente activada y desactivada. Cada lámpara debe tener su pulsador para prender y apagar en forma independiente.

4. Diseñe un circuito de conmutación para prender y apagar una lámpara desde dos sitios diferentes. El circuito es similar a tener dos conmutadores de tres vías. Para el Ladder implementar con dos pulsadores que al accionar cualquier pulsador cambiará de estado la lámpara.

https://acortar.link/fbRZi2

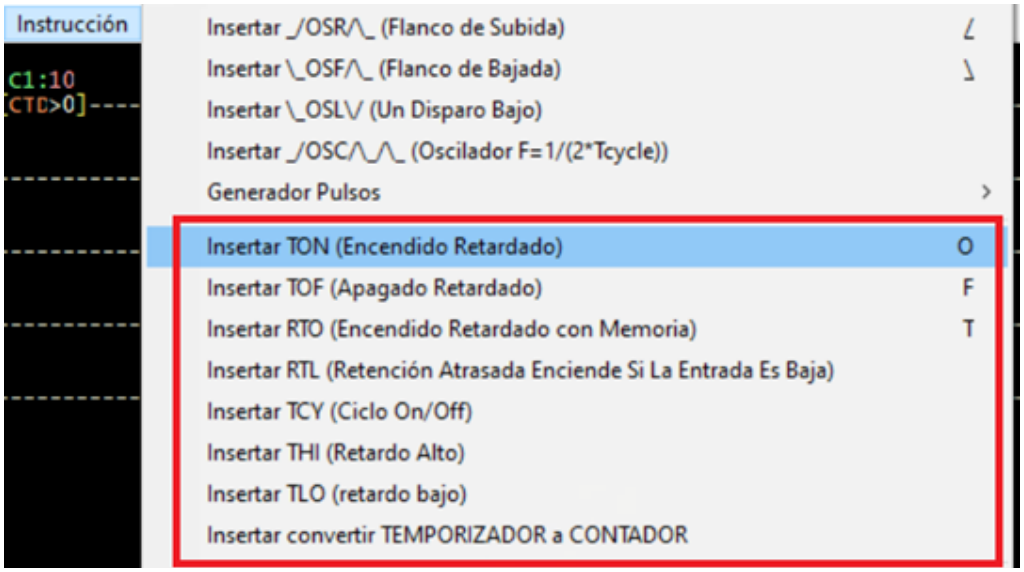
CAPÍTULO III

Temporizadores

Los temporizadores en circuitos de control industrial son muy importantes porque permiten controlar eventos en tiempos diferentes. LDmicro tiene una variedad de temporizadores, que son comunes a los de uso frecuente con los PLCs. En el menú desplegable opción Instrucción se encuentran los comandos para insertar en el diagrama Ladder los temporizadores.

Figura 48

Opciones para insertar temporizadores en LDmicro

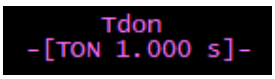


Temporizador encendido retardo TON

Es el temporizador de retardo a conexión, que comúnmente se conoce en control industrial y en los PLCs. Cuando la señal de entrada es verdadera, empieza el temporizador a contar el tiempo establecido previamente, activa a la salida asociada una vez finalizado el tiempo de retardo. Si la señal de entrada es falsa antes de que finalice el tiempo de retardo, la salida asociada al temporizador nunca se activará (CASTAÑEDA, 2015).

Figura 49

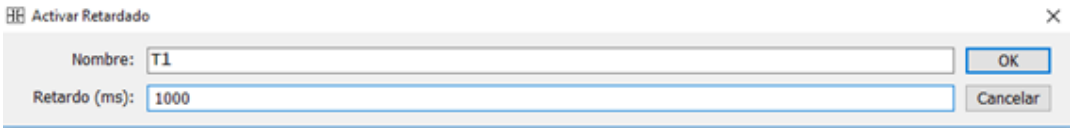
Temporizador encendido retardado



El tiempo puede ser establecido en milisegundos de 0 a 65536. Los valores del tiempo de retardo se fijan en la caja de datos del temporizador. A esta se accede al hacer doble clic en el símbolo del temporizador.

Figura 50

Caja de datos del temporizador TON



Ejemplo:

Implementar el diagrama Ladder para activar una lámpara L1, después de 5 segundos de haber accionado un pulsador. Comprobar el funcionamiento en Proteus.

Componentes:

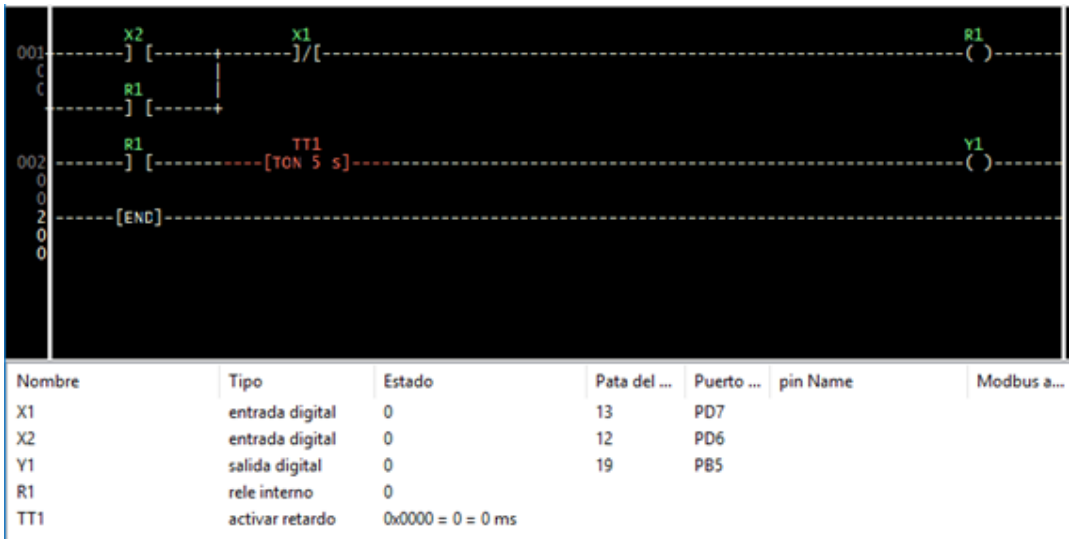
X1, pulsador para apagar.

X2, pulsador para activar la lámpara L1, transcurridos los 5 s.

Y1, salida para la lámpara L1.

Figura 51

Diagrama Ladder, aplicación del temporizador encendido retardado

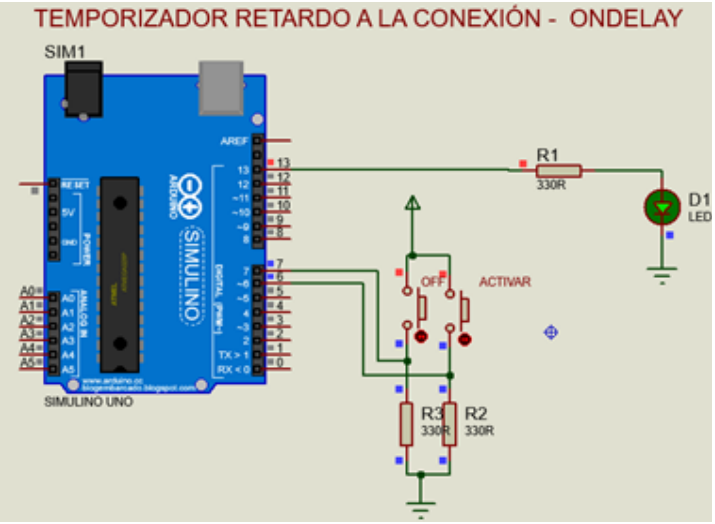


Nombre	Tipo	Estado	Pata del ...	Puerto ...	pin Name	Modbus a...
X1	entrada digital	0	13	PD7		
X2	entrada digital	0	12	PD6		
Y1	salida digital	0	19	PB5		
R1	rele interno	0				
TT1	activar retardo	0x0000 = 0 = 0 ms				

Al accionar X2, se enclava la salida R1 y activa el temporizador TT1, que empieza a contar 5s. Transcurrido este tiempo activa la salida Y1, cumpliendo las condiciones del ejercicio. En el simulador Proteus, también podemos ver el funcionamiento. Después de 5 segundos, de accionar el pulsador activar, el LED se activa. El pulsador OFF apaga el circuito.

Figura 52

Temporizador retardo a la conexión, simulado en ISIS

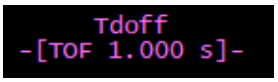


Temporizador apagado retardo TOF

En los PLC y relés programables, es el relé de retardo a la desconexión. Cuando la señal de entrada es verdadera, la salida asociada al temporizador se activa inmediatamente. Cuando la señal de entrada es falsa, el temporizador empieza a contar el tiempo de retardo y transcurrido este tiempo, la salida asociada al temporizador se desactiva (Castañeda, 2015).

Figura 53

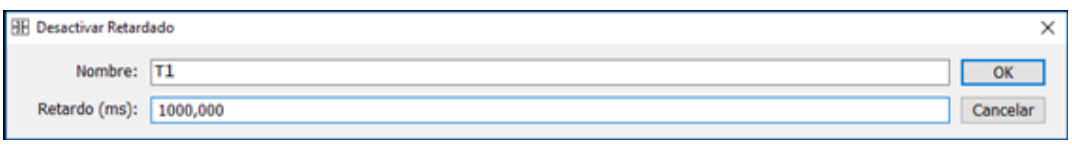
Temporizador apagado retardado



Al igual que el caso anterior, el tiempo puede ser establecido en milisegundos de 0 a 65536. Los valores del tiempo de retardo se fijan en la caja de datos del temporizador. A esta se accede al hacer doble clic en el símbolo del temporizador.

Figura 54

Caja de datos del temporizador TOF



Ejemplo:

Una lámpara L1 se activa al accionarse un pulsador P1. Una vez que está activada la lámpara, se acciona un pulsador P2. La lámpara continúa prendida por 5 segundos y se apaga.

Componentes:

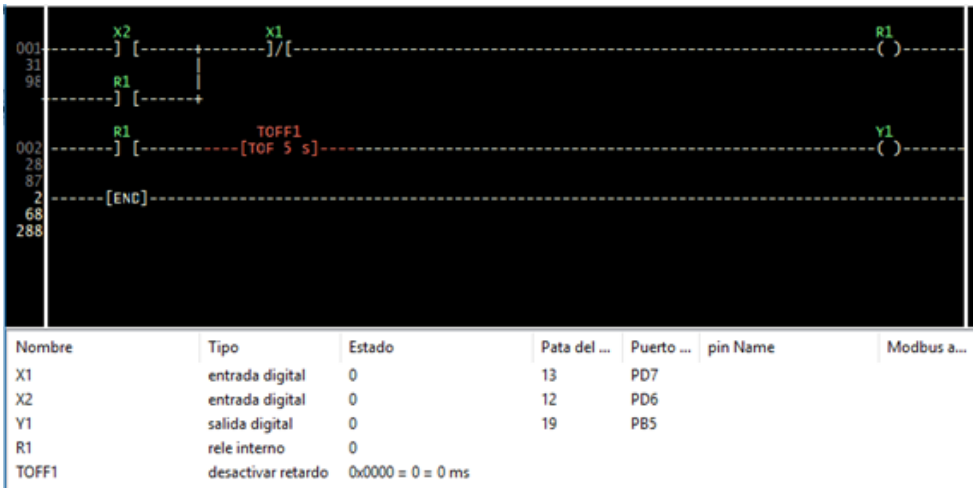
X1, pulsador OFF para desactivar la lámpara L1.

X2, pulsador ON para activar la lámpara L1.

Y1, salida para conectar la lámpara L1.

Figura 55

Diagrama Ladder, aplicación del temporizador apagado retardado

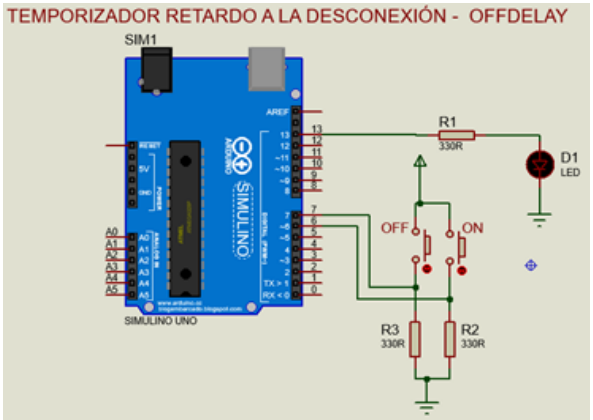


Nombre	Tipo	Estado	Pata del ...	Puerto ...	pin Name	Modbus a...
X1	entrada digital	0	13	PD7		
X2	entrada digital	0	12	PD6		
Y1	salida digital	0	19	PB5		
R1	rele interno	0				
TOFF1	desactivar retardo	0x0000 = 0 = 0 ms				

Mediante la simulación en el simulador Proteus, se puede visualizar de una mejor forma el funcionamiento del temporizador TOFF. Un pulso en el pulsador ON activa al temporizador y al LED. Al accionar el pulsador OFF, transcurre 5 segundos y el LED se apaga.

Figura 56

Circuito para aplicación del temporizador apagado retardado

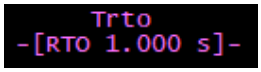


Temporizador encendido retardado con memoria RTON

Cuando hay señal en la entrada, el temporizador empieza a contar el tiempo de retardo establecido. Transcurrido ese tiempo, se activa la salida asociada al temporizador. En este estado si la señal de entrada es falsa, la salida sigue activada y para desactivarla debemos aplicar un pulso de RESET.

Figura 57

Temporizador encendido retardado con memoria



Ejemplo:

Activar una salida Y1 después de 5 s., de accionar un pulsador P1. Con un pulsador P2, desactivar el circuito.

Tabla 3

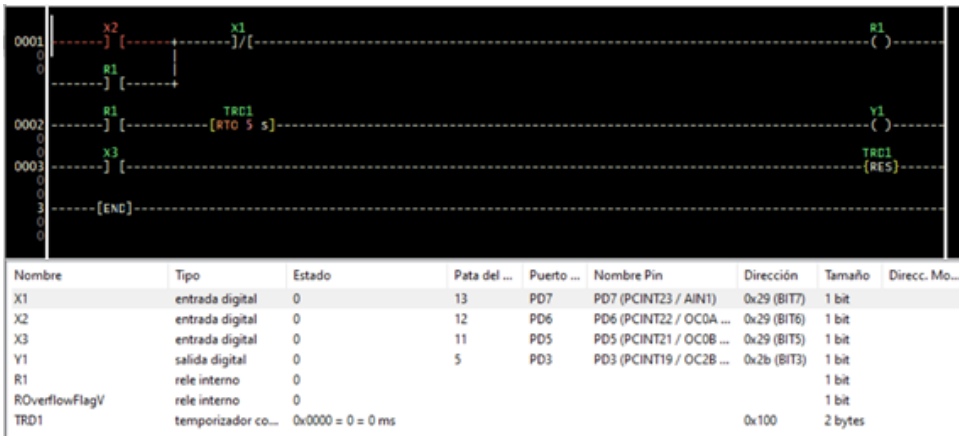
Asignación de pines

ENTRADAS SALIDAS	COMPONENTES	FUNCIÓN
X1	P1	PULSADOR DE PARO
X2	P2	PULSADOR DE INICIO
Y1	L1	LÁMPARA 1

En este caso, el temporizador solo necesita en su entrada un pulso para activarse y no es necesario mantener la señal de entrada como en el temporizador retardado a la conexión. Con un pulso en la bobina de RESET, que debe estar asociada al temporizador, permite desactivar al componente.

Figura 58

Aplicación de un temporizador encendido retardo con memoria



Al accionar X2, se enclava el relé interno R1, energizando al temporizador TRD1, transcurrido el tiempo de temporización (3s), se acciona la salida Y1. Al accionar X1, desactiva al relé interno R1, pero Y1 continúa activado debido a la acción de memoria del temporizador. La salida Y1 se desactiva al resetear el temporizador, mediante la entrada X3.

Ejercicios:

1. Prender dos lámparas en secuencia. La lámpara L1 se acciona mediante un pulsador P1 y transcurridos 10 s., se acciona la lámpara L2. Se dispone de un pulsador de paro para apagar las lámparas en cualquier momento.

Empecemos realizando la tabla de asignaciones para los componentes.

Tabla 4

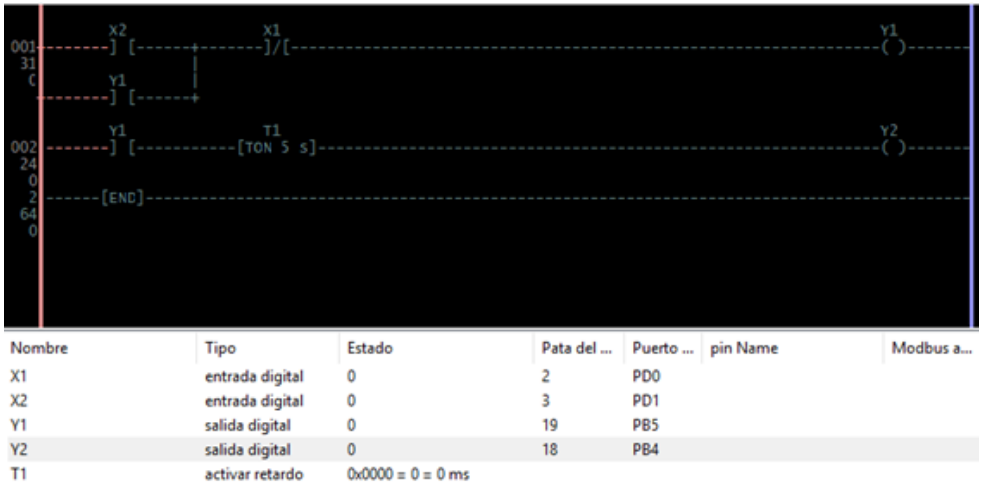
Asignación de pines

ENTRADAS SALIDAS	COMPONENTES	FUNCIÓN
X1	P1	PULSADOR DE PARO
X2	P2	PULSADOR DE INICIO
Y1	L1	LÁMPARA 1
Y2	L2	LÁMPARA 2

Interpretando el enunciado necesitamos enclavar la salida Y1 para activar la lámpara L1, esto lo hacemos con el pulsador P1 que está conectado a la entrada X1. Al mismo tiempo debe funcionar un temporizador con retardo a la conexión para que despues de 10 s., active la salida Y2. El contacto en serie de Y1 activa al temporizador y luego de 10 s., el contacto de T01 activa Y2. Al accionar X2, desconecta Y1 y esta a su vez al temporizador y por tanto a Y2.

Figura 59

Activado temporizado de salida Y2



Nombre	Tipo	Estado	Pata del ...	Puerto ...	pin Name	Modbus a...
X1	entrada digital	0	2	PD0		
X2	entrada digital	0	3	PD1		
Y1	salida digital	0	19	PB5		
Y2	salida digital	0	18	PB4		
T1	activar retardo	0x0000 = 0 = 0 ms				

2. Prender una lámpara L1 con un pulsador P2 y transcurridos 10 s., se acciona la lámpara L2 y se apaga la lámpara L1. Se dispone de un pulsador de paro para apagar las lámparas en cualquier momento.

Empecemos realizando la tabla de asignaciones para los componentes.

Tabla 5

Asignación de pines

ENTRADAS SALIDAS	COMPONENTES	FUNCIÓN
X1	P1	PULSADOR DE PARO
X2	P2	PULSADOR DE INICIO
Y1	L1	LÁMPARA 1
Y2	L2	LÁMPARA 2

Este ejercicio es similar al anterior, sólo que la lámpara L1, se apaga cuando la lámpara L2 se prende, por tanto, el temporizador debe enclavar a la salida Y2 y desactivar a Y1.

Figura 60

Activado temporizado de Y2 y apagado de Y1

Nombre	Tipo	Estado	Pata del ...	Puerto ...	pin Name	Modbus a...
X1	entrada digital	1	2	PD0		
X2	entrada digital	0	3	PD1		
Y1	salida digital	0	19	PB5		
Y2	salida digital	0	18	PB4		
R1	rele interno	0				
T1	activar retardo	0x0000 = 0 = 0 ms				

3. Activar dos salidas Y1 y Y2 mediante un pulsador P2. Desactivar Y1 mediante un pulsador P1 y partir de este instante, Y2 queda activada por 5 s más, apagándose automáticamente transcurrido este tiempo.

De acuerdo al enunciado, procedemos a realizar la tabla de asignaciones:

Tabla 6

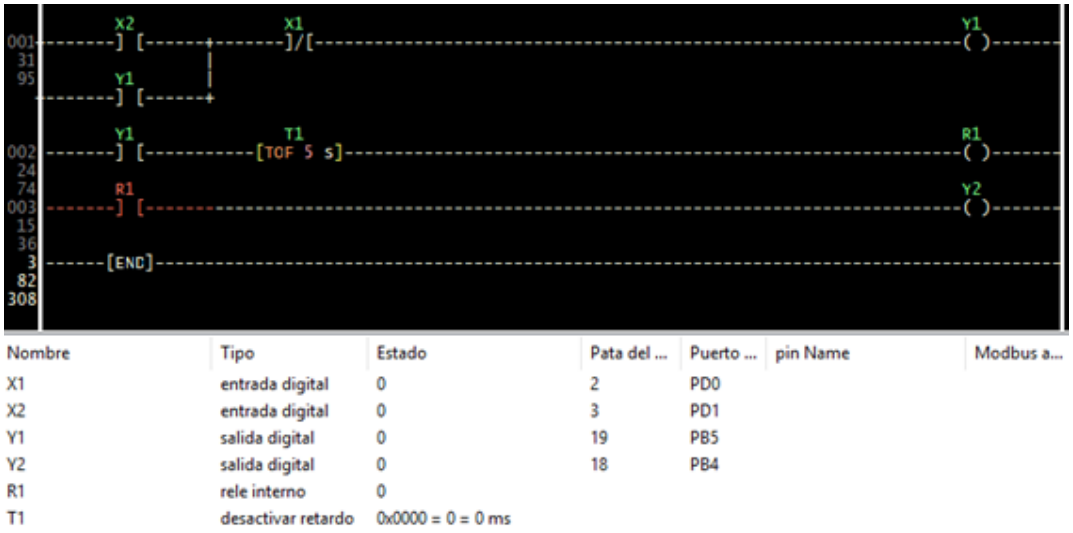
Asignación de pines

ENTRADAS SALIDAS	COMPONENTES	FUNCIÓN
X1	P1	PULSADOR DE PARO
X2	P2	PULSADOR DE INICIO
Q1	L1	SALIDA 1
Q2	L2	SALIDA 2

Diagrama Ladder

Figura 61

Desactivación de la salida Y2, mediante el temporizador retardo a la desconexión



Nombre	Tipo	Estado	Pata del ...	Puerto ...	pin Name	Modbus a...
X1	entrada digital	0	2	PD0		
X2	entrada digital	0	3	PD1		
Y1	salida digital	0	19	PB5		
Y2	salida digital	0	18	PB4		
R1	rele interno	0				
T1	desactivar retardo	0x0000 = 0 = 0 ms				

Funcionamiento:

Al accionar X2 se enclava Y1, al mismo tiempo cierra el contacto abierto de Y1 que activa al temporizador T1 y el relé R1. El contacto R1, también se cierra activando a Y2. Al accionar el pulsador de la entrada X1, desactiva a Y1 y el temporizador T1, el mismo que empieza a contar (se abre Y1 y activa el conteo del temporizador, el relé continúa activado). Transcurrido los 5 s., apaga al relé R1, abriendo el contacto R1 y apaga la salida Y2.

4. Activar una salida Y1 y Y2 simultáneamente al accionar un pulsador P1. Transcurridos 10 s., se apaga Y1 y Y2 sigue funcionando por 10 s apagándose automáticamente. Para realizar el Ladder usar un temporizador retardo a la conexión y uno a la desconexión.

Para este ejemplo se utiliza un temporizador retardo a la conexión para, mediante un contacto cerrado, desactivar Y1 y el temporizador retardo a desconexión apagar a la salida Y2.

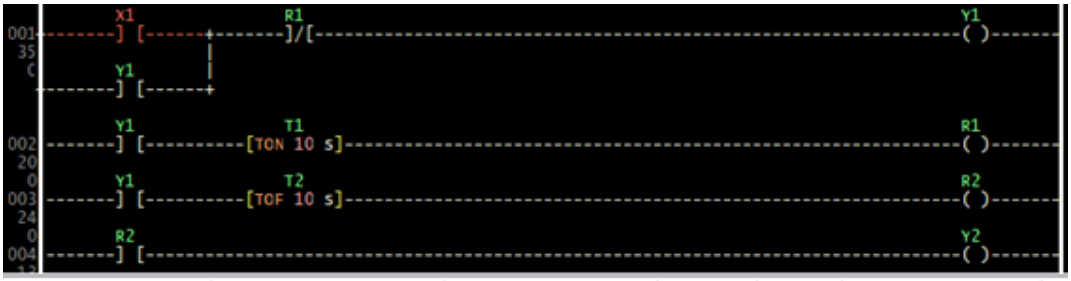
Tabla 7

Asignación de pines

ENTRADAS SALIDAS	COMPONENTES	FUNCIÓN
X1	P1	PULSADOR DE INICIO
Y1	L1	SALIDA 1
Y2	L2	SALIDA 2

Figura 62

Desactivado de salidas mediante temporizadores



Nombre	Tipo	Estado	Pata del ...	Puerto ...	pin Name
X1	entrada digital	0	2	PD0	
Y1	salida digital	0	19	PB5	
Y2	salida digital	0	18	PB4	
R1	rele interno	0			
R2	rele interno	0			
T1	activar retardo	0x0000 = 0 = 0 ms			
T2	desactivar retardo	0x03e8 = 1000 = 10 s			

Funcionamiento:

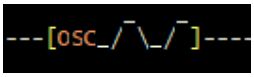
Al recibir un pulso la entrada X1 enclava Y1, activa a los temporizadores T1 y T2. T1 TON comienza a contar los 10 s., T2 TOF, activa al relé R2 y el contacto de este relé activa a Y2. Al pasar los 10 s., se energiza R1 y su contacto cerrado asociado a la rama de la salida Y1, se abre momentáneamente desactivando esta salida y el temporizador T2, empieza a contar el tiempo de retardo. Transcurrido los 10 s., se desactiva R2 y apaga a Y2.

Oscilador

La función oscilador si la señal de entrada es verdadera, esta instrucción produce una señal con período T= (2 * Tiempo de ciclo del PLC). Si la señal de entrada la instrucción es falsa, entonces la señal de salida es falsa (Westhues, 2009).

Figura 63

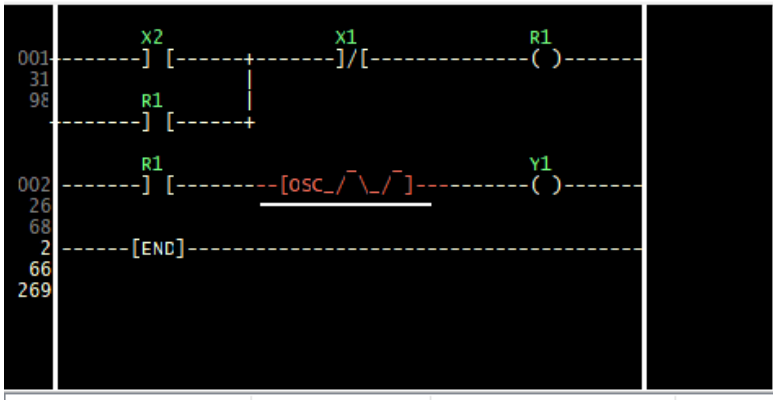
Función oscilador



En la siguiente figura se indica el circuito para demostrar la función del oscilador.

Figura 64

Función oscilador



Nombre	Tipo	Estado	Pata del ...
X1	entrada digital	0	12
X2	entrada digital	0	13
Y1	salida digital	0	19
R1	rele interno	1	

Funcionamiento:

Al recibir un pulso la entrada X1 enclava R1, activa al oscilador, que empezará a generar una señal cuadrada de frecuencia igual a la mitad del reloj del sistema.

La simulación del circuito en Proteus muestra la onda generada.

Figura 65

Circuito para la simulación en Proteus de la función Oscilador

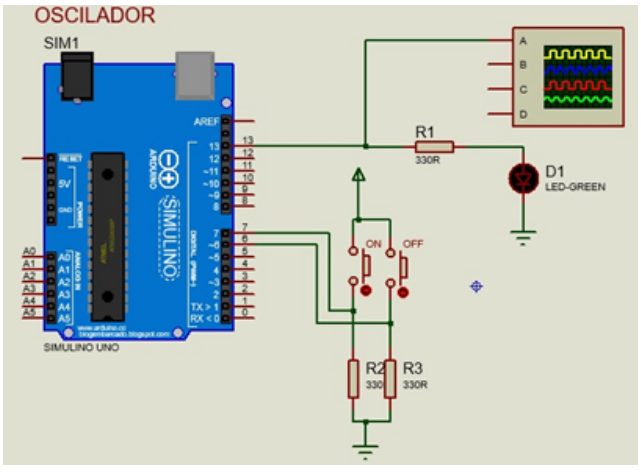
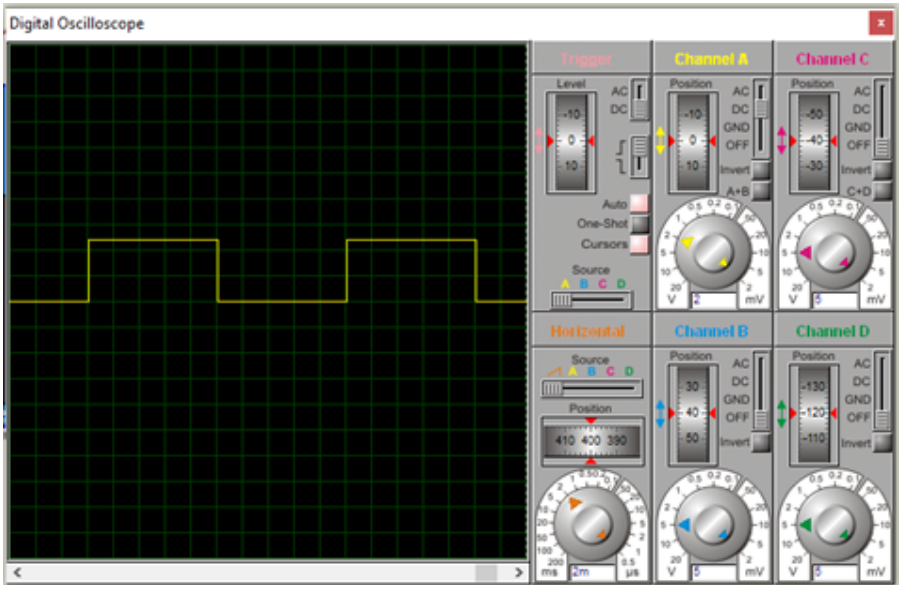


Figura 66

Onda cuadrada, simulación en Proteus

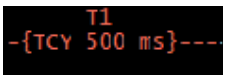


Temporizador cíclico

Genera una señal cuadrada de 1s. y una frecuencia de 1Hz, si la señal de entrada en la caja del símbolo es verdadera, y permanece en 0, si la entrada es 0 o falsa (Westhues, 2009).

Figura 67

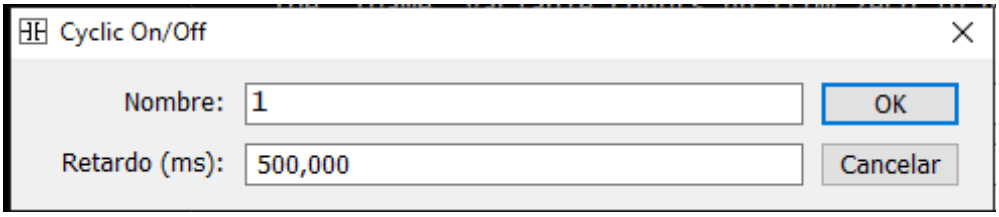
Símbolo de la instrucción temporizador cíclico



En la caja de texto de esta instrucción se debe definir el nombre y el periodo de retardo del temporizador.

Figura 68

Definición de parámetros para el temporizador cíclico



Ejemplo:

El programa más famoso "Hola mundo", escrito inicialmente por Brian W. Kernighan en 1973 para computadoras, en microcontroladores o PLCs su equivalente es prender y apagar un diodo LED, es decir generar una onda cuadrada. La instrucción Temporizador cíclico permite fácilmente realizar la generación de una onda cuadrada con una frecuencia de 1 HZ.

Figura 69

Diagrama Ladder para activar y desactivar una salida que maneja un LED

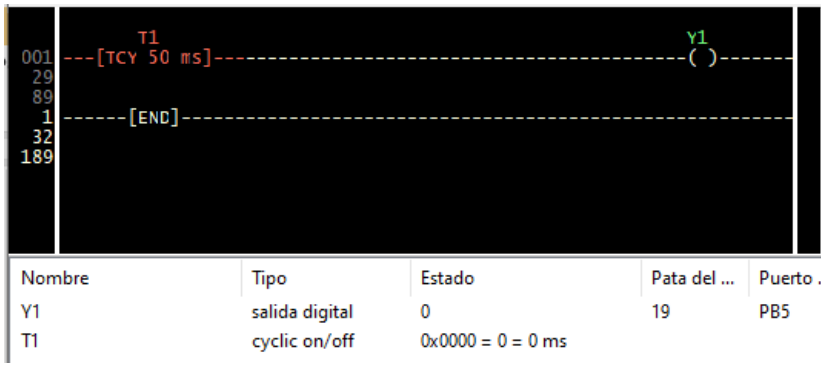


Figura 70

Circuito en Proteus para titilar un LED

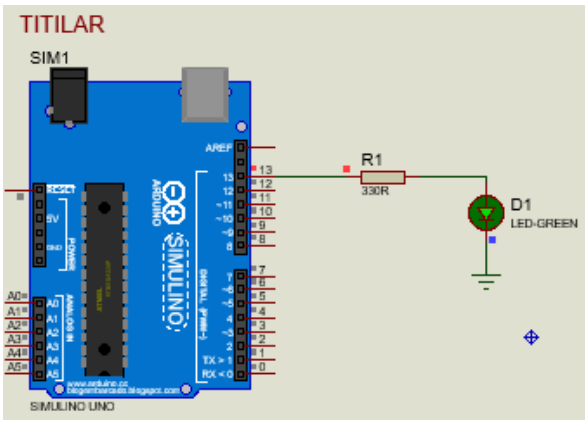
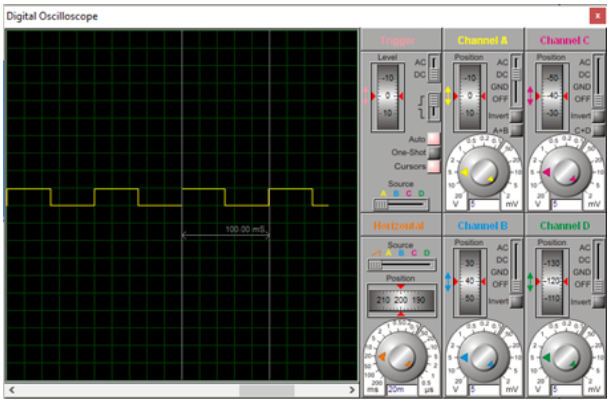


Figura 71

Señal generada por el temporizador cíclico



Ejercicios propuestos

1. Conectar tres salidas sucesivamente cada 5 s. primero Y1, luego Y2 y finalmente Y3. Las tres salidas quedan activadas. Se tiene un pulsador de marcha y otro paro. Realizar tres soluciones con temporizadores diferentes.

2. Similar al anterior, pero al final sólo quedará encendida la última salida.

3. Realizar el circuito para controlar dos semáforos. Disponer de pulsadores de marcha y paro.

4. Realizar el diagrama Ladder para automatizar el arranque de dos motores M1 y M2, 1ue funcionan de la siguiente forma:
   - El motor M1, arranca con se acciona simultáneamente dos pulsadores P1 y P2.
   - El motor M2 funciona al pulsar P1 y P2, pero si M1 ha funcionado 10s. Los dos motores en este momento deben estar en funcionamiento normal.
   - Los dos motores tendrán su respectivo pulsador de paro, así como las protecciones térmicas correspondientes. Luces pilotos deben activarse cuando los motores funcionan.

CAPÍTULO IV

Contadores y registros

Otras de las funciones de los PLC son el uso de contadores y registros, muy útiles para realizar el control de distintas variables en automatismos industriales. LDmicro posee las funciones necesarias que permiten implementar diferentes contadores como incrementales, decrementales, circular, circular reversible y registros de desplazamiento.

Contadores

Los principales contadores que LDmicro dispone y la mayor parte de PLCs, son los siguientes:

Contador incremental/decremental

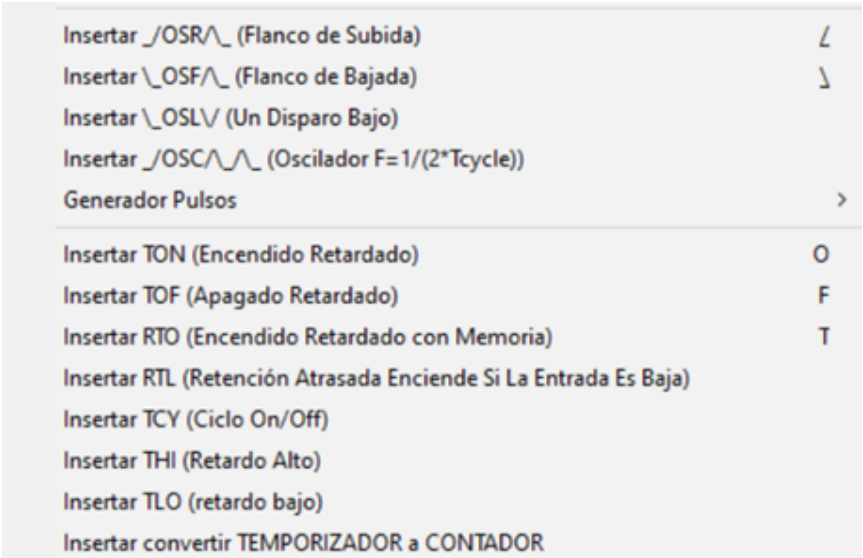
El contador incrementa (CTU, contador ascendente) o decrementa (CTD, contador descendente), el conteo asociado en cada flanco ascendente de un pulso de entrada. La condición de salida del contador es verdadera si cumple con la variable establecida y es falsa en caso contrario (Westhues, 2009).

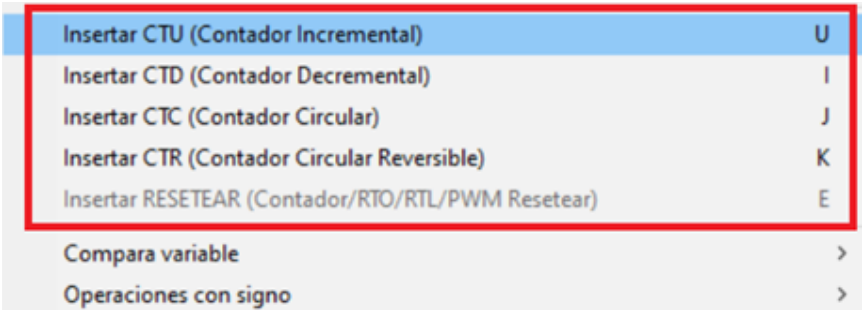
En la caja de datos del contador se estable el nombre, que empezará por defecto con la letra C que identifica al contador, un valor inicial y la condición donde la salida será verdadera.

En el menú desplegable del programa LDmicro opción Instrucción, encontramos los comandos para insertar en el diagrama Ladder los contadores.

Figura 72

Opciones para insertar los contadores



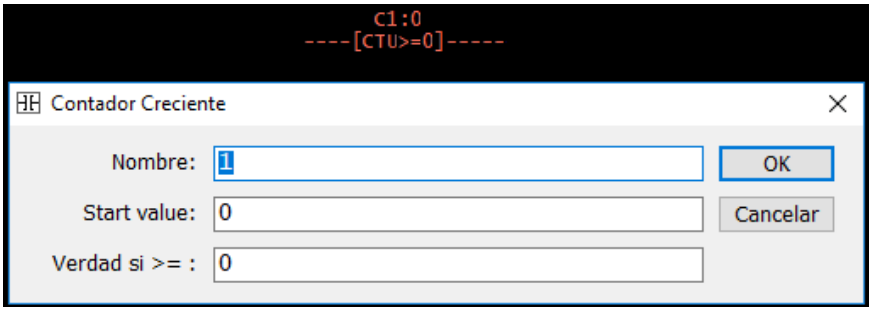


Contador incremental o creciente CTU

El contador incremental o creciente (CTU), la variable de conteo incrementa su cuenta cuando en la entrada de la línea de condición del contador hay un cambio en el flanco ascendente (de 0 a 1). Como la condición del contador es >= los pulsos que son ingresados en este estado seguirán actualizando la variable del conteo. El contador puede ser reiniciado mediante el comando con una línea de RES (RESET) (Westhues, 2009).

Figura 73

Parámetros de un contador creciente y decreciente



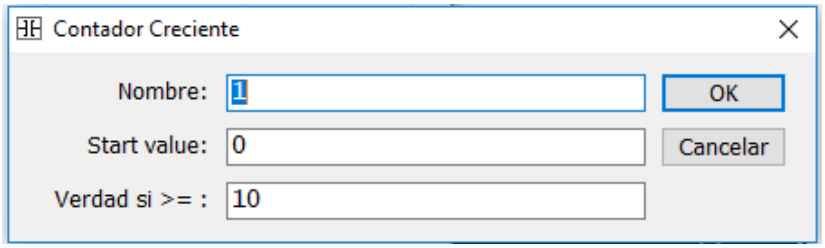
Ejemplo: Contador incremental.

Figura 74

Diagrama LDR para inicialización de parámetros de un contador creciente

Figura 75

Definición de parámetros de un contador incremental



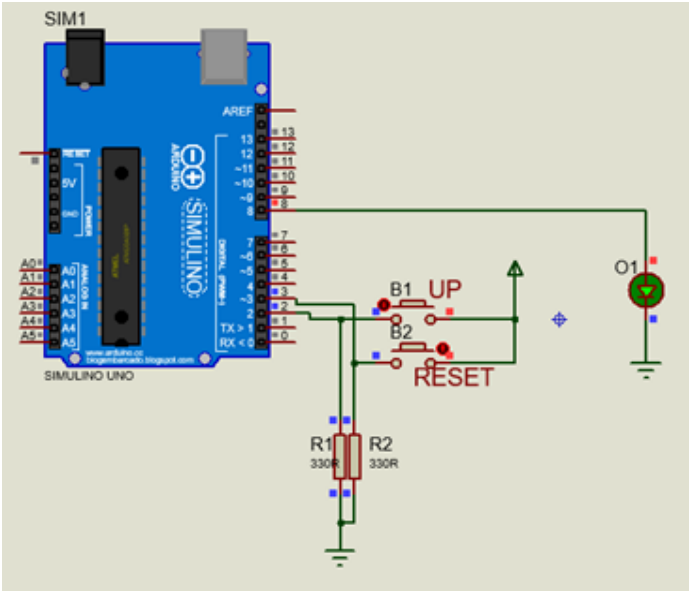
En el diagrama LDR del ejemplo del contador incremental, la entrada X1 es por donde ingresen los pulsos, cuando el contador sea mayor o igual a 10 activa a la salida Y1. X2 sirve para reiniciar el contador.

En el simulador Proteus también podemos simular el funcionamiento del contador incremental.

Los pulsadores B1 sirve para generar los pulsos, cuando se de 10 pulsos, la salida Y1 se prende, un pulso en el botón B2, restablece el contador a 0 y se apaga la salida Y1.

Figura 76

Circuito en Proteus para simular el contador decreciente



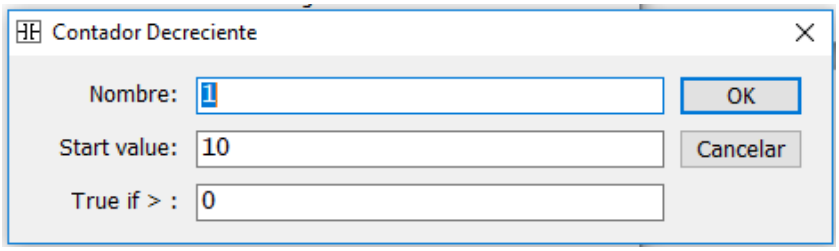
Contador decremental o decreciente CTD

Un decremento en el contador (CTD, cuenta hacia abajo) es el conteo asociado en cada flanco ascendente (de 0 a 1) de la entrada de línea condición del contador. Para este caso la condición es >, por lo que esta línea cumplirá solo

cuando la condición sea verdadera. El contador puede ser reiniciado mediante el comando con una línea de RES (RESET).

Figura 77

Parámetros de un contador decremental

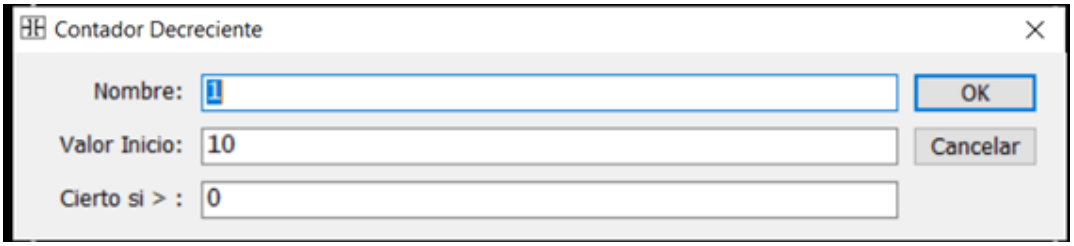


Ejemplo: Contador decremental.

Se fija en el contador el valor inicial en 10 y el valor verdadero si es mayor a 0. En el diagrama Ladder primero debemos cargar el valor de 10 a la variable C1, dando un pulso en la entrada X1, porque esta variable inicialmente está en 0. Si no hacemos esto el contador, al recibir los pulsos empezará a contar -1,-2,-3,-4,… y nunca se cumplirá la condición verdadera. Con la entrada X0 cargamos el valor de 10 a la variable C1.

Figura 78

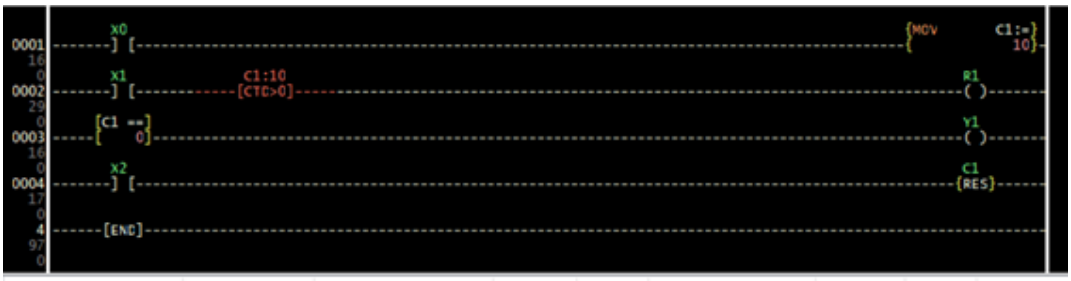
Definición de parámetros de un contador decremental



Una vez cargado el valor C1=10, el relé interno R1 está activado. Asociado a la salida Y1 está incluido una condición si el valor de conteo de C1==0. Al recibir los pulsos el contador decrementa el conteo a partir de 10 y al llegar a 0, cumple la condición C1==0, activando la salida Y1. Con un pulso en X2, el contador se restablece a 0.

Figura 79

Circuito con un contador decremental

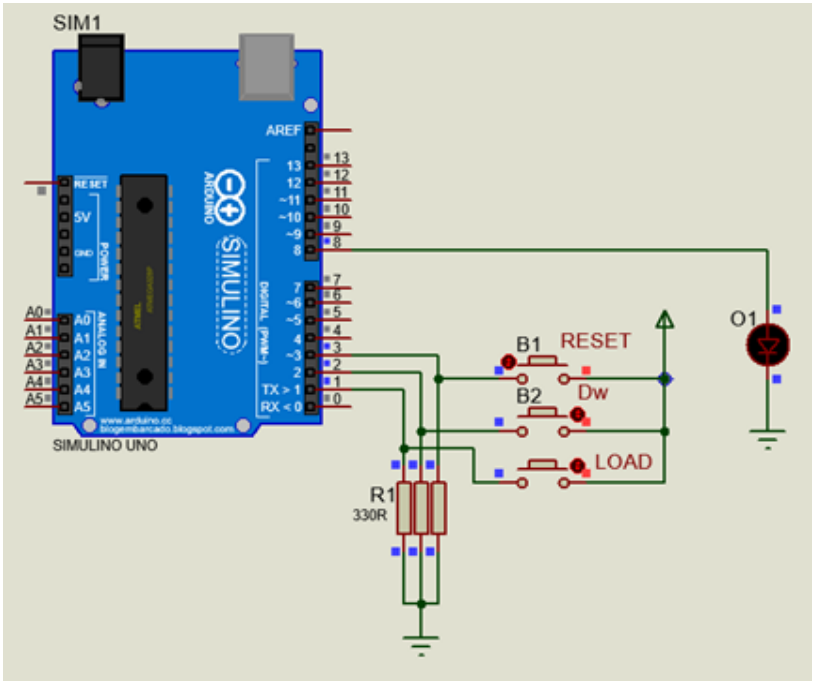


Nombre	Tipo	Estado	Pata del ...	Puerto ...	Nombre Pin	Dirección	Tamaño	Direcc. Mo...
C1	contador	0x0000 = 0				0x102	2 bytes	
X0	entrada digital	0	3	PD1	PD1 (PCINT17 / TXD)	0x29 (BIT1)	1 bit	
X1	entrada digital	0	4	PD2	PD2 (PCINT18 / INT0)	0x29 (BIT2)	1 bit	
X2	entrada digital	0	5	PD3	PD3 (PCINT19 / OC2B ...	0x29 (BIT3)	1 bit	
Y1	salida digital	0	14	PB0	PB0 (PCINT0 / CLKO / ...	0x25 (BIT0)	1 bit	
R1	rele interno	1				0x101 (BIT0)	1 bit	
ROverflowFlagV	rele interno	0				0x101 (BIT1)	1 bit	

En el simulador Proteus podemos simular el funcionamiento del contador decreciente con el circuito indicado. El pulsador LOAD carga al contador con el valor de10, los pulsos se generan con el pulsador DW, cuando se cumple la condición de C1= 0 el LED se activa. Al accionar RESET, el contador se restablece a 0.

Figura 80

Circuito en Proteus para simular el contador decreciente



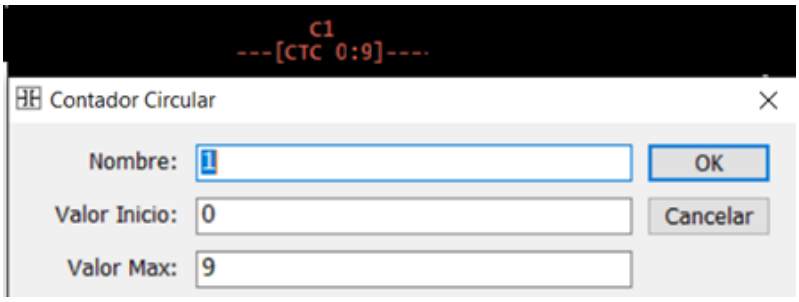
Contador circular (CTC)

El contador circular permite establecer un valor de inicio y un valor máximo. El contador circular tiene el mismo funcionamiento que el contador CTU, pero después de alcanzar su límite máximo, restablece la variable de conteo a 0 y al recibir los pulsos empezará una nueva cuenta La variable C1 se incrementa cuando recibe un pulso en flanco de subida (Westhues, 2009).

Por ejemplo, el contador indicado en la figura contaría del 0 al 9, continuamente.

Figura 81

Contador circular



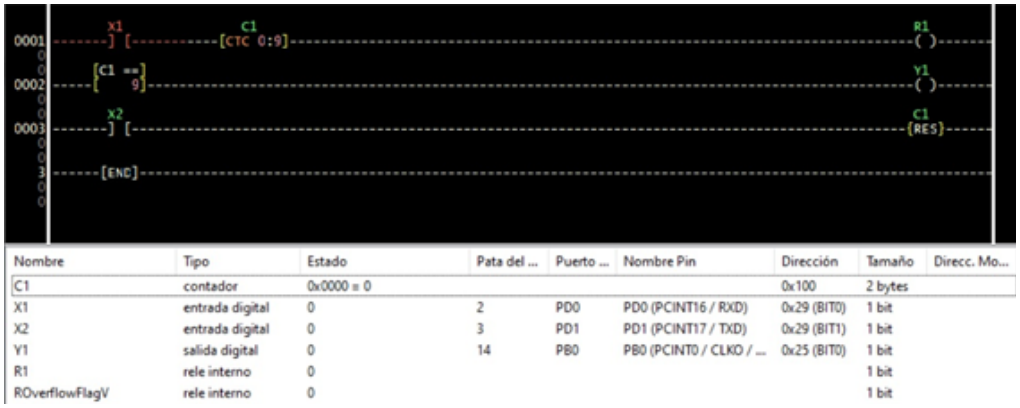
Ejemplo:

Utilizando un contador circular activar una salida Y1, cada vez que el contador reciba 10 pulsos.

Los pulsos que ingresan por la entrada digital X1 ingresan al contador y la variable C1, se incrementará cada vez que ingresa un pulso. Cuando el contador es igual a 9, el comparador de igualdad activara a la salida Y1. El contador al siguiente pulso retornará a 0 y la cuenta continuará de esta forma.

Figura 82

Aplicación de un contador circular



Nombre	Tipo	Estado	Pata del ...	Puerto ...	Nombre Pin	Dirección	Tamaño	Direcc. Mo...
C1	contador	0x0000 = 0				0x100	2 bytes	
X1	entrada digital	0	2	PD0	PD0 (PCINT16 / RXD)	0x29 (BIT0)	1 bit	
X2	entrada digital	0	3	PD1	PD1 (PCINT17 / TXD)	0x29 (BIT1)	1 bit	
Y1	salida digital	0	14	PB0	PB0 (PCINT0 / CLKO / ...	0x25 (BIT0)	1 bit	
R1	rele interno	0					1 bit	
ROverflowFlagV	rele interno	0					1 bit	

Contador circular reversible (CTR)

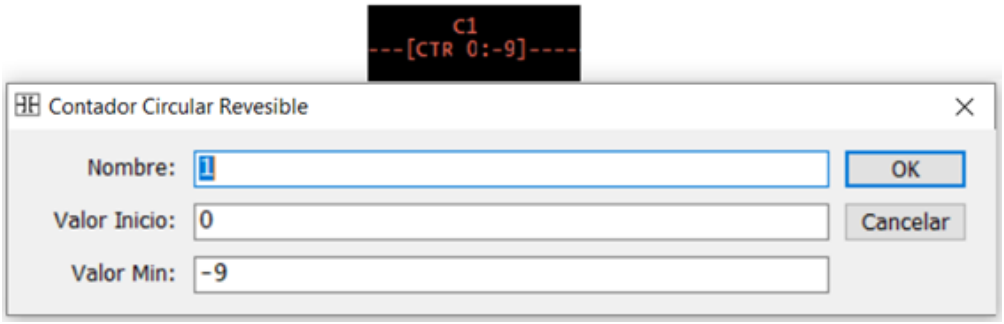
El contador circular reversible permite establecer un valor de inicio y un valor mínimo.

El funcionamiento del contador circular es similar al contador CTD, con la excepción que el valor mínimo la variable del contador se reinicia a 0 y el conteo comenzará desde este valor. La variable C1 se decrementa cuando recibe un pulso en flanco de subida (Westhues, 2009).

Por ejemplo, el contador mostrado contaría del 0 al -9, repetidamente.

Figura 83

Definición del contador circular reversible



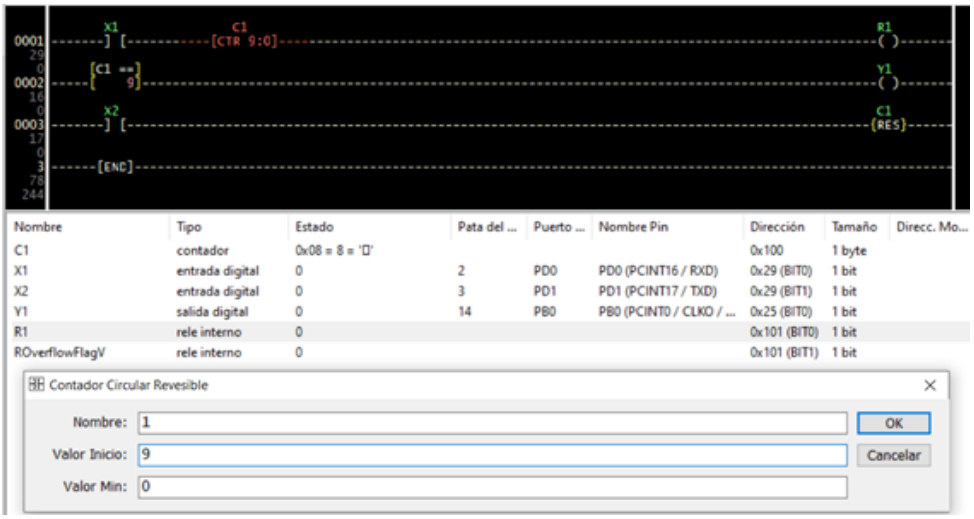
Ejemplo:

Utilizando un contador circular reversible activar una salida Y1, cada vez que el contador reciba 10 pulsos y el valor de contador sea 0.

Para este caso definimos el valor de inicio en 9 y el valor mínimo en 0.

Figura 84

Aplicación del contador circular reversible



Los pulsos que ingresan por la entrada digital X1 ingresan al contador y la variable C1, se incrementará cada vez que ingresa un pulso. Cuando el contador es igual a 9, el comparador de igualdad activará a la salida Y1. El contador al siguiente pulso retornará a 0 y la cuenta continuará de esta forma.

Registros

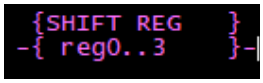
Registro de desplazamiento

Un registro de desplazamiento está asociado con un conjunto de variables: reg0, reg1, reg2 y reg3. Estas variables son de propósito general tipo int unsigned. La entrada de datos al registro de desplazamiento es por la variable reg0.

En cada flanco ascendente en la línea de entrada del registro, el registro de desplazamiento cambiará su valor a la derecha. Eso significa que asigna el valor de reg0 a reg1, el valor de reg1 a reg2, reg2 a reg3, y así sucesivamente, si tuviéramos más variables. Se debe tener en cuenta que un registro de desplazamiento grande consume mucha memoria (Westhues, 2009).

Figura 85

Símbolo de un registro de desplazamiento

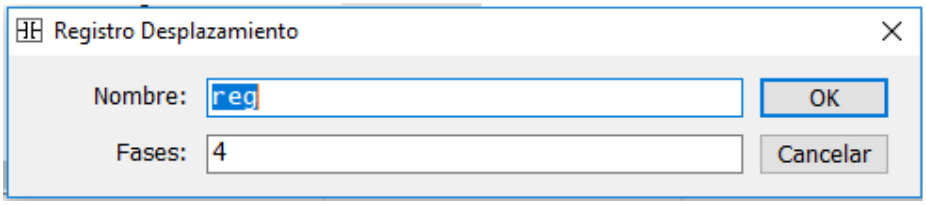


Ejemplo:

En el siguiente ejercicio definimos el nombre y el número de variables asociadas al registro. Se generan 4 variables reg0, reg1, reg2 y reg3.

Figura 86

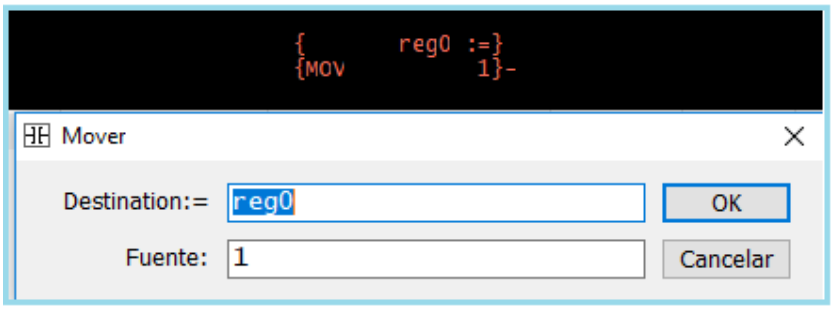
Generación de nombre y variables del registro



Mediante la instrucción MOV (Mover), cargamos a la variable reg0 con 1.

Figura 87

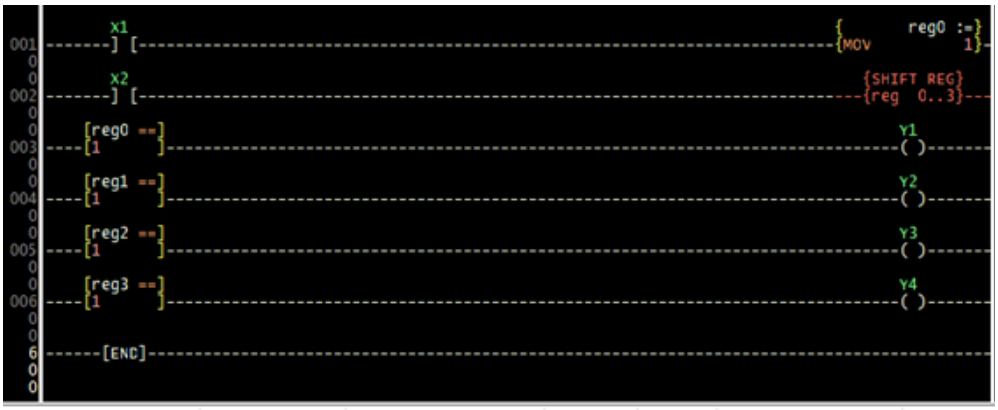
Instrucción Mover. Símbolo y asignación de valores



Cuando reg0, reg1, reg2 y reg3 sean igual a 1, activarán las salidas Y1, Y2, Y3 y Y4. Con un pulso en X1 se carga la variable reg0, se activa Y1. Cada vez que ingrese un pulso por X2 se irán cargando las variables reg1, reg2, reg3 y activándose las salidas asociadas, con lo que estamos teniendo el efecto de un registro de desplazamiento.

Figura 88

Registro de desplazamiento



Nombre	Tipo	Estado	Pata del ...	Puerto ...	pin Name	Modbus a...
reg0	var general	0x0000 = 0				
reg1	var general	0x0000 = 0				
reg2	var general	0x0000 = 0				
reg3	var general	0x0000 = 0				
X1	entrada digital	0	3	PD1		
X2	entrada digital	0	4	PD2		
Y1	salida digital	0	14	PB0		
Y2	salida digital	0	15	PB1		
Y3	salida digital	0	16	PB2		
Y4	salida digital	0	17	PB3		

Ejemplo:

Realizar la secuencia indicada en la tabla. Debe haber un pulsador de inicio para empezar la secuencia y uno de paro para detener. Cuando se detenga la secuencia al accionar el pulsador de paro, al pulsar el pulsador inicio la secuencia continuará normalmente.

Tabla 8

Asignación de pines

Y1	Y2	Y3	Y4	Y5	Y6
0	0	0	0	0	0
1	0	0	0	0	0
1	1	0	0	0	0
1	1	1	0	0	0
1	1	1	1	0	0
1	1	1	1	1	0
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
0	1	1	1	1	1
0	0	1	1	1	1
0	0	0	1	1	1
0	0	0	0	1	1
0	0	0	0	0	1
Repite la secuencia					

El diagrama LDR siguiente muestra la solución al ejercicio planteado.

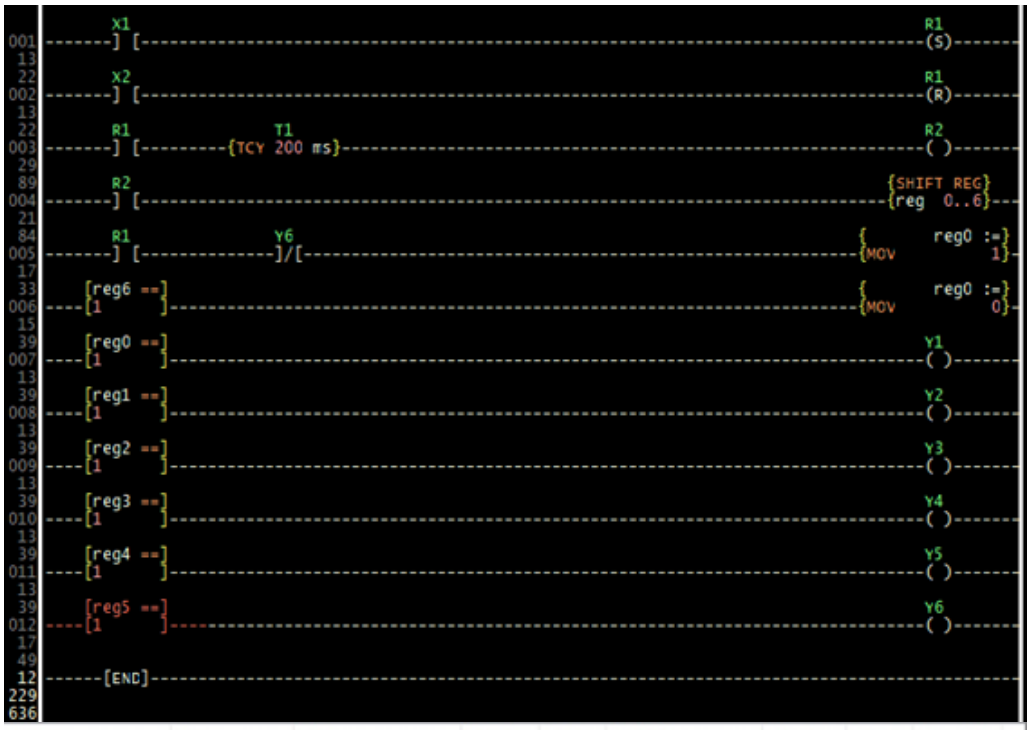
Funcionamiento:

El funcionamiento del circuito es como sigue:

- Un relé interno R1, actúa como interruptor ON/OFF (SET/RESET). X1, activa R1 habilitando al relé interno R2 que está asociado al temporizador TCY genera una señal cuadrada de 200 ms en alto. Por tanto, el relé R2, a través de un contacto servirá como la salida de un generador de pulsos.
- Los pulsos del generador ingresan al registro, donde se han definido las variables reg0, reg1, reg2, reg3, reg4 y reg5, para que los datos empiecen a desplazarse.
- Cuando R1 se activa, carga con reg0= 1, y activa a la salida Y1. Este dato estará presente siempre en la entrada del registro.
- Cada vez que se genere en R2 un pulso las variables del registro van cargándose con 1 y activando las salidas que están asociadas al registro.

Figura 89

Diagrama Ladder ejemplo del registro de desplazamiento



Nombre	Tipo	Estado	Pata del ...	Puerto ...	Nombre Pin	Dirección	Tamaño	Direcc. Mo...
reg0	var general	0x0000 = 0				0x100	2 bytes	
reg1	var general	0x0000 = 0				0x102	2 bytes	
reg2	var general	0x0000 = 0				0x104	2 bytes	
reg3	var general	0x0000 = 0				0x106	2 bytes	
reg4	var general	0x0000 = 0				0x108	2 bytes	
reg5	var general	0x0000 = 0				0x10a	2 bytes	
reg6	var general	0x0000 = 0				0x10c	2 bytes	
X1	entrada digital	0	12	PD6	PD6 (PCINT22 / OC0A ...	0x29 (BIT6)	1 bit	
X2	entrada digital	0	13	PD7	PD7 (PCINT23 / AIN1)	0x29 (BIT7)	1 bit	
Y1	salida digital	0	14	PB0	PB0 (PCINT0 / CLKO / ...	0x25 (BIT0)	1 bit	
Y2	salida digital	0	15	PB1	PB1 (OC1A / PCINT1)	0x25 (BIT1)	1 bit	
Y3	salida digital	0	16	PB2	PB2 (SS / OC1B / PCIN...	0x25 (BIT2)	1 bit	
Y4	salida digital	0	17	PB3	PB3 (MOSI / OC2A / P...	0x25 (BIT3)	1 bit	
Y5	salida digital	0	18	PB4	PB4 (MISO / PCINT4)	0x25 (BIT4)	1 bit	
Y6	salida digital	0	19	PB5	PB5 (SCK / PCINT5)	0x25 (BIT5)	1 bit	
R1	rele interno	0					1 bit	
R2	rele interno	0					1 bit	
ROverflowFlagV	rele interno	0					1 bit	
T1	ciclo on/off	0x0000 = 0 = 0 ms				0x10e	2 bytes	

- Cuando se activa Y6, el contacto que está asociado a mover el dato de reg0 se abre, por consiguiente, bloquea la entrada a R1 y carga con 0 a la variable reg0. Con esto empieza a ingresar 0 en las demás variables apagando las salidas externas.

Figura 90

Operación de carga de datos al registro de desplazamiento

- Al desactivarse Y6, su contacto se cierra y nuevamente reg0 = 1, empieza a desplazarse el 1 a las demás variables repitiéndose el ciclo.
- Al accionar la entrada X2 desactiva el relé interno R1 y detiene la secuencia del circuito.

Para la simulación en Proteus asignamos los componentes como se muestra en la tabla.

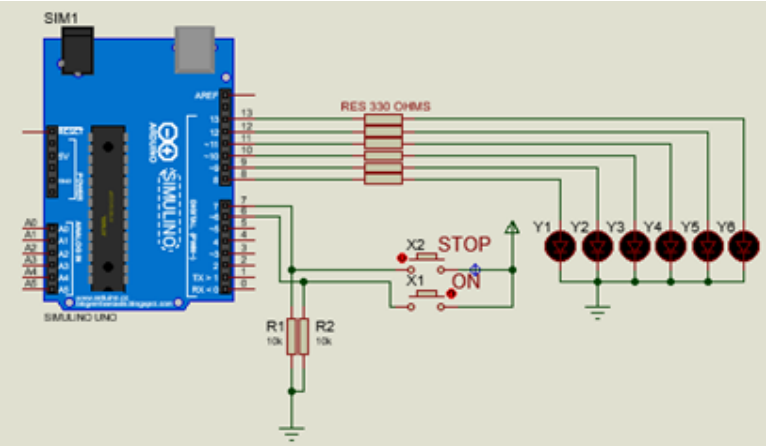
Tabla 9

Asignación de pines

ENTRADAS SALIDAS	COMPONENTES	FUNCIÓN
X1	STOP	Detiene la secuencia.
X2	ON	Inicia – continua la secuencia
Y1	SALIDA 1	LED1
Y2	SALIDA 2	LED2
Y3	SALIDA 3	LED3
Y4	SALIDA 4	LED4
Y5	SALIDA 5	LED5
Y6	SALIDA 6	LED6

Figura 91

Simulación de la secuencia de encendido y apagado de los LEDS usando el registro de desplazamiento



Ejercicios propuestos

1. Fijar el contador para que conectar una salida Y1 a los 10 pulsos, otra Y2 a los 15 pulsos y al llegar a los 20 pulsos apagarse las dos salidas y el contador poner en 0. Realizar el ejercicio para dos soluciones con contadores diferentes.

2. Realice el diagrama Ladder para obtener la secuencia indicada en la tabla. Disponer de pulsantes de marcha y de paro.

Tabla 10

Secuencia de activado y desactivado de las salidas

Y1	Y2	Y3	Y4	Y5	Y6
0	0	0	0	0	0
1	0	0	0	0	0
0	1	0	0	0	0
0	0	1	0	0	0
0	0	0	1	0	0
0	0	0	0	1	0
0	0	0	0	0	1
Repite la secuencia					

3. Realice el diagrama Ladder para obtener la secuencia indicada en la tabla. Disponer de pulsantes de marcha y de paro.

Tabla 11

Secuencia de activado y desactivado de las salidas

Y1	**Y2**	**Y3**	**Y4**	**Y5**	**Y6**
0	0	0	0	0	0
1	0	0	0	0	0
1	1	0	0	0	0
1	1	1	0	0	0
1	1	1	1	0	0
1	1	1	1	1	0
1	1	1	1	1	1
1	1	1	1	1	0
1	1	1	1	0	0
1	1	1	0	0	0
1	1	0	0	0	0
1	0	0	0	0	0
Repite la secuencia					

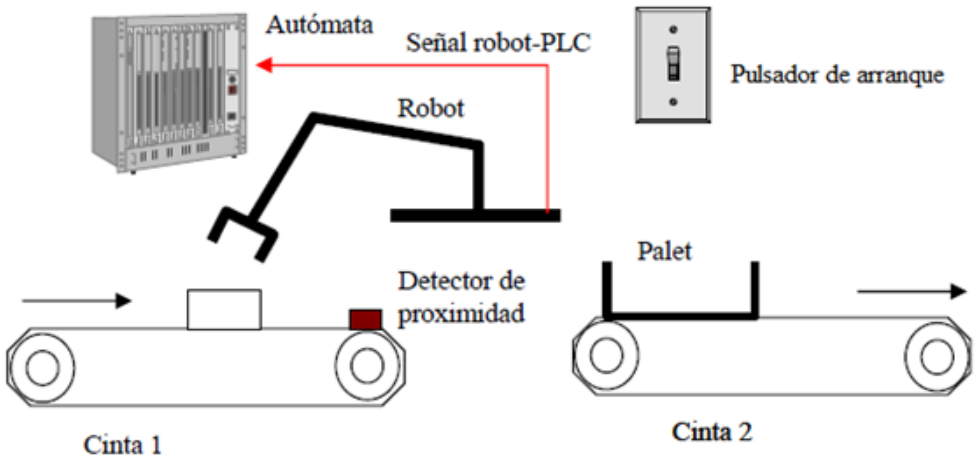
4. Se tiene un entorno de fabricación constituido por dos cintas transportadoras, un robot y un Palet según se representa en la figura. El autómata controla el sistema que evoluciona de la siguiente manera:

- Pulsando un botón de arranque se inicia el proceso.
- Por la cinta 1 llegan cajas. El detector de proximidad detecta una caja y el autómata para la cinta.

- El autómata activa el programa del robot.
- Cuando el robot ha colocado 4 cajas en el palet, el autómata activa la cinta 2 durante 5 s.
- Al final, todo vuelve a la posición de reposo.

Figura 92

Ejercicio robot – palet



https://acortar.link/vPXYhn

CAPÍTULO V

Conversor análogo digital y modulación de ancho de pulso

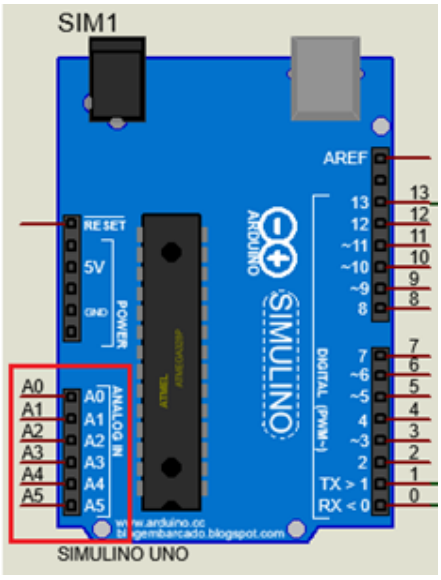
Entradas analógicas

Para los microcontroladores que disponen de conversores A/D, LDmicro genera el código necesario para manejar estos módulos de una forma eficiente y precisa. El A/D funciona cuando la condición de entrada de la instrucción es verdadera, entonces se obtiene una sola muestra del convertidor A/D y almacena en una variable "Name". Esta variable puede ser posteriormente manipulada con operaciones generales (menor que, mayor que, aritméticas, etc.) (Westhues, 2009).

Para el tratamiento de las señales analógicas Arduino en la mayoría de las tarjetas se dispone de varias entradas analógicas, y LDmicro dispone de las funciones para leer estos valores y el conversor análogo digital (ADC) del microcontrolador transforma a valores digitales. La resolución del ADC es de 10 bits.

Figura 93

Entradas analógicas de Arduino Uno

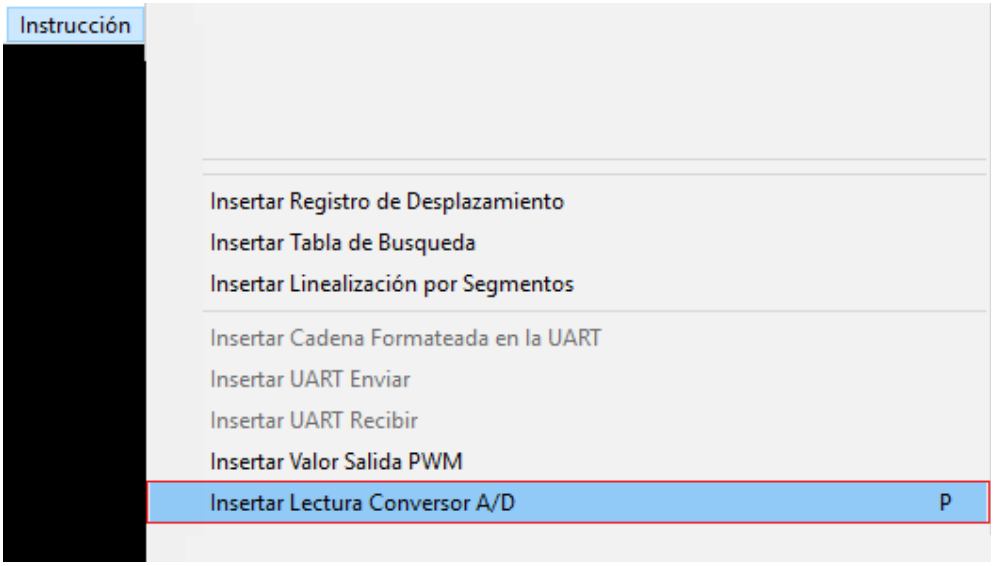


Lectura de la señal analógica

En el menú Instrucción, encontramos el comando *Insertar Lectura Conversor A/D*, para leer el valor analógico de una entrada.

Figura 94

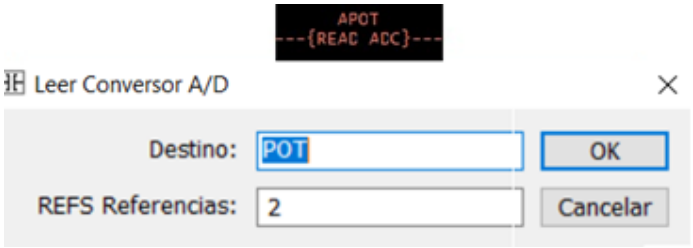
Comando Insertar Conversor A/D



Insertamos en el diagrama Ladder y definimos el nombre. Por defecto la letra A acompañará al nombre que asignemos. Por ejemplo, si damos el nombre de POT, en el Ladder tomará el nombre de la variable será APOT.

Figura 95

Símbolo del ADC y caja para designar el nombre



A partir de la versión 5 de LDmicro, se incluye el parámetro configurable REFS Referencias, que pertenece a los bits de referencia del del conversor. Con el valor de REFS= 0, indica la compatibilidad de LDmicro con versiones más bajas. REFS= 1, la referencia interna es de 2.51 y REFS= 5V, para el caso que el voltaje de alimentación del microcontrolador sea de 5V.

El lector interesado en conocer en detalle sobre el voltaje de referencia del microcontrolador AVR y como trabaja REFS, puede ingresar al link:

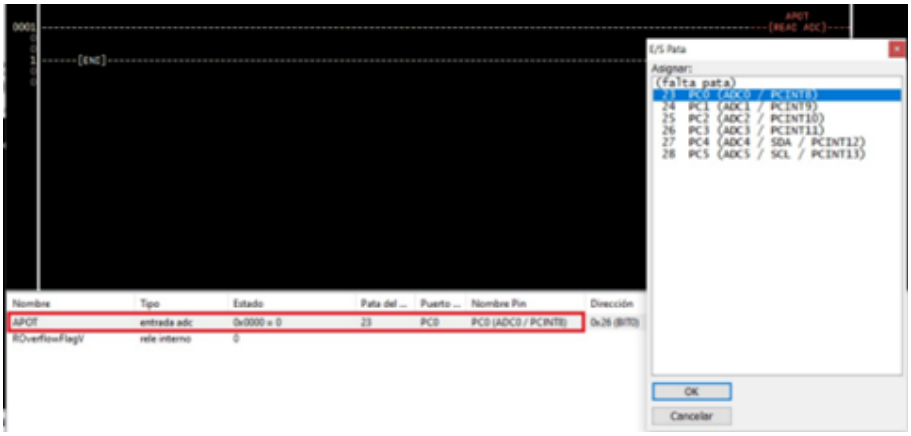
https://github.com/LDmicro/LDmicro/wiki/ADC-Voltage-Reference

Una vez insertado en el diagrama LDR podemos seleccionar el pin por donde se realizará la lectura. En la ventana de asignaciones al hacer doble clic so-

bre el nombre del pin que por defecto asigna al PCO se desplegara la ventana con las demás entradas habilitadas.

Figura 96

Pines de lectura para señales analógicas del ADC



Ejemplo: Leer la entrada de una señal analógica que varía de 0 a 5V, mediante la activación de LEDS verificar el estado de la entrada de la siguiente forma:

- Cuando la señal sea menor o igual al 25% del valor de entrada prender un LED ROJO.
- Prender un LED AMARILLO cuando la señal de entrada este entre el 25% y el 50%.
- Prender un LED VERDE cuando la señal de entrada sea mayor al 50% hasta al 100%.
- Prender un LED BLANCO cuando la señal llegue al máximo valor.

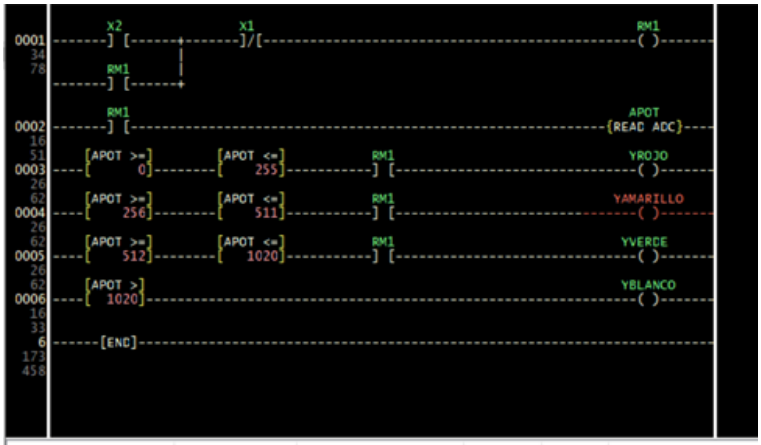
En las cajas de condición colocar los valores indicados en el diagrama de la Figura 97 y en la fijar REFS en 2, en la caja del APOT.

Como se indicó el conversor tiene una resolución de 10 bits, es decir que el rango de conversión estará entre 0 y 1023. Por tanto, cuando la señal de entrada sea el 25%, el conversor será aproximadamente 256, para el 50% el conversor tendrá un valor de 512 y para el 100% será 1023.

Se incluyen entradas digitales X1 y X2 sirven para desactivar y activar el circuito.

Figura 97

Diagrama Ladder para leer una señal analógica

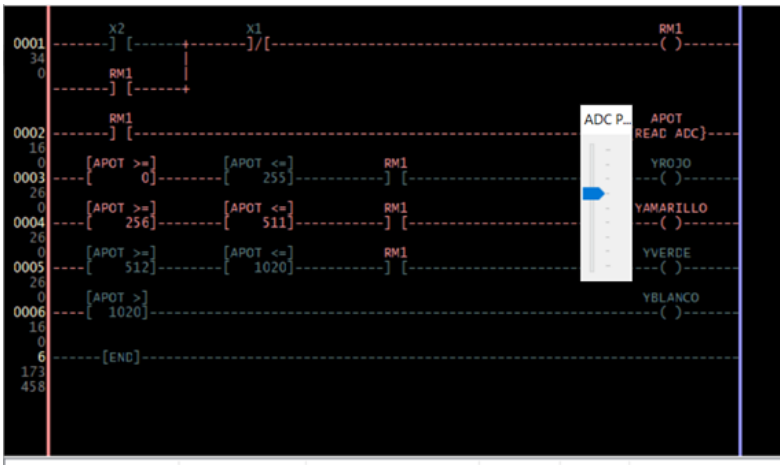


Nombre	Tipo	Estado	Pata del ...	Puerto ...	Nombre Pin
X1	entrada digital	0	13	PD7	PD7 (PCINT23 / AIN1)
X2	entrada digital	0	12	PD6	PD6 (PCINT22 / OC0A ...
YAMARILLO	salida digital	0	18	PB4	PB4 (MISO / PCINT4)
YBLANCO	salida digital	0	16	PB2	PB2 (SS / OC1B / PCIN...
YROJO	salida digital	0	19	PB5	PB5 (SCK / PCINT5)
YVERDE	salida digital	0	17	PB3	PB3 (MOSI / OC2A / P...
APOT	entrada adc	0x0000 = 0	23	PC0	PC0 (ADC0 / PCINT8)
RM1	rele interno	0			
ROverflowFlagV	rele interno	0			

Al simular en el diagrama LDmicro y visualizar el potenciómetro se debe dar doble clic en la caja del AD. Al mover el potenciómetro podemos ver los valores de las variables y la activación de las salidas.

Figura 98

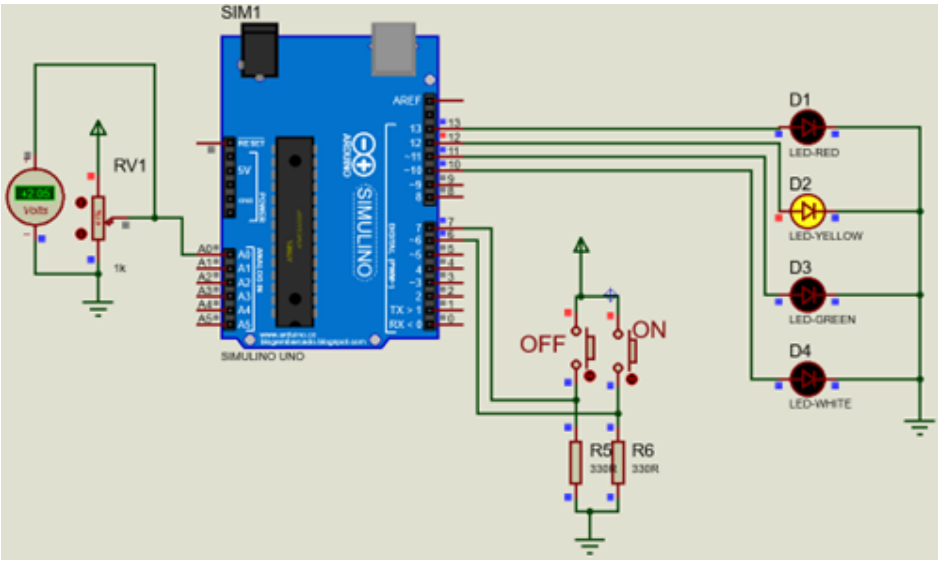
Simulación en LDmicro del diagrama Ladder para la visualización del potenciómetro para el manejo de la señal de entrada



Nombre	Tipo	Estado	Pata del ...	Puerto ...	Nombre Pin
X1	entrada digital	0	13	PD7	PD7 (PCINT23 / AIN1)
X2	entrada digital	0	12	PD6	PD6 (PCINT22 / OC0A ..
YAMARILLO	salida digital	1	18	PB4	PB4 (MISO / PCINT4)
YBLANCO	salida digital	0	16	PB2	PB2 (SS / OC1B / PCIN..
YROJO	salida digital	0	19	PB5	PB5 (SCK / PCINT5)
YVERDE	salida digital	0	17	PB3	PB3 (MOSI / OC2A / P...
APOT	entrada adc	0x0181 = 385	23	PC0	PC0 (ADC0 / PCINT8)
RM1	rele interno	1			
ROverflowFlagV	rele interno	0			

Figura 99

Simulación en Proteus del módulo AD



Generación de señal PWM

LDmicro puede generar código para obtener señales PWM. Los módulos PWM vienen incorporados en ciertos microcontroladores como es el caso del ATmega328P, que utiliza la tarjeta Arduino. Si la condición de entrada a esta instrucción es verdadera, entonces el ciclo de trabajo (duty cycle) de la señal PWM se establece en el valor de la variable duty_cycle. El duty cycle se define con valores comprendidos entre 0 y 100; 0 corresponde al valor más bajo, y 100 es el valor más alto (Westhues, 2009).

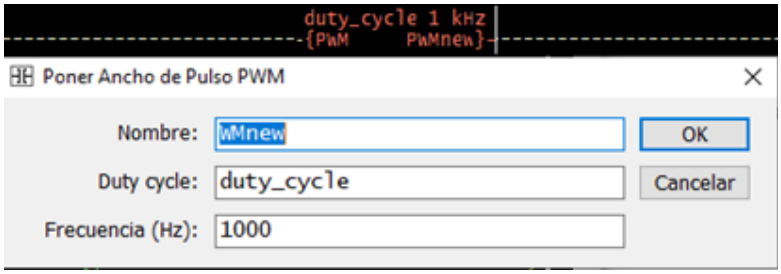
La frecuencia de la señal PWM se fija en Hz. Cuando se especifica una frecuencia que puede no ser generada, el programa LDmicro aproxima a una frecuencia la más cercana posible; si el error es grande, el programa entrega un mensaje de advertencia sobre este problema. Las velocidades más rápidas pueden sacrificar la resolución.

La selección de frecuencias, según indica el LDmicro, debe estar entre 61,275 HZ y 62,745KHZ con un porcentaje de error >5% (Westhues, 2009).

La instrucción Insertar Valor Salida PWM permite insertar en el diagrama Ladder la función PWM. La salida del PWM se obtiene por defecto puerto PB3 del microcontrolador correspondiente al pin11 de la tarjeta Arduino 1.

Figura 100

Definición de los parámetros del PWM

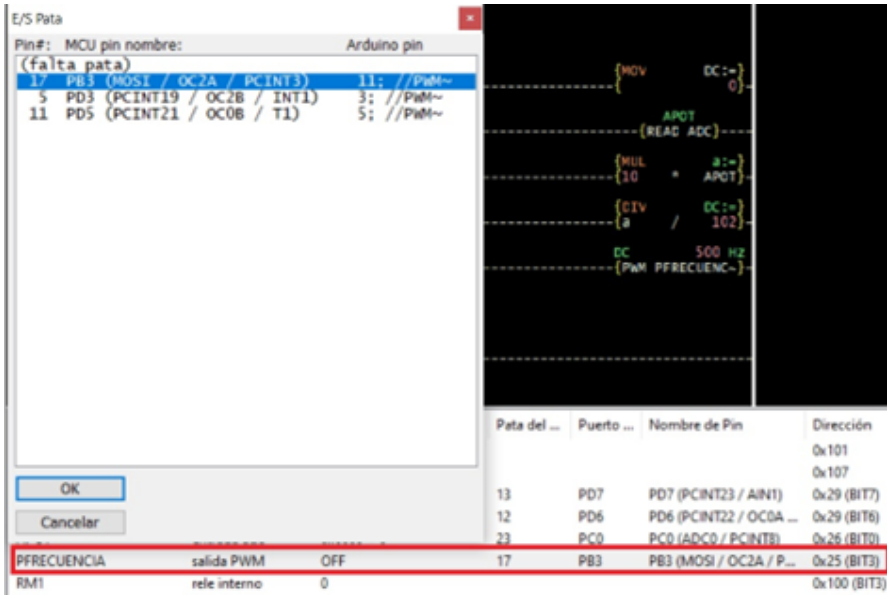


A continuación, se debe definir el nombre, el ciclo de trabajo y la frecuencia.

Si deseamos cambiar el pin de salida del PWM, en la ventana de asignaciones podemos de pines podemos seleccionar un pin diferente al 11. Está disponible el pin 3 y 5.

Figura 101

Pines para la señal PWM



En la versión 5, LDmicro incluye el valor de la Resolución seleccionable en valores de:

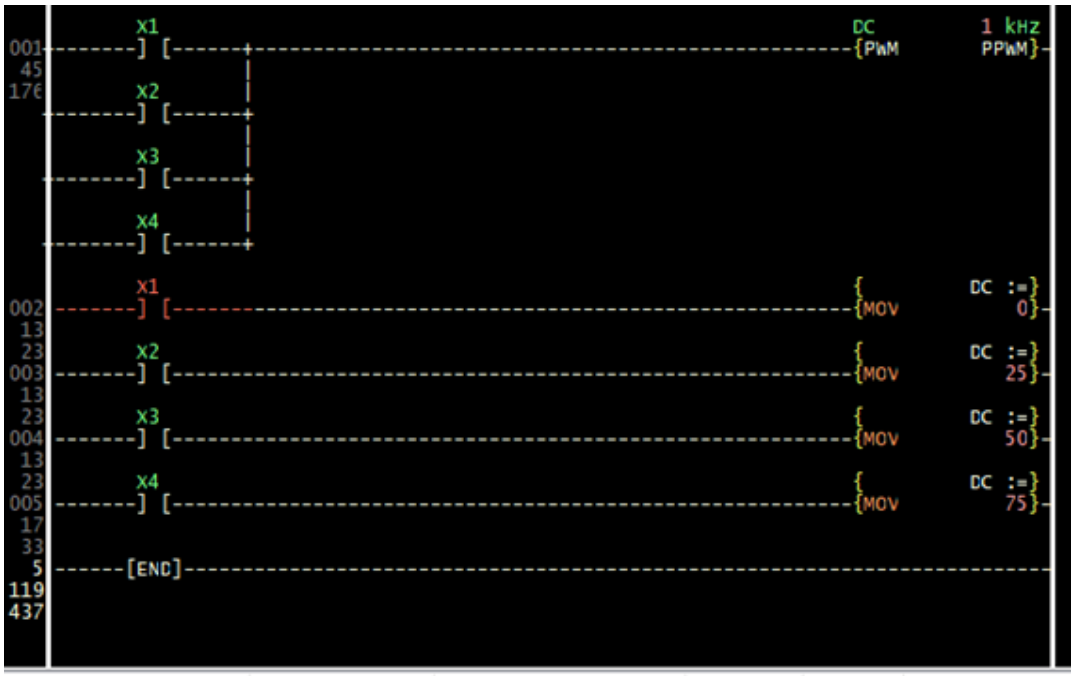
- 6 o 7 bits, el ancho de pulso corresponde de 0 al 100%.
- 8 bits, el ancho de pulso corresponde de 0 al 256.
- 9 bits, el ancho de pulso corresponde de 0 a 512.
- 10 bits, el ancho de pulso corresponde de 0 al 1024.

El valor que se seleccione dependerá si el microcontrolador soporta o tiene pines para resoluciones altas. En caso de Arduino que usa el AVR328P, podemos utilizar las opciones 6 o 7 y 8 bits.

Ejemplo: En el siguiente ejercicio vamos a generar una señal PWM de 1 KHz, para con DC= 25%, DC= 50% y DC= 75%.

Figura 102

Diagrama Ladder para generar una señal PWM de 1KHZ con diferentes DC



Nombre	Tipo	Estado	Pata del ...	Puerto ...	pin Name
DC	var general	0x0000 = 0			
X1	entrada digital	0	13	PD7	
X2	entrada digital	0	12	PD6	
X3	entrada digital	0	11	PD5	
X4	entrada digital	0	6	PD4	
PPWM	salida PWM	0x0000 = 0	17	PB3	

La Figura 103 de la página siguiente muestra el diagrama utilizado para la simulación. Es importante tener el osciloscopio para verificar las señales que se generan en el PWM.

Figura 103

Circuito de simulación para generar la señal PWM

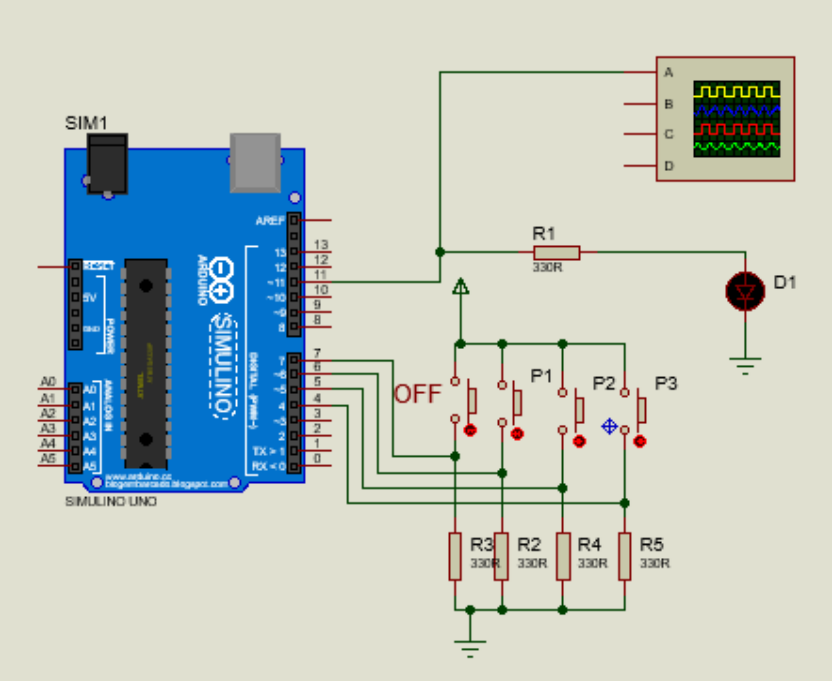


Tabla 12

Asignación de pines

ENTRADAS SALIDAS	COMPONENTES	FUNCIÓN
X1	OFF	PWM= 0
X2	P1	PMW= 25%
X3	P2	PWM= 50%
X4	P3	PWM= 75%
Pin 11	Salida	Salida del PWM

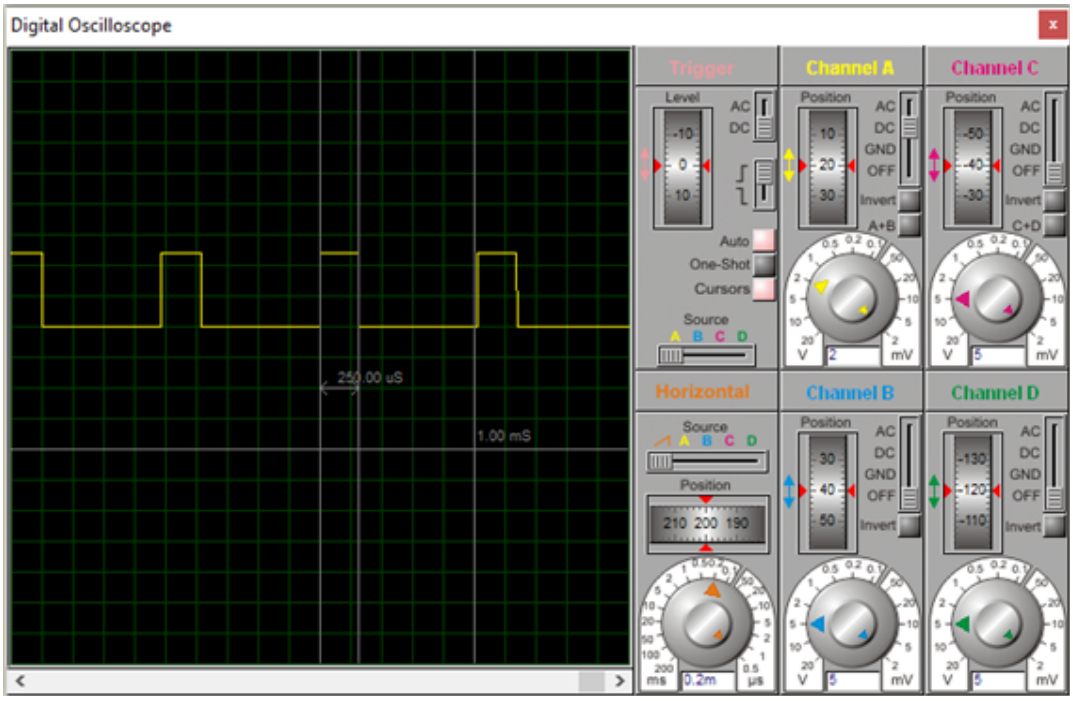
Funcionamiento:

Todas las entradas deben activar el módulo PWM y están definidas para generar 1HZ. La salida del PWM es en el PIN PB3 del microcontrolador y este en la tarjeta Arduino es el Pin 11. Al accionar las entradas X1, X2, X3 y X4 actuarán de la siguiente forma:

- Al accionar X1, la señal del PWM es "0".
- Al accionar X2, se genera una señal PWM= 1KHZ y factor de trabajo DC= 25%. Es decir, 250 us en alto y 750 us en bajo, dando un periodo total de 1ms, que corresponde a la frecuencia de 1KHZ.

Figura 104

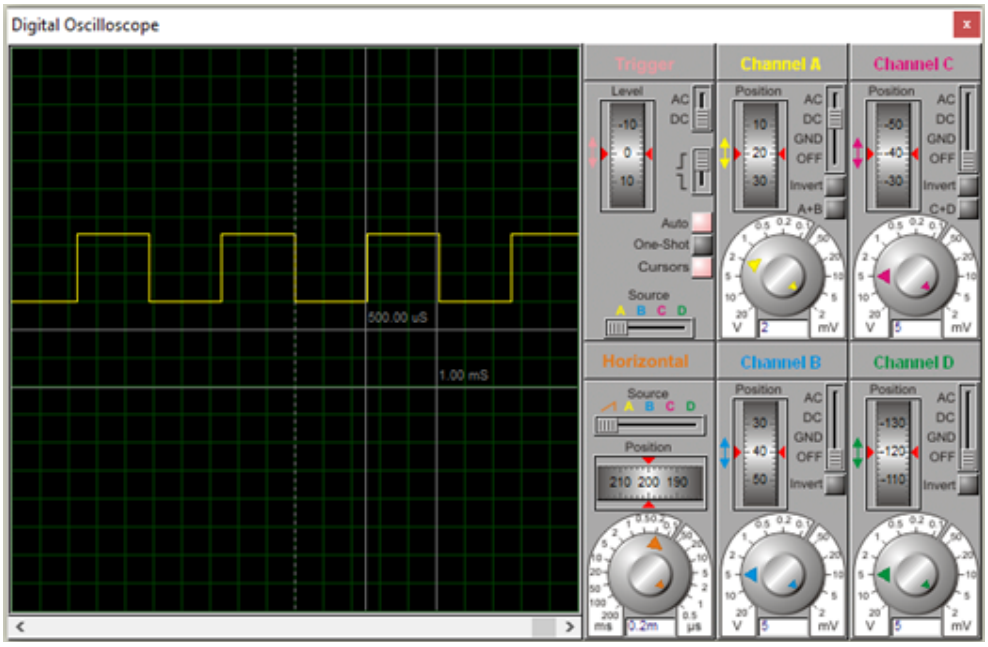
Señal del PWM con DC=25%



- Con X3, se genera un PWM de 1KHZ, y DC= 50%, o sea, una onda cuadrada.

Figura 105

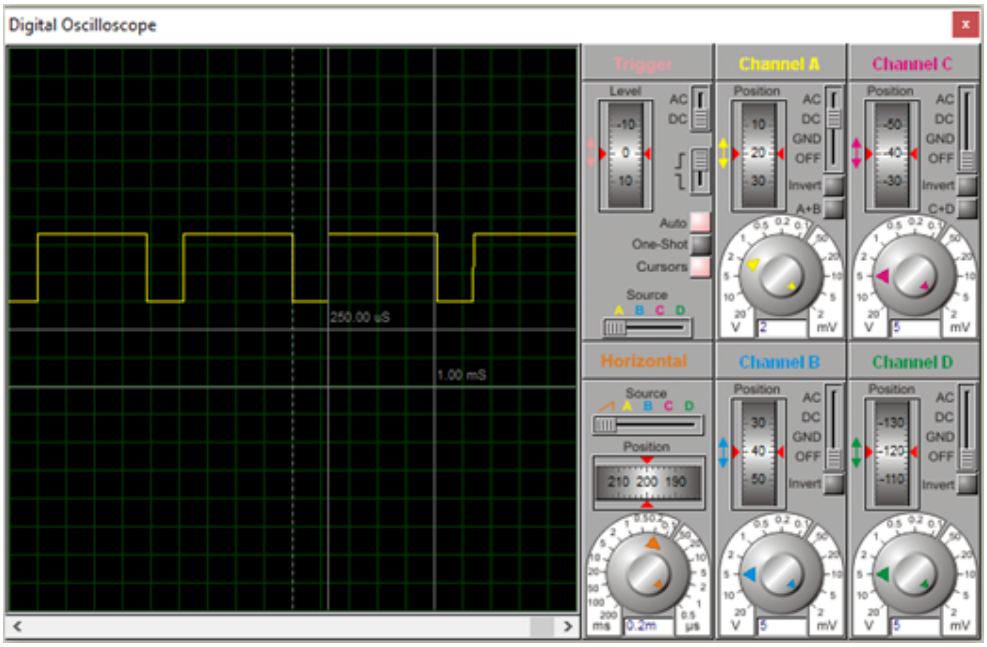
Señal del PWM con DC=50%



- Con X4, se genera un PWM de 1KHZ, y DC= 75%, o sea, una onda de 750 us, en alto y 250 us. en bajo.

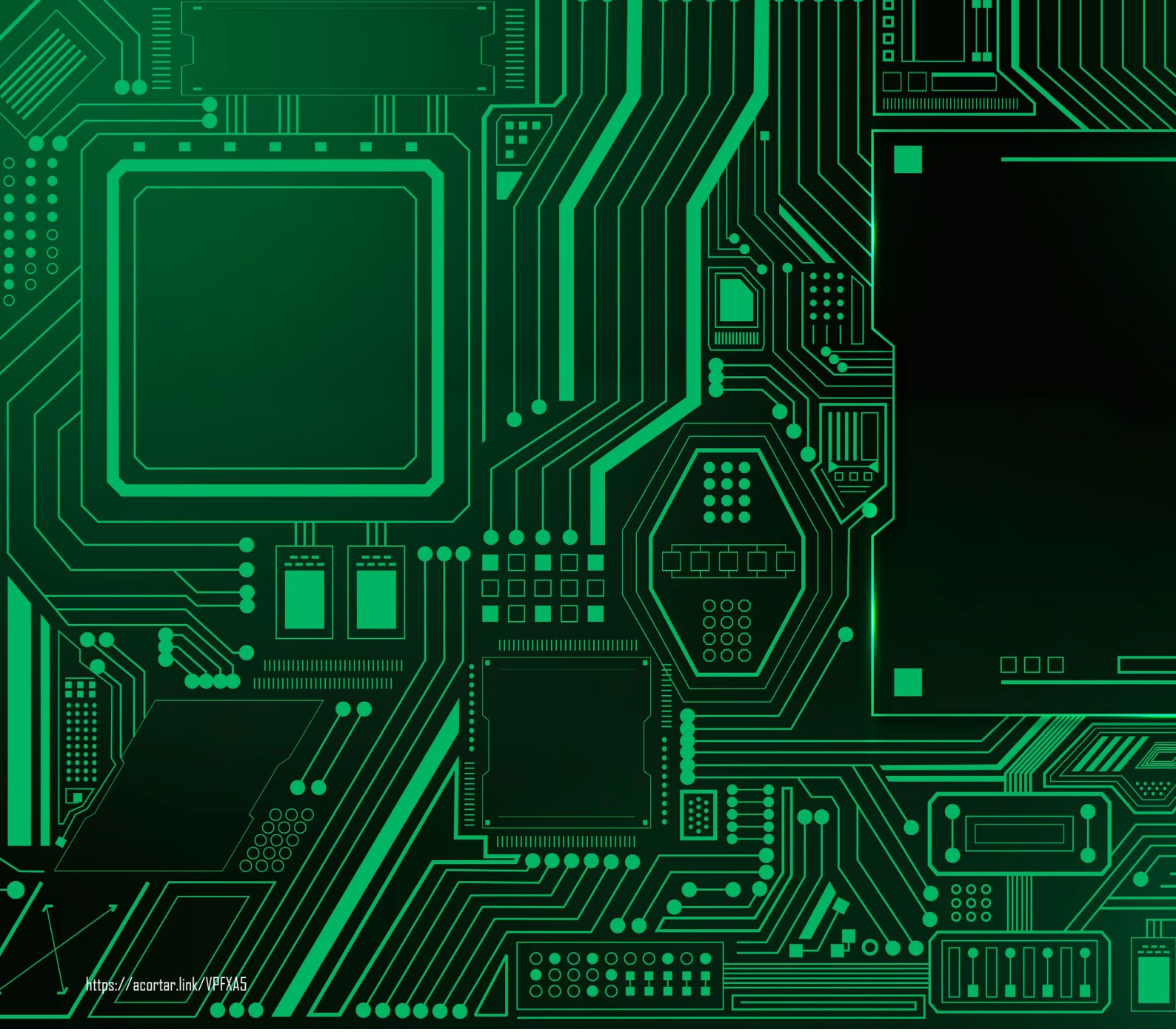
Figura 106

Señal del PWM con DC=75%



Ejercicios propuestos

1. Realizar un programa para leer dos señales de entrada simultáneamente. Mediante LEDS indicadores señalar las lecturas media y máxima de cada señal.
2. Realizar el circuito para generar una onda de 2KHZ con un DC= 75%, con una resolución 8 bits. Simular el circuito y verificar la onda mediante el osciloscopio virtual.



CAPÍTULO VI

Aplicaciones

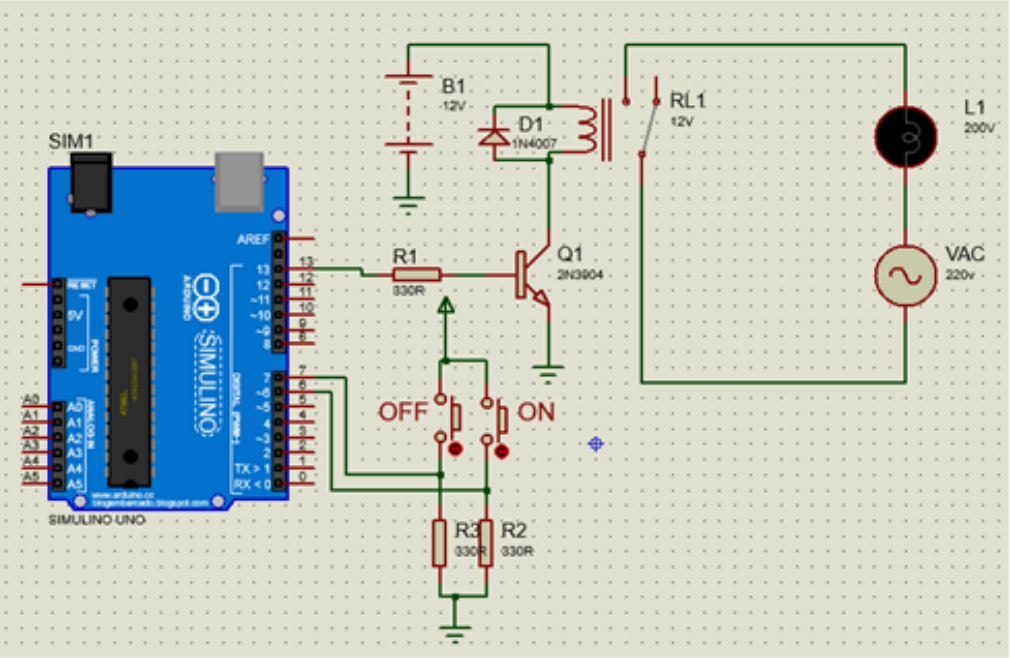
Una de las aplicaciones más comunes de los PLC es el control de motores eléctricos de CA. En este capítulo abordaremos los circuitos de control industrial de uso más frecuente en la industria.

Circuito de interfase para salida con relé

La tarjeta Arduino entrega en sus salidas digitales niveles lógicos de 0 y 5 V, por lo que estos valores no pueden manejar relés o contactores de CC o CA de tensiones superiores a 120V. Por esto es necesario realizar circuitos de interfase que permitan manejar este tipo de cargas.

Figura 107

Interfase para salida por relé



Cuando la salida del Arduino sea alta satura al transistor el mismo que actúa como interruptor cerrado activando al relé. Un contacto del relé puede utilizarse para activar cualquier tipo de carga. Cuando el estado de la salida del Arduino sea bajo, el transistor actuará como interruptor abierto por lo que el relé estará apagado. La capacidad de los transistores dependerá de la corriente de carga que se maneje. Los PLC que tienen salidas por contacto de relé utilizan esta interfase.

Simulación en CADe SIMU con la tarjeta Arduino

El programa CADe-SIMU y la tarjeta Arduino se puede trabajar de manera offline (desconectado la tarjeta Arduino) y online (conectada la tarjeta Arduino).

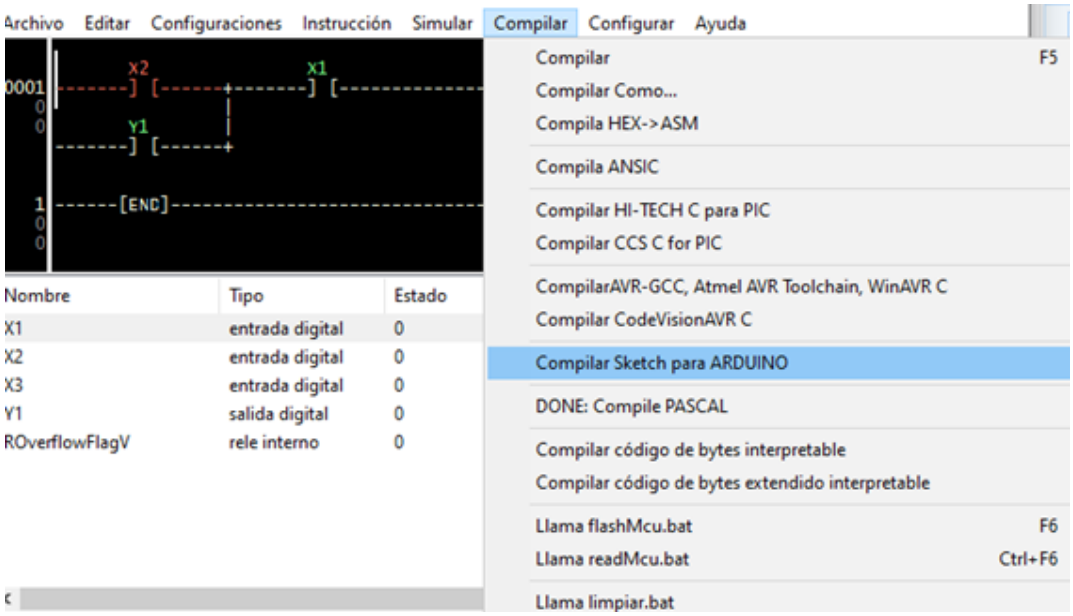
En forma offline en CADe-SIMU podemos crear los esquemas de control eléctrico, de potencia y el diagrama Ladder. En este caso solo realizaríamos la simulación del circuito, no necesita estar conectado la tarjeta Arduino, ni el programa de LDmicro. Si trabajamos de esta forma lo que necesitamos es dibujar todos los circuitos en CADe-SIMU y en el diagrama Ladder identificar las entradas y salidas con los pines que utiliza la tarjeta Arduino en CADe-SIMU como se explicó en el Capítulo 1, sección "Uso de CADe-SIMU y PC-SIMU".

Para interactuar con el programa CADe SIMU con la tarjeta Arduino, o sea para trabajar en forma online, se necesita realizar los siguientes pasos:

1. Crear el "Sketch" para el IDE de Arduino. Para esto procedemos a ir al programa LDmicro en la pestaña Compilar/Compilar Sketch para Arduino.

Figura 108

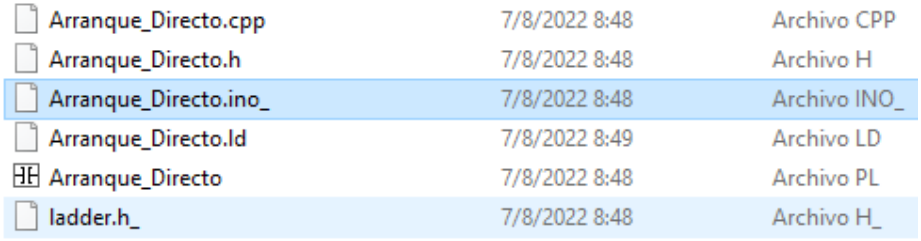
Obtención del Sketch para el IDE de Arduino



Se debe tener la precaución de guardar en una carpeta que tenga el mismo nombre del archivo. Con esto se generan varios archivos con diferentes extensiones.

Figura 109

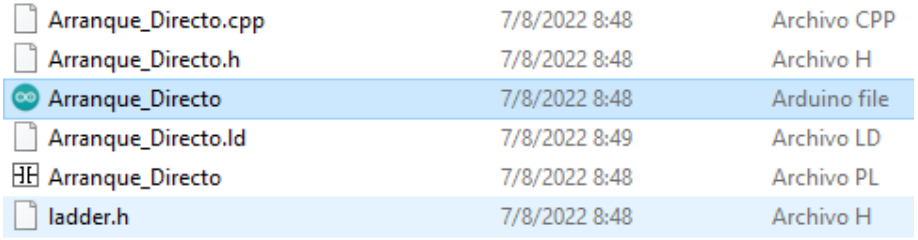
Archivos generados para el Sketch de Arduino



Nos interesa el archivo con extensión **.ino_** y el **ladder.h_**. Estos dos archivos, para poder utilizarlos debemos quitar el _ (guion bajo), que aparece en las extensiones de los archivos. Con esto el archivo **.ino** ya podemos abrirlo en el IDE de Arduino.

Figura 110

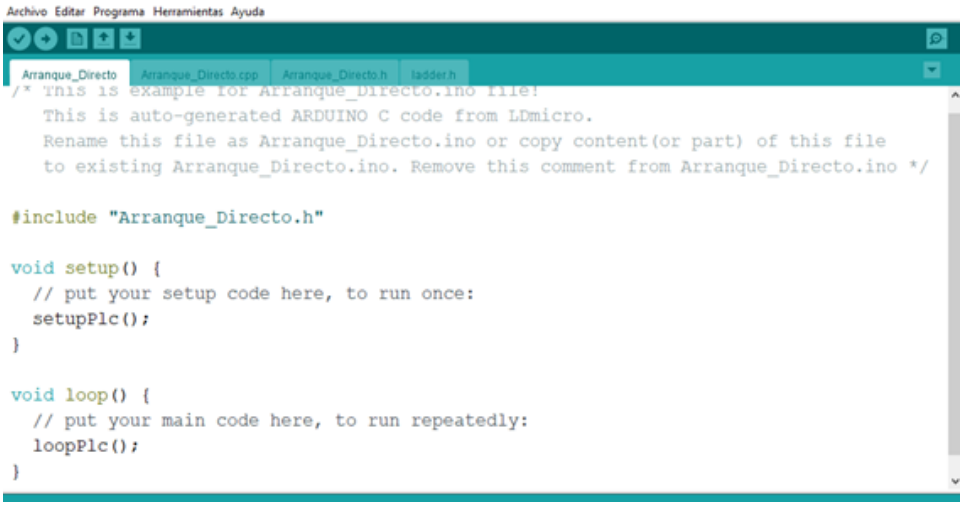
Archivos modificados para el Sketch de Arduino



2. Verificar y agregar el código de PC-SIMU en el Sketch de Arduino. Abrimos el programa generado utilizando el ID de Arduino (EJEMPLO: Arranque_Directo.ino).

Figura 111

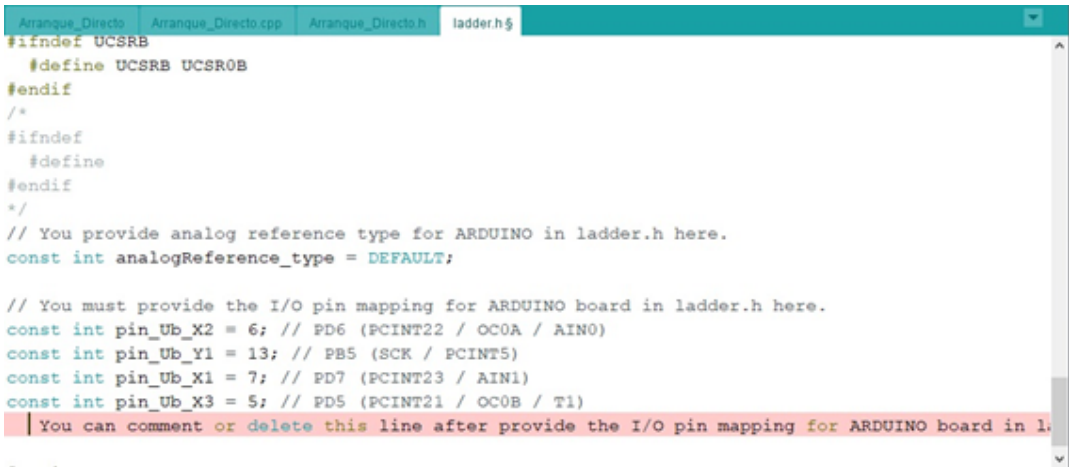
Programas del Sketch de Arduino



Verificamos el código generado y nos da un error. Esta línea simplemente podemos eliminarla o dejarla como comentario. Con esto el código está preparado para ser utilizado.

Figura 112

Error en el Sketch de Arduino



Los errores que aparecen en el código, dependen de la versión del LDmicro que se utilice. Para eliminarlos tememos que verificar y dar las acciones solicitadas en el programa. En este ejemplo estamos utilizando la versión 4.3.8.3. En la versión 5.4.1.1, el error hace referencia a que falta una librería. Estas librerías son necesarias cuando se va realizar algún tipo de modificación u otras funciones que son propias de esta versión. La librería indicada podemos eliminarla del programa, si no es necesaria, o incluirla. Para esto, en la carpeta dónde están los archivos de LDmicro, viene adjunta la carpeta "LIBRARIES_FOR", buscamos la librería que nos solicita, copiamos y pegamos en la carpeta donde guardamos en sketch.

A continuación, agregamos las líneas de código al inicio del programa:

#include <PC_SIMU.h>

PC_SIMU.INI

Y dentro de la función setup(), incluimos

PLC.INI();

Con esto el programa está listo para ser transferir a la tarjeta e interactuar con el programa CADe-SIMU.

Figura 113

Código agregado en el programa principal del Sketch, para vincular Arduino con CA-De-SIMU

```
#include  "Arranque_Directo.
#include  <PC_SIMU .h>
PC_SIMU  PLC;

void  setup () {
  // put your setup code here, to run once:
  setupPlc ();
  PLC.INI ();
}

void  loop () {
  // put your main code here, to run repeatedly:
  loopPlc ();
}
```

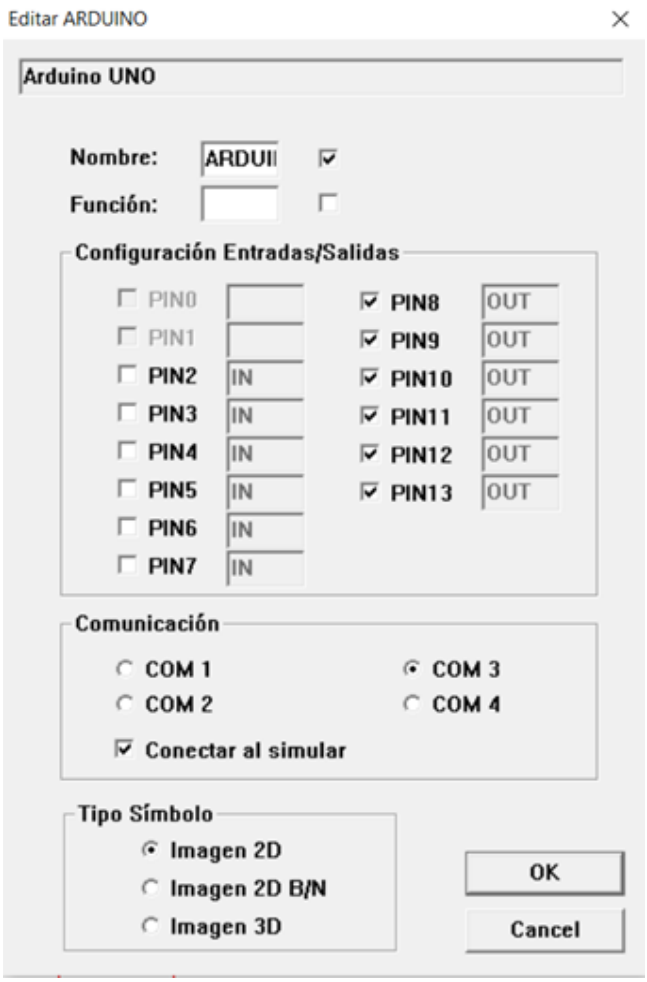
3. Conectar la Tarjeta Arduino UNO al computador y transferir el programa. Conectamos la tarjeta Arduino Uno a uno de los puertos del USB del computador. Se genera un puerto serial virtual COMxx. Desde IDE de Arduino, seleccionar el respectivo puerto y transferir el programa a la tarjeta.
4. Crear el esquema de control en CADe-SIMU. En CADe-SIMU, hacer el esquema de control con Arduino y el circuito de potencia para interactuar en tiempo real con la tarjeta Arduino. Los pulsadores que se conecten en las entradas de la tarjeta Arduino con los pulsadores que realizados en el esquema en CADe-SIMU, el programa asume como contactos en paralelo. Por tanto, hay que tomar en cuenta esta situación, especialmente cuando se trabaje con contactos normalmente cerrados, ya al accionarlos no permiten desactivar el circuito desde el programa.
5. Configurar Arduino en CADe-SIMU. Antes de interactuar con la tarjeta Arduino desde CADe-SIMU, la tarjeta debemos configurarle, indicando los pines de entrada y salida, seleccionando las casillas según el caso, indicar el puerto de comunicación que esta utilizando y seleccionar la casilla “conectar al simular”. Para acceder al cuadro de dialogo indicado, basta con hacer “doble clic” con el mouse en el simbolo de la tarjeta Arduino de CADe-SIMU.

Se debe tener en cuenta que cuando se establece la comunicación entre CA-De-SIMU y Arduino, el puerto queda utilizado por el programa y no se puede

efectuar ninguna modificación desde el IDE de Arduino. Se debe cerrar el programa CADe-SIMU para que quede libre el puerto.

Figura 114

Configuración de Arduino en CADe-SIMU



Arranque directo de un motor trifásico

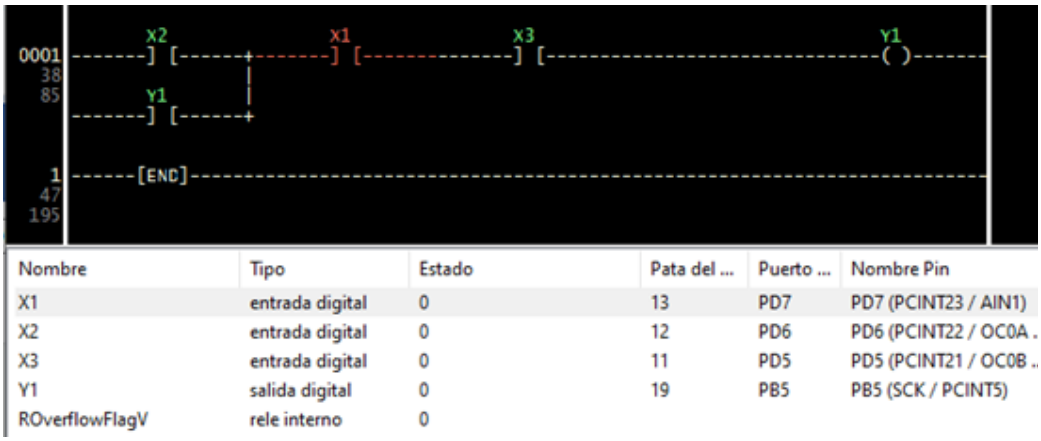
Para el arranque directo de cualquier tipo de motor, es un simple circuito de enclavamiento. En el diagrama Ladder el pulsador conectado a entrada digital del pin 3, al ser accionado activa la salida Y1, asociada con el pin 19. Esta salida manejará un contactor de tres polos para conectar las tres fases de corriente alterna al motor.

Se ha incluido una entrada digital, X3, para conectar un contacto de un relé térmico. Considerando que el contacto de protección del térmico es normalmente cerrado, para el Ladder debemos ponerlo con un contacto abierto, con

lo cual, actuará como cerrado cuando esté funcionando normalmente el relé térmico y se abrirá ante una sobrecarga en el motor. En la entrada digital X1 se conectará un pulsante de paro que es normalmente cerrado y en la entrada X2, un pulsador normalmente abierto para poner en marcha al motor mediante la salida Y1, la cual manejará a un relé y este a su vez al contactor de línea.

Figura 115

Diagrama Ladder para arranque directo de un motor AC



Nombre	Tipo	Estado	Pata del ...	Puerto ...	Nombre Pin
X1	entrada digital	0	13	PD7	PD7 (PCINT23 / AIN1)
X2	entrada digital	0	12	PD6	PD6 (PCINT22 / OC0A .
X3	entrada digital	0	11	PD5	PD5 (PCINT21 / OC0B ..
Y1	salida digital	0	19	PB5	PB5 (SCK / PCINT5)
ROverflowFlagV	rele interno	0			

Tabla 13

Asignación de pines

ENTRADAS SALIDAS	PINES ARDUINO	COMPONENTES	FUNCIÓN
X1	7	OFF	APAGAR
X2	6	ON	ARRANQUE
X3	5	TÉRMICO	PROTECCIÓN
Y1	13	MOTOR	CARGA TRIFÁSICA

Cuando se trabaja con circuitos de control industrial, los esquemas básicos son el circuito de control y el circuito de fuerza. En circuitos electrónicos basados con componentes programables, el programa y la tarjeta Arduino constituyen el diagrama de control, pero como este dispositivo maneja voltajes y corrientes bajas es necesario tener el circuito de interfase, para manejar a los dispositivos de maniobra con las señales que provienen del circuito de control.

La Figura 116 de la página siguiente muestra los circuitos realizados en Proteus para la simulación del circuito realizado en el diagrama Ladder del programa LDmicro.

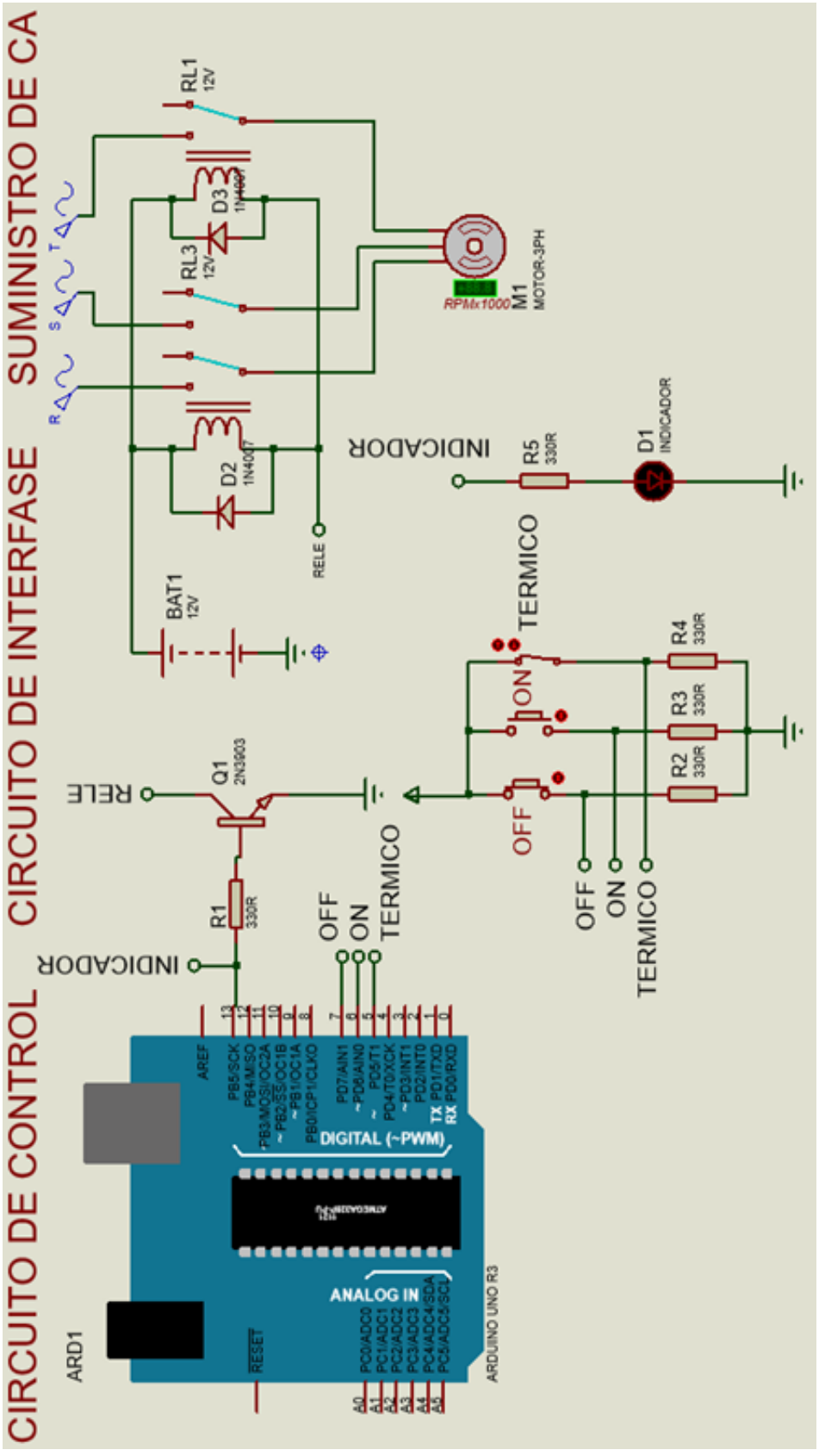


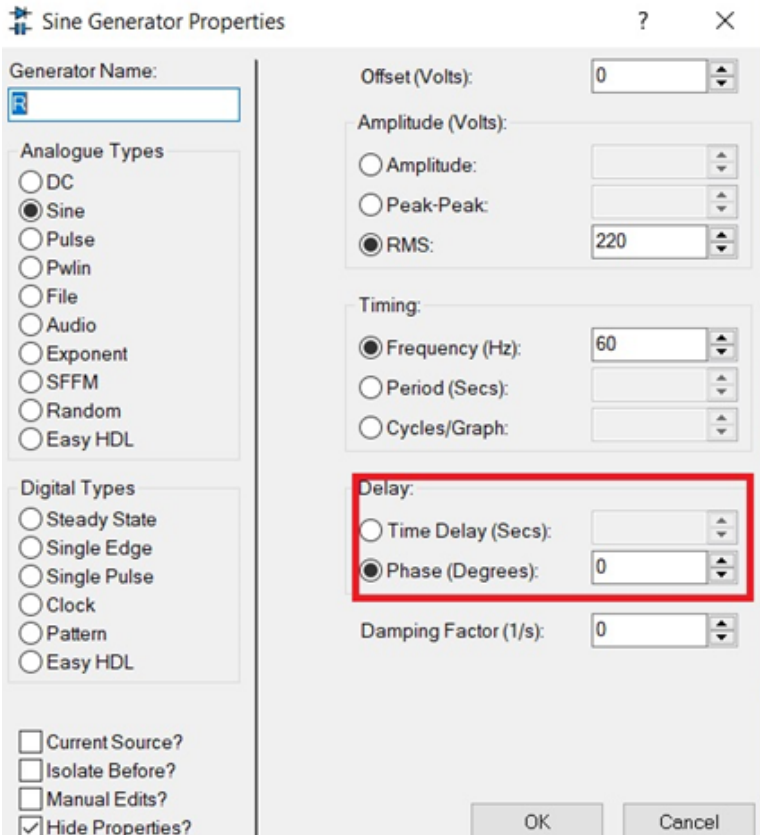
Figura 116
Circuito en Proteus para la simulación de arranque directo de un motor AC.

Consideraciones para la simulación

- La fuente de CA trifásica obtenemos de tres fuentes monofásicas, modificando en las propiedades el ángulo de fase. Así en la fase R, el ángulo de fase 0o; para S, 180º y la fase T, 240º. El voltaje RMS entre líneas es 220V.

Figura 117

Fijar el ángulo de fase para obtener una fuente trifásica

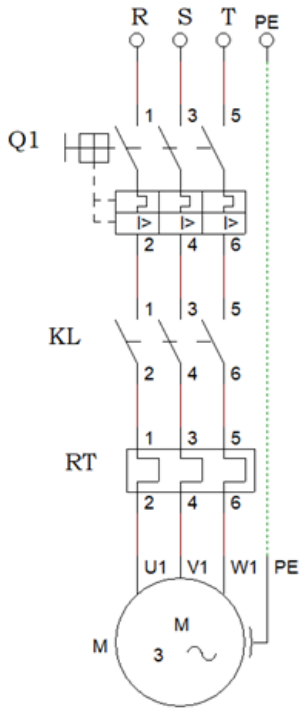


- Proteus no dispone de un relé de 3 polos, por lo que se utiliza uno de 2 polos y otro de 1 polo que funcionan simultáneamente para conectar la fuente trifásica que alimenta al motor.

El circuito de fuerza práctico, estaría formado por un disyuntor trifásico Q1, para conectar–desconectar manualmente las fases del motor. El contactor de línea KL para la maniobra del motor un relé térmico trifásico como protección al sobrecalentamiento de las bobinas del motor. Un contacto de este relé servirá para desactivar el circuito de control cuando exista un sobrecalentamiento en las bobinas del motor.

Figura 118

Diagrama de fuerza para arranque de motor trifásico



Simulación en CADe-SIMU

Los problemas de simulación con Proteus, especialmente en la etapa de potencia, se solventan mediante el uso de CADe-SIMU, el cual provee todas las herramientas necesarias para realizar este trabajo.

La figura 119 muestra los diagramas realizados en el programa CADe-SIMU. El diagrama la Ladder es el mismo del LDmicro, con la excepción que las entradas y salidas han sido establecidos de acuerdo a las asignaciones de CADe-SIMU.

Primero realizaremos la simulación en CADe-SIMU de forma offline.

Tabla 14

Asignación de pines

ENTRADAS SALIDAS	PINES ARDUINO	COMPONENTES	FUNCIÓN
I0.7	7	S1	APAGAR
I0.6	6	S2	ARRANQUE
I0.5	5	RT1	PROTECCIÓN
Q1.5	13	RELE	CARGA TRIFÁSICO

En la Figura 120 se muestra, el resultado de la simulación para el arranque directo del motor trifásico. Los resultados obtenidos muestran claramente el funcionamiento correcto de todos los circuitos.

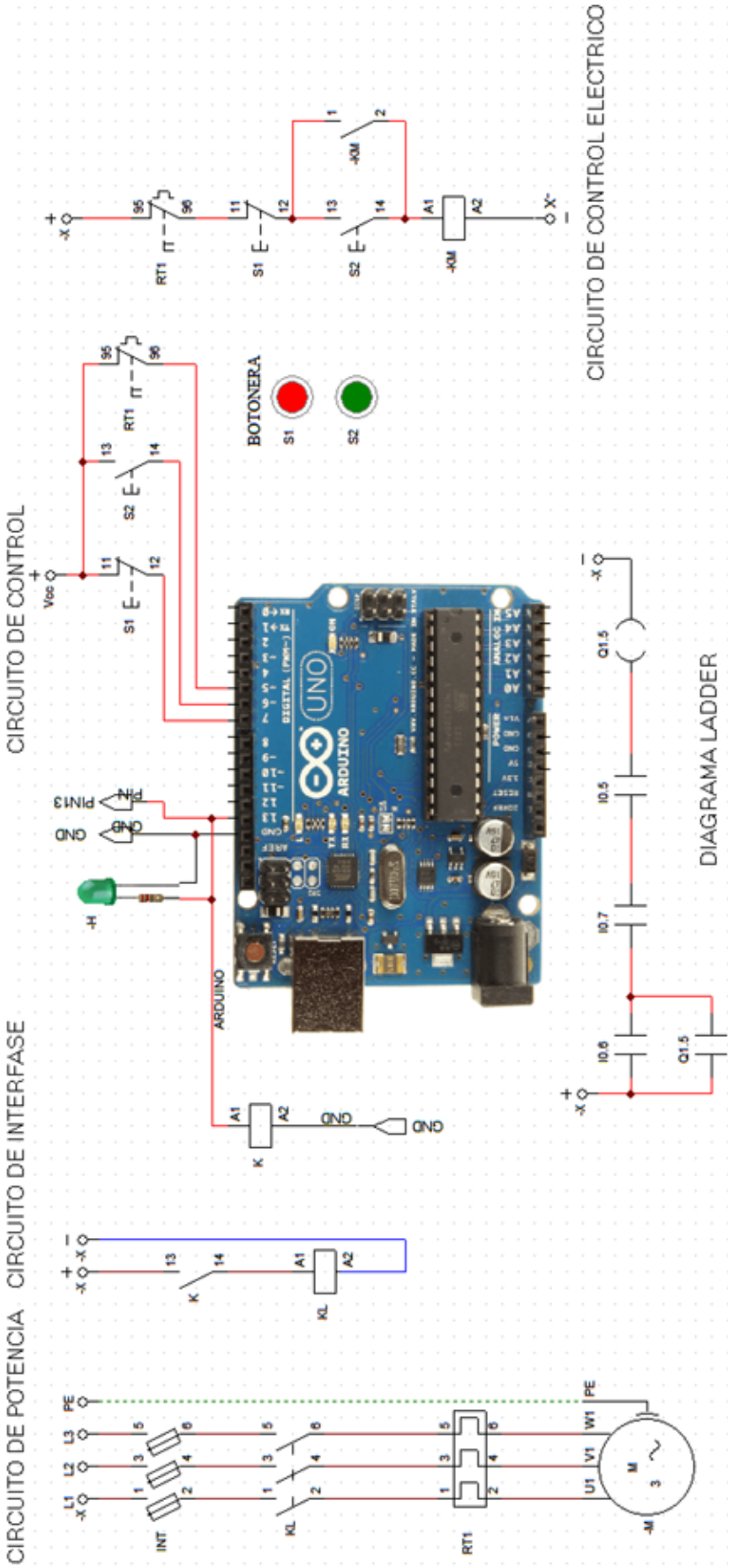


Figura 119

Esquemas de simulación en CADe-SIMU, sin conexión con la tarjeta Arduino. Arranque directo de motor trifásico

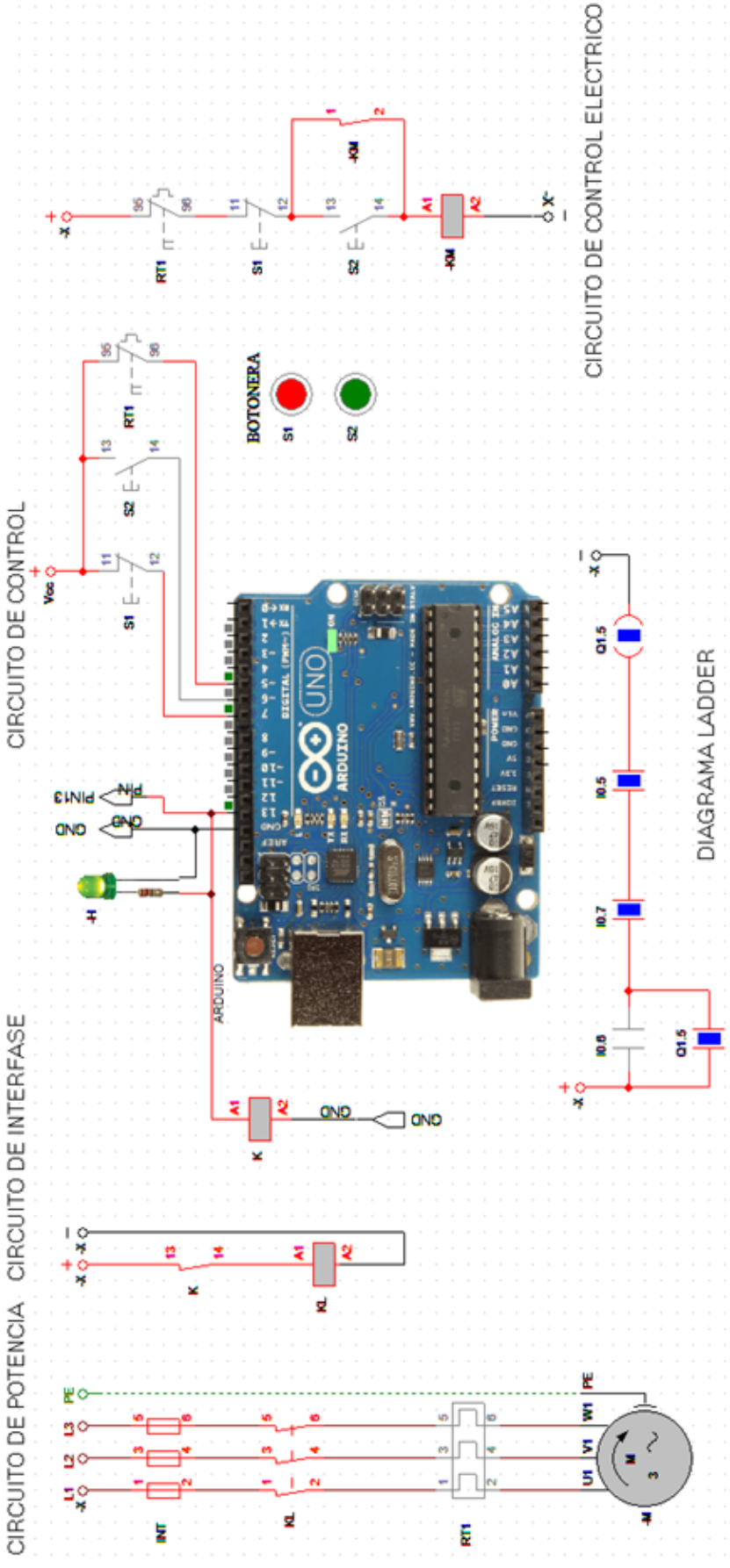


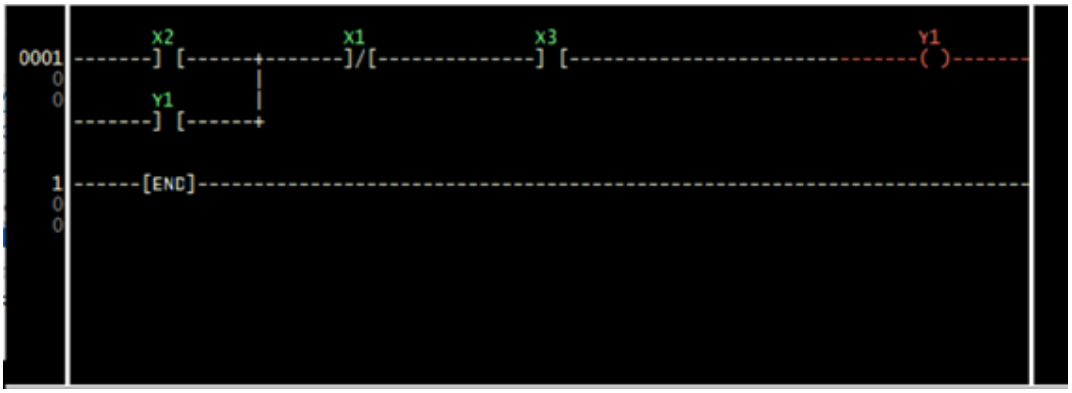
Figura 120

Resultados de la simulación en CADe-SIMU, sin conexión con la tarjeta Arduino. Arranque directo de motor trifásico

Para la simulación online, recordamos que los contactos de las entradas en el programa CADe-SIMU con los de la tarjeta Arduino actúan como conexión en paralelo. Tomando esta situación, en el diagrama Ladder de LDmicro modificamos la entrada del pulsador de apagado como un contacto normalmente cerrado, con esto en el Arduino y en CADe-SIMU colocamos pulsadores normalmente abiertos.

Figura 121

Diagrama Ladder para el arranque directo de un motor trifásico



Seguimos todos los pasos descritos, para obtener el Sketch del IDE de Arduino y transferir a la tarjeta Arduino. Aquí debemos conectar todos los pulsadores y elementos utilizados en forma real.

A continuación, dibujamos todos los esquemas necesarios para la simulación en CADe-SIMU. Para el presente ejemplo, hemos realizado los siguientes circuitos:

Circuito de potencia. El esquema de potencia, el mismo que consta de un interruptor magnético INT puede ser automático o manual. El contactor KL, para manejar la carga trifásica, el relé térmico RT1 y el motor trifásico de inducción.

Circuito de interfase. Dado que la tarjeta Arduino en sus salidas proporcionan niveles lógicos de 0 o 5V, lo máximo que puede manejar es un relé de corriente continua de 5V. Generalmente las bobinas de los contactores que manejan a los motores funcionan con 120V o 220V. Entonces, el pequeño relé K, que está conectada a la salida del pin 13, mediante un contacto abierto manejara al contactor de línea KL. Un LED o una luz piloto se activa cuando el pin 13 está en alto.

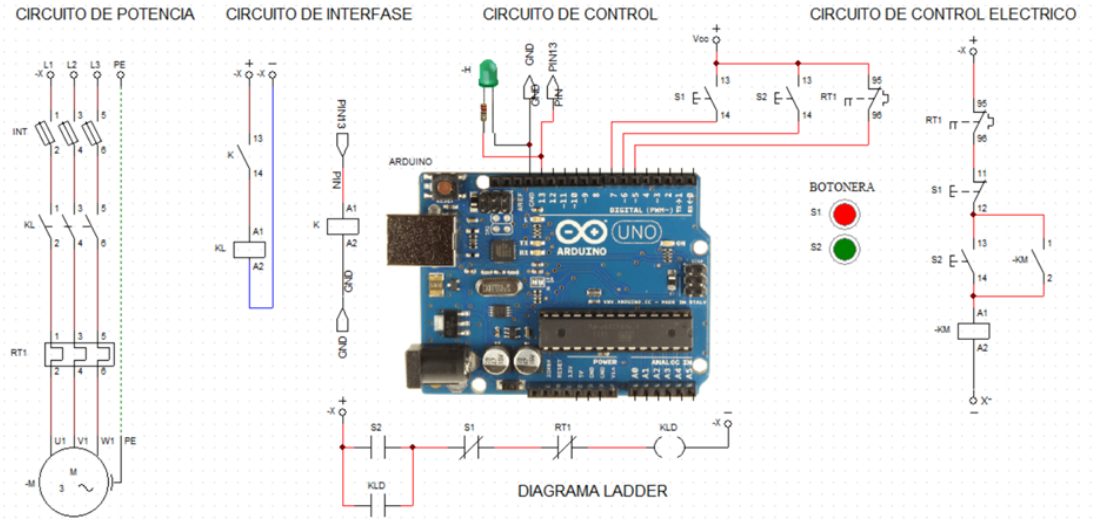
Circuito de control. El pulsador S2, sirve para poner en marcha al motor. Según el diagrama Ladder diseñado en el LDmicro, al accionar este pulsador la salida Y1 se activa, esta salida está vinculada al pin 13 de Arduino. En

este momento que enclavado el circuito y el motor se pone en marcha. En CADe-SIMU también se puede realizar el diagrama Ladder y comprobar el funcionamiento. CADe-SIMU también provee en forma gráfica de pulsadores en 2D y 3D, y con estos botones hemos realizado una botonera, para desde allí accionar los pulsadores. Es común también trabajar con los contactos de los relés y realizar los esquemas de control eléctrico basado en contactos. Este circuito también es útil para verificar el funcionamiento.

En el esquema de simulación (Figura 123), se observa el accionamiento de todos los componentes. El pulso en S2 activa las bobinas de los relés y contactor poniendo en funcionamiento el motor. La flecha en el símbolo del motor indica el sentido de giro. Una luz LED actúa como piloto, indicando que el circuito está accionado. De igual forma se verifica con el diagrama Ladder y el circuito de control eléctrico. Recuerde que previamente la tarjeta Arduino debe estar conectado con el Computador.

Figura 122

Esquemas en CADE SIMU conexión online con la tarjeta Arduino. Arranque directo del motor trifásico



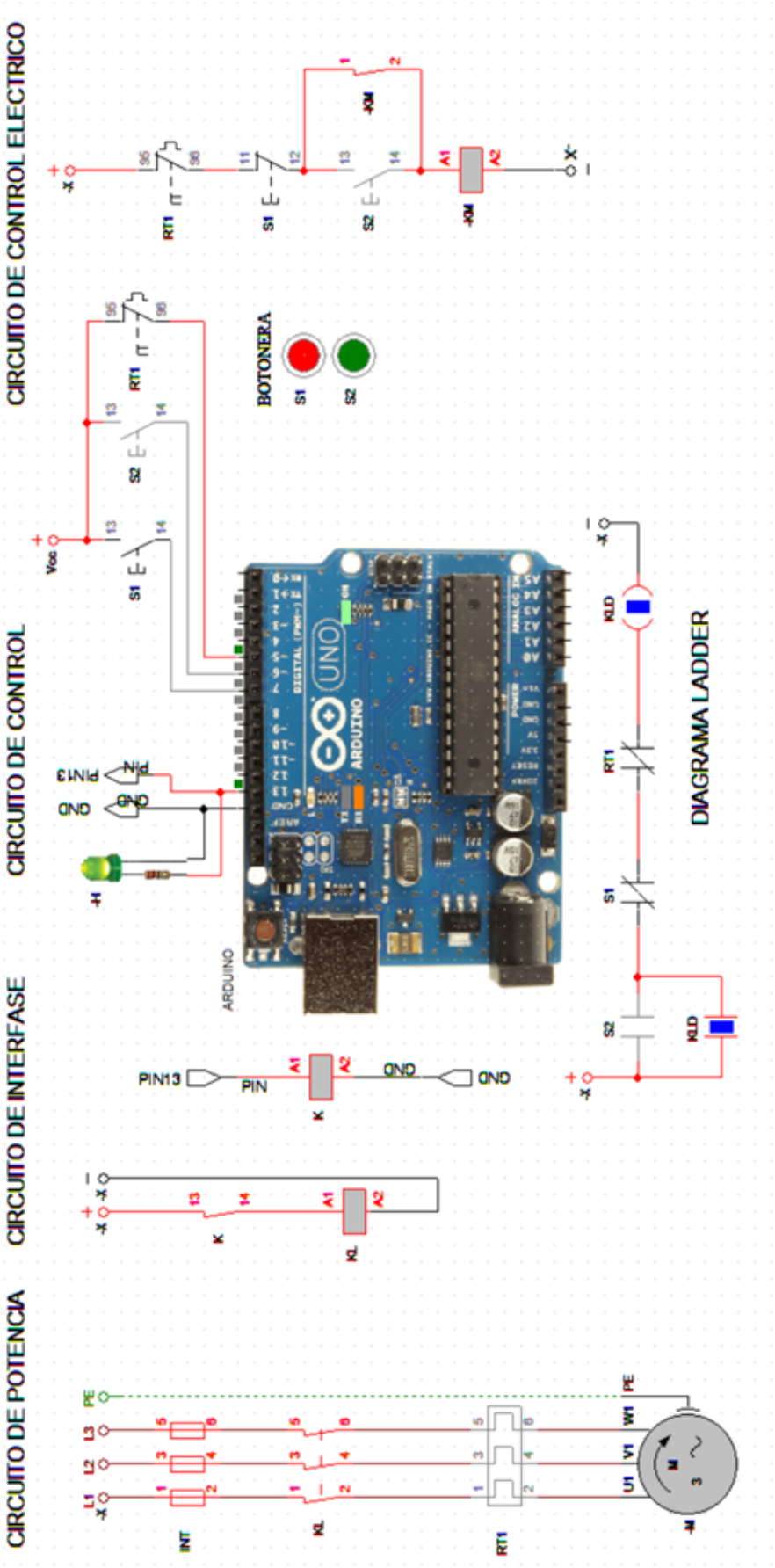


Figura 123
Vista de la simulación en CADe-SIMU. Arranque directo

Inversión de giro de un motor trifásico

Para cambiar el sentido de giro de un motor trifásico se cambia dos de las tres fases que ingresan al motor. Con dos salidas de la tarjeta Arduino podemos manejar a dos relés que pueden accionar a dos contactores de línea. Para este Caso las salidas Y1 y Y2 se energizan mediante los pulsadores X2 y X3. De igual forma adicionamos una entrada X4 para un relé de protección térmica.

Figura 124

Diagrama LDR para la inversión de giro de un motor trifásico

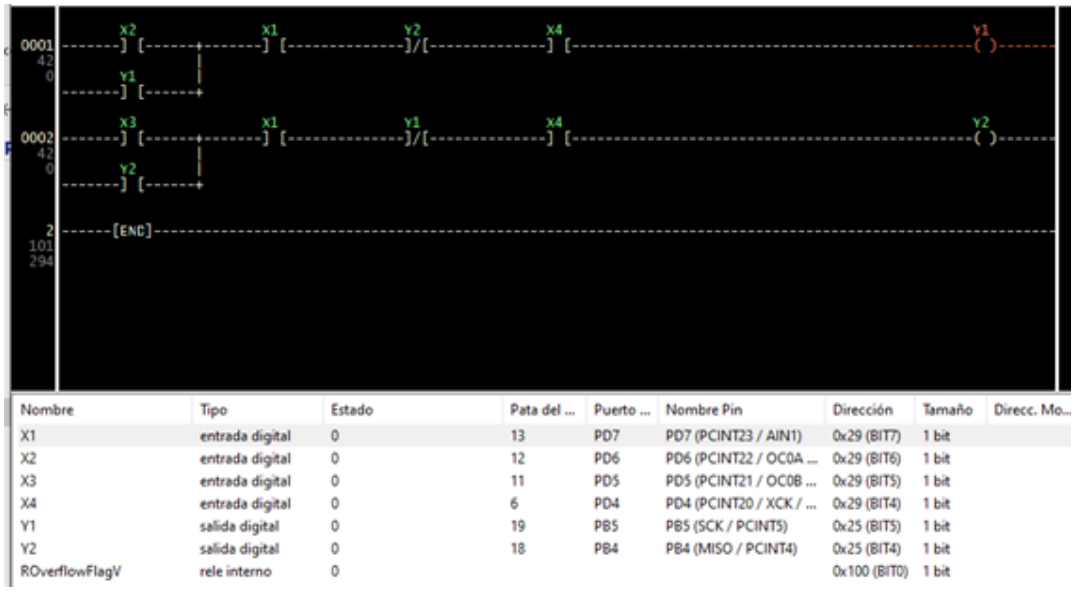


Tabla 15

Asignación de pines

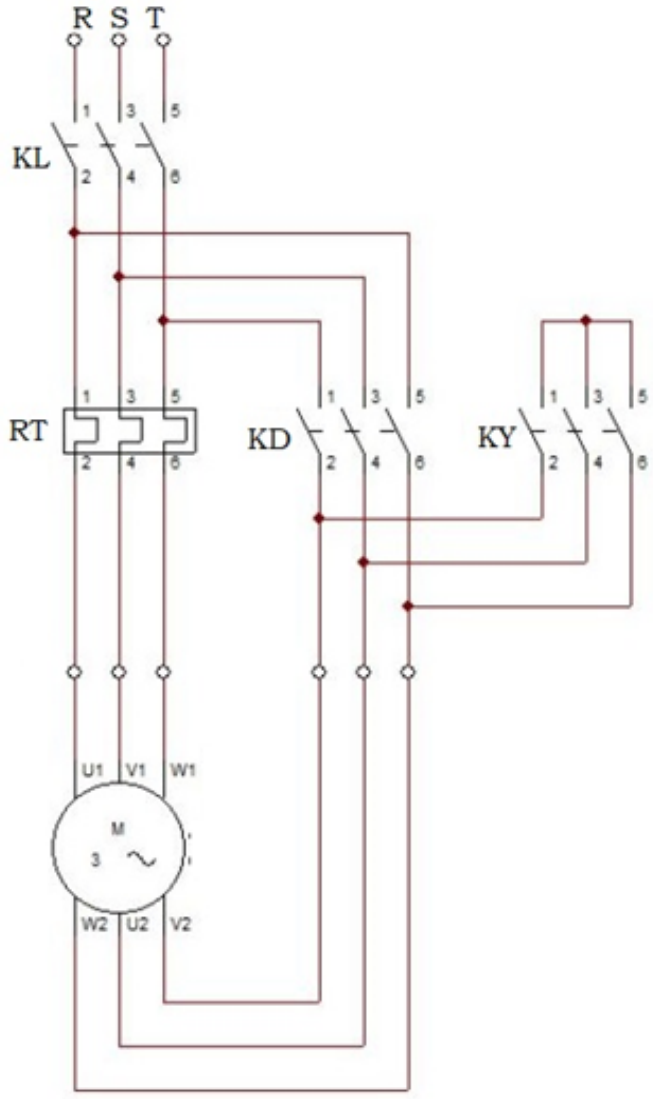
ENTRADAS SALIDAS	PINES ARDUINO	COMPONENTES	FUNCIÓN
X1	7	OFF	APAGAR
X2	6	IZQ	PUL. IZQUIERDA
X3	5	DER	PUL. DERECHA
X4	4	TÉRMICO	PROTECCIÓN
Y1	13	RELÉ IZQ	Accionar contactor - KI
Y2	12	RELÉ DER	Acciona contactor - KD

Al accionar el pulsador de izquierda (IZQ), el motor arranca y gira en un sentido (en nuestro ejemplo a la izquierda). Si accionamos el pulsador de derecha (DER), el motor sigue girando en el mismo sentido. Este circuito es denominado: "sistema de llamado de apagado", porque para cambiar el sentido de giro del motor es necesita apagar el motor con el pulsador deparo (OFF) para poder accionar el pulsador de derecha (DER) y que gire en otro sentido. De esta forma evitamos tracciones en el rotor que dañarían al eje del motor y posibles cortocircuitos en los contactores de línea.

La figura muestra el circuito de fuerza típico de conexiones de los contactos para la inversión de giro del motor trifásico.

Figura 125

Circuito de fuerza para inversión de giro de motor trifásico



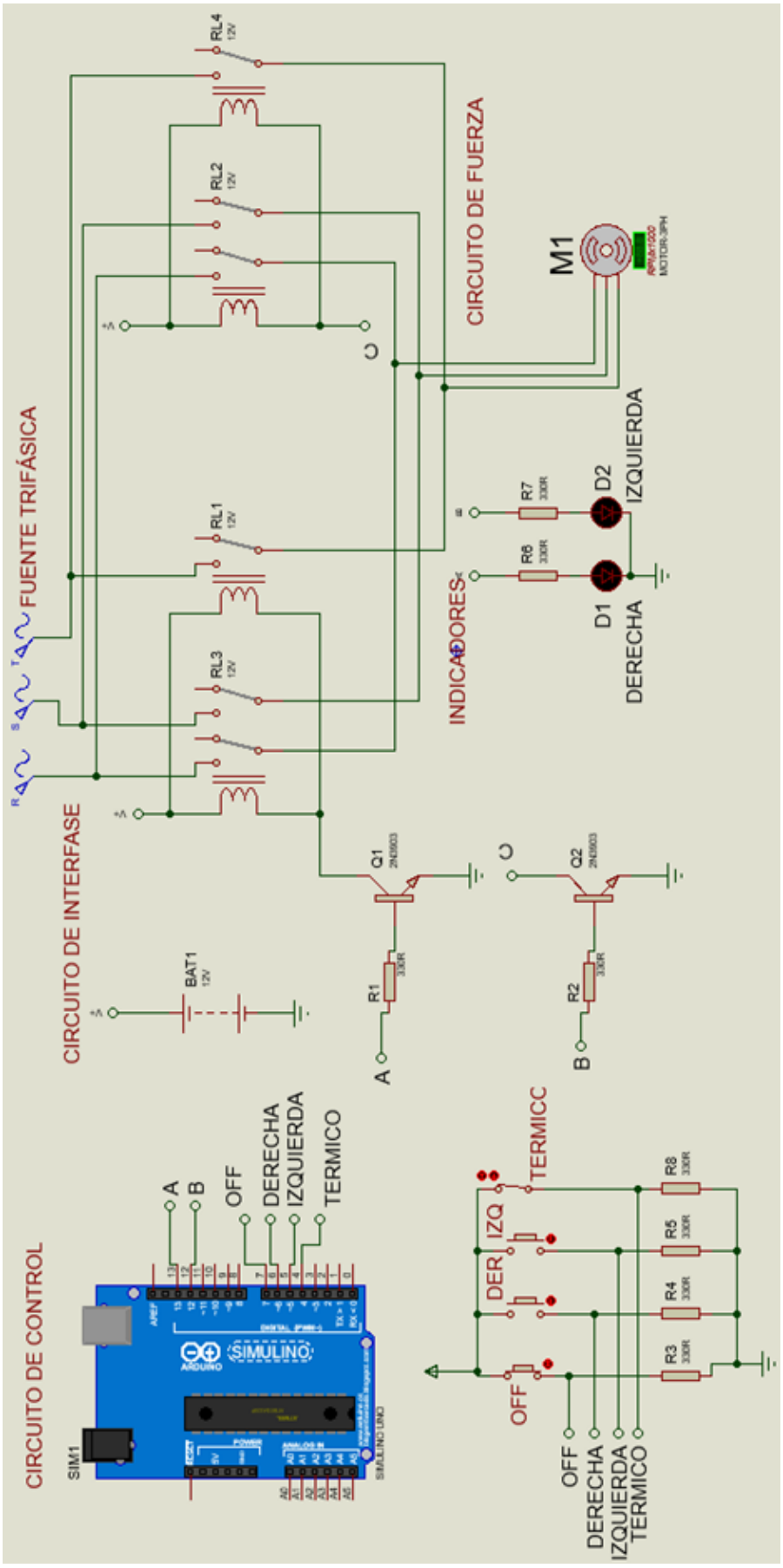
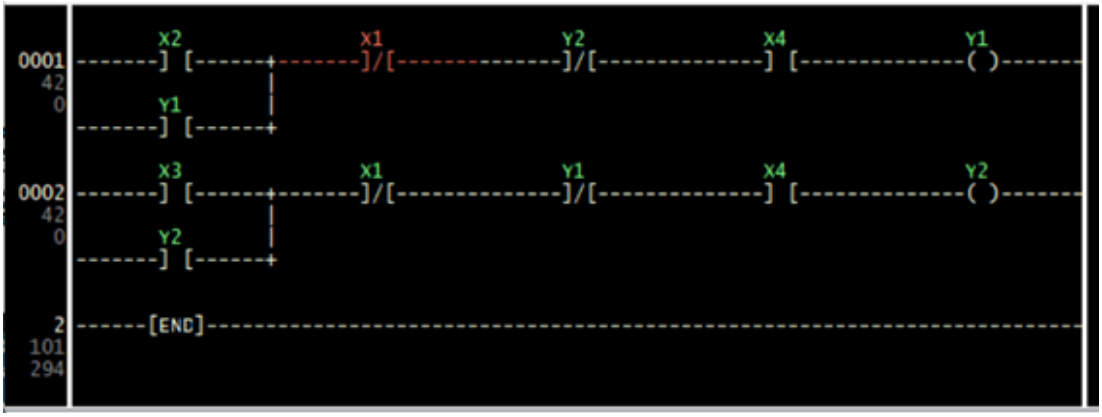


Figura 126
Circuito para la simulación para la inversión de giro de un motor trifásico

Para la simulación en CADe-SIMU, utilizaremos un pulsante normalmente abierto como pulsador de paro. Por tanto, en el diagrama Ladder, se coloca un contacto normalmente cerrado.

Figura 127

Diagrama Ladder para la simulación CADe-SIMU para la inversión de giro de un motor trifásico



* Por motivo de legibilidad se coloca la figura siguiente en la próxima página

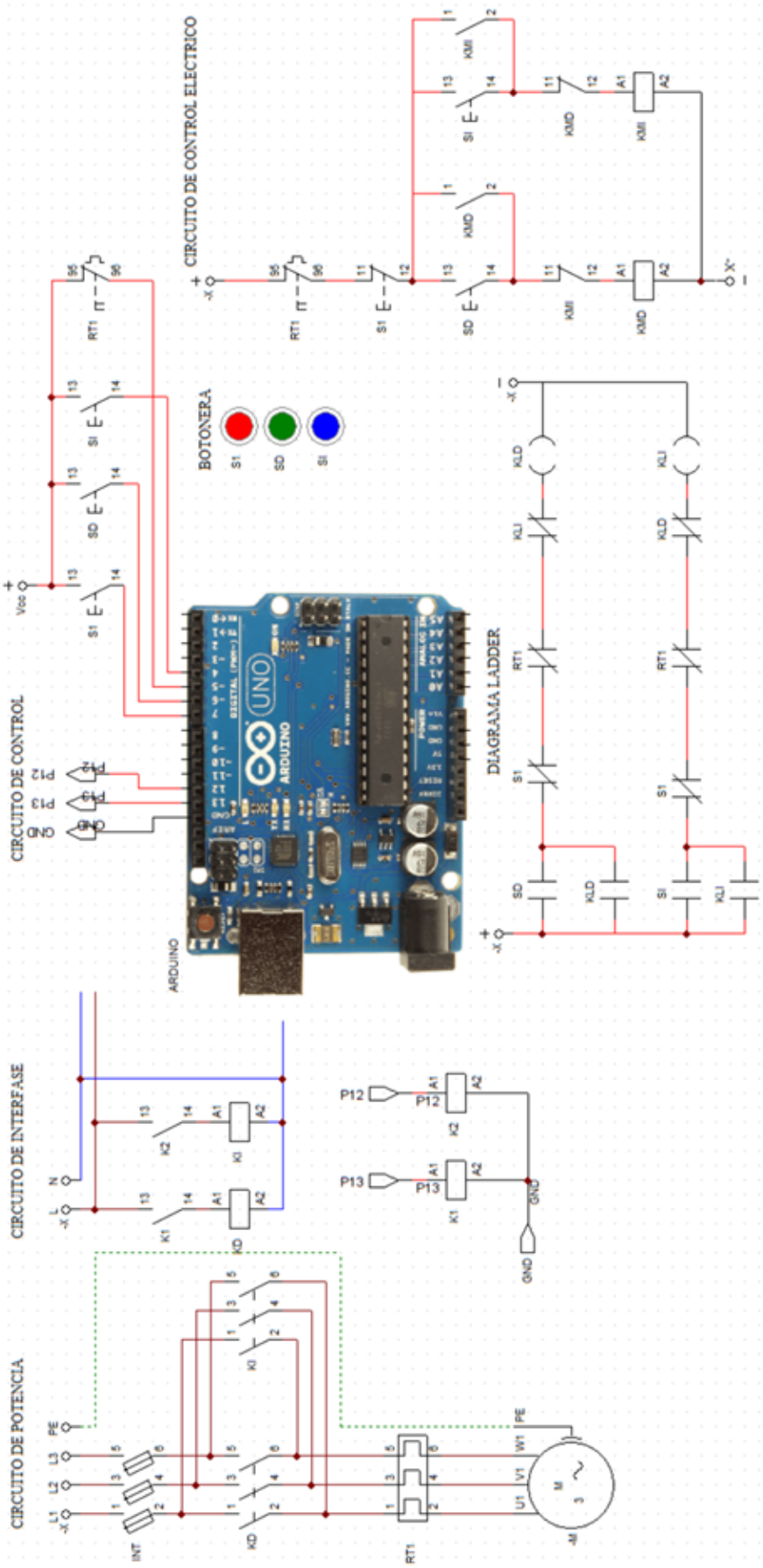


Figura 128
Circuito para la simulación CADe-SIMU para la inversión de giro de un motor trifásico

Arranque estrella triángulo

Cuando el motor de inducción trifásico dispone los 6 terminales de las bobinas para conectar exteriormente es común realizar el arranque estrella – triángulo o arranque a tensión reducida.

Figura 129

Diagrama pictórico de un arranque estrella – triangulo

Nota. Tomado de: https://electrotec.pe/blog/ArduinoEstrellaTriangulo

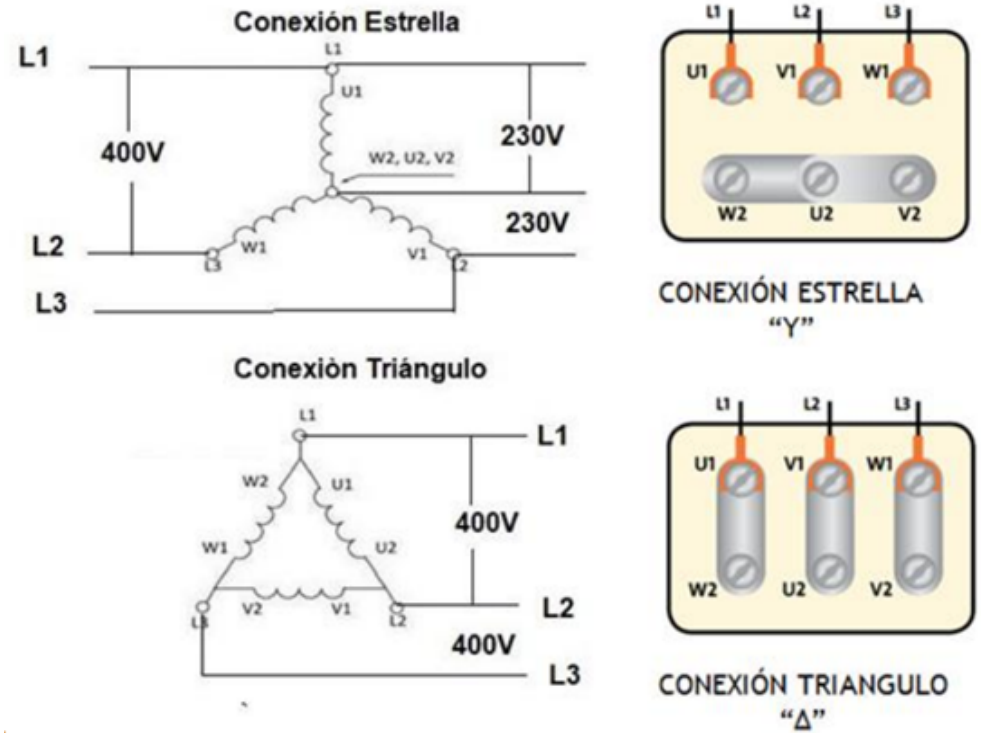
Recordando que el arranque estrella–triángulo de un motor trifásico de inducción, significa que primero se conecta las bobinas en estrella, es decir, todas las salidas de las bobinas se cortocircuitan, esto se hace para reducir la tensión de la línea en las bobinas y disminuir la corriente de arranque del motor. Recuerde que la tensión de fase es igual a:

$$Vf = \frac{1}{\sqrt{3}} VL$$

Una vez que ha alcanzado una velocidad prudente el motor, se procede a cambiar a una conexión denominada en triángulo, que no es más que conectar una salida con una entrada de una bobina diferente del motor para que funcione a su tensión nominal y alcance el máximo torque. Estas conexiones serán realizadas automáticamente por los contactos de los contactores en el circuito de fuerza, entonces todo el control será realizado por el programa Ladder de la tarjeta Arduino y con sus respectivas interfases para manejar a los contactores del circuito de fuerza del motor.

Figura 130

Conexiones de las bobinas del motor en estrella y triángulo



A continuación, se muestra el diagrama Ladder de control para el arranque estrella–triangulo del motor trifásico. Los circuitos control frecuentemente se realizaban con contactores y relés de tiempo. Esto ha sido desplazado con la aparición de los PLCs y ahora, también podemos utilizar la tarjeta Arduino programando en Ladder para obtener los mismos resultados que si realizamos con un PLC.

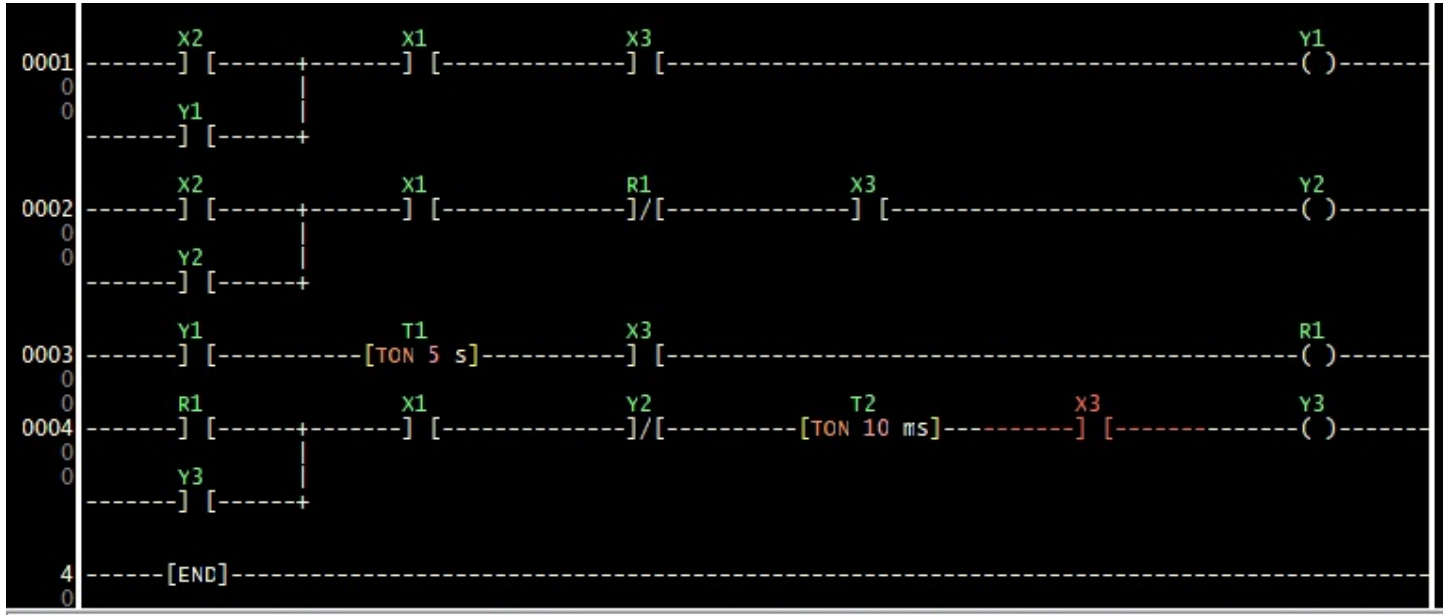
Tabla 16

Asignación de pines

ENTRADAS SALIDAS	PINES ARDUINO	COMPONENTES	FUNCIÓN
X1	7	PULSADOR CNC	PARO
X2	6	PULSADOR CNA	MARCHA
X3	4	TÉRMICO CNC	PROTECCIÓN
Y1	13	RELÉ LINEA	CONTACTOR DE LINEA
Y2	12	RELÉ ESTRELLA	CONTACTOR ESTRELLA
Y3	11	RELÉ TRIANGULO	CONTACTOR TRIANGULO

Figura 131

Circuito LDR para arranque estrella-triángulo de un motor trifásico de inducción



Nombre	Tipo	Estado	Pata del ...	Puerto ...	Nombre Pin	Dirección
X2	entrada digital	0	12	PD6	PD6 (PCINT22 / OC0A ...	0x29 (BIT6)
X3	entrada digital	0	11	PD5	PD5 (PCINT21 / OC0B ...	0x29 (BIT5)
Y1	salida digital	0	19	PB5	PB5 (SCK / PCINT5)	0x25 (BIT5)
Y2	salida digital	0	18	PB4	PB4 (MISO / PCINT4)	0x25 (BIT4)
Y3	salida digital	0	17	PB3	PB3 (MOSI / OC2A / P...	0x25 (BIT3)
R1	rele interno	0				
ROverflowFlagV	rele interno	0				
T1	activar retardo	0x0000 = 0 = 0 ms				0x100
T2	activar retardo	0x0000 = 0 = 0 ms				0x102

Al accionar el pulsador de arranque (ON), conectado a la entrada X2 en el diagrama Ladder, se activan las salidas Y1 y Y2 que a su vez energizan los relés de línea y de estrella del circuito de control, que activan a los contactores de línea (KL) y de estrella (KY) que forman parte del circuito de fuerza. Luego de un tiempo (5 segundos) establecido en el circuito del control (programa Ladder) el relé retardo a la energización T1 desconecta a la salida Y2 que maneja al relé de control de estrella y pone en funcionamiento al temporizador T2, el mismo que en 10 ms activa la salida Y3 que controla al relé de triángulo. Es conveniente tener un tiempo de retardo (tiempo muerto) entre la desconexión del relé de estrella y la conexión del relé de triángulo, para evitar posibles cortocircuitos por la velocidad de los contactos mecánicos de los relés de control y los contactores de fuerza.

El pulsador de paro que está conectado a la entrada X1, sirve para desactivar el circuito en cualquier momento. En este caso, el pulsador será un NC para que el contacto que está abierto en el diagrama Ladder se cierre y pueda permitir el paso de la corriente y al accionarlo, interrumpirá la corriente abriendo el circuito. Se prevé una entrada X3, para un contacto NC de un relé térmico, que se accionara ante posibles recalentamientos o sobrecargas del motor. De la misma forma, los contactos del relé térmico en el circuito de potencia harán su trabajo quitando la alimentación al motor.

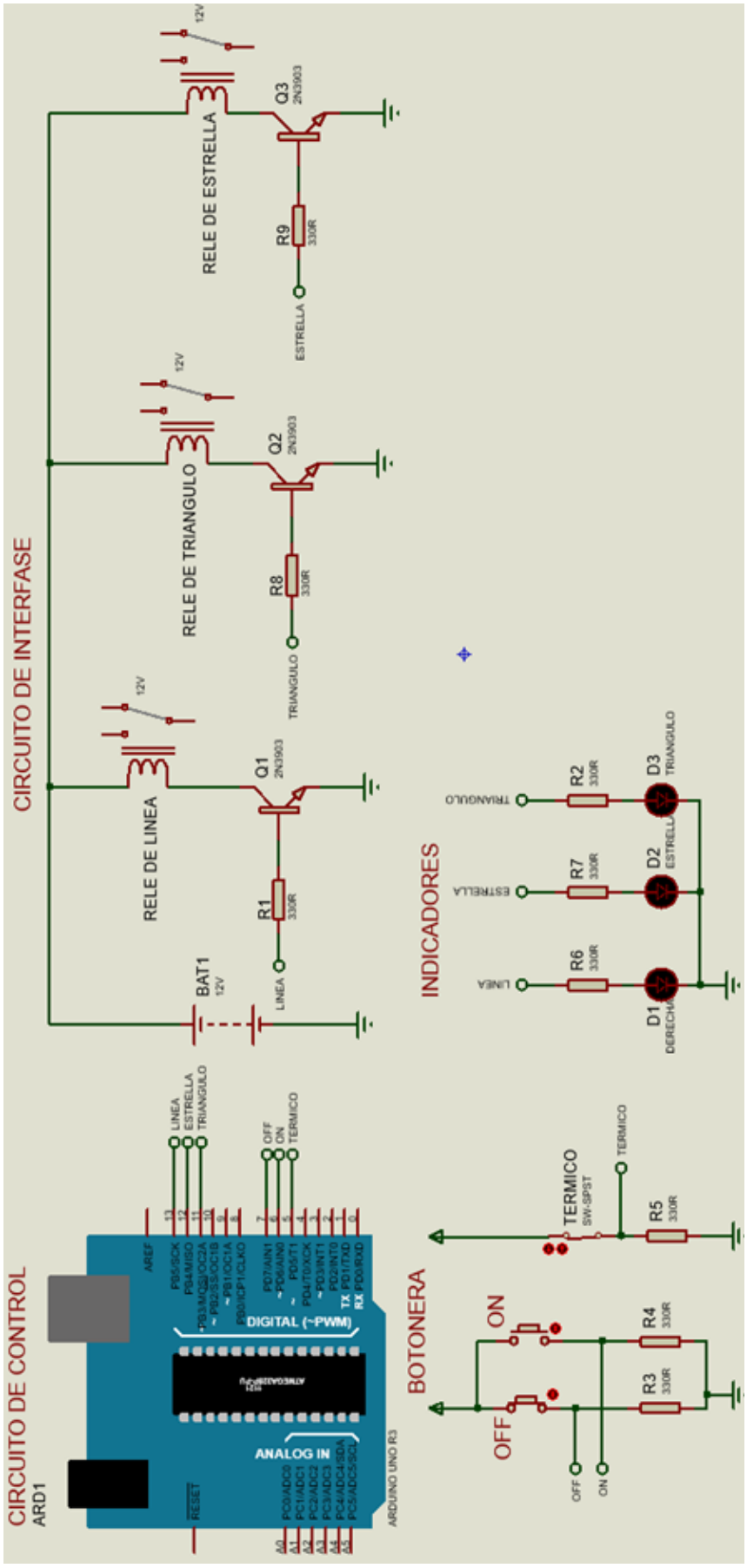


Figura 132
Circuito de simulación arranque estrella triángulo

Figura 133

Circuito de fuerza y de control eléctrico para arranque estrella - triangulo para un motor trifásico de inducción

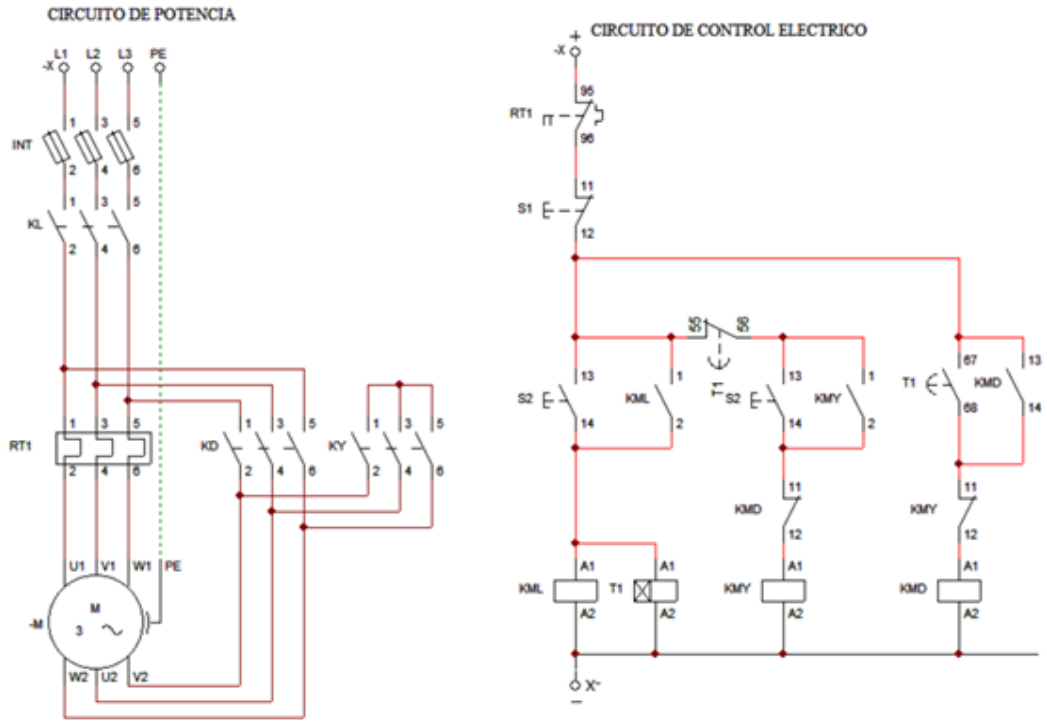
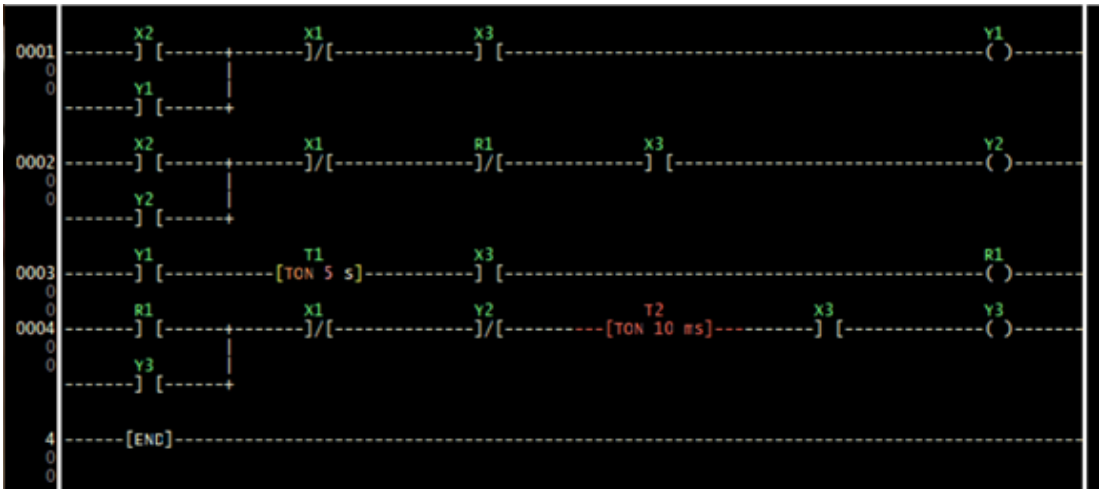


Figura 134

Diagrama Ladder para la simulación en CADE SIMU para arranque estrella - triangulo para un motor trifásico de inducción



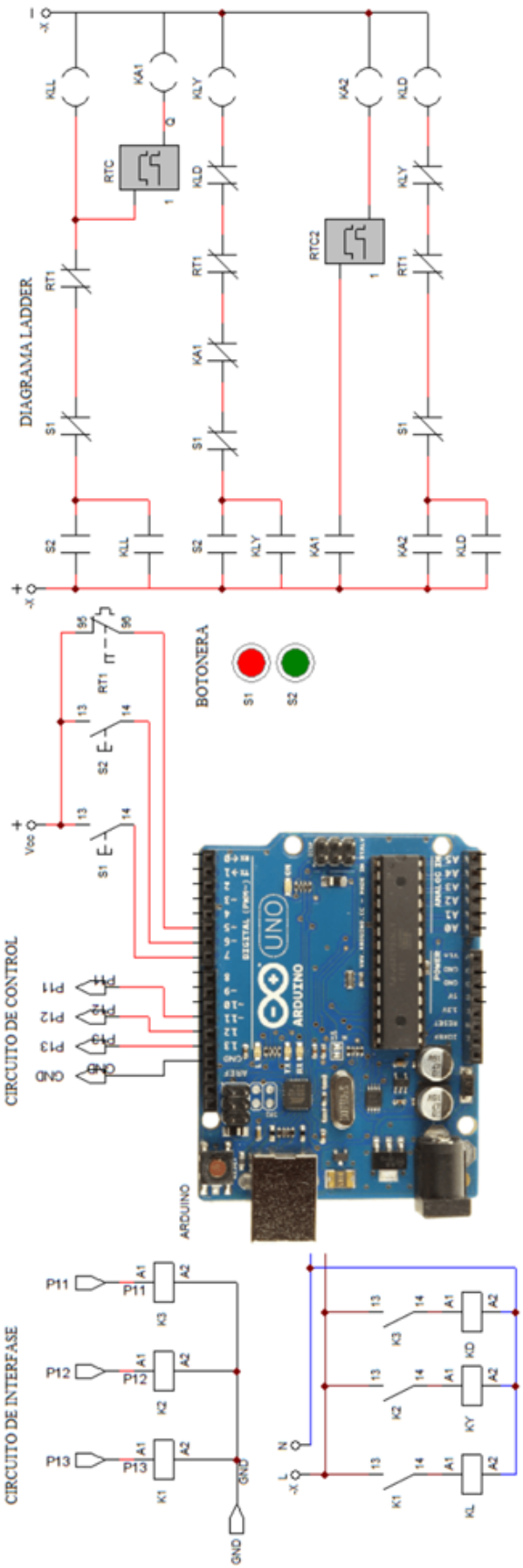


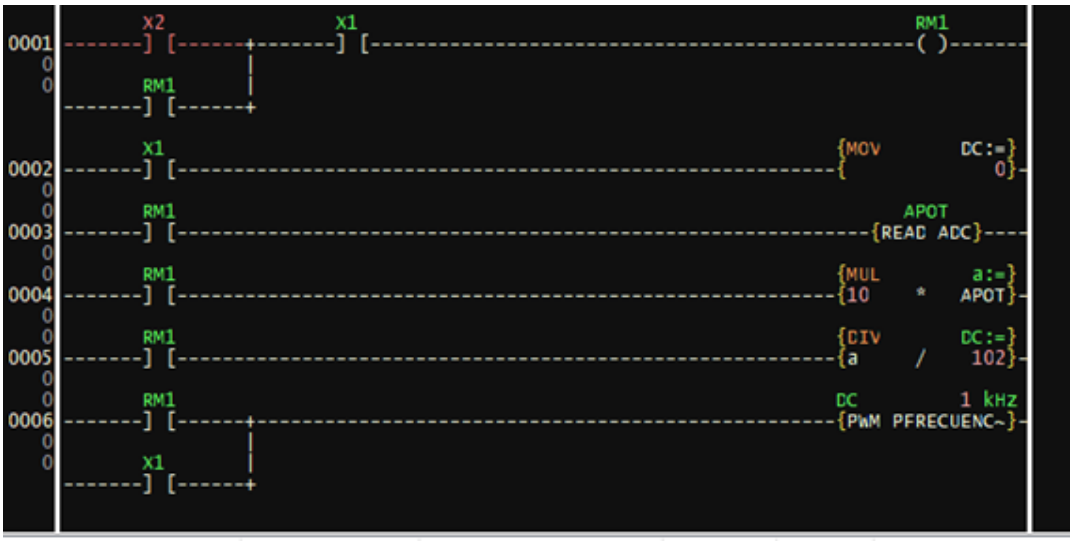
Figura 135
Circuitos de simulación en CADE SIMU para arranque estrella - triangulo para un motor trifásico de inducción

Control de velocidad de motor DC, con señal PWM

Una de las aplicaciones típicas de PLCs que disponen de salidas PWM, es el control de velocidad de motores DC.

Figura 136

Diagrama LDR para control de velocidad de un motor DC con señal PWM



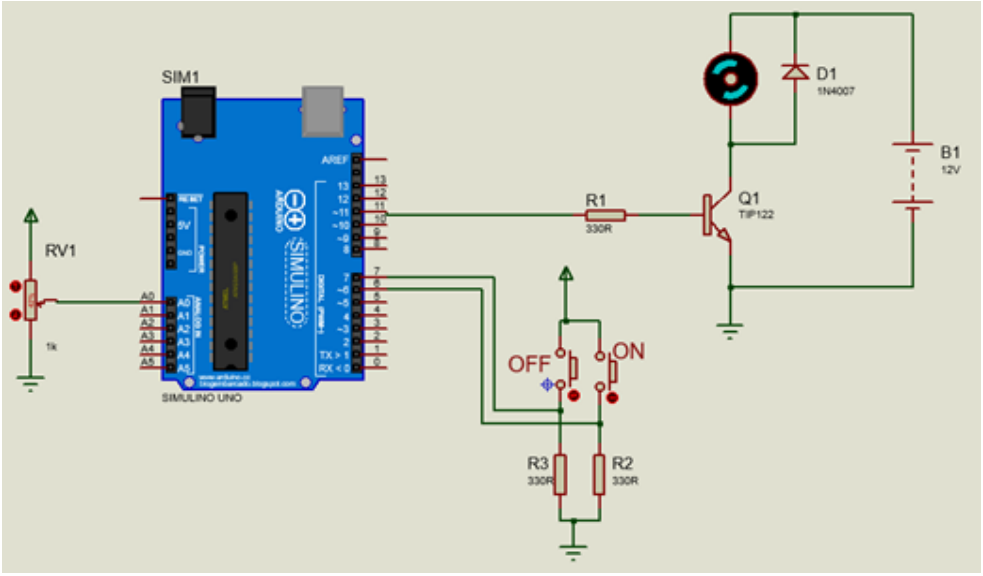
Nombre	Tipo	Estado	Pata del ...	Puerto ...	Nombre Pin
DC	var general	0x0000 = 0			
a	var general	0x0000 = 0			
X1	entrada digital	0	13	PD7	PD7 (PCINT23 / AIN1)
X2	entrada digital	0	12	PD6	PD6 (PCINT22 / OC0A
APOT	entrada adc	0x0000 = 0	23	PC0	PC0 (ADC0 / PCINT8)
PFRECUENCIA	salida PWM	OFF	17	PB3	PB3 (MOSI / OC2A / P
RM1	rele interno	0			
ROverflowFlagV	rele interno	0			

En el diagrama LDR, al recibir una señal en X1 apaga el circuito y coloca el ciclo de trabajo de la señal PWM en 0. Cuando ingresa una señal en X2 se activa el relé interno RM1, permitiendo operar a todo el circuito. Con esto queda habilitada la línea de lectura del ADC para determinar el valor entrada de la señal analógica. Al valor de la conversión entregado (0 a 1023) lo multiplicamos por 10, entonces obtendremos un rango de 0 a 10023 y dividimos para 102, esto lo hacemos para reducir el error y tener un valor aproximado de 0 a 100, que corresponde al ciclo de trabajo (DC) de la señal PWM. De esta forma tenemos un control total de la señal de salida que alimenta al motor mediante el control con el potenciómetro.

Para simular el circuito de acuerdo a las asignaciones en el diagrama LDR, utilizamos un potenciómetro y le conectamos en la entrada analógica A0 con 5V. Todos los demás componentes se conectan como se indica en la figura.

Figura 137

Circuito para simular el control de velocidad de un motor DC con señal PWM



Control de un semáforo

Otro de los circuitos típicos de control industrial es el circuito para control de un semáforo. Con esta aplicación demostramos lo versátil y útil que resulta realizar mediante la programación en Ladder de la tarjeta Arduino y que en muchos de los casos resulta más económico al utilizar este dispositivo.

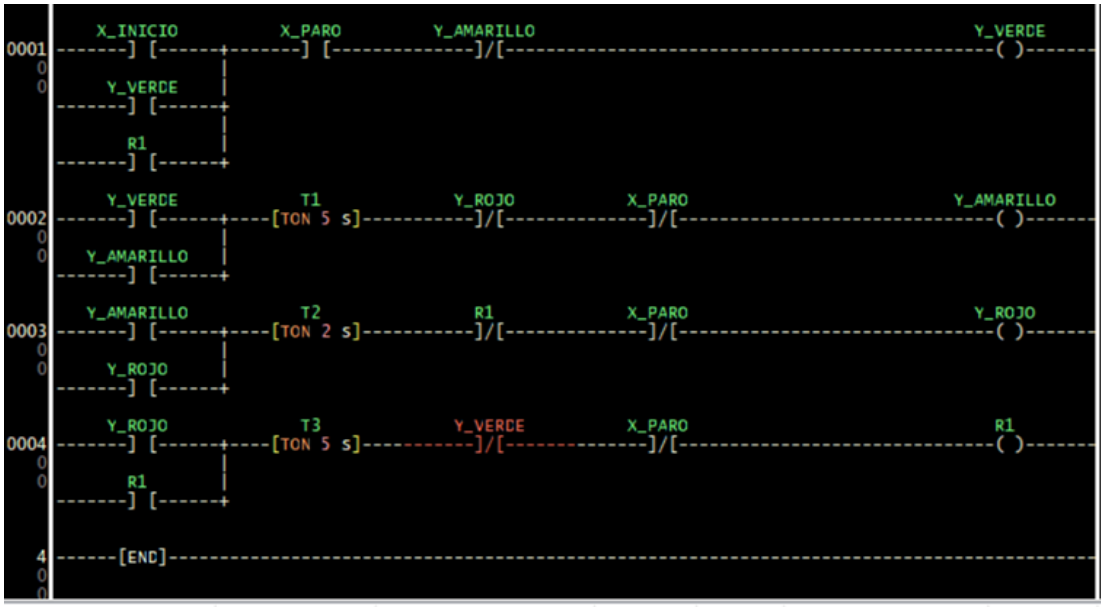
El diagrama LDR presenta el circuito de control de un semáforo. Consta de dos pulsadores para inicio y paro del circuito. Las salidas se han etiquetado con los nombres de las luces que corresponden al semáforo.

Al accionar el pulsador conectado en la entrada X_INICIO, se activa la salida Y_VERDE que maneja a LUZ VERDE, transcurrido el tiempo establecido en el temporizador T1 (5 segundos), desactiva a la LUZ VERDE y activa la LUZ AMARILLA. Transcurrido el tiempo fijado en el temporizador T2 (2 segundos), apaga la luz AMARILLA y prende la luz ROJA. Esta luz permanece prendida por el tiempo calibrado en el temporizador T3 (5 segundos). En este instante activa un relé interno R1 y con un contacto de este relé activamos a la luz VERDE para repetir el de forma automática y continua. Recuerde que los

temporizadores en LDmicro no es una salida, por loque es necesario manejar en caso un relé interno.

Figura 138

Diagrama Ladder para control de un semáforo

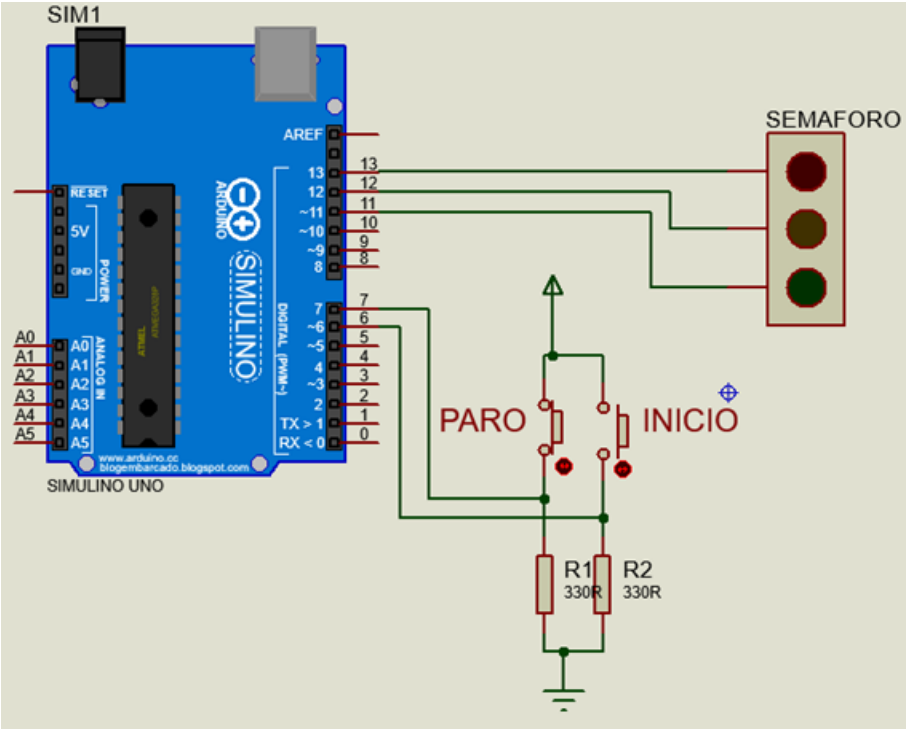


Nombre	Tipo	Estado	Pata del ...	Puerto ...	Nombre Pin	Dirección
X_INICIO	entrada digital	0	12	PD6	PD6 (PCINT22 / OC0A ...	0x29 (BIT6)
X_PARO	entrada digital	0	13	PD7	PD7 (PCINT23 / AIN1)	0x29 (BIT7)
Y_AMARILLO	salida digital	0	18	PB4	PB4 (MISO / PCINT4)	0x25 (BIT4)
Y_ROJO	salida digital	0	19	PB5	PB5 (SCK / PCINT5)	0x25 (BIT5)
Y_VERDE	salida digital	0	17	PB3	PB3 (MOSI / OC2A / P...	0x25 (BIT3)
R1	rele interno	0				

Para ver de una mejor forma, se ha simulado el diagrama Ladder del semáforo utilizando el simulador Proteus. Se incluyen dos pulsadores de inicio y paro, que podrían ser reemplazados por un interruptor, con un previo cambio en el diagrama Ladder.

Figura 139

Circuito para simular el funcionamiento de un semáforo



En PC-SIMU, se ha preparado una animación para comunicarse con la tarjeta Arduino. El HMI realizado permite interactuar con la tarjeta Arduino.

Figura 140

Semáforo animación en PC-SIMU y comunicación con Arduino

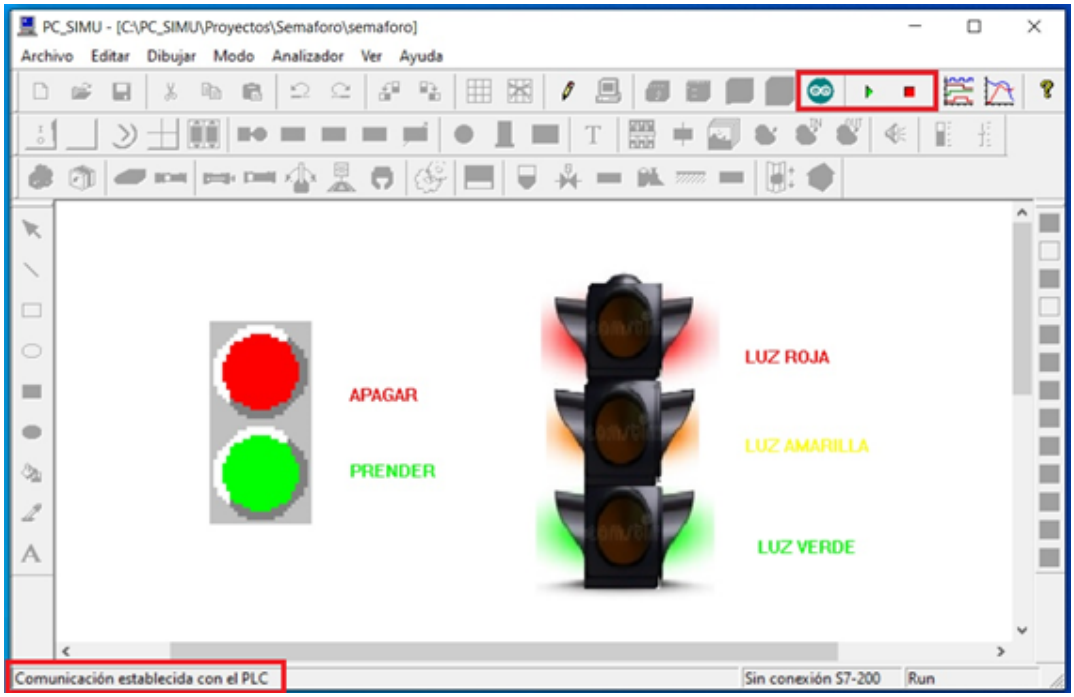


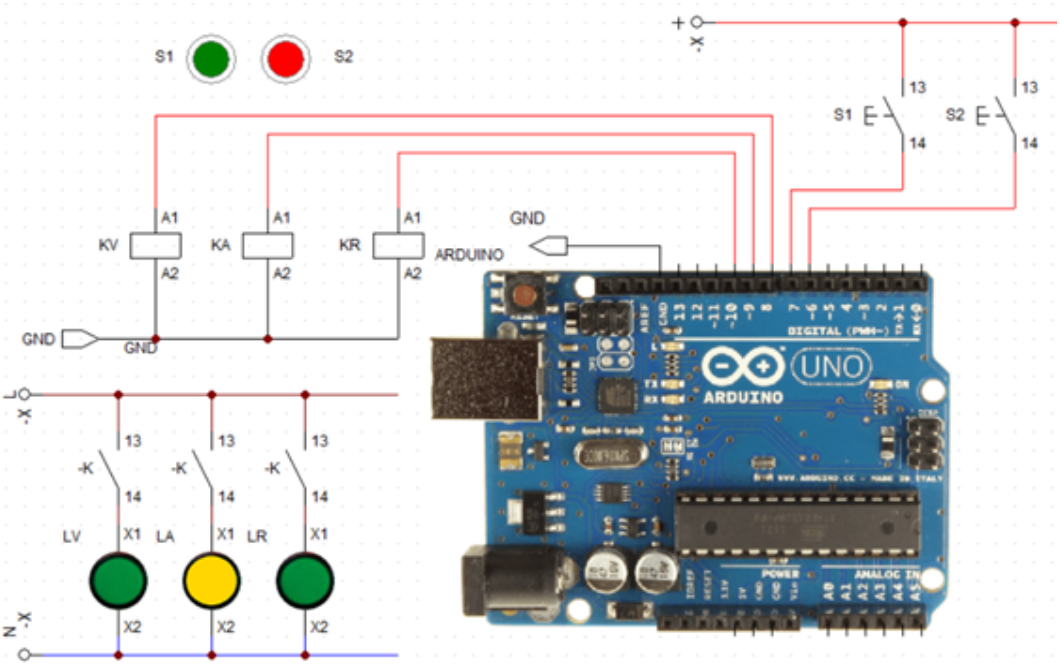
Figura 141

Resultados de la animación en PC-SIMU y comunicación con Arduino

La Figura 142, indica las conexiones de los componentes en CADe-SIMU y para la tarjeta Arduino física.

Figura 142

Diagrama de conexiones del semáforo



El funcionamiento del semáforo puede visualizarlo en el blog web del autor del texto:

https://rafaelreinososrv.blogspot.com/

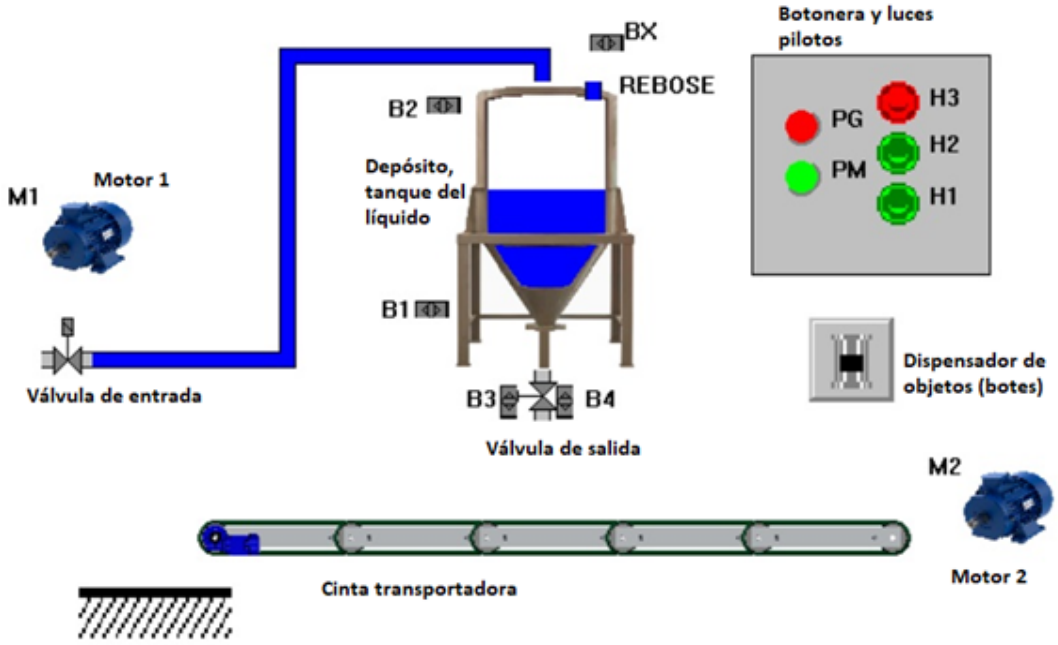
Llenado de botes

El proceso inicia al accionar el pulsador PM, activa el motor M1 que está asociado a una bomba de agua, la válvula de entrada se abre y el tanque que

inicialmente está vacío, empieza a llenarse. Al mismo tiempo del alimentador de botes cae el recipiente en la cinta, la misma que es movida a la izquierda por el motor M2. Al llegar el recipiente el sensor capacitivo (B3) en centro de la válvula de salida detiene la cinta y el motor 2. La válvula de salida se abre empezando a llenar el bote. Cuando el sensor capacitivo (B4) detecta el nivel del líquido en el recipiente y cuando está lleno, el bote, continúa hacia la izquierda. Otro bote está listo para continuar el proceso, de forma automática.

Figura 143

Proceso de llenado de botes



El funcionamiento del presente ejercicio puede visualizarlo en el blog web del autor del texto:

https://rafaelreinososrv.blogspot.com/

Lista de componentes usados

A continuación, se describen todos los componentes usados en el proceso de llenado de botes.

Sensores y pulsadores

- B1, sensor capacitivo NA, detecta el nivel mínimo de líquido.
- B2, sensor capacitivo NA, detecta el nivel máximo de líquido.
- B3, sensor capacitivo NA, detecta el extremo izquierdo del bote.
- B4, sensor capacitivo NA, detecta el nivel del líquido en el bote. Envía la señal de 1 cuando el bote está lleno.

- BX, sensor capacitivo NA, auxiliar del sensor B4, por alguna anomalía el sensor B4, no detecto que el tanque está lleno, detecta el sensor BX que el líquido del depósito empieza a rebosar.
- PG, pulsador NC, paro general del proceso.
- PM, pulsador NA, puesta en marcha del proceso.

Motores y luces pilotos

- M1, motor 1, acciona la bomba del líquido para llenar el depósito.
- M2, motor 2, acciona la cinta transportadora.
- H1, luz piloto que indica el funcionamiento del motor 1.
- H2, luz piloto que indica el funcionamiento del motor 2.
- H3, luz piloto que indica alguna falla en los sensores del depósito.

Asignación de los pines para PC-SIMU y CADE SIMU

En este punto se deben asignar correctamente los pines de la tarjeta Arduino para los dos programas, como se indicó en el Capítulo I, sección "Simuladores CADe-SIMU y PC-SIM". La Tabla 17 muestra las asignaciones respectivas.

Tabla 17

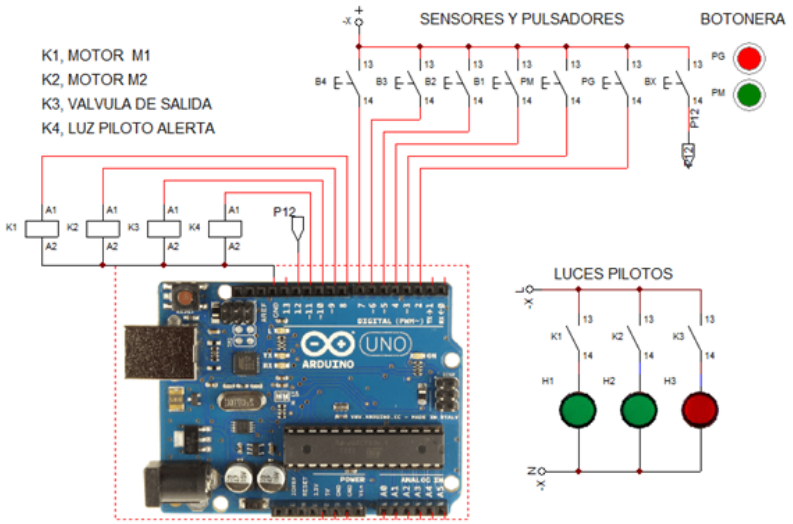
Asignación de pines de Arduino en los programas PC-SIMU y CADe-SIMU

PIN ARDUINO	PC-SIMU		CADE SIMU		COMPONENTES
13					
12	I1.4		I1.4	BX	SENSOR NIVEL REBOSE DEPÓSITO
11		Q1.3	Q1.3	H3	LUZ PILOTO ALERTA
10		Q1.2	Q1.2	VS	VÁLVULA DE SALIDA
9		Q1.1	Q1.1	M2	MOTOR 2
8		Q1.0	Q1.0	M1	MOTOR 1
7	I0.7		I0.7	B4	SENSOR NIVEL LLENO BOTE
6	I0.6		I0.6	B3	SENSOR BOTE
5	I0.5		I0.5	B2	SENSOR NIVEL MÁXIMO DEPÓSITO
4	I0.4		I0.4	B1	SENSOR NIVEL BAJO DEPÓSITO
3	I0.3		I0.3	PM	PULSADOR DE MARCHA
2	I0.2		I0.2	PG	PULSADOR DE PARO

Diagrama de conexiones

Figura 144

Diagrama de conexiones de sensores y actuadores en Arduino



En CADe-SIMU, por defecto vienen los pines 2 al 7 como entradas y los pines del 8 al 13 como salidas. En el pin 12 se conectará el sensor de rebose, es decir, es un pin como entrada, entonces desmarcamos la opción en el PIN 12 para que se convierta en IN en el cuadro de edición de la tarjeta Arduino; de igual manera, activamos el puerto de comunicación y conectar al simular para interactuar con la tarjeta Arduino física.

Figura 145

Configuración del Arduino en CADe-SIMU

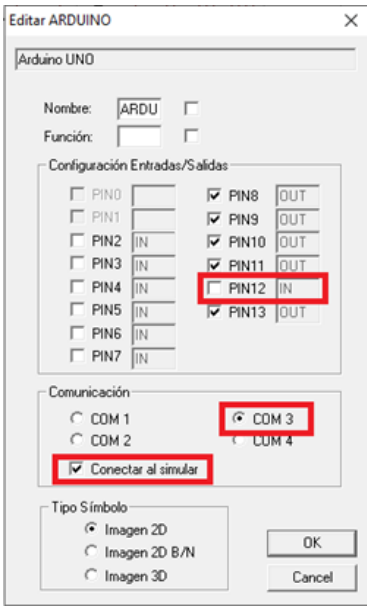


Diagrama Ladder

Para elaborar el diagrama Ladder en LDmicro, se mantiene la misma nomenclatura de las designaciones de la tabla en Arduino. Se debe tener en cuenta que el programa asigna directamente la primera letra del símbolo según el tipo de componente.

Figura 146

Diagrama LDR para el proceso de llenado de botes

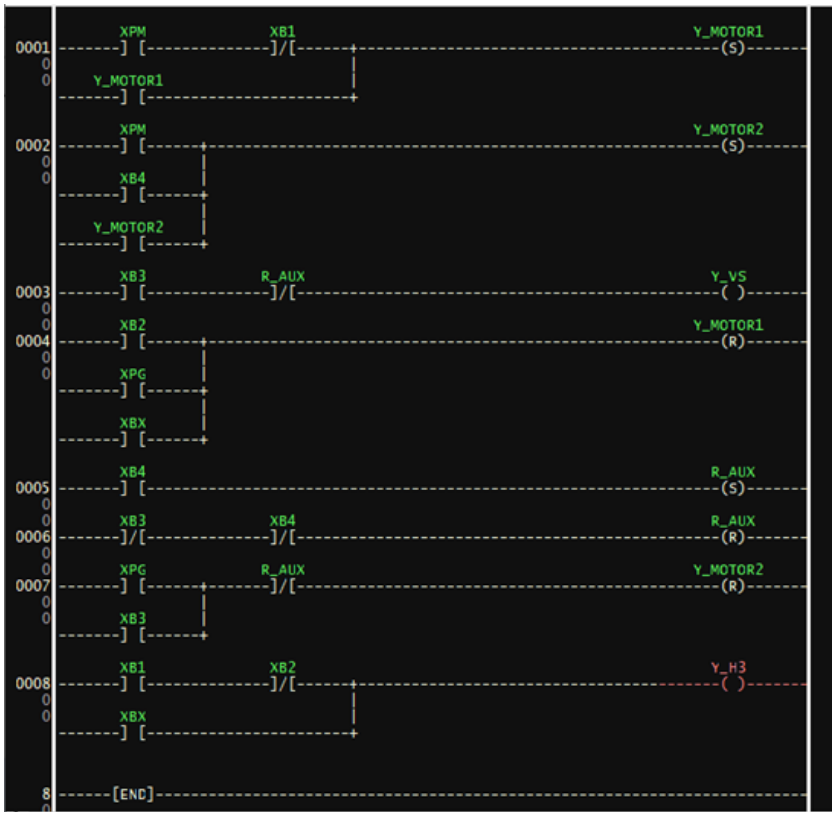


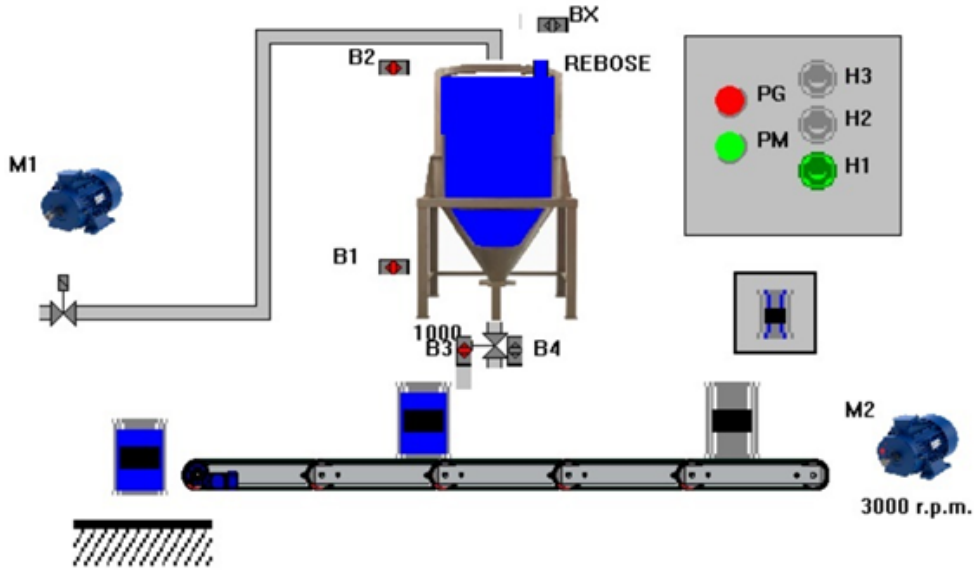
Figura 147

Asignación de pines en LDmicro para la tarjeta física

Nombre	Tipo	Estado	Pata del Micro	Puerto del Micro	Nombre Pin
XB1	entrada digital	0	6	PD4	PD4 (PCINT20 / XCK / T0)
XB2	entrada digital	0	11	PD5	PD5 (PCINT21 / OC0B / T1)
XB3	entrada digital	0	12	PD6	PD6 (PCINT22 / OC0A / ...
XB4	entrada digital	0	13	PD7	PD7 (PCINT23 / AIN1)
XBX	entrada digital	0	18	PB4	PB4 (MISO / PCINT4)
XPG	entrada digital	0	4	PD2	PD2 (PCINT18 / INT0)
XPM	entrada digital	0	5	PD3	PD3 (PCINT19 / OC2B / I...
Y_H3	salida digital	0	16	PB2	PB2 (SS / OC1B / PCINT2)
Y_MOTOR1	salida digital	0	14	PB0	PB0 (PCINT0 / CLKO / IC...
Y_MOTOR2	salida digital	0	15	PB1	PB1 (OC1A / PCINT1)
Y_VS	salida digital	0	17	PB3	PB3 (MOSI / OC2A / PCI...
ROverflowFlagV	rele interno	0			
R_AUX	rele interno	0			

Figura 148

Captura de imagen del funcionamiento del proceso de llenado de botes



Vaivén de un cilindro de doble efecto

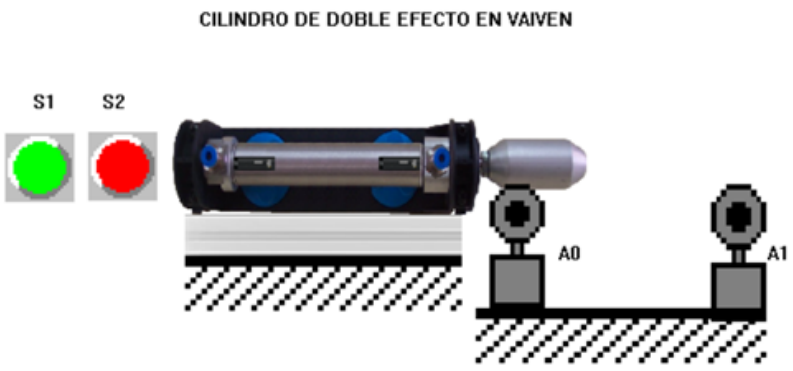
Dentro del campo de la automatización el control de elementos neumáticos y electroneumáticos es otra de las aplicaciones donde ampliamente se utilizan los PLCS.

En este ejercicio realizaremos el control de un cilindro de doble efecto para que realice la secuencia A+A-, que se conoce como vaivén.

La Figura 149 muestra la disposición de los componentes para el ejercicio propuesto en PC SIMU.

Figura 149

Disposición de los componentes en PC SIMU



Funcionamiento

Al accionar el pulsador de marcha S1, el cilindro avanza (A+). Cuando el vástago llega al final de carrera A1, el cilindro retrocede (A-). El final de carrera A0, detecta que el vástago está contraído, nuevamente repite el ciclo. Con el pulsador S2, detenemos la secuencia, empezando siempre con el cilindro contraído.

La Tabla 18 muestra las asignaciones realizadas con las nomenclaturas correspondientes a los programas y sus equivalentes a los pines de la tarjeta Arduino Uno.

Tabla 18

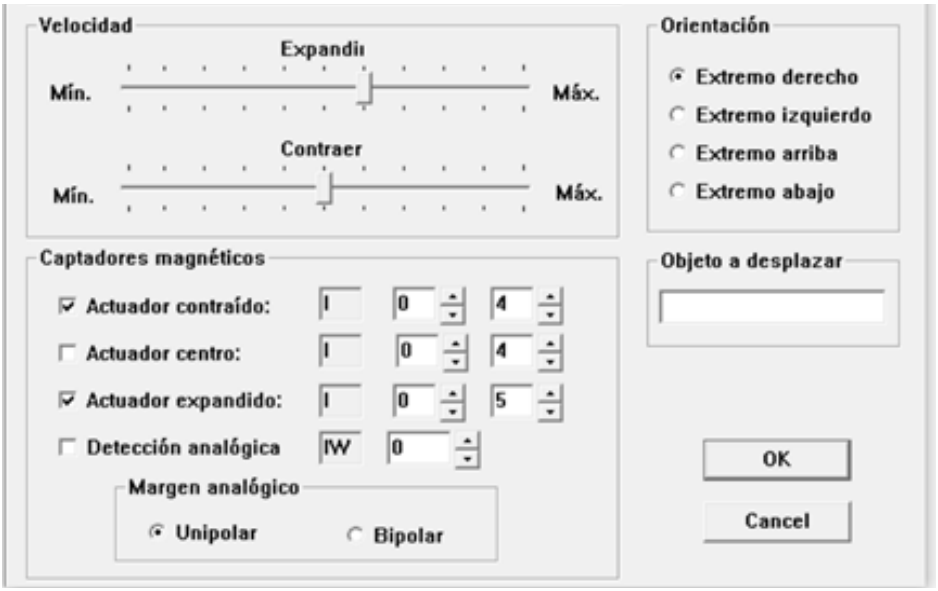
Asignación de pines de Arduino en los programas PC-SIMU y CADe-SIMU

PIN ARDUINO	PC-SIMU	CADE SIMU		COMPONENTES
13				
12				
11	Q1.3			
10	Q1.2	Q1.2	EV2	ELECTROVALVULA A-
9	Q1.1	Q1.1	EV1	ELECTROVALVULA A+
8	Q1.0	Q1.0	KM1	RELE: MARCHA – PARO
7	I0.7			
6	I0.6			
5	I0.5	I0.5	A2	FINAL DE CARRERA-CONTRAIDO
4	I0.4	I0.4	A1	FINAL DE CARRERA-EXPANDIDO
3	I0.3	I0.3	S2	PULSADOR DE PARO
2	I0.2	I0.2	S1	PULSADOR DE MARCHA

Para PC SIMU, fijamos los parámetros para cada componente. Para expandir el cilindro la salida Q1.1, para contraer el cilindro la salida Q1.2, que corresponde a los pines 9 y 10 respectivamente de la tarjeta Arduino Uno. En la Figura 150 se muestra los parámetros del cilindro de doble efecto.

Figura 150

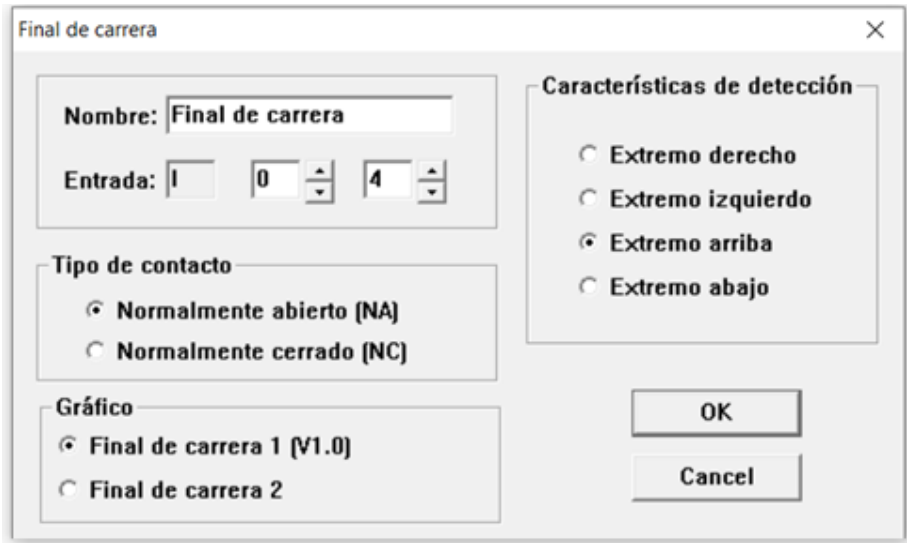
Parámetros del actuador neumático, cilindro de doble efecto



Para los captadores magnéticos se designan las entradas I0.4, para detectar cuando el cilindro está contraído y la entrada I0.5 para censar cuando el cilindro está expandido. En estado de reposo los dos sensores son contactos normalmente abiertos. Se puede usar un sensor magnético o un final de carrera.

Figura 151

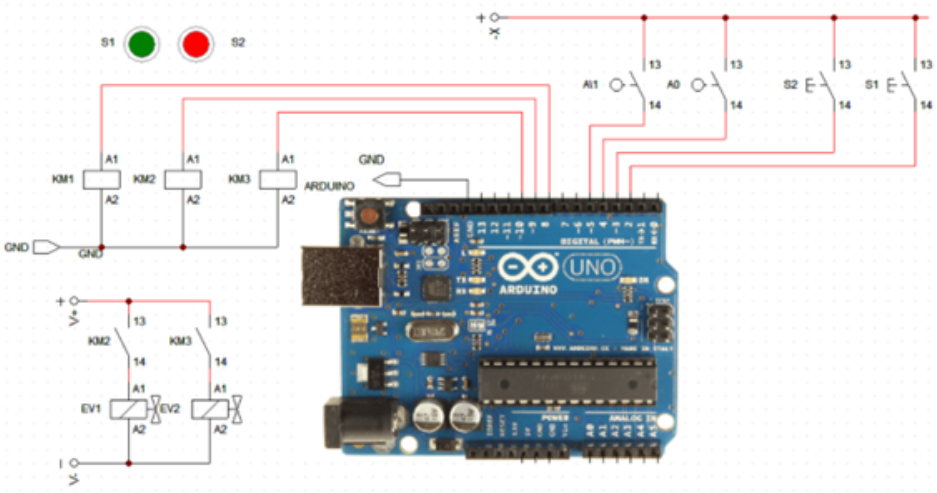
Parámetros del final de carrera



En CADe-SIMU para la conexión de componentes es importante tener en cuenta los pines que utiliza el programa en la tarjeta Arduino Uno como se indicó en la tabla 6.5. El relé KM1, se utiliza para poner en marcha o parar el circuito. Los relés KM2 y KM3, manejan las electroválvulas para el suministro del aire comprimido hacia el cilindro.

Figura 152

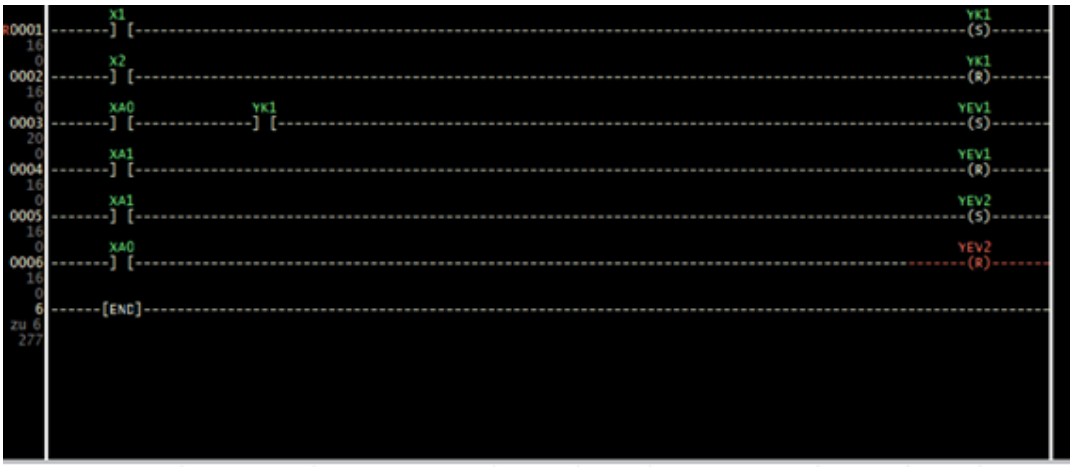
Diagrama de conexiones en CADe-SIMU para el control del cilindro



Finalmente, el diagrama Ladder en LDmicro, tomando en cuenta los usados en los programas PC SIMU y CADe-SIMU, se vinculan a los correspondientes pines del AVR328P y que corresponden a la tarjeta Arduino. como muestra la Figura 153.

Figura 153

Diagrama Ladder para controlar el avance y retroceso del cilindro



Nombre	Tipo	Estado	Pata del ...	Puerto ...	Nombre de Pin	Direccion	Tamano	Direccion ...
X1	entrada digital	0	4	PD2	PD2 PCINT18/INT0	0x29 (BIT2)	1 bit	
X2	entrada digital	0	5	PD3	PD3 PCINT19/OC2B/I...	0x29 (BIT3)	1 bit	
XA0	entrada digital	1	6	PD4	PD4 PCINT20/XCK/T0	0x29 (BIT4)	1 bit	
XA1	entrada digital	0	11	PD5	PD5 PCINT21/OC0B/T1	0x29 (BIT5)	1 bit	
YEV1	salida digital	1	15	PB1	PB1 OC1A/PCINT1	0x25 (BIT1)	1 bit	
YEV2	salida digital	0	16	PB2	PB2 _SS/OC1B/PCINT2	0x25 (BIT2)	1 bit	
YK1	salida digital	1	14	PB0	PB0 PCINT0/CLKO/ICP1	0x25 (BIT0)	1 bit	

El funcionamiento del control del vaivén del cilindro se puede observar ingresando al blog web del autor:

https://rafaelreinososrv.blogspot.com/

Ejercicios propuestos

1. Realizar el diagrama Ladder en LDmicro; el circuito de potencia, circuito de control eléctrico, diagrama Ladder y la interfase del Arduino, en CADe-SIMU; para el arranque estrella–triángulo de un motor trifásico de inducción con inversión de giro. Utilice los pulsadores y protecciones necesarias para el circuito.
2. Realizar la simulación con CADe-SIMU y Arduino para el control de dos semáforos en una intersección. Elabore también un HMI en PC-SIMU.
3. Implemente un Semáforo de control de tráfico y de paso peatonal, como se indica en la figura. El diagrama de control se realizará en LDmicro y el HMI respectivo en PC-SIMU.

Figura 154

Imagen de un semáforo de tráfico y peatones

4. En el ejercicio "Llenado de botes", realice las siguientes modificaciones:
   - Agregue un panel de prueba de fallas. Incluya pulsadores para todos los sensores.
   - La luz piloto debe funcionar cuando ocurra un fallo en los sensores B1 y B2.
   - Agregue una luz a piloto H3, que se active cuando haya fallas en los sensores B3 y B4.
   - Modifique el diagrama Ladder para que cumpla con las especificaciones pedidas.
5. Realice todos los diagramas y circuitos necesarios para el control de dos cilindros de doble efecto que trabajen en la siguiente secuencia: A+ B+ B- A-

Referencias

Asin, F. (24 de Octubre de 2021). *Componentes Electrónicos*. https://fjasin.wixsite.com/electronica-asin/arduino

Arranque estrella triangulo con Arduino. (s.f.). *Electrotec.* Recuperado el 12 de junio de 2022, de: https://electrotec.pe/blog/ArduinoEstrellaTriangulo

Castañeda, B. S. (14 de diciembre de 2015). fdocuments. https://fdocuments.mx/document/taller-pre-conocimiento-sobre-arduino.html?page=1

¿Cuántos tipos diferentes de Arduino hay? (s.f). Recuperado el 15 de abril de 2021, de: https://hacedores.com/cuantos-tipos-diferentes-de-arduino-hay/

Elecplc. (MAyo de 2016). Electplc. *Manual de uso para PLC con LDmicro*: http://electgpl.blogspot.com/2016/05/manual-de-uso-para-plc-con-ldmicro.html

Otiniano, C. (s.f.). *EDOCS.* https://qdoc.tips/problemaspropuestos-plc-pdf-free.html

Proyectos con Arduino. (s.f). Recuperado el 22 de mayo de 2022, de: http://proyectosconarduino.com/curso/arduino-uno-r3/

Westhues, J. (Febrero de 2009). *LDmicro*. https://github.com/LDmicro/LDmicro/blob/master/ldmicro/manual-es.txt

Programando Arduino como PLC

El presente libro trata de cómo programar la tarjeta Arduino utilizando diagramas Ladder, que es el lenguaje de programación mayoritariamente usado por los PLCs industriales. Arduino es una tarjeta de desarrollo muy versátil que se puede aplicar en proyectos de mediana complejidad y realizar funciones de un PLC de gama baja o un relé programable, pero con un costo muy inferior a un PLC comercial.

LDmicro es el software que se usa para programar y es de licencia libre. Este software es un editor de diagramas Ladder, un Compilador (genera archivos: HEX, sketch para trabajar en el IDE del Arduino y códigos para Pascal y varios compiladores C para PIC y AVR) y simulador en tiempo real.

Además, se incluyen en el texto HMI y simulaciones con Proteus, CADe-SIMU y PC-SIMU, estos dos últimos permiten interactuar en tiempo real con la tarjeta y también son software de licencia libre.

Para abordar la temática, el texto se ha organizado en 6 capítulos que tratan varios temas:

En el capítulo 1 se da la descripción del hardware y software utilizado: Arduino Uno, LDmicro, ISIS de Proteus, CADe-SIMU y PC SIMU.

El capítulo 2 detalla los circuitos con contactos, memorias, enclavamientos y funciones lógicas, manejo de entradas y salidas, memorias internas, configuraciones, simulación en LDmicro, compilación y simulación en Proteus.

En el capítulo 3 se describe el uso de los temporizadores. Los temporizadores que LDmicro posee son muy similares a la mayoría de los que tienen los PLCs. Esto hace la diferencia al programar Arduino con el IDE.

Los contadores y registros son otras de las funciones que LDmicro posee, y sus usos y aplicaciones se enfocan en el capítulo 4.

La lectura de señales analógicas y el uso de PWM en los PLCs modernos es muy común. El capítulo 5 abarca estos temas.

Finalmente, en el capítulo 6 se realizan aplicaciones comunes de control industrial. Mandos de motores trifásicos, ejemplos de procesos industriales y control de cilindros. Las interfaces usadas y la simulación hacen que se vea como con PLC real. Se realiza una amplia explicación de forma de generar códigos para el sketch y uso en el Arduino IDE y el uso con los programas CADe-Simu y PC SIMU.

En todos los temas se incluyen ejemplos y ejercicios para que el lector pueda practicar y desarrollar nuevos proyectos.

Los autores

ISBN: 978-9942-765-83-3

ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA