PRÁCTICAS DE AUTOMATIZACIÓN INDUSTRIAL

SIEMENS Simatic
National Instruments LabVIEW

MAGDA JUDITH MORALES TAVERA
ELKIN YESID VESLIN DÍAZ

PRÁCTICAS DE AUTOMATIZACIÓN INDUSTRIAL

SIEMENS SIMATIC
NATIONAL INSTRUMENTS LABVIEW

Magda Judith Morales Tavera
Elkin Yesid Veslin Díaz

DIRECCIÓN TÉCNICA

M. en C. Omar Lengerke Pérez

UNIVERSIDAD AUTÓNOMA DE BUCARAMANGA
ESCUELA DE CIENCIAS NATURALES E INGENIERÍA
FACULTAD DE INGENIERÍA MECATRÓNICA
BUCARAMANGA

2005

ÍNDICE

INTRODUCCIÓN

"No basta saber, se debe también aplicar. No es suficiente querer, se debe también hacer."

Johann Wolfgang von Goethe
Poeta y dramaturgo alemán

Con el propósito de contribuir en la formación de los futuros Ingenieros Mecatrónicos, la Universidad Autónoma de Bucaramanga invirtió una buena parte de su presupuesto en la adquisición de equipos de control industrial y otros materiales afines, el compendio de estos permitió la creación del Laboratorio de Automatización Industrial, un proyecto que se catalogaba indispensable por los directivos de la facultad. Si bien, la creación de un establecimiento donde se puedan realizar pruebas y ejercicios con equipos profesionales era importante, también lo era el desarrollo de guías que permitieran al estudiante acceder a él. Actualmente, encontrar manuales de este tipo es muy difícil, además, es imposible adaptar los pocos que existen a los materiales que disponen el laboratorio. Por eso, se vio la necesidad de crear este trabajo.

Este manual no sólo está dirigido a los estudiantes de Ingeniería Mecatrónica de la UNAB; profesionales, tecnólogos, y estudiantes de ingenierías afines pueden referirse a las prácticas aquí contenidas para complementar su conocimientos o sencillamente para aprender. Es importante que la persona que utilice esta obra, tenga formación en los temas que engloban el esquema de Automatización Industrial, como lo son, electrónica digital y análoga, programación y neumática.

Las prácticas aquí contenidas están divididas en dos capítulos, el primero, trata sobre la configuración y programación de PLC's de la serie Simatic 300 de Siemens, haciendo especial énfasis en la implementación de funciones especiales que incluyen los equipos de esta serie, y finalizando el capítulo con la introducción hacia una metodología de programación, cuya autoría se debe al Ing. Ricardo López, y que se puede aplicar a muchos autómatas programables de diferentes marcas, cerrando con una actividad final la cual integra todos los conceptos explicados. El segundo capítulo, se enfatiza en

el desarrollo de proyectos a través de LabVIEW, un sistema SCADA desarrollado por National Instruments. Las prácticas abordan temas sobre su entorno de programación, las herramientas que dispone y con las cuales se pueden desarrollar múltiples proyectos.

Cada práctica está compuesta por los objetivos, que definen los propósitos de la misma, un marco teórico que introduce al lector al Desarrollo Metodológico, el cual es el cuerpo de la práctica, donde el estudiante a través de un proceso aprende los conceptos básicos y resuelve un problema de forma asistida. Al final, asume un reto denominado Actividad Complementaria, cuyo propósito es el de llevar al estudiante más allá, las actividades complementarias exigen un tiempo diferente del dado para la solución de la práctica, son actividades que el estudiante tendrá de forma opcional, de acuerdo al nivel deseado de capacitación. Por otro lado están las Actividades previas, cuestionamientos y reflexiones que estudiante deberá hacer como introducción a la práctica, de esta forma, ampliará sus conceptos y podrá acceder con facilidad al Desarrollo Metodológico. Las prácticas tienen un desarrollo lineal, por lo que se exigirá para su elaboración haber realizado la actividad anterior.

El trabajo mostrado en esta práctica es el resultado de nuestra experiencia, teniendo como antecedentes la elaboración del Manual de Prácticas de Puerto Paralelo, junto con otros compañeros, que fuera desarrollado para la Universidad Autónoma de Bucaramanga y dirigido por el Ing. Eduardo Rincón, el cual es considerado material de referencia por muchos estudiantes; además hicimos parte del grupo de investigación que trabajó junto con el Dr. Dante Dorantes y demás docentes del Instituto Tecnológico de Monterrey Campus Estado de México (que de antemano agradecemos su amabilidad y acogida), en el desarrollo del libro AUTOMATIZACIÓN Y CONTROL, Prácticas de Laboratorio, publicado por la editorial Mc Graw Hill.

Esperamos que la obra aquí desarrollada llene todas sus expectativas.

Ingeniera Mecatrónica, Magda Morales Tavera
Ingeniero Mecatrónico, Elkin Yesid Veslin Díaz

Facultad de Ingeniería Mecatrónica
Universidad Autónoma de Bucaramanga

PARA EL DOCENTE

Si bien, el desarrollo de las prácticas por el estudiante, juegan un papel importante en la formación de su desempeño, es necesario crear parámetros que permitan evaluar ese trabajo. Es cierto que el criterio de evaluación difiere entre cada docente, sin embargo, se pueden crear parámetros o bases, a partir de los cuales, el docente puede aplicar su metodología de calificación. En el momento en que se ponga en marcha los manuales, la forma de evaluación sugerida para cada punto que compone la práctica es:

- ***Actividad previa***

La actividad previa es un requisito indispensable para acceder a la sesión, se sugiere que tenga un peso sobre el ponderado final del 10%. Se deben responder la totalidad de las preguntas de forma puntual (no texto inútil), y estas respuestas se deben entregar (antes de iniciar el desarrollo metodológico, al docente) en hojas tamaño carta a puño y letra del estudiante.

- ***Desarrollo metodológico***

Es la actividad que se realiza durante la sesión. Por esto, al concluir este ítem, el alumno debe notificar al docente la culminación del ejercicio y demostrarle su correcto funcionamiento. Se sugiere que tenga un peso sobre el ponderado final del 30%. El alumno durante todo el desarrollo, debe tomar nota de los puntos clave, esta información será la base del documento que forma parte del reporte final. La duración del desarrollo metodológico es de una sesión, un tiempo extra incurre en la nota.

- ***Actividad complementaria***

La actividad complementaria es el reto final que se le propone al alumno al culminar el desarrollo metodológico, en algunos casos, la extensión de este último puede ocasionar la necesidad de una sesión adicional (con o sin el docente a cargo), en donde el alumno no sigue pasos, sino que los desarrolla de la misma manera que se le indicó en la fase anterior. Cuando la solución al problema planteado sea la adecuada o cumpla los requisitos propuestos, se debe mostrar al docente el funcionamiento correcto como garantía del desarrollo. Se sugiere un peso ponderado del 10%.

Para concluir la práctica se debe documentar con un reporte escrito, el cual completa el 100% de la nota, este se entregará al docente en la sesión próxima, el documento debe contener:

- ***Portada***

Página informativa del documento que incluye: Título del trabajo, nombre (s) del (de los) autor (es), clase de trabajo realizado (Informe Práctica #), nombre con el título académico o cargo del docente encargado, precedido del término "Profesor", institución, facultad, departamento, división sección o área que representa el autor del trabajo, según el orden jerárquico interno de la entidad, ciudad y año[1].

- ***Materiales***

Se debe enlistar detalladamente cada uno de los elementos utilizados para dar solución al problema.

- ***Desarrollo metodológico de la actividad complementaria***

Es la descripción paso a paso de la solución al problema planteado, o al que el alumno mismo planteó, se deben incluir las imágenes y los gráficos necesarios para el buen entendimiento del trabajo por parte docente.

- ***Conclusiones***

Son las impresiones y pareceres por parte del estudiante sobre el problema solucionado, deben ser acordes a los objetivos propuestos en la práctica, concisos y de alto criterio.

Finalmente, la calificación dependerá de:

- La correcta implementación de los conocimientos adquiridos.
- El funcionamiento físico del proyecto.
- Concordancia entre el funcionamiento del proyecto y su planteamiento.
- Orden en el programa.

[1] ICONTEC, Compendio Tesis y otros trabajos de grado.

CD DIDÁCTICO

El CD didáctico es una recopilación de todas las prácticas desarrolladas en el manual. Su función, es la de ofrecer una perspectiva diferente a la dada por las prácticas, brindando de esta forma, una mejor alternativa al concepto de la guía realizada en un formato de documento de texto. Es accesible y de entorno amable. Adicional a las prácticas, ofrece imágenes del montaje final de la misma y referencias de los elementos (dícese de sensores y actuadores) que la conforman, como también, información adicional acerca de la teoría que abarca este manual.

El CD didáctico se convierte en una herramienta muy importante para la difusión de este manual, ya sea a través de un medio físico como un disco compacto, o a través de una autopista de información como lo es Internet.

CAPÍTULO 1
SIEMENS SIMATIC

- Configuración del PLC Siemens Simatic
- Lenguajes de programación
- Entradas y salidas digitales
- Operaciones lógicas con bits
- Lógica secuencial – uso de Flip-Flop's
- Temporizadores
- Transferencia de paquetes de datos
- Contadores y comparadores
- Funciones
- Método de programación estructurada
- Entradas y salidas análogas
- Trabajo final

PRÁCTICA 0
CONFIGURACIÓN DEL PLC SIEMENS SIMATIC

PRÁCTICA 0
CONFIGURACIÓN DEL PLC SIEMENS SIMATIC

1. OBJETIVO

Desarrollar habilidades para la configuración del hardware del PLC *Siemens Simatic S7-300.*

2. CONCEPTOS FUNDAMENTALES

Mencionaremos algunos datos técnicos y bondades del ***SIMATIC S7-300.***

El PLC SIMATIC S7-300 es una versión mejorada de la serie 200 de la misma compañía. Entre sus nuevas características está una memoria de programa de hasta 48KB de instrucciones. Un puerto de Profibus DP integrado. Es pequeño, potente, rápido y modular. Necesita solo 0.1ms para ejecutar 1024 instrucciones binarias.

Simatic es el único sistema de automatización actualmente en el mercado que permite materializar la integración total en automatización, es decir, nos permite no sólo el control automático de procesos, sino que también la integración entre componentes a través de redes industriales y sistemas de interfaz con el PC.

El sistema modular comprende varias CPU's para distintas exigencias, módulos de señal para entradas/salidas digitales y analógicas, módulos de función para contaje rápido, posicionamiento en lazo abierto y lazo cerrado así como módulos de comunicación para el acoplamiento a redes en bus. Fuentes de alimentación de carga y módulos de interconexión (interfaces) para configuración en varias filas.

3. ACTIVIDAD PREVIA

Antes de proceder al desarrollo de esta práctica es necesario que elabore el siguiente cuestionario:

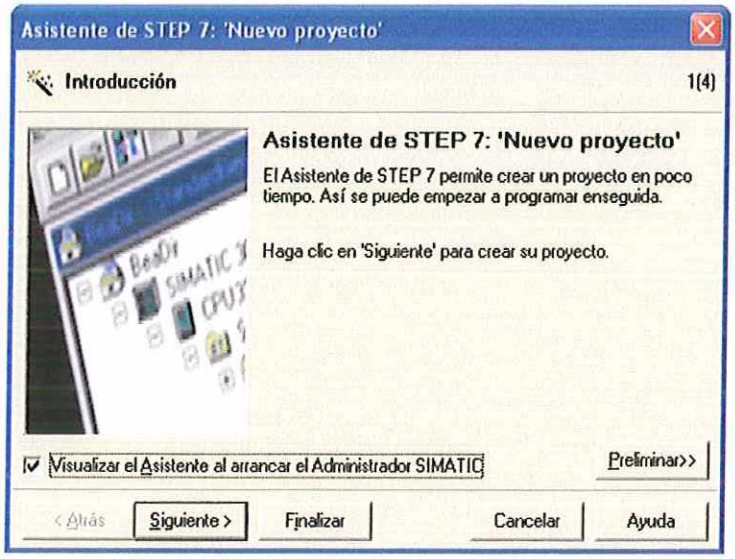
1) Qué es un PLC?
2) Cuáles son los componentes básicos de un PLC?
3) Qué fabricantes de PLC existen en la actualidad?
4) Qué módulos adicionales existen para el PLC Simatic S7-300?

5) Qué es la interface PG/PC?
6) Analice el documento extra que viene con esta práctica, **Conexiones al PLC**.

4. DESARROLLO METODOLÓGICO

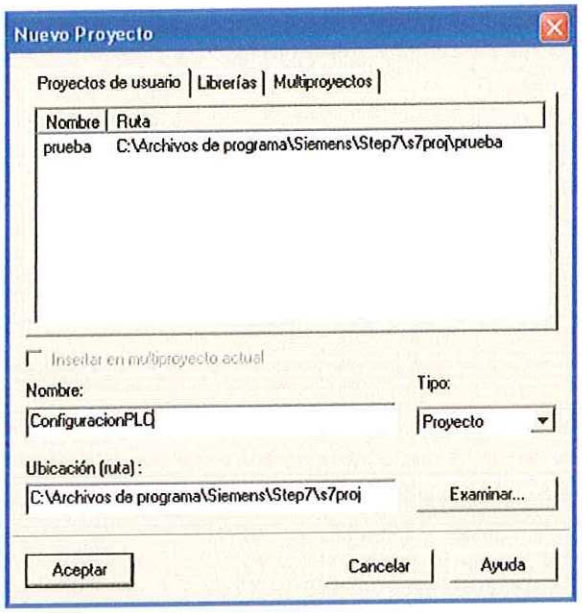
Abrir ***SIMATIC/Administrador Simatic***, el cual se encargará de la configuración de todos los elementos que componen al PLC para su correcto funcionamiento.

Dar clic en cancelar en la ventana emergente del Asistente de *Step7*.

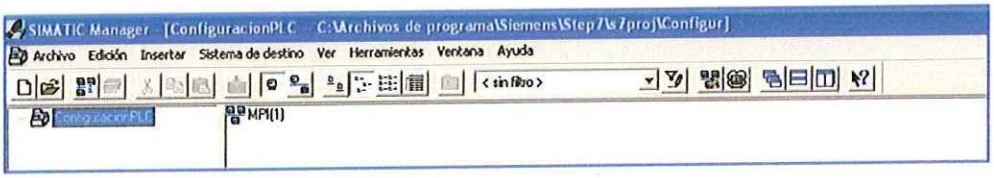


En esta ventana se le da al usuario la opción de configurar el PLC de forma automática, sin embargo se considera de vital importancia el aprender a configurar cada uno de los elementos para poder identificarlos, que es el objetivo principal de esta práctica. Ahora procederemos a realizar la **configuración manual del equipo.**

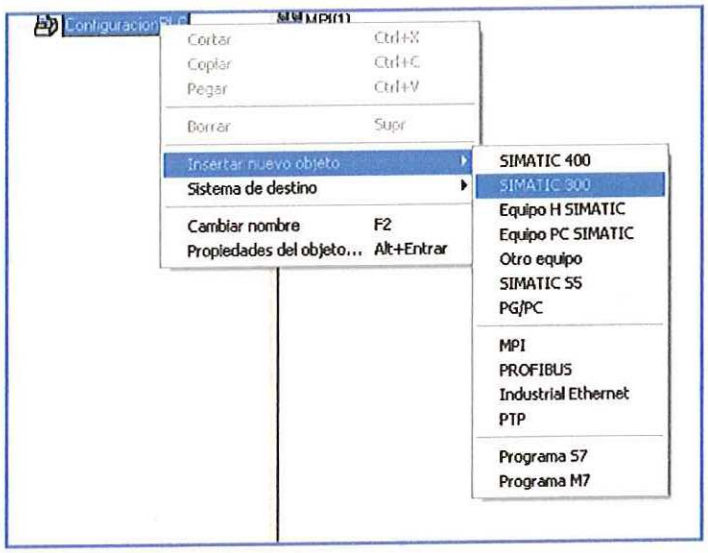
En el ***Administrador SIMATIC***, dar clic en ***Archivo/Nuevo***. Se abrirá una ventana ***Nuevo Proyecto***, en el submenú ***Proyectos de Usuario*** podrá ver la dirección donde se guardará su proyecto, como también podrá asignarle un nombre, verifique estas opciones como se indica a continuación y acepte la operación.



Al dar aceptar se abrirá el ***Simatic Manager***, observará su proyecto habilitado en el workspace.



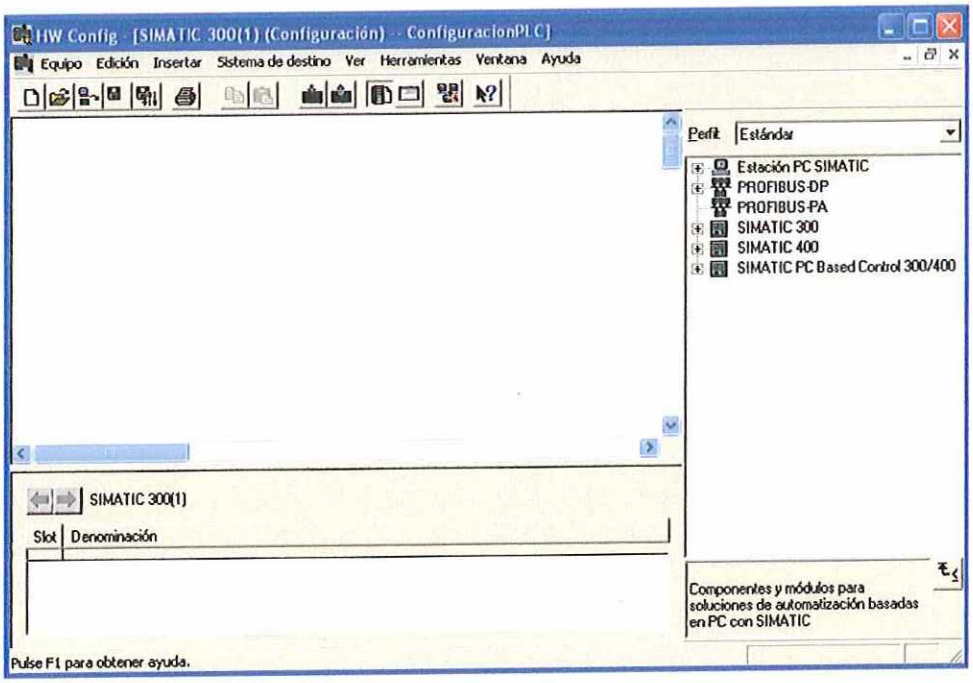
El PLC a configurar es un *Simatic S7-300* en nuestro caso. El último número define la serie del equipo. Para agregarlo al proyecto, de clic derecho sobre el icono que tiene el nombre que le asignó y elija ***Insertar Nuevo Objeto/SIMATIC 300***.



Al agregar debe observar que debajo de su proyecto aparece un nuevo icono, este representa al PLC que va a configurar.

De doble clic sobre ***SIMATIC 300(1)***, observe que en la ventana derecha aparezca un enlace llamado Hardware.

De doble clic sobre este nuevo icono ***Hardware***. El cual desplegará la ventana ***HW-Config***.



Antes de configurar el PLC es necesario que haya identificado sus componentes, para que tenga una idea de los elementos que agregará al

hardware.

Nota: *el orden de anexo de los componentes va de acuerdo a la instalación real que estos tengan en su mesa de trabajo. Puede referirse al orden aquí propuesto para hacer su propia configuración. Para agregar el elemento deseado se debe ver la referencia en la carcasa de cada uno de ellos.*

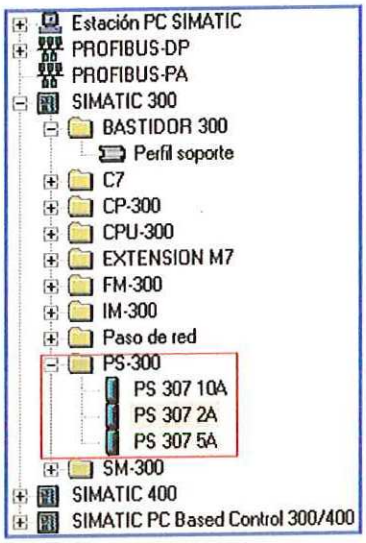
Pasos para la configuración del hardware del PLC:

a) Agregar el bastidor. Doble clic en ***SIMATIC 300***, Doble Clic en ***BASTIDOR 300***, seleccionar *Perfil soporte* y arrastrar hacia el área en blanco. Suelte el botón. Aparecerá un nuevo elemento sobre el área. Sobre este elemento se montará toda la configuración del PLC.

Nota: *SIEMENS tiene predeterminado para todos sus PLC que debe existir un* ***orden en la configuración*** *de los elementos, este orden es el siguiente:*

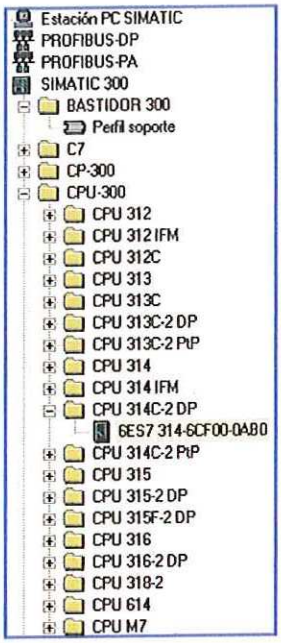
1) Power Supply (Fuente de poder).
2) CPU
3) Elementos externos como pantallas, este es un espacio que se puede dejar en blanco.
4) En adelante Módulos del PLC (I/O Digitales y Análogas, etc).

b) Agregar la fuente. En la misma carpeta ***SIMATIC 300***, doble clic en la capeta ***PS-300***. Seleccionar la fuente que corresponda a su PLC (En este caso ***PS 307 2A***). Arrástrela hacia la cuadrícula que cambia de color "casilla 1", indicando la posición que esta debe tomar.



c) Agregar la CPU. De la misma forma que los anteriores elementos. Ahora de doble clic en ***CPU-300***, ahora seleccione la CPU que corresponde a su PLC (Para nuestro caso de doble clic en ***CPU 314 C-2DP*** y arrastre el componente que despliega esta carpeta a la casilla 2).

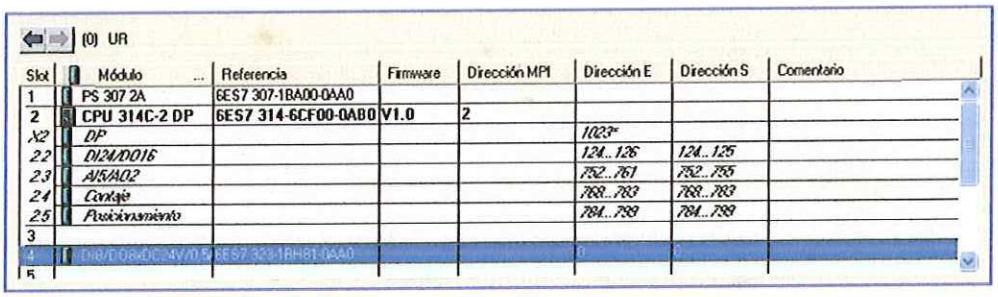
Nota: *A la ventana emergente, de nombre Propiedades – Interface Profibus DP de clic en cancelar, puesto que no se va a utilizar una interface de red.*
Este PLC tiene los módulos de entradas/salidas análogas y digitales predefinidos dentro de la CPU, es decir que estos módulos no necesitan ser configurados por aparte.



d) Agregar módulos de simulación. Módulos como el *SM 374*, constan de entradas/salidas digitales simuladas, este módulo cuenta con 16 bits los cuales pueden ser configurados de 3 formas:
1) 16 bit de salidas.
2) 16 bit de entradas.
3) 8 bits de entrada y 8 bits de salida.
Para poder hacer estos cambios, el módulo cuenta con una perilla para ajustar cualquiera de las 3 opciones.

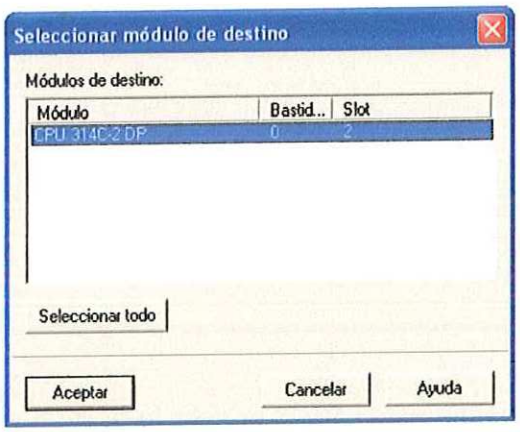
Se configurará el módulo para que simule 8 entradas y 8 salidas, en este caso la perilla debe estar en la parte central. Una vez hecho esto, en ***HW Config*** busque la carpeta ***SM 300/DI-DO 300/SM323DI8DO8XDC24V/0,5A***. Arrástrelo hacia la primera casilla de las que cambiaron de color "casilla 4".

Al haber desarrollado los pasos anteriores detenidamente; en la ventana ***HW-Config*** se debe apreciar una configuración como la mostrada a continuación:

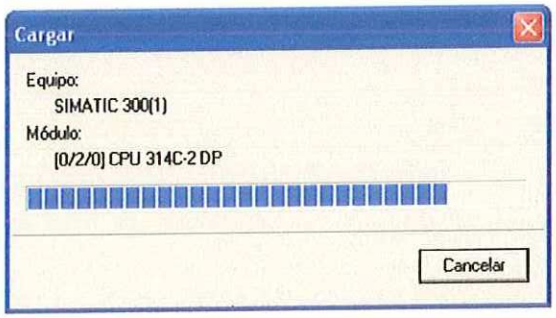


Una vez hecho esto, guarde "***Guardar***" y compile "***Guardar y Compilar***" estos cambios. Posteriormente verifique que el PLC está encendido y que la conexión **(Cable PG-PC)** esté. Cargue la configuración desarrollada a lo largo de esta practica a través del icono ***Cargar en módulo***.

Mientras se está cargando la configuración al módulo se desplegaran ventanas emergentes como ***Seleccionar módulo de destino*** y ***Seleccionar dirección de estación***, la cuales hay que aceptar teniendo en cuenta que la información que ellas contengan sea coherente.



Al aceptar las dos operaciones anteriores observaremos el mensaje en donde se muestra cuando la información se esta transfiriendo al módulo en la ventana emergente ***Cargar***.



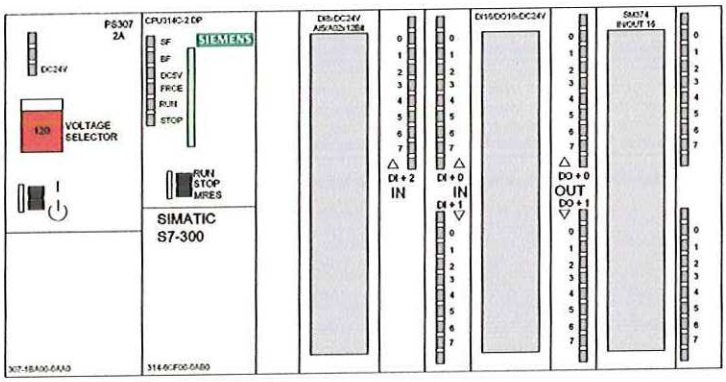
Una vez terminado este proceso, ubique la perilla de la CPU del PLC en ***Run*** y verifique que no se presentaron errores de configuración, esto significa que el aviso SF (System Failure) en el PLC no está encendido.

En caso de que el aviso *SF* se ilumine intermitentemente, ubique la perilla en ***MRES*** para resetear la memoria, esto borrará la configuración anterior, déjelo en ***STOP***, y revise nuevamente el programa. Una vez hecho esto vuelva a cargar la configuración desarrollada.

MATERIAL EXTRA No. 1

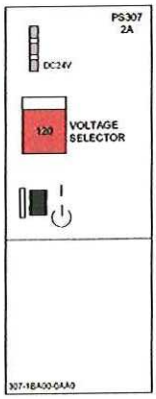
Conexiones del PLC

Antes de iniciar la práctica de configuración de Hardware del PLC, es necesario repasar un elemento que, a pesar de que no se hará todo el tiempo, es muy necesario conocerlo, se trata de las conexiones de alimentación eléctrica del PLC.

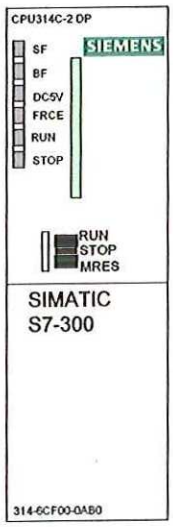


El PLC Siemens Simatic, se alimenta de una fuente de Corriente Alterna de 120 V ó 240 V dependiendo del modelo, además a esta alimentación, requiere de una conexión a tierra como en los computadores caseros. El PLC viene con su referencia de puesta a tierra, el cual se trata una lata metálica, sobre la cual este se sostiene, esta tierra se debe conectar con el polo a tierra del establecimiento para tener una referencia común.

Las conexiones eléctricas con puesta a tierra se reconocen por tener tres aberturas, y su cable de potencia correspondiente por tener sus tres respectivas conexiones, en el módulo de alimentación eléctrica del PLC (identificado por las siglas PS) están los tres espacios para conectar estos cables, L1, N y GND. Entre el L1 y N está el potencial requerido (120 V ó 240 V) y GND es la tierra. Ayúdese de un multímetro para identificar estos dos cables.



Una vez hecho esto conecte los tres cables al PLC y enciéndalo. La PS se encarga de suministrar una alimentación en Corriente Continua adecuada para el funcionamiento del equipo, esta se encuentra entre los terminales L y M (neutro) quienes suministran 24 VDC, note que la CPU tiene unos terminales iguales, conéctelos y así estará alimentada.



Con la alimentación ya establecida, es necesario conectar el cable PG-PC entre el computador (a través del puerto serial COM1 o COM2) y el PLC (en la bahía de conexión de la CPU denominada MPI).

PRÁCTICA 1
LENGUAJES DE PROGRAMACIÓN

PRÁCTICA 1
LENGUAJES DE PROGRAMACIÓN

1. OBJETIVOS

- ✓ Conocer los diferentes lenguajes de programación existentes para el desarrollo de proyectos en los PLC.
- ✓ Desarrollar algoritmos en los lenguajes de programación existentes.
- ✓ Implementar los diferentes lenguajes de programación en la elaboración de tareas propuestas.

2. CONCEPTOS FUNDAMENTALES

Lenguajes de programación

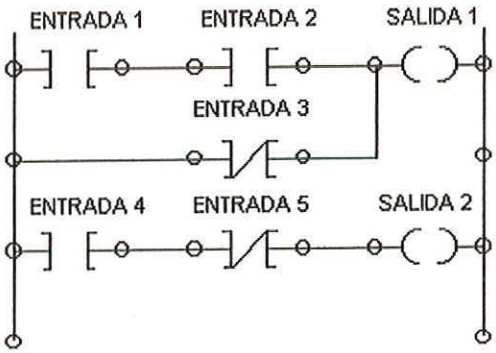
Los lenguajes de programación fueron creados para facilitar el diseño de programas para los PLC, para esto, se basan en recursos gráficos que describen funciones y características propias de este dispositivo, de tal forma, que con sólo mirarlo, se puede deducir que acción va a realizar y bajo que condiciones. Un lenguaje de programación va acompañado de un compilador, el cual se encarga de transformarlo a términos que un PLC si puede procesar (lenguaje de máquina) y luego, cargarlo a éste equipo.

Los programas hechos bajo cualquiera de estos lenguajes pueden ser modificados fácilmente, y una vez editados se pueden cargar nuevamente al equipo, el cual los interpreta y ejecuta las acciones de acuerdo a las órdenes, se pueden crear muchos programas distintos para que hagan lo mismo, el éxito depende del número de instrucciones o pasos que diseñe el usuario, entre más recursivo sea, un PLC se tiene que esforzar menos, esta habilidad sólo se desarrolla con el paso del tiempo a medida que se tenga mayor experiencia.

Los lenguajes de programación más usados para la programación de PLC son:

Lenguaje de Contactos (LD ó KOP)

También conocido como lenguaje de escalera (o ladder en inglés), su forma gráfica es similar a la usada por los electricistas para el diseño de esquemas de contacto, se basa en la activación de una salida a partir de la activación o desactivación de los contactos que la gobiernan.



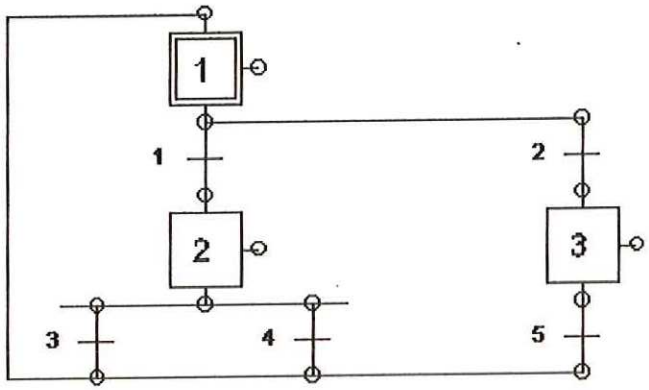
Para la imagen anterior la salida 1 se activará si las entradas 1 y 2 están conectadas, también se activará cuando la entrada 3 se desactive. En cambio la salida 2 sólo se activará si la entrada 4 esta conectada y la entrada 5 este desconectada.

Lenguaje por lista de instrucciones (IL ó AWL)

Es la forma más básica en la cual se puede programar un PLC, sus comandos se basan en palabras que representan acciones como activar o desactivar, cargar, etc. Se caracteriza también por ser la más potente ya que requiere de menos tiempo de procesamiento.

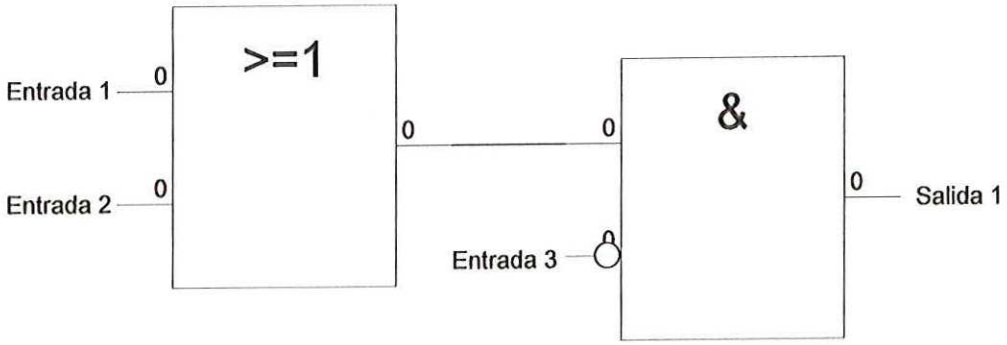
Grafcet

Es el llamado Gráfico de Orden Etapa Transición. Ha sido especialmente diseñado para resolver problemas secuenciales. Las acciones son asociadas a las etapas (cuadros) y las condiciones a cumplir en las transiciones (segmentos). Este lenguaje resulta enormemente sencillo de interpretar por operarios sin conocimientos de programación.



Lenguaje de funciones (FUP o FBD)

Lenguaje basado en operaciones lógicas booleanas, resulta especialmente cómodo de utilizar a técnicos habituados a trabajar con circuitos de compuertas lógicas.



3. ACTIVIDAD PREVIA

Antes de proceder al desarrollo de esta práctica es necesario que elabore el siguiente cuestionario:

1) ¿Qué es un lenguaje de programación?
2) ¿Para qué son útiles los lenguajes de programación?
3) ¿Cuáles son las funciones lógicas existentes en el álgebra booleana?
4) ¿Qué lenguajes de programación maneja el PLC Simatic 300?

4. DESARROLLO METODOLÓGICO

Nota: *para el desarrollo de esta actividad es necesario haber elaborado la Práctica 0, Configuración del PLC Siemens Simatic.*

Abrir ***SIMATIC/Administrador Simatic***, cree un nuevo proyecto con el nombre *Práctica 1.*

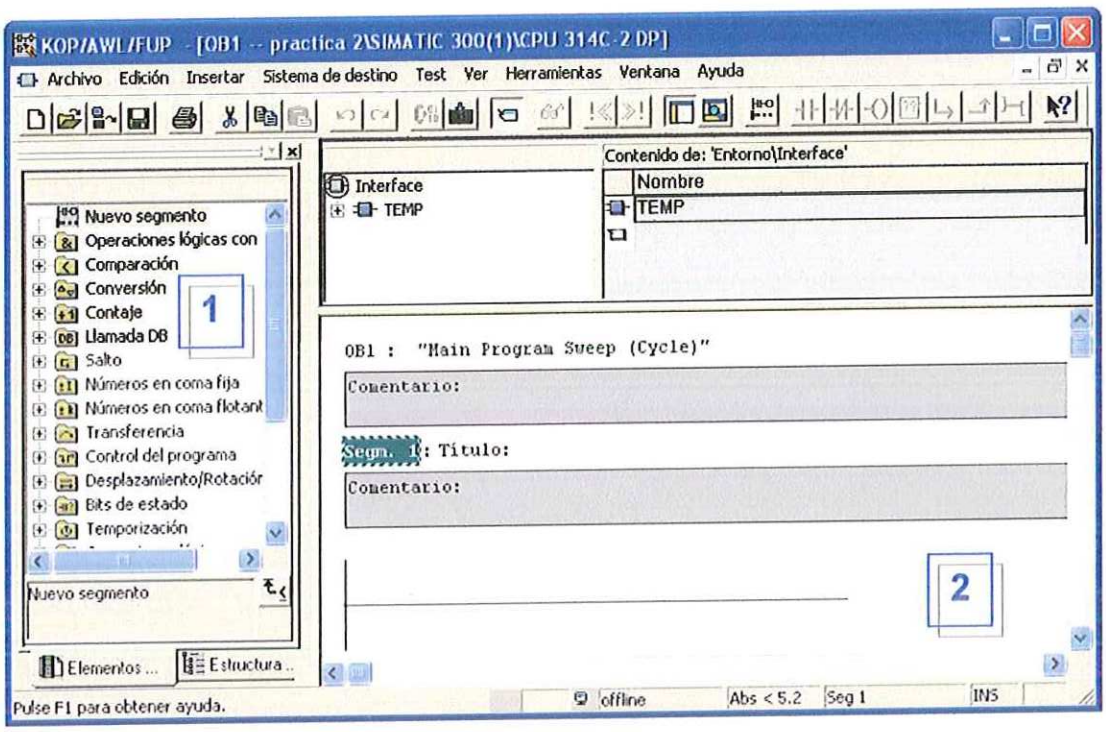
En la ventana ***SIMATIC Manager***, de doble clic en el icono *Práctica 1* y así consecutivamente hasta encontrar un icono llamado ***Bloques***, selecciónelo, y ahora de doble clic sobre ***OB1***, abrirá así un nuevo entorno.

Nota: *OB1 (Organization Block 1)es el bloque principal de programación, algo así como un Main en lenguaje C, aquí deberá estar almacenado el programa principal del proyecto,como soporte existen otros bloques que también son importantes, como los* ***FB*** *(Funcion Block), los* ***DB*** *(Data Block) y los* ***FC****, los cuales se explicaran en otras prácticas.*

En esta nueva ventana SIMATIC ofrece un entorno que le facilitará el ***desarrollo de proyectos para el PLC***, la ventaja que tiene es que todos los proyectos que realice aquí le servirá para cualquier máquina de este tipo de la compañía Siemens. Este entorno permite el desarrollo de programas en cinco tipos diferentes de lenguaje *AWL, KOP, FUP, GRAFCET y HIGRAPH* cada uno de estos, ofrece un listado de funciones (excepto *AWL*) que realizan operaciones específicas, muchas de las cuales son similares a las existentes en las compuertas y chips de lógica secuencial, como FLIP-FLOPS, contadores, temporizadores, etc.

Puede compilar los programas diseñados, y si existen errores, los señala, de tal forma que no permitirá cargar el programa al PLC hasta que no esté totalmente corregido. Una vez cargado permite también la visualización de su

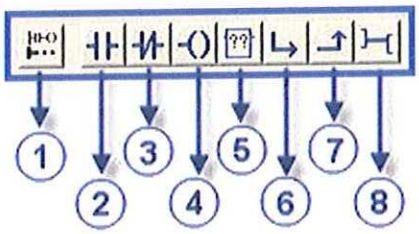
funcionamiento desde el PLC.



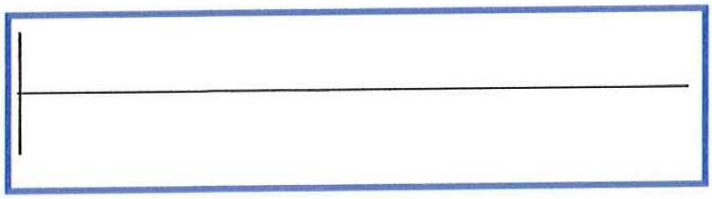
La ventana está dividida en 3 espacios, de los cuales (son más importantes) el primero (1) ofrece el listado de funciones disponibles para la elaboración de los programas y el segundo (2) es el entorno de trabajo, donde podrá diseñar los proyectos. Como todos los programas, tiene una barra de herramientas y un Menú, la barra de herramientas varía dependiendo del lenguaje seleccionado.

- **LENGUAJE KOP**

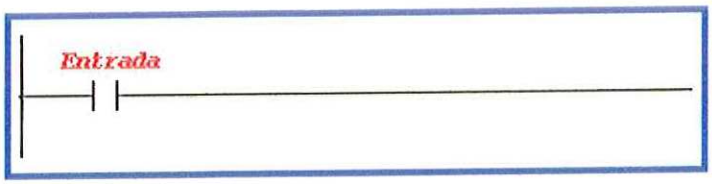
Para seleccionar el lenguaje KOP en el Menú ***VER/KOP***. Generalmente esta opción es la predefinida por el programa. La siguiente es la barra de herramientas que se despliega cuando se desea programar en lenguaje KOP.



1. Inserta el segmento de trabajo.

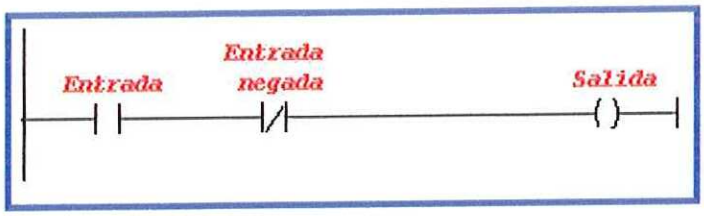


2. Agrega al segmento de trabajo una bobina normalmente abierta, la cual puede representar entradas, salidas y espacios de memoria. ***Es análoga a la función YES*** de las compuertas lógicas.

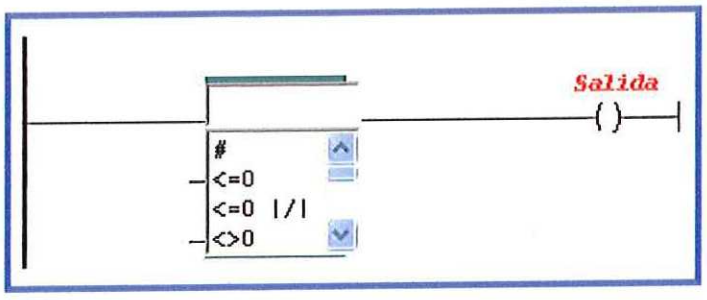


3. Agrega al segmento de trabajo una bobina normalmente cerrada, la cual puede representar entradas, salidas y espacios de memoria. ***Es análoga a la función NOT*** de las compuertas lógicas.

4. Inserta una salida, la cual se activará siempre y cuando se cumplan las condiciones establecidas por las bobinas.

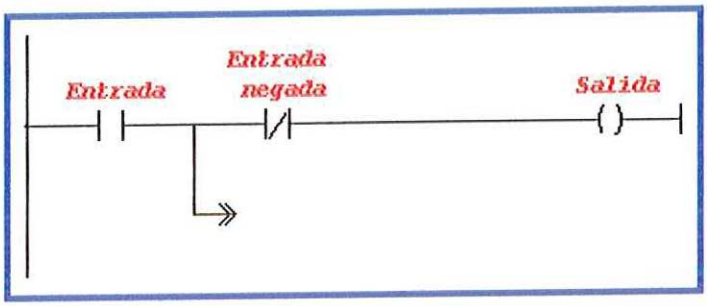


5. Inserta una función predeterminada del programa.

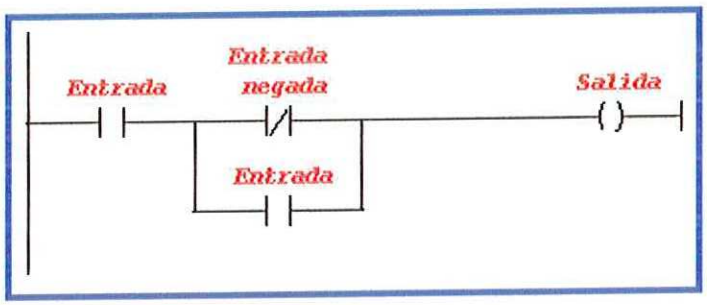


6. Crea un camino adicional al segmento, estableciendo una segunda opción para que la salida se pueda activar, y en la cual se pueden agregar elementos de la misma forma como se haría en la línea principal, ***es análoga***

a la función OR. Para crear un nuevo tramo es necesario señalar el lugar donde empieza, en este caso entre las bobinas normalmente cerrada y abierta.



7. Permite cerrar el camino creado en la opción 6, es necesario especificar el lugar donde se debe cerrar el segmento. En este caso en el espacio entre la bobina normalmente cerrada y la salida.

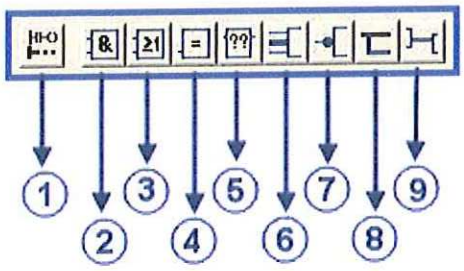


8. Conecta una entrada con una salida.

- **Lenguaje FUP**

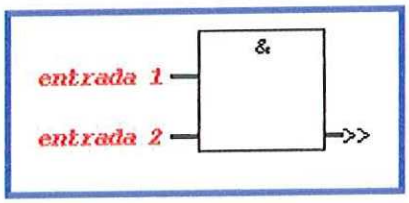
Para seleccionar el lenguaje FUP en el Menú ***VER/FUP.***

La siguiente es la barra de herramientas que se despliega cuando se desea programar en lenguaje FUP.

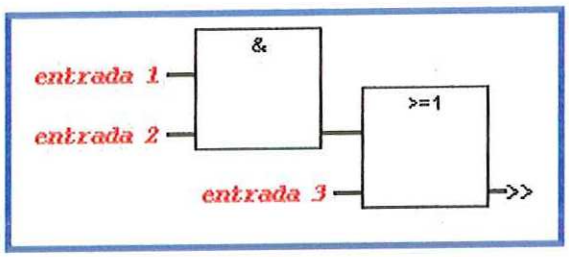


1. Habilita un espacio de trabajo.

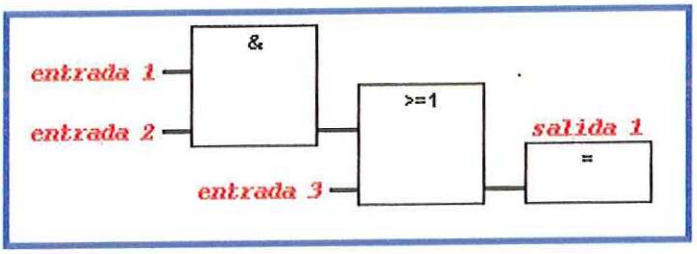
2. Inserta la función AND **(&)**.



3. Inserta la función OR **(>=1)**.

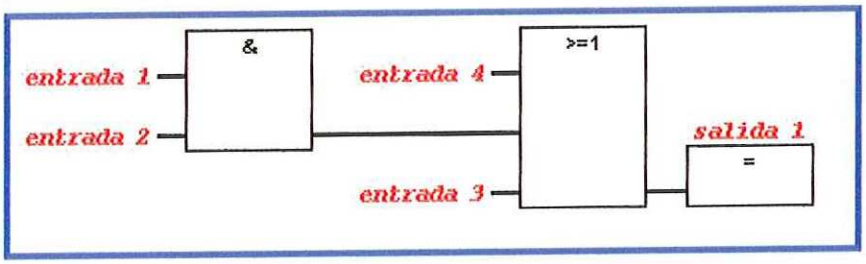


4. Habilita una salida la cual será regida por la lógica combinacional que la precede **(=)**.

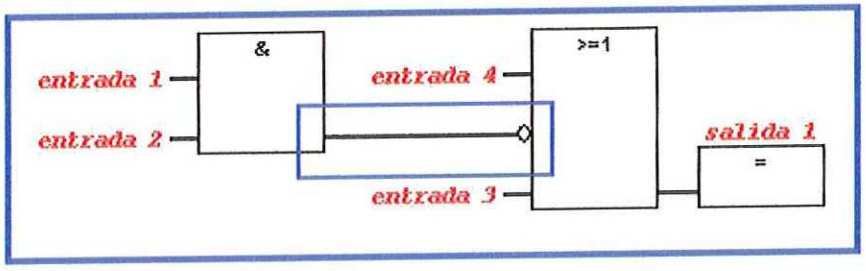


5. Inserta una función predeterminada del programa.

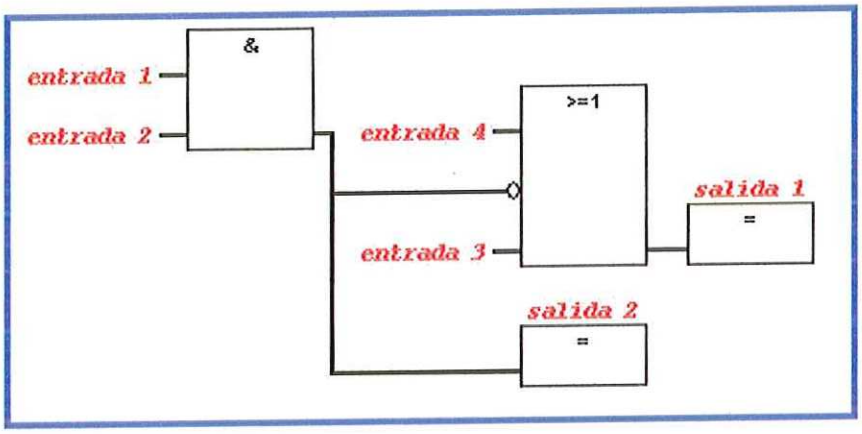
6. Agrega una tercer entrada a cualquiera de las funciones.



7. Niega una entrada seleccionada, es análoga a la función bobina normalmente cerrada del lenguaje KOP.



8. Crea un nuevo tramo, permitiendo el uso de una misma señal en otras instancias. En este caso la respuesta de la compuerta AND habilita una salida (salida 2) directamente, y a su vez actúa sobre la compuerta OR que rige a la salida 1.



- **Lenguaje AWL**

Para seleccionar el lenguaje AWL en el Menú ***VER/AWL***.

Este lenguaje no utiliza simbología, se basa en programación mediante caracteres. Cada línea de instrucción representa una condición para activar o desactivar una salida.

- Para activar una salida cuya condición este regida por una sola entrada se escribe en el espacio de trabajo:

U	A	***entrada 1***
=	E	***salida 1***

El signo = hace referencia a la salida sobre la cual se desea aplicar todas las condiciones que son necesarias para su activación. ***U*** es el análogo a ***AND*** en un lenguaje de programación.

***Nota: A** indica una entrada del PLC, dependiendo del idioma puede cambiar a la letra **i** (idioma inglés). De igual manera sucede con la salida representada por la letra **E** (en alemán) la cual puede variar a la letra **Q** usada en inglés. Tome atención del lenguaje con el que el Administrador Simatic esté trabajando.*

- Para activar una salida cuya condición este regida por una entrada negada se escribe en el espacio de trabajo:

```
UN    A    entrada 1
=     E    salida 1
```

UN significa entrada negada.

- Para activar una salida cuya condición este regida por dos entradas se escribe en el espacio de trabajo:

```
U     A    entrada 1
U     A    entrada 2
=     E    salida 1
```

- Para activar una salida cuya condición este regida por dos entradas y una esta negada cualesquiera que sea, se escribe en el espacio de trabajo:

```
UN    A    entrada 1
U     A    entrada 2
=     E    salida 1
```

- Si para activar una salida cuya condición esté regida por la activación de una u otra entrada (OR) se usa el siguiente código:

```
O     A    entrada 1
O     A    entrada 2
=     E    salida 1
```

O es el análogo a OR en un lenguaje de programación.

- Un grupo de condiciones para la activación de un salida se puede establecer de la siguiente forma:

```
UN    A     entrada 1
U     A     entrada 2
O
U     A     entrada 3
U     A     entrada 4
=     E     salida 1
```

El código anterior consta de dos condiciones separadas por la letra ***O***, lo que significa que la salida se activará si la entrada 1 (negada) y la entrada 2 están activas o si las entradas 3 y 4, están activas.

- Observe el siguiente código:

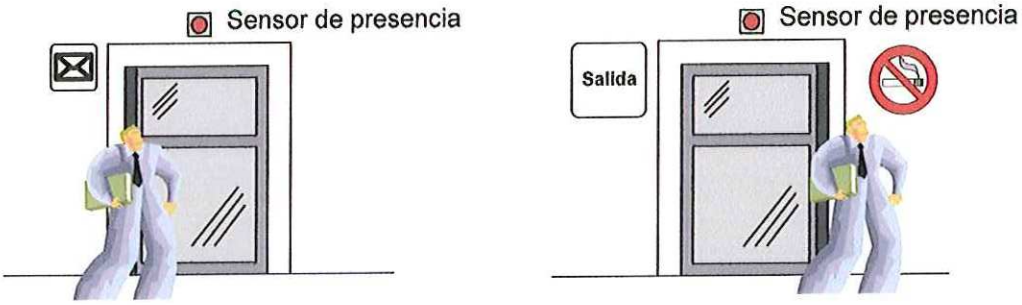
```
U(
ON    A     entrada 1
O     A     entrada 2
)
U     A     entrada 3
U     A     entrada 4
=     E     salida 1
```

Son tres condiciones de encendido para una salida, pero una (encerrada entre paréntesis) consta de dos opciones.

Las funciones propias del PLC tienen su versión en AWL, para poder utilizarlas es necesario conocer su estructura y el ordenamiento de los datos que se requiera, esta información la puede encontrar en la ayuda que suministra el programa, así como también otras instrucciones o comandos propios de este lenguaje de programación.

Ahora se usarán los conceptos aquí utilizados para resolver un problema.

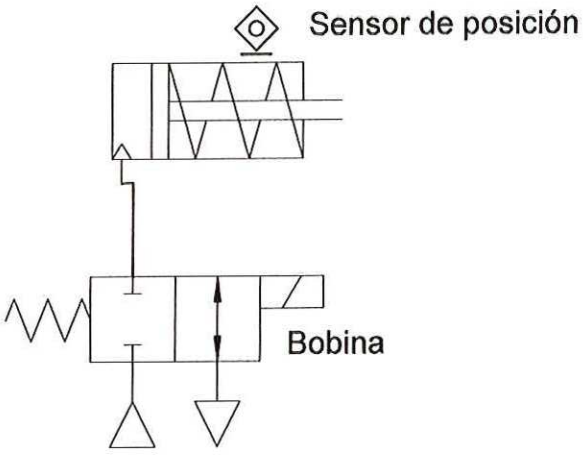
La entrada de una oficina de correos es una puerta automática accionada por un pistón neumático, muy similar a las existentes en los autobuses. Para abrir la puerta, el sistema cuenta con un sensor de presencia de tipo on-off a ambos lados de ella, uno avisa la llegada de un cliente, mientras el otro notifica su posterior salida. Existe un tercer sensor que notifica si la puerta esta abierta totalmente. ¿Cómo sería el control de la puerta hecho por un PLC?



Obviamente utilizar un solo PLC para controlar una sola puerta sería bastante excesivo. Sin embargo, puede ser útil si en el mismo edificio existe una mayor cantidad de sistemas del miso tipo. Y no sólo eso, se podría usar también para controlar el encendido de las luces, el accionamiento de las bandas transportadoras que llevan las cartas, etc.

Es necesario analizar el problema antes de desarrollar cualquier programa, un esquema gráfico sería muy útil, este esquema le permite conocer la situación del entorno y a partir de su análisis desarrollar una solución. Analizando el texto propuesto y las imágenes que lo acompañan se puede conocer que:

El sistema cuenta con tres sensores dos de presencia de tipo on-off, y uno de posición para el pistón que acciona la puerta, un pistón y su correspondiente válvula de accionamiento. El pistón a utilizar sería de simple efecto y la válvula que lo controlaría sería una 2/2 con accionamiento eléctrico y retorno por muelle (si existe en el mercado). El siguiente diagrama muestra el equipo a utilizar.



Es necesario determinar el funcionamiento del sistema, para el caso, cuando una persona se acerque a la puerta, ya sea de entrada o de salida, el sensor

de presencia de la puerta la detecta, inmediatamente manda la señal al PLC el cual la procesa de acuerdo al programa diseñado, activa la bobina de la válvula y el pistón sale, abriendo la puerta. Una vez el sistema no detecta a la persona, desactiva la bobina cerrando la alimentación de aire al pistón, ocasionando que este se regrese y cierre la puerta, quedando en espera de un nuevo ingreso o salida del edificio.

Nota: *Por ahora no es necesario desarrollar el programa en la ventana analizada anteriormente, sin embargo, la puede utilizar para hacer el esquema (de tal forma que se familiarice con el programa), procurando* ***no salvar los cambios realizados****.*

Se hará el programa en lenguaje KOP. Para esto es necesario que este habilitado en el Menú ***VER/KOP.***

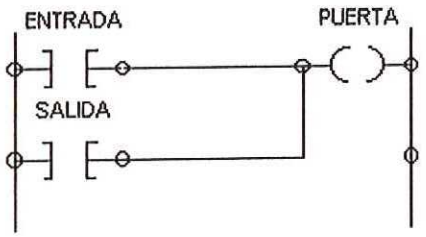
Las entradas del sistema son: dos sensores de presencia, uno para la entrada y otro para la salida. Y un sensor de posición. La característica de estos sensores es su condición de on-off, la cual emula dos estados encendido y apagado, La salida del sistema sería la bobina de la válvula que controla al pistón.

El programa sería el siguiente:

1. Para activar la puerta a la entrada del edificio.

Esta condición establece que la salida (**Puerta**) que controla a la puerta sólo se activará cuando el sensor de presencia a la entrada (**Entrada**) detecta a una persona.

2. Para activar la misma puerta a la salida del edificio.



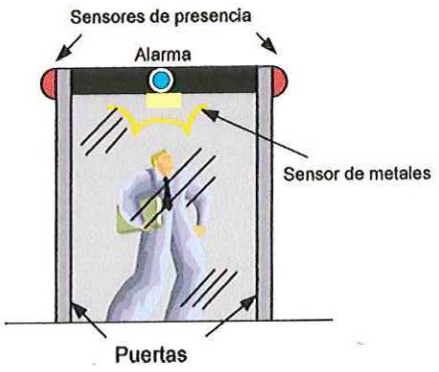
Ahora la salida (**Puerta**) se puede accionar si el sensor de presencia detecta a una persona que quiera salir del edificio. La conexión tipo OR realizada en el programa permite a la puerta abrirse en dos condiciones, si cualquiera se cumple la puerta se abrirá, lo mismo sucederá si las condiciones se cumplen a la vez, caso que sería muy apropiado porque el sistema seguiría a la persona mientras pasa a través de la puerta, cerrándola sólo hasta que este por fuera del campo de visón del sensor de presencia. Con un no-cumplimiento de las condiciones sencillamente la puerta permanecería cerrada.

Vale notar que en ningún momento se usó el sensor de posición de la puerta, sin embargo esto no lo hace inútil, podría ser útil para detectar una posible falla en le sistema, por ejemplo, una obstrucción que los sensores de presencia no puedan detectar, el PLC podría observar esta falla y emitir cualquier aviso.

Realice el mismo programa en los otros lenguajes de programación disponibles en el PLC.

5. ACTIVIDAD COMPLEMENETARIA

Para el problema anterior se ha diseñado un sistema anti-robo, el cual fue diseñado a partir de un sensor detector de metales, sin embargo para implementarlo, fue necesario diseñar un nuevo sistema de entrada y salida. El nuevo sistema es similar al que utilizan los bancos, cuando una persona desee entrar o salir la puerta correspondiente se abre, una vez esté adentro, el detector de metales hará su trabajo, si no detecta algo, permite abrir la otra puerta, de lo contrario, deja la puertas cerradas y activa un alarma. ***Nunca las dos puertas están abiertas a la vez.*** Las puertas son del mismo tipo que las usadas en el trabajo del *Desarrollo Metodológico*. Enumere la cantidad de elementos necesarios y diseñe su solución en los tres lenguajes. ***Nota:*** *el sistema requerirá de más salidas.*



PRÁCTICA 2
ENTRADAS Y SALIDAS DIGITALES

PRÁCTICA 2
ENTRADAS Y SALIDAS DIGITALES

1. OBJETIVOS

- ✓ Identificar los elementos y sus respectivas conexiones.
- ✓ Desarrollar habilidad en la manipulación de entradas y salidas digitales en lo referente a: identificación del módulo, direccionamiento y programación.
- ✓ Aprender a etiquetar direcciones, para así facilitar la programación y el diseño de soluciones de problemas.

2. CONCEPTOS FUNDAMENTALES

ENTRADAS DIGITALES

En las entradas digitales que encontramos en los PLC's se pueden conectar captadores de tipo todo o nada como finales de carrera, pulsadores y sensores. Estos trabajan con señales de tensión, por ejemplo cuando por una vía llegan 24 voltios se interpreta como un "1" y cuando llegan cero voltios se interpreta como un "0".

- El proceso de adquisición de la señal digital consta de varias etapas:

  Protección contra sobre tensiones
  Filtrado
  Puesta en forma de la onda
  Aislamiento por optoacoplador.

SALIDAS DIGITALES

Un módulo de salida digital permite al PLC actuar sobre los preaccionadores y accionadores (bobinas, motores, luces, etc.) que admitan ordenes de tipo todo o nada. El valor binario de las salidas digitales se convierte en la apertura o cierre de un relé interno del PLC.

- El proceso de envío de la señal digital consta de varias etapas:

  Puesta en forma
  Aislamiento
  Circuito de mando (relé interno)
  Protección electrónica
  Tratamiento cortocircuitos

El PLC Simatic 314C-2DP tiene a su disposición en el módulo de entradas y salidas digitales 16 entradas y 16 salidas; a su vez el módulo análogo cuenta con 8 entradas extras. También cuenta con un módulo simulador de entradas y salidas digitales que permiten representar elementos.

3. ACTIVIDAD PREVIA

Antes de proceder al desarrollo de esta práctica es necesario que elabore el siguiente cuestionario:

1) ¿Qué es un bit, byte, word y double word?
2) ¿Que tipo de elementos existentes en el laboratorio se pueden utilizar con las entradas y salidas digitales? Nómbrelos y dé una breve explicación de cada uno (que es, como funciona...).
3) Examine el material extra sobre la **Conexión de sensores**. Nota: refiérase a la sección 4, componentes.
4) Examine el material extra sobre la **Alimentación del módulo de entradas y salidas digitales**.

4. LISTA DE MATERIALES

1) Dos sensores de presencia inductivos.
2) Un cilindro de simple efecto.
3) Una electroválvula de accionamiento de una bobina con retroceso por muelle.

5. DESARROLLO METODOLÓGICO

Nota: *para el desarrollo de esta actividad es necesario haber elaborado las Práctica 0 y 1.*

Abrir ***SIMATIC/Administrador Simatic***, cree un nuevo proyecto con el nombre *Práctica 2.*

Para poder utilizar las entradas y salidas digitales, Simatic le asigna direcciones, para esto, las divide en paquetes de bytes (8 bits) y le da a cada una un número decimal como dirección, para este PLC hay un total de 3 bytes de entradas (24 bits de entradas en total) y 2 bytes de salidas (16 bits de salidas en total). **Para identificar las direcciones asignadas al módulo,** es necesario ir al ***HW Config*** y referirse a la ventana de configuración, el módulo a utilizar es el ***DI24/DO16*** y sus direcciones correspondientes se muestran al frente, las de las entradas son las referenciadas en la columna *Dirección E* y para las salidas en la columna *Dirección S*.

Slot	Módulo	...	Referencia	Firmware	Dirección MPI	Dirección E	Dirección S	Comentario
1	PS 307 2A		6ES7 307-1BA00-0AA0					
2	CPU 314C-2 DP		6ES7 314-6CF00-0AB0	V1.0	2			
X2	DP					1023*		
2.2	DI24/DO16					124..126	124..125	
2.3	AI5/AO2					752..761	752..755	
2.4	Contaje					768..783	768..783	
2.5	Posicionamiento					784..799	784..799	
3								
4	DI8/DO8xDC24V/0.5		6ES7 323-1BH80-0AA0			0	0	
5								

El hecho de que a un byte, como por ejemplo, el de salidas, se le haya asignado la dirección 124, permite que cada uno de los 8 bits que lo componen tenga un nombre o una dirección propia que lo haga diferenciable no sólo en su propio grupo, sino también de otro byte de salidas. En tal caso ***los bits que componen el byte 124 tienen establecida las direcciones***:

Bit	Dirección
0	124.0
1	124.1
2	124.2
3	124.3
4	124.4
5	124.5
6	124.6
7	124.7

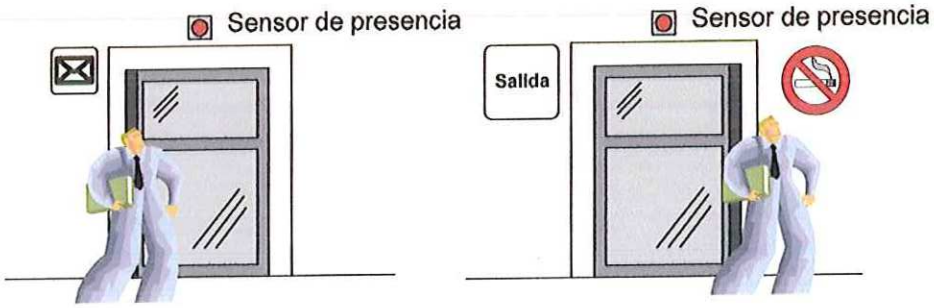
Aplicando también la misma distribución para los demás bytes de entrada (125 y 126) y los de salida (124 y 125).

Nota: *estas direcciones son predefinidas, por lo que pueden variar entre diferentes PLC's, a su vez se pueden modificar según las necesidades del usuario.*

Ahora se implementará el ejercicio de la práctica anterior haciendo tanto

montaje como programa, de esta manera se estará retro-alimentado los conceptos anteriores. Aquí está otra vez:

La entrada de una oficina de correos es una puerta automática accionada por un pistón neumático, muy similar a las existentes en los autobuses. En el momento en que una persona desea ingresar a la oficina existe un sensor (Sensor inductivo 1) que lo detecta, igualmente en el momento que alguna persona desee salir existe otro sensor (Sensor inductivo 2) ubicado dentro de la oficina.



Antes de continuar con esta parte, es necesario que conecte los sensores y la bobina de la válvula al módulo de entradas y salidas digitales. Las alimentaciones y referencias a tierra correspondientes realícelas tal como lo investigó en la *Actividad preliminar, pregunta 3.* **Examine si los módulo de entradas y salidas digitales están alimentados.**

Elemento	Número de cables	Conexión	
Sensor (1,2)	3	Cable 1	GND
		Cable 2	24 V
		Cable 3	Señal
Válvula	2	Negro	GND
		Rojo	Salida

Recomendamos que se realicen las conexiones de entradas y salidas según la referencia de la siguiente tabla:

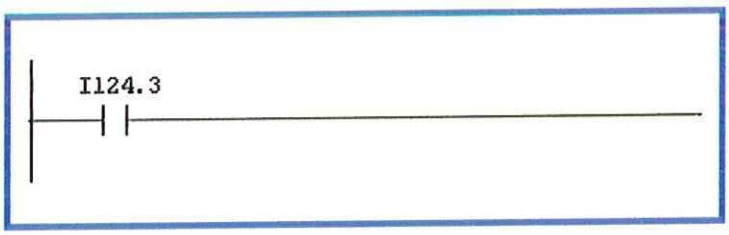
Elemento	Bit	E/S
Sensor Inductivo 1	124.3	Entrada
Sensor Inductivo 2	124.4	Entrada
Válvula	125.0	Salida

Ya conectados los elementos se procederá a realizar el desarrollo paso a paso del programa.

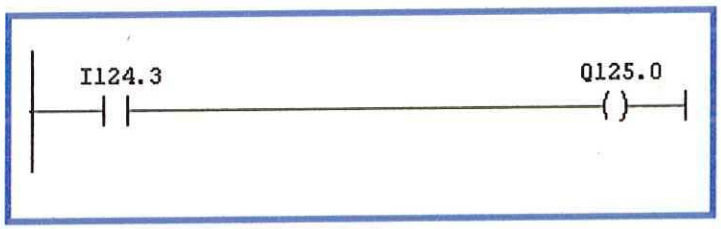
Desde el icono OB1 ingrese al entorno de trabajo para diseñar programas, con sólo dar doble clic. Retomando los conceptos de la *Práctica 1* se diseñará el programa utilizando el lenguaje KOP.

Pasos para el desarrollo del programa:

1) En el segmento de trabajo se ubica el ***sensor de presencia 1*** (normalmente abierto), para direccionarlo basta con dar clic sobre él y escribir la letra I (que especifica que se trata de una entrada) y la dirección donde el sensor esta conectado.

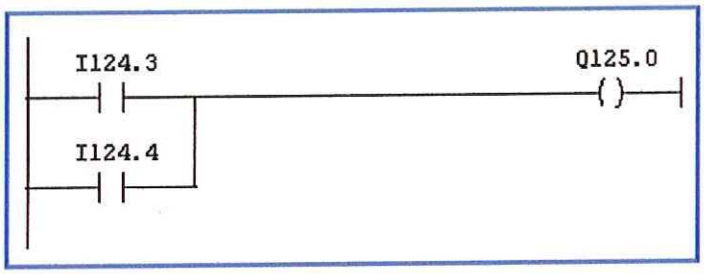


2) Ahora se ubicará la ***bobina de la válvula***, en este caso se especifica con la letra Q (indicando que se trata de una salida), y se direcciona de igual manera.

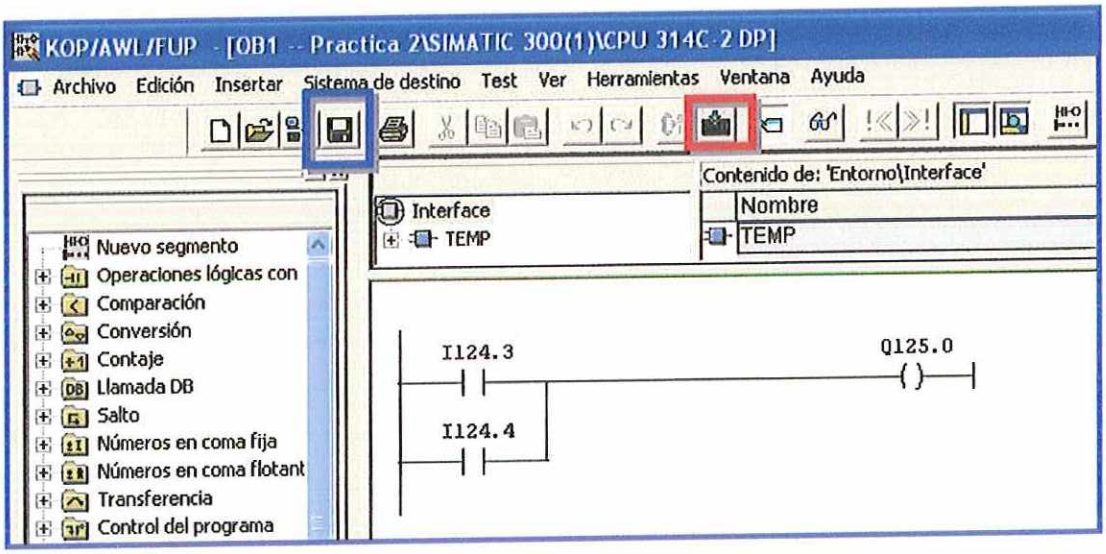


3) De forma paralela al sensor 1, se agrega el ***sensor de presencia 2***, como se muestra en la siguiente gráfica.

   Recordando que esta disposición se debe a que la puerta se abrirá si hay una persona entrando, o una persona saliendo o la condición en que las dos estén al mismo tiempo.



4) Después de haber finalizado el programa se guarda en el proyecto con el icono ***Guardar*** o través del menú ***Archivo/Guardar***. Hecho esto, se procederá a su descarga en el módulo con el icono ***Cargar en módulo***



5) Para facilitar el entendimiento de los programas desarrollados se cuenta con una forma de documentar las variables del proyecto por medio del menú *herramientas/tabla de símbolos*.

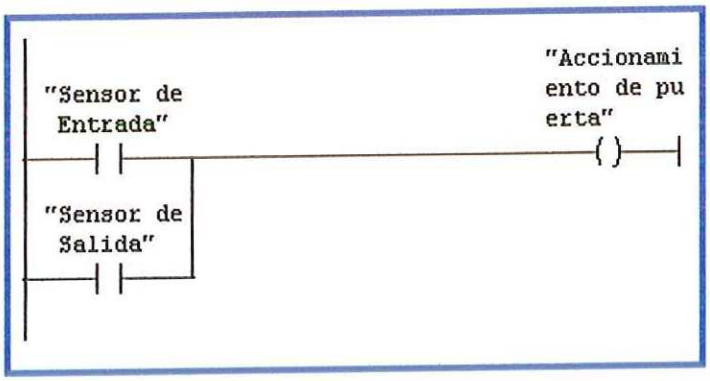
6) La tabla se llena de la siguiente manera:

	Estado	Símbolo /	Dirección	Tipo de dato	Comentario
1		Sensor de Entrada	I 124.3	BOOL	Sensor Inductivo
2		Sensor de Salida	I 124.4	BOOL	Sensor Inductivo
3		Accionamiento de puerta	Q 125.0	BOOL	Valvula

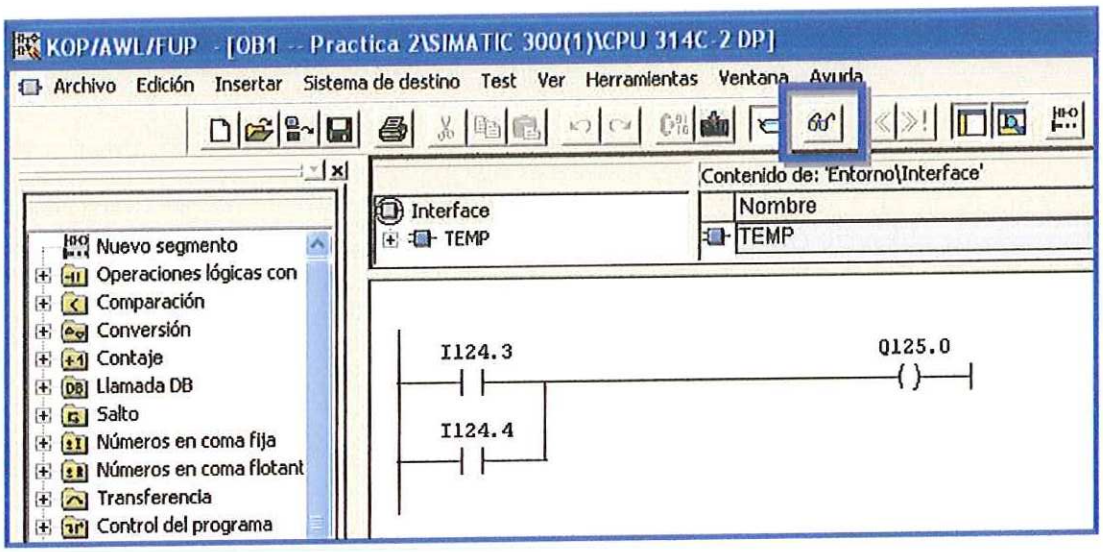
- El texto escrito bajo el campo *Símbolo* será el que reemplazará en el entorno de trabajo a la dirección del elemento.
- La *dirección* es la que corresponde a cada símbolo según como se requiera.

- *Tipo de Dato* expresa el tipo de variable que la dirección maneja.
- El *Comentario* es una forma de ampliar la información referente al símbolo.

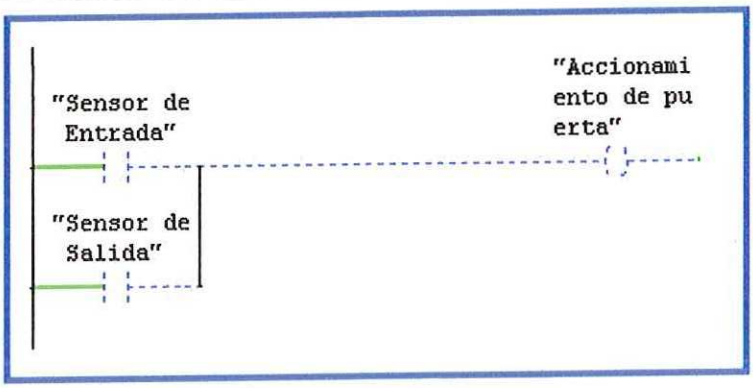
7) Guarde los datos y cierre la ventana, una vez hecho esto podrá apreciar los cambios en el entorno de trabajo y de esta forma identificar e interpretar más fácilmente las variables del sistema.



8) Para poder comprobar los resultados del proyecto y asegurarnos de que todo esté funcionando bien, existe la opción ***Observar***, esta pone en línea al PLC con el entorno de trabajo.

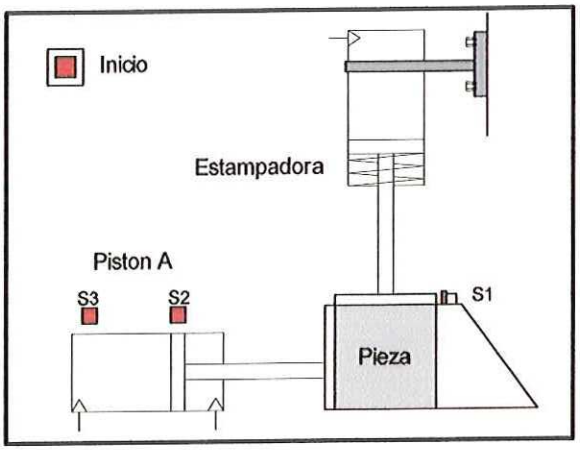


9) **Ponga el PLC en RUN**, el programa interpreta este cambio mostrando en tiempo real los estados de los elementos del programa, para probar, active uno de los sensores y observará que se ilumina su elemento correspondiente. La opción conlleva a una reacción en este caso se accionará la puerta.



6. ACTIVIDAD COMPLEMENTARIA

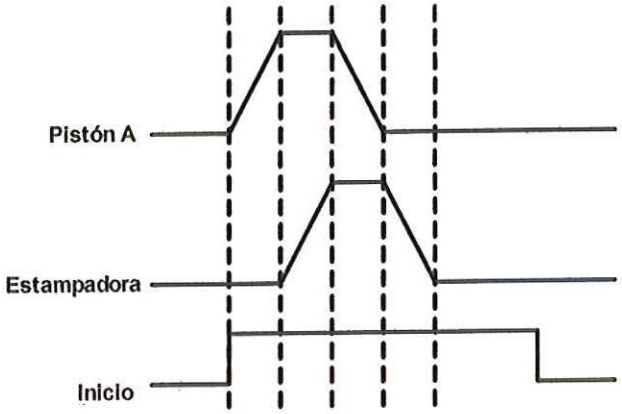
Una empresa tiene varias máquinas de estampado, las cuales hacen el mismo proceso pero con diferentes figuras, el accionamiento de este proceso consta de dos pistones, el primero (A) se encarga de sostener la pieza, el segundo (B) estampa la figura sobre la pieza. La orden de estampado inicia una vez se presiona un interruptor, al terminar todo vuelve a la posición inicial y queda en espera para iniciar nuevamente una vez se vuelva a presionar el interruptor.



Hay tres sensores de posición para determinar el estado de los pistones. Dos para A que es de doble efecto, y uno para la estampadora que es de simple efecto.

El interruptor de inicio es con enclavamiento.

El siguiente diagrama de tiempos describe el trabajo del proceso:



Diseñe el programa que pueda controlar este proceso.

MATERIAL EXTRA No. 2

1) Conexión de Sensores y Actuadores

Lo sensores son dispositivos que detectan cambios en un sistema que están analizando. Estos dispositivos son de carácter eléctrico por lo que necesitan alimentación, su funcionamiento de tipo todo o nada, generan un nivel de tensión cuando están activos y generalmente cero cuando están en estado inactivo. Es necesario que el Sensor y el PLC estén referenciados a un mismo punto para que funcionen armónicamente, para los sensores de dos cables como interruptores no hay dificultades en su conexión, ya que uno de los cables se conecta a la entrada y el otro a una línea de alimentación.

Cuando el sensor tiene tres cables la conexión no es tan simple, ya que es necesario identificar entre los tres cuál es la señal, la alimentación y la referencia. Para esto es necesario valernos de un multímetro.

Lo primero que hay que identificar es el neutro o referencia, este se haya con el multímetro en modo de lectura de resistencia eléctrica, el neutro es aquél que marca resistencia al conectar con los otros dos. Una vez hallado el neutro es necesario determinar cual cable es la señal y cual es la alimentación, para esto es necesario hacerlo al tanteo, conecte uno de los cables al módulo de entradas mientras el otro a 24 VDC o a la alimentación requerida, si el sensor queda en estado activo de forma permanente la conexión se debe invertir.

Una vez este conectado el sensor correctamente pruébelo para determinar su correcto funcionamiento.

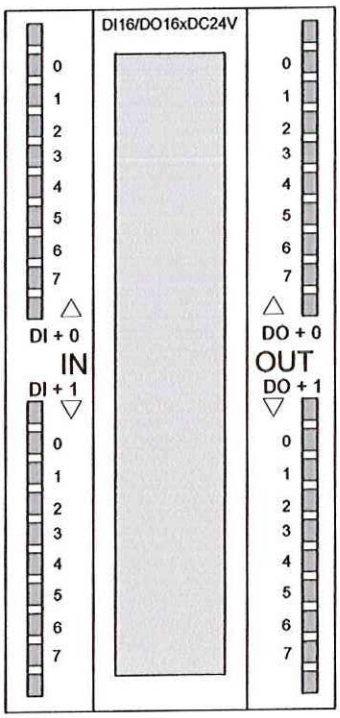
Los actuadores como las válvulas que controlan a los pistones, son dispositivos ON-OFF que no tienen polaridad. Funcionan a través de las salidas digitales del PLC y la referencia (la cual es totalmente diferente a la usada para la alimentación de 120 VAC), para este propósito tiene dos cables. El hecho de que no tengan polaridad permite realizar la conexión sin que se tengan que identificar los cables.

Un tratamiento similar los tiene los interruptores que se utilizarán a través de todas las prácticas (normalmente abiertos, normalmente cerrados, con enclavamiento o sin él) y las lámparas (dos conexiones sólo que una va a la salida digital y la otra a tierra).

2) Alimentación del módulo de entradas y salidas digitales

El módulo de entradas y salidas digitales permite la conexión de elementos cuya alimentación varía entre los 10 VDC a los 30VDC, el cual se puede suministrar externamente a través de una fuente de corriente directa, es necesario que este potencial este referenciado con el del PLC para que funcionen correctamente, para esto, el PLC cuenta con puntos de conexión para estas alimentaciones.

Los puntos son {L1 - M1}, {L2 - M2} y {L3 - M3}; los dos primeros son para las salidas digitales y el último es para las entradas (se identifican por ser entradas sin numeración). L1, L2 y L3 es la conexión positiva proveniente de la fuente, mientras M1, M2 y M3 sus respectivas tierras. La posición de estas conexiones se pude apreciar en la tapa del módulo. Así por ejemplo, si se necesita conectar sensores que trabajan con 24VDC es conecta a las terminales L3 y M3 una alimentación de 24VDC.



PRÁCTICA 3
ENTRADAS Y SALIDAS DIGITALES

PRÁCTICA 3
OPERACIONES LÓGICAS CON BITS

1. OBJETIVOS

- ✓ Interpretar los conceptos sobre lógica booleana en el lenguaje de programación de PLC.
- ✓ Utilizar las diferentes operaciones lógicas para la solución de problemas.

2. CONCEPTOS FUNDAMENTALES

Las compuertas lógicas son los circuitos lógicos más fundamentales que pueden describir su operación mediante el uso de álgebra booleana. El álgebra booleana difiere de manera importante del álgebra ordinaria en que las constantes y variables booleanas sólo pueden tener dos valores posibles 0 y 1. Por ejemplo, en cualquier sistema digital el valor booleano de 0 podría ser asignado a cualquier voltaje en el intervalo de 0 a 8 voltios, en tanto que el valor booleano de 1 podría ser asignado en el voltaje de 2 a 5 voltios. Por lo tanto el 0 y el 1 booleanos no representan números sino; el estado de una variable de voltaje o bien lo que se conoce como su nivel lógico.

En el campo de la lógica digital, se emplean otros términos como sinónimo de 0 y 1, estos son algunos de los más comunes[1]:

Cero lógico	Uno lógico
Falso	Verdadero
Desactivado	Activado
Bajo	Alto
No	Sí
Interruptor Abierto	Interruptor Cerrado

El álgebra booleana se utiliza para expresar los efectos que los diversos circuitos digitales ejercen sobre las entradas lógicas y manipular variables

[1] Extraído de: Sistemas Digitales, Principios y Aplicaciones de Ronald J. Tocci.

lógicas con el objeto de determinar el mejor método de ejecución de cierta función de un circuito. Ya que solo son dos valores, el álgebra booleana es relativamente fácil de manejar en comparación con la ordinaria. En el álgebra booleana no existen fracciones, decimales, números negativos, raíces cuadradas, cúbicas, logaritmos, números imaginarios, etc. De hecho en el álgebra booleana solo existen tres operaciones lógicas NOT, AND y OR.

3. ACTIVIDAD PREVIA

Antes de proceder al desarrollo de esta práctica es necesario que elabore el siguiente cuestionario:

1) Describa brevemente el funcionamiento de las tres operaciones básicas AND, NOT y OR
2) ¿En el PLC Simatic que valor representa el 1 y el 0 lógico?
3) Averigüe el modo correcto de conectar el ventilador, el cual se utilizará en la práctica.

4. LISTA DE MATERIALES

1) Ventilador.

Nota: *los sensores e interruptores serán representados con el módulo simulador de entradas y salidas digitales.*

5. DESARROLLO METODOLÓGICO

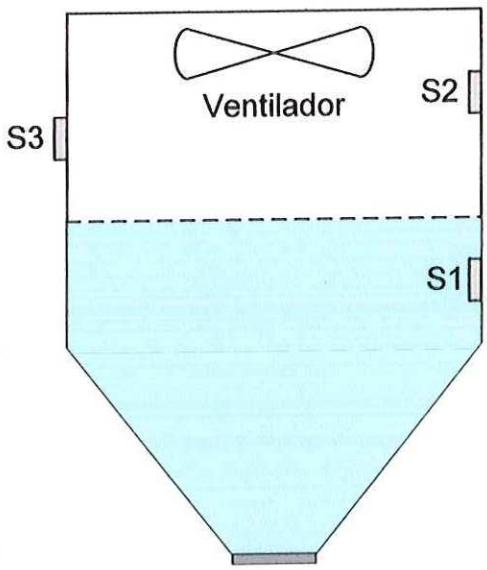
Abrir ***SIMATIC/Administrador Simatic***, cree un nuevo proyecto con el nombre *Práctica 3.*

Ahora se elaborará un programa en el ***OB1*** que resuelva la siguiente situación.

En una planta de combustible, existe un tanque de almacenamiento el cual tiene a su disposición un sistema de control que consta de:

1) Un sensor de temperatura para el combustible (S1).
2) Un sensor de temperatura para el vapor emitido (S2).
3) Un sensor de temperatura para las paredes del tanque (S3).
4) Un interruptor de operación para el ventilador en caso de

emergencias.



La planta está ubicada en una ciudad a nivel del mar, y los vapores del combustible son altamente explosivos, poniendo en riesgo la vida de los trabajadores. Para enfrentar este problema se le ha adaptado al tanque un sistema de ventilación que se activa cuando la temperatura se eleva a puntos críticos previamente establecidos.

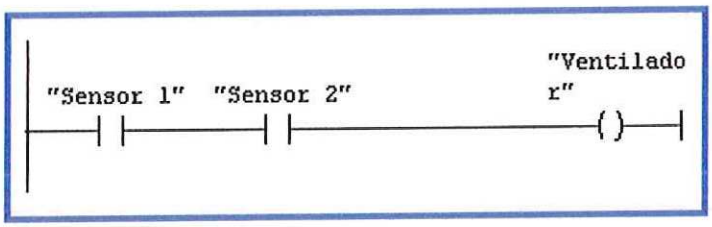
Existen ciertos parámetros para la activación del ventilador determinados empíricamente por los operadores de la planta y que aseguran el funcionamiento óptimo de esta.

Se utilizará el módulo de simulación de entradas y salidas digitales para representar a los sensores de temperatura y el botón de accionamiento de emergencia. El ventilador se conectará al módulo de salidas digitales. Antes de iniciar con el programa, **es preferible asignar nombres para hacer más sencilla la programación**, para esto entre al menú *herramienta/tabla de símbolos* y copie la siguiente tabla:

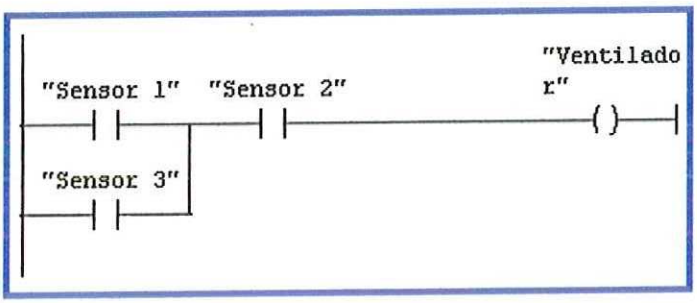
	Estado	Símbolo	Dirección	Tipo de dato	Comentario
1		Sensor 1	I 0.0	BOOL	Sensor de temperatura del liquido
2		Sensor 2	I 0.1	BOOL	Sensor de temperatura del vapor
3		Sensor 3	I 0.2	BOOL	Sensor de temperatura del tanque
4		emergencia	I 0.3	BOOL	emergencia
5		Ventilador	Q 124.0	BOOL	
6					

Nota: *haga referencia de las direcciones que va a utilizar en su proyecto.*

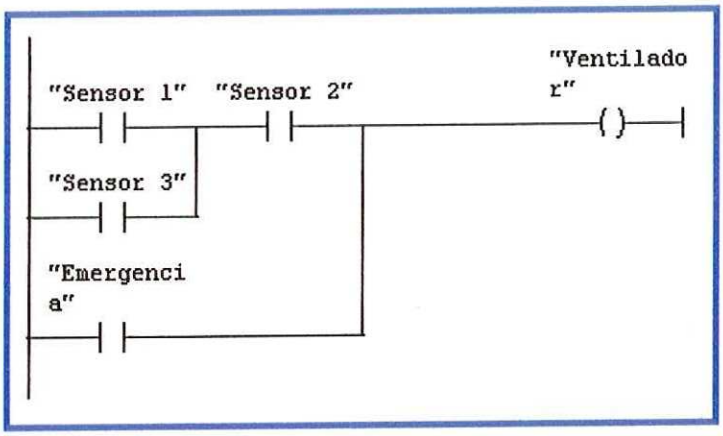
Ahora, se iniciará el desarrollo del programa, la primera condición de activación del ventilador, se da cuando el sensor de temperatura del líquido y el sensor de temperatura del vapor se activen. Es decir cuando estos dos medios estén cerca a su punto crítico.



La segunda condición de activación se da en el momento que tanto el sensor del vapor como el sensor de temperatura del tanque se activen.

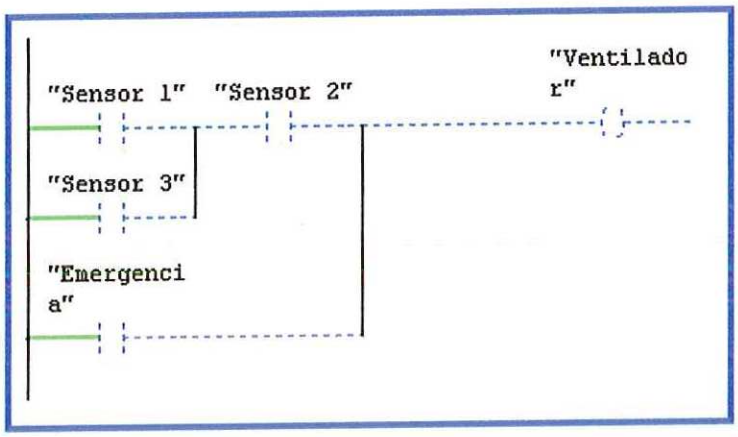


Por último existe un interruptor que en caso de algún daño en los sensores, puede activar directamente el sistema de ventilación, esta tarea es realizada por un operario.

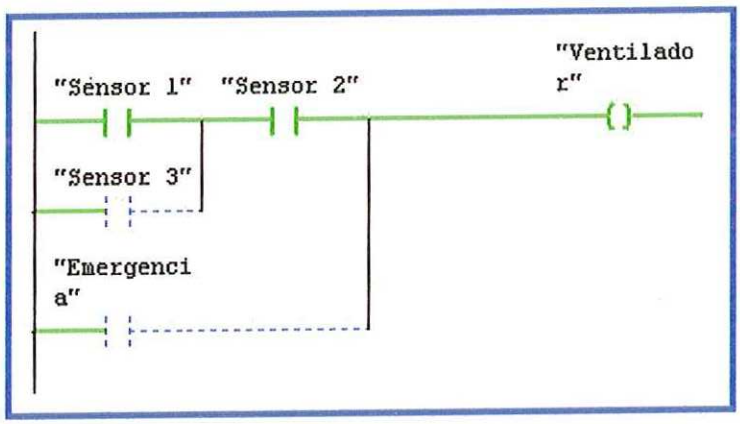


Ahora se procederá a cargar el programa realizado al PLC, a través del icono *cargar al módulo*. Una vez el PLC este listo, de clic sobre el icono *observar* y con el módulo de simulación de entradas y salidas digitales procederemos hacer pruebas.

En el momento en que ninguno de los sensores haya detectado un punto crítico el ventilador se mantendrá desactivado.

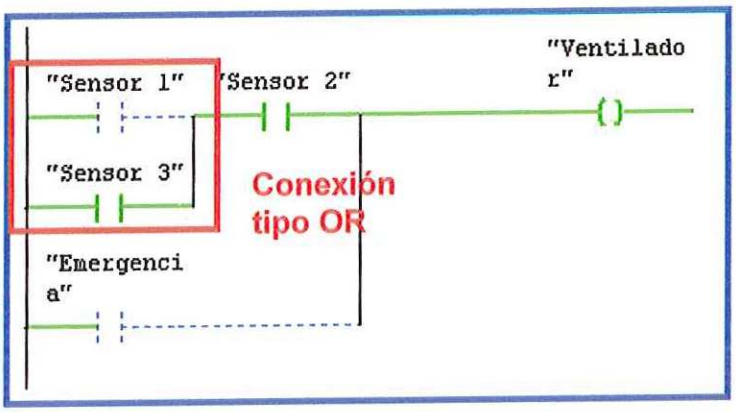


En el momento en que el sensor de temperatura del líquido **"y"** el sensor de temperatura del vapor hayan alcanzado su punto crítico emitirán la señal de activación al dispositivo de ventilación. Observe que esta condición es de tipo AND la cual expresa que la salida será 1 sólo si todas las entradas son 1.

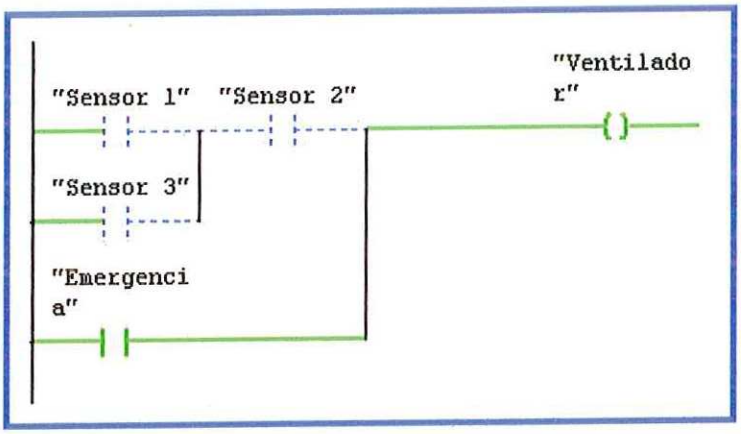


Los operarios establecen que una alta temperatura en el vapor del líquido no es crítica para que ella por si sola active el ventilador, sin embargo, un elevamiento en la temperatura de las paredes del tanque **"o"** del líquido son suficientemente alarmantes como para generar una señal de activación en el dispositivo. Observe que la condición entre el sensor 1 y el sensor 3, estando activo el sensor 2, es de tipo OR, la cual expresa que el resultado será 1 si una o la otra variable es 1.

Observe la siguiente gráfica que ilustra lo explicado.

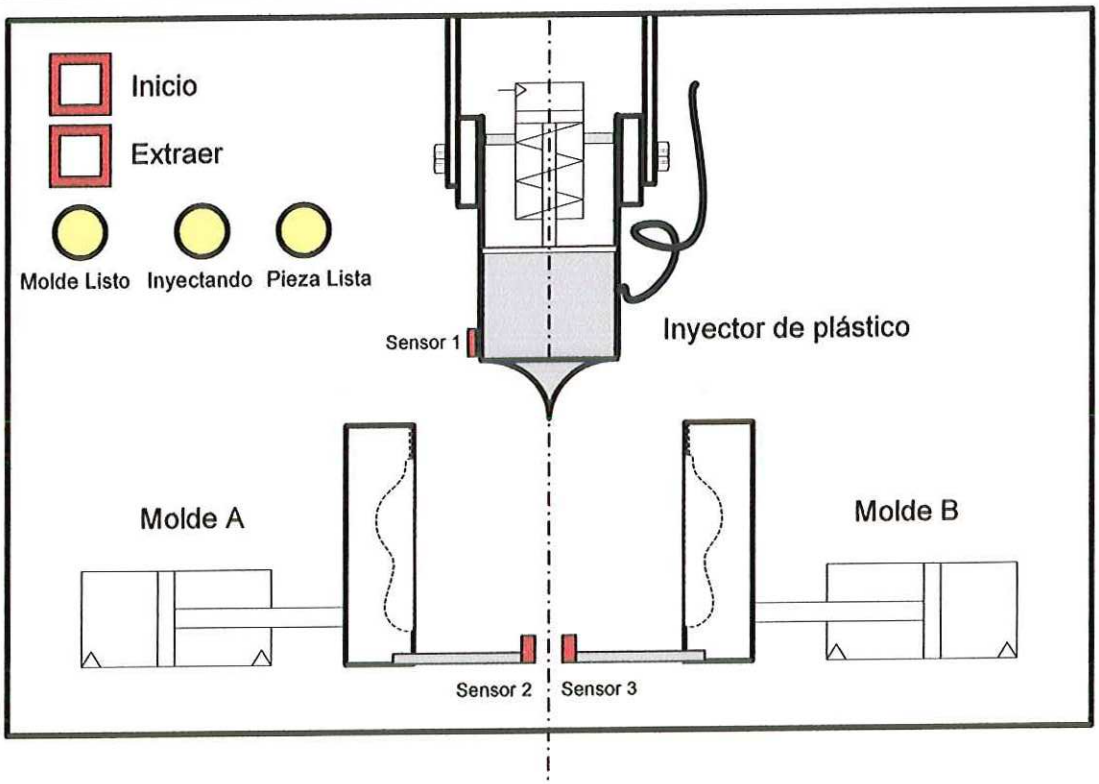


Si por alguna circunstancia el operario considera que es necesario encender el ventilador, lo puede hacer directamente activando el interruptor de emergencia, observe que todo el sistema está gobernado por una condición tipo OR que enciende el ventilador sin importar la condición en que se encuentran los sensores.



6. ACTIVIDAD COMPLEMENTARIA

Una máquina de moldeado de plástico consta de tres pistones, dos se encargan de unir un molde metálico que da la forma, mientras el otro desde arriba, inyecta una cantidad determinada de material hacia este (Ver figura del proceso). Una vez está lista la pieza, un operario da la orden para que la máquina la suelte (con un interruptor), El proceso no es continuo y la misma máquina es utilizada para dar forma a otras piezas. Así que cada vez que se requiere, el operario monta el molde y con otro interruptor da la orden para que inicie a trabajar.



Existen luces indicadoras que señalan tres momentos:

1) Molde listo, el cual indica que el molde ya esta listo para ser inyectado por plástico (Iniciar el proceso).
2) Inyectando, el cual indica que se esta inyectando el plástico a la pieza.
3) Pieza lista, el cual indica que la pieza ya esta lista.

Las tres luces nunca están encendidas a la vez, e indican el orden del proceso y sirven para que el operario pueda conocer el estado del trabajo, no se debe dar la orden para sacar la pieza sino hasta que la tercera luz está encendida.

Como se mencionó existen dos interruptores (con enclavamiento):

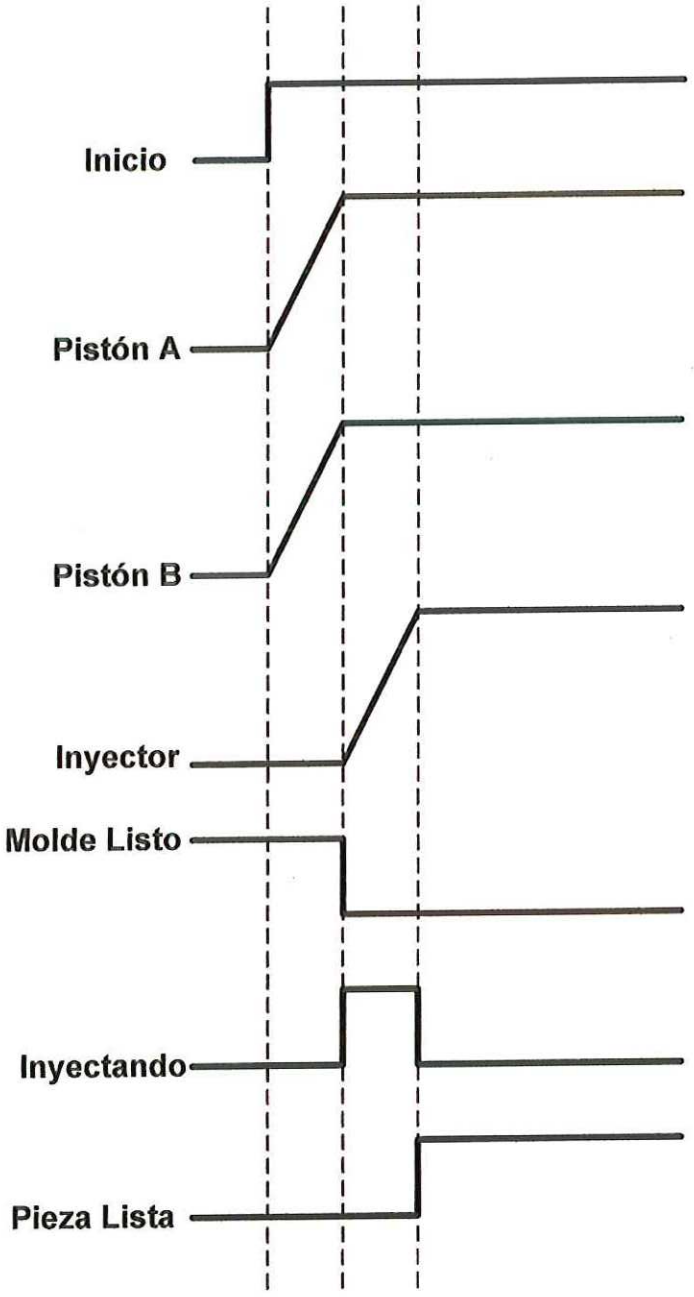
1) Inicio, que es el encargado de iniciar el proceso.
2) Extraer, que es el encargado de regresar los pistones a la posición inicial.

Al terminar el proceso es necesario desenclavar los interruptores.

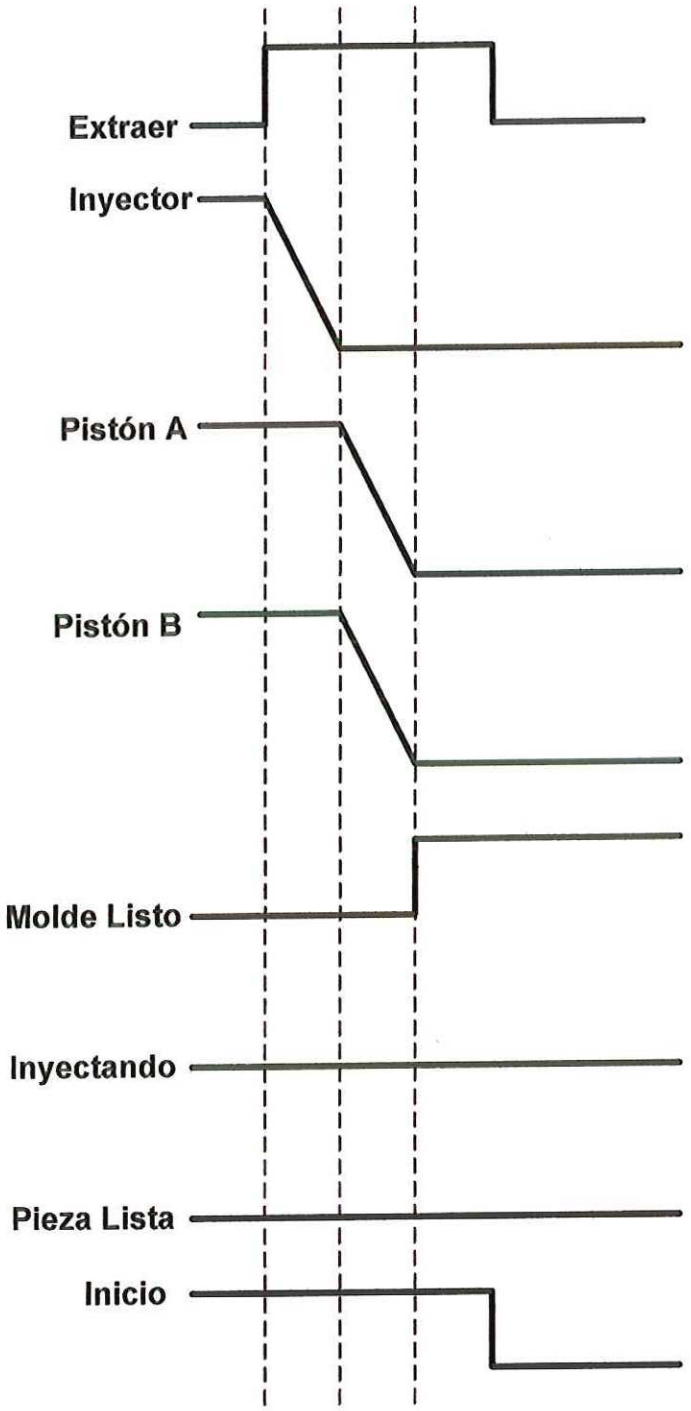
El proceso consta de tres sensores para determinar si los pistones están expandidos.

El siguiente diagrama de tiempo muestra la forma en que trabaja el proceso:

Al presionar el botón Inicio:



Al presionar el botón Extraer:



Diseñe el programa que realiza este proceso.

PRÁCTICA 4
LÓGICA SECUENCIAL – USO DE FLIP – FLOP'S

PRÁCTICA 4
LÓGICA SECUENCIAL – USO DE FLIP-FLOP'S

1. OBJETIVOS

- ✓ Configurar y utilizar los recursos de lógica secuencial disponibles en el PLC.
- ✓ Implementar elementos de lógica secuencial en la elaboración de proyectos en el PLC.
- ✓ Utilizar las marcas de memoria como elementos útiles para el desarrollo de proyectos en el PLC.

2. CONCEPTOS FUNDAMENTALES

Lógica Secuencial

Hasta el momento se ha estudiado el diseño de proyectos a través de la lógica combinacional, aquí se exploró el uso de combinaciones tipo AND, OR y NOT para el desarrollo de soluciones y la elaboración de algoritmos usando los diferentes lenguajes de programación.

Los circuitos de lógica combinacional tienen la característica de dar una respuesta que depende de los niveles existentes en las entradas en ese instante. Los circuitos combinatorios no tienen memoria, así que las condiciones establecidas anteriormente no afectan a las salidas actuales. La memoria es un recurso utilizado para almacenar datos, el elemento más importante de ella es el FLIP-FLOP.

Ahora bien, en la electrónica digital se usan tipos distintos de Flip-Flop's, como los , S-C. J-K, o los tipo D, los cuales dependen de una señal de reloj (clock o CLK) que da la "orden" para almacenar un dato. Step 7, el software de programación utilizado para los PLC's de SIEMENS, sólo tiene Flip-Flop's

de tipo S-C (su similar) el cual no requieren de una señal temporizada para su funcionamiento por lo que son más sencillos de implementar.

3. ACTIVIDAD PREVIA

Antes de proceder al desarrollo de la práctica es necesario que elabore el siguiente cuestionario:

1) ¿Cómo funcionan los Flip-Flop's tipo S-C?
2) Examine el complemento disponible al final de la práctica: **Marcas.**

4. LISTA DE MATERIALES

1) Dos sensores de presencia inductivos.
2) 1 electroválvula de una bobina.
3) 1 pistón de simple efecto.
4) 1 pulsador normalmente abierto sin enclavamiento.
5) 1 pulsador normalmente cerrado sin enclavamiento.

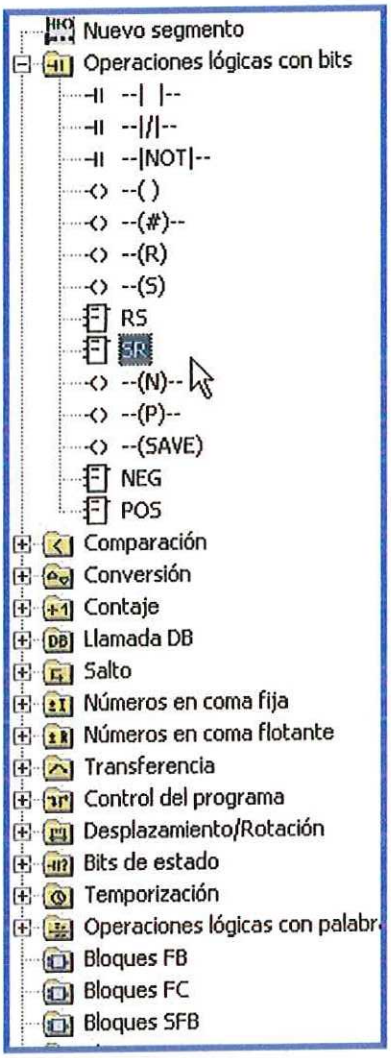
5. DESARROLLO METODOLÓGICO

Nota: *para el desarrollo de esta actividad es necesario haber elaborado la Práctica 3, Operaciones Lógicas con Bits .*

Abrir ***SIMATIC/Administrador Simatic***, cree un nuevo proyecto con el nombre *Práctica 4*.

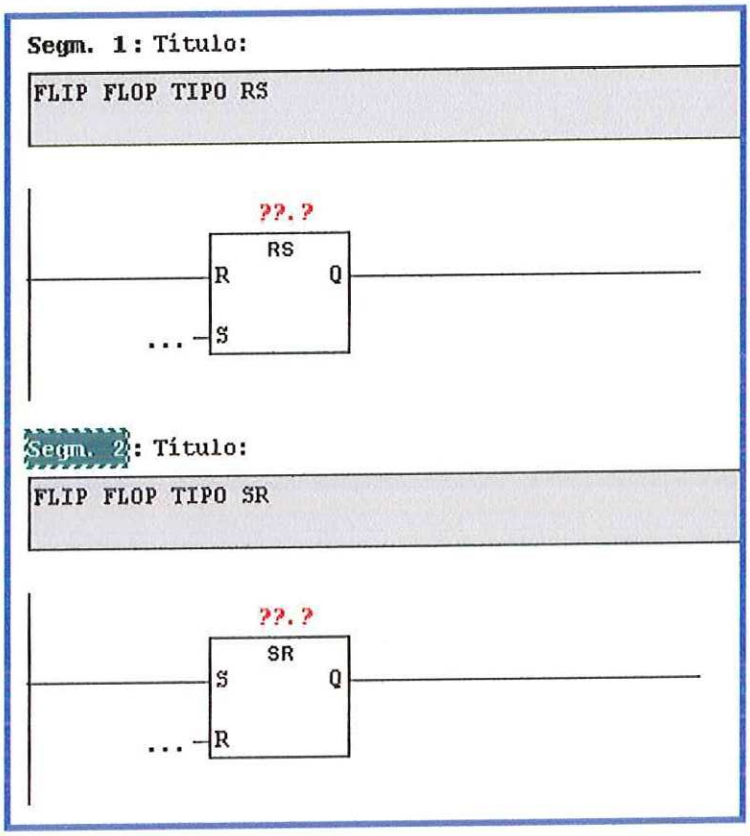
Entre al Bloque de organización No.1 (OB1), antes de proceder con la actividad de práctica es necesario explicar el funcionamiento de los elementos de lógica secuencial disponibles en Step 7, el programador de PLC's Simatic.

Cerciórese que la ventana de *Elementos de Programa* esté abierta, para esto vaya al menú *VER* y verifique que la opción *Vista General* esté seleccionada.



Agregue un segmento al entorno de trabajo, en la librería de funciones despliegue la carpeta *Operaciones lógicas con bits*, Step 7 tiene a su disposición dos tipos diferentes de F-F (Flip-Flop's) los SR y los RS. Para habilitarlos basta con seleccionar su figura de la librería y desplazarla hacia el segmento de trabajo.

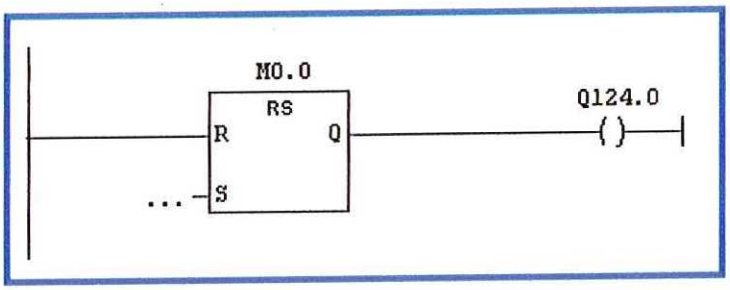
Los F-F son elementos que responden a cambios en sus entradas R (Reset) y S (Set), activar y desactivar en español, estos cambios se visualizan consecutivamente en dos partes: la salida (Q) y la **etiqueta**, representada por los signos de interrogación en la parte superior de la gráfica.



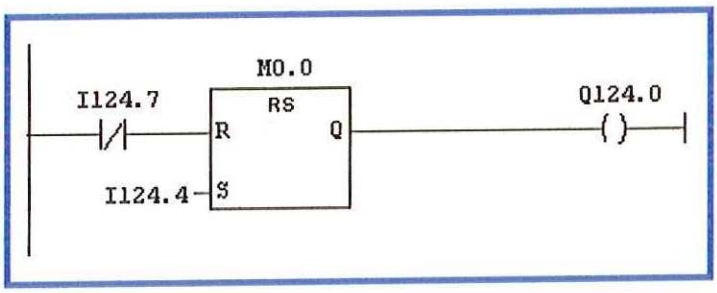
A continuación, se procederá a explicar el funcionamiento de estas herramientas, pero antes de iniciar con este paso, **conecte los elementos** mencionados en el punto 4 *Lista de materiales*, conéctelos a las direcciones de entradas y salidas que les parezcan convenientes ya que no existe ninguna restricción para este punto.

Ahora se procederá a la configuración del F-F RS.

1) Asigne a Q la dirección correspondiente a la salida conectada a la válvula de accionamiento. A la etiqueta asigne el bit de marca M0.0.

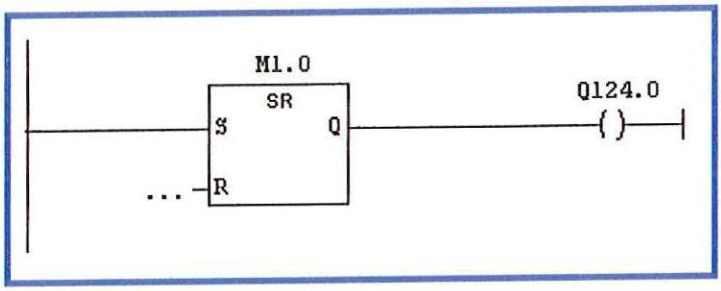


2) Ahora **asigne a S la dirección correspondiente al interruptor normalmente abierto** (en nuestro caso se asignó a I124.4) y a **R asigne el interruptor normalmente cerrado** (en este caso I124.7). Para insertar las bobinas, seleccione el espacio correspondiente en el F-F y de clic sobre le icono insertar bobina. Una vez hecho esto, guarde los cambios y cárguelos al PLC. A continuación, de clic en el icono observar.

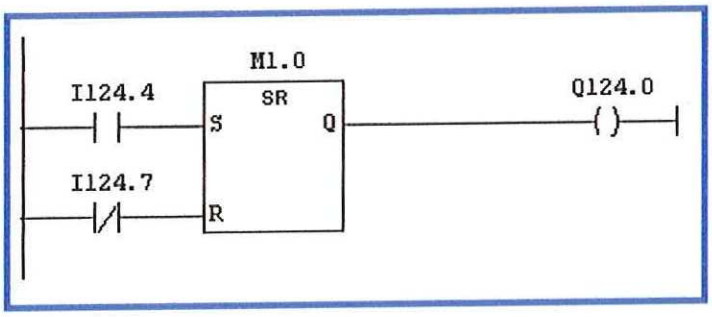


Este F-F funciona de la siguiente forma, al presionar el interruptor asignado a S, la salida Q y la etiqueta (M0.0) se activarán y conservarán este estado aunque S vuelva a 0 (recuerde que este interruptor es normalmente abierto, por lo que su estado natural es desconectado). Para desactivarlo (Q = 0) accione el pulsador asignado a R. Como resultado de este experimento, y ya con los elementos conectados verá que cuando presiona el accionamiento normalmente abierto (S) el pistón saldrá y no volverá a su posición inicial sino hasta que presione el botón que es normalmente cerrado (R).

3) Ahora, borre estos cambios, para hacer esto, basta con seleccionar el F-F y pulsar *Supr* en el teclado. Inserte ahora el F-F tipo SR y realice los mismos pasos del punto 1.



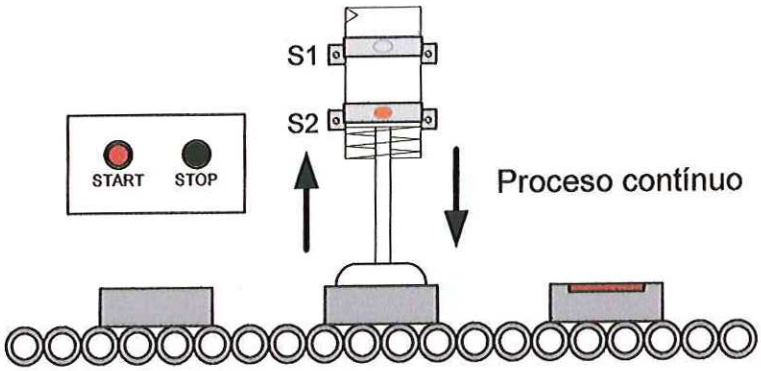
4) Ahora bien, como puede apreciar, la diferencia de este F-F con su homólogo está en la posición de S y R, la configuración completa se hace de la misma forma como se hizo con el RS.



El objetivo de este punto no fué solo la puesta a prueba del funcionamiento de los F-F, sino también el de mostrar una de sus utilidades como elemento de memoria. Hasta el momento, en los sistemas de accionamiento de las ejercicios y actividades, se han utilizado interruptores con enclavamiento; la virtud de estos elementos radica en la capacidad de conservar su estado una vez ha sido activado. Los interruptores sin enclavamiento conservan su cambio de estado siempre y cuando se mantengan presionados, una vez se sueltan, vuelven a la posición original y por ende a su estado inicial. Los F-F almacenan este cambio de estado y pueden interpretar esto como una orden, la cual se conservará hasta que sea activado el Reset.

Con el siguiente ejemplo se ilustrará lo anteriormente mencionado.

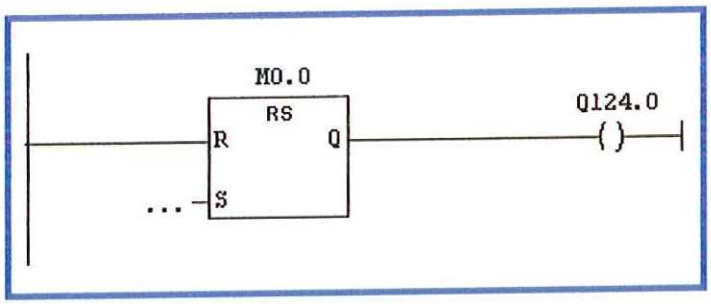
En una empresa de alimentos se tiene como proceso final, una etiqueteadora que se encarga de estampar la marquilla al producto ya terminado. La etiqueteadora esta compuesta de un pistón de simple efecto el cual es controlado por una válvula de una bobina. Para conocer la posición del pistón, el sistema cuenta con dos sensores de presencia inductivo (S1 y S2) ubicados al inicio y al final de la carrera del actuador. El proceso inicia cuando se presiona START (normalmente abierto sin enclavamiento) el pistón saldrá y entrará continuamente, se tiene calculado que a la salida del pistón ya está la pieza lista para ser etiquetada. El proceso se detiene cuando se presiona STOP (normalmente cerrado sin enclavamiento)



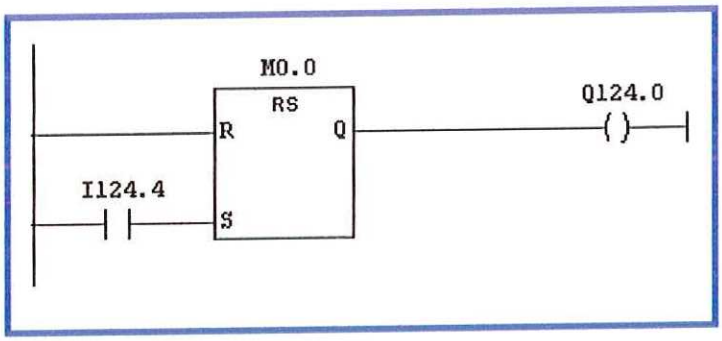
Nota: *cuando se detiene el sistema, el pistón debe volver a la posición inicial.*

Para la solución de este problema se usarán los elementos de lógica secuencial para almacenar las órdenes hechas por el operario.

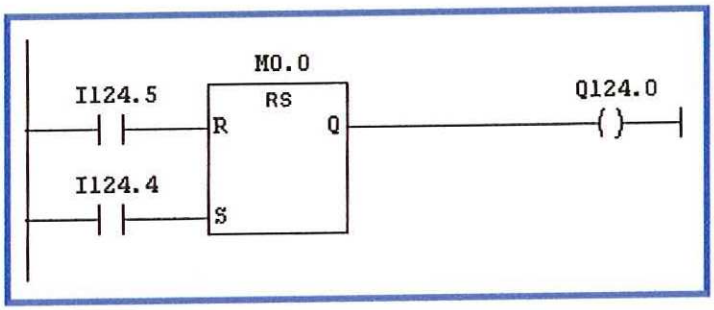
1) Para controlar la válvula (conectada a la salida Q124.0 para este caso) se utilizará un F-F tipo RS.



2) START (direccionado a la entrada I124.4), se conectará directamente a la entrada S del F-F, de esta forma cuando se presione, la válvula se activará y el pistón saldrá.

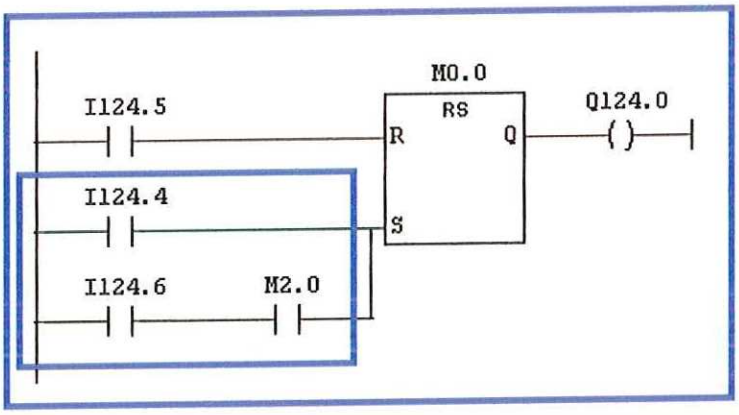


3) Para que el pistón se devuelva automáticamente, se ha conectado el sensor de posición S2 a R, (S2 está direccionado a la entrada I124.5). De esta forma, cuando el actuador sale, la señal de S2 hará que el F-F se resetee, provocando que el pistón regrese a la posición inicial,



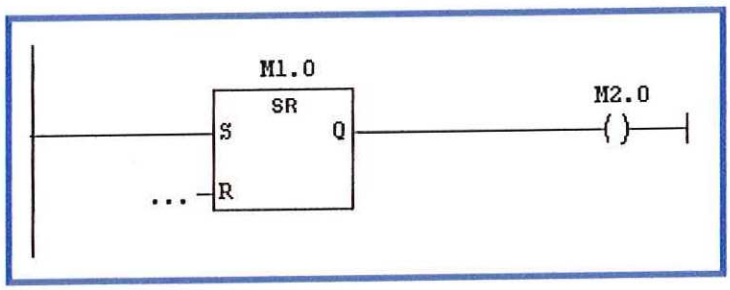
Sin embargo el programa aún no se ha terminado, porque una de las exigencias es que sea de accionamiento continuo y se detenga cuando se presiona el botón STOP.

4) Para solucionar esto, **con una conexión tipo OR** se establece una segunda condición de activación de S, esta es, la señal de S1 (conectado a la entrada I124.6), la cual se activa cuando el pistón vuelve a su estado inicial. Cuando esto sucede, Q se vuelve a activar y el pistón vuelve a salir y así se cumplirá el ciclo de forma continua.

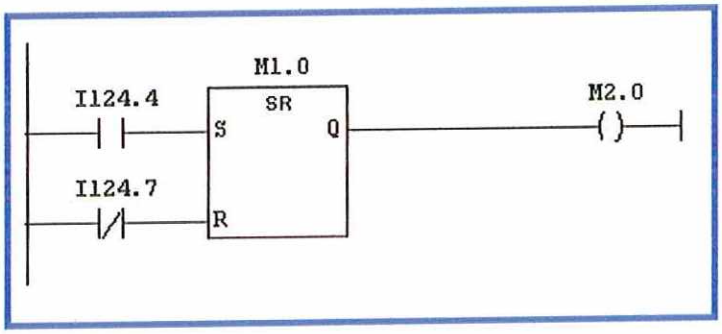


Sin embargo, con esta modificación el programa no queda listo, ya que gracias a este cambio surge un nuevo problema, presentado cuando el programa inicia, en este punto la válvula está desactivada y por ende el pistón retraído, esto significa que (S1 = 1). Generalmente un PLC interpreta los datos que llegan a sus entradas cuando se encuentra en modo *RUN*, en modo *STOP* son insensibles a los cambios que estas presenten, un cambio de modo *STOP* a *RUN* hará que el pistón salga automáticamente sin presionar START, ya que S esta leyendo el 1 proveniente de S1. Para solucionar esto, se hace **una conexión tipo AND** en la línea donde está el sensor de posición S1, usando la marca M2.0, la cual es un indicador de estado que evitará el accionamiento automático de la válvula.

5) Para controlar el estado de la Marca se utilizará un F-F tipo SR.



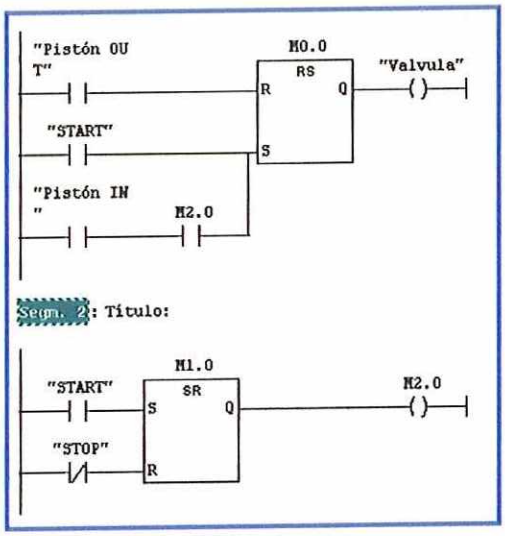
6) La Marca es una bandera que se habilita cuando se presiona START (I124.4), mientras se conserve en ese estado, el programa correrá automáticamente, y sólo se desactivará cuando se da STOP (conectado a la entrada I124.7) al hacer esto, el sistema se detendrá automáticamente y sólo volverá a iniciar cuando se presione START nuevamente.



7) Para una mejor comprensión del funcionamiento del programa, etiquete las direcciones utilizadas con los elementos con que están conectados

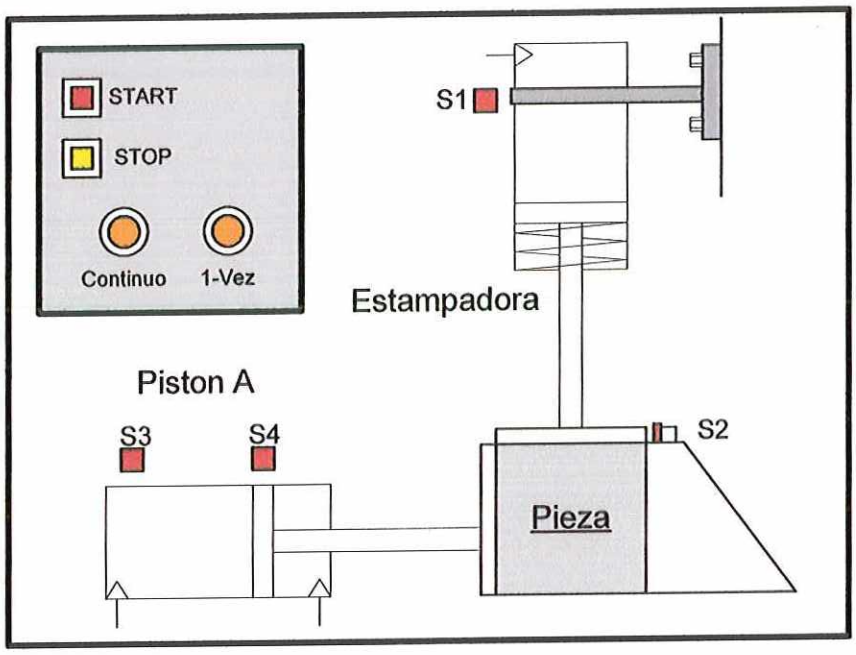
	Estado	Símbolo /	Dirección	Tipo de dato	Comentario
1		START	I 124.4	BOOL	Interruptor normalmente Abierto
2		Pistón OUT	I 124.5	BOOL	Sensor Inductivo de presencia
3		Pistón IN	I 124.6	BOOL	Sensor Inductivo de presencia
4		STOP	I 124.7	BOOL	Interruptor normalmente Cerrado
5		Valvula	Q 124.0	BOOL	Activa a el pistón
6					

8) El programa final quedará de la siguiente forma.



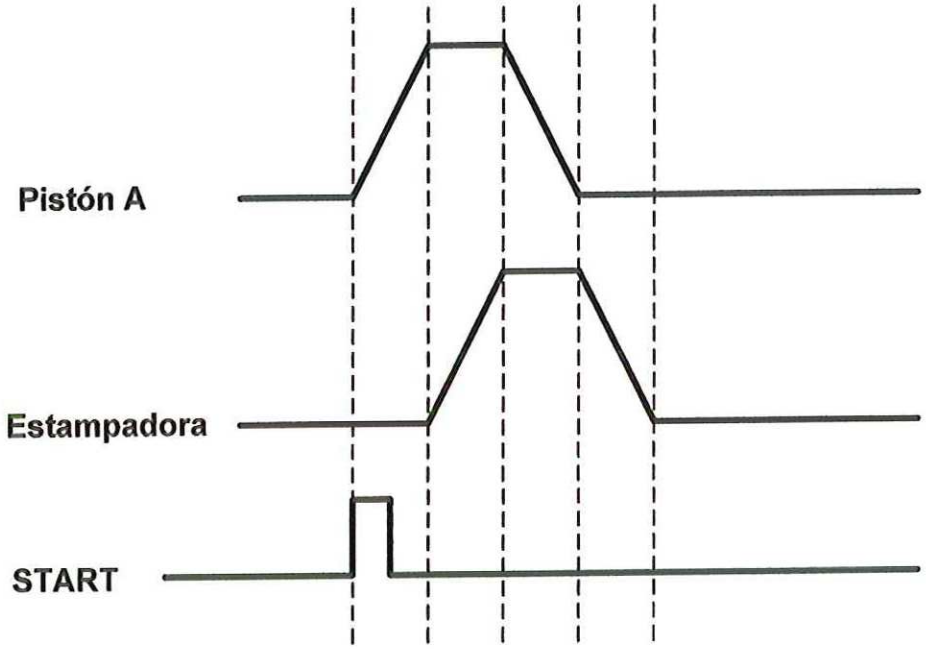
5. ACTIVIDAD COMPLEMENETARIA

Retomando la actividad realizada en la *Práctica 2*, la estampadora, a la cual se le han hecho varias modificaciones, por lo que es necesario modificar el software. Recordemos que la estampadora está compuesta por un cilindro de doble efecto **A** (controlado por una válvula de dos bobinas) que realiza la labor de sujeción de la pieza y un cilindro de simple efecto (controlado por una válvula de una bobina) que realiza la operación de estampado, ambos actuadores tienen sensores de posición (S1,S2 y S3,S4) que leen permanentemente el estado de los pistones.



El sistema trabaja ahora en dos modos los cuales son seleccionados con la ayuda de dos interruptores normalmente abiertos sin enclavamiento.

El primer modo es denominado **Proceso Único**, cuya operación se describe en el siguiente diagrama de tiempos. **Proceso Único** se habilita cuando se presiona el botón **1-vez**, al hacer esto, el sistema queda en espera a que se presione START (normalmente cerrado sin enclavamiento), cuando esto sucede, **se realiza la secuencia una sola vez.** Cuando esta termina, el sistema queda en espera de la orden de START o que se cambie al otro modo.



Este modo es denominado **Proceso Continuo**, el cual se ejecuta cuando se presiona **Continuo,** al hacer esto, el sistema queda en espera de que se presione START, cuando esto sucede, **realizará el proceso descrito en el Proceso Único de forma indefinida**, y sólo se detendrá cuando se presione STOP (normalmente cerrado sin enclavamiento). Si se vuelve a presionar START el sistema volverá a trabajar en el mismo modo.

No se debe presionar el botón **1-vez** mientras el proceso este trabajando en modo continuo, en caso de que se desee hacer el cambio a modo único, se debe detener el sistema con STOP, y luego se procede a activar en el otro modo. La selección de un modo de operación deshabilita la otra. **El Proceso Único y el Proceso Continuo nunca deben estar habilitados al mismo tiempo**.

Diseñe el programa que ejecuta el proceso descrito teniendo en cuenta las consideraciones establecidas.

MATERIAL EXTRA No. 3

MARCAS

Las marcas son espacios virtuales que están disponibles para el uso en el diseño de programas, son representaciones binarias las cuales el entorno de programación interpreta de la misma forma que a las entradas y las salidas digitales.

Ocupan un espacio de 240 Bytes (dependiendo del modelo) dentro de la CPU del PLC, cada Byte tiene a su disposición los 8 bits que lo componen, lo que da un total de 1928 bits que pueden ser utilizados plenamente para cualquier proyecto. Para hacer uso de ellas en el entorno de programación, basta con escribir en la parte correspondiente a la dirección del elemento la letra M y la dirección que se desee. Por ejemplo una bobina con la dirección M10.4, hace referencia a que se va a usar el bit 4 del Byte 10 de las marcas de memoria para guardar algún tipo de información binaria. Es importante no asignar la misma marca (igual dirección) para varias tareas de un mismo proyecto esto puede generar conflictos en el programa y el PLc no funcionaría correctamente, gráficamente (y en el entorno de trabajo) una memoria puede ser representada por una bobina o una salida.

La marcas no ocupan un espacio físico, están almacenadas dentro del CPU del PLC y disponibles para cualquier uso, su utilidad radica en la capacidad de almacenamiento de información en tiempo real, que, a diferencia de los Flip-Flop's los cuales no son gobernados por decisiones en su operación, puede variar de la misma forma en que varían las situaciones en el entorno de trabajo. También pueden ser usados como banderas, elementos que se activan o desactivan según nuestra conveniencia y que pueden ser útiles para el desarrollo de proyectos en el entorno de trabajo.

PRÁCTICA 5
TEMPORIZADORES

PRÁCTICA 5
TEMPORIZADORES

1. OBJETIVOS

- ✓ Identificar los diferentes tipos de temporizadores disponibles en Step 7.
- ✓ Implementar los Temporizadores en la elaboración de proyectos en el PLC dependiendo de su función.

2. CONCEPTOS FUNDAMENTALES

El temporizado es una función que trabaja de acuerdo a un periodo de tiempo para realizar una acción, la acción consiste en la activación o desactivación de un bit, el cual puede ser una salida digital o una marca, el periodo de tiempo es determinado por el usuario, estos retardos trabajan en periodos de tiempo que van desde los minutos (sin llegar a la hora), hasta los milisegundos (sin ser mas rápidos que el tiempo de lectura y procesamiento del PLC mismo)

La acción de temporizado está implícitamente involucrada con los procesos, su inclusión en proyectos determina los retardos, tiempos de activación y desactivación y de espera. Su existencia data desde antes del nacimiento de los PLC's, siendo para ese entonces dispositivos eléctricos que trabajaban de acuerdo a la posición de una perilla que los controlaba, funcionaban en el instante que recibieran la orden y retardaban la activación (o desactivación) de una salida.

En los procesos continuos es importante la programación en base al mínimo y al máximo tiempo permisible para la creación de un producto, también es importante tener en cuenta la resistencia de los elementos que son accionados eléctricamente, es decir sus capacidades mecánicas, y muchos procesos trabajan en función de un tiempo determinado, estos son datos que influyen en el momento que se desea diseñar un programa para su control.

3. ACTIVIDAD PREVIA

Antes de proceder al desarrollo de esta práctica es necesario que elabore el siguiente cuestionario:

Con sus palabras responda:

a) ¿Ve usted en las prácticas preliminares la necesidad de implementar tiempos para la ejecución de los procesos?
b) Enumere los casos y las razones por las cuales necesitaría del uso de temporizadores.
c) Cite ejemplos de procesos que requerirían del uso de temporizadores.

4. LISTA DE MATERIALES

1) 4 sensores de posición.
2) 2 electroválvulas de doble bobina.
3) 2 pistones de doble efecto.
4) 1 pulsador normalmente abierto sin enclavamiento.
5) 1 pulsador normalmente cerrado sin enclavamiento.
6) 1 ventilador (cooler).

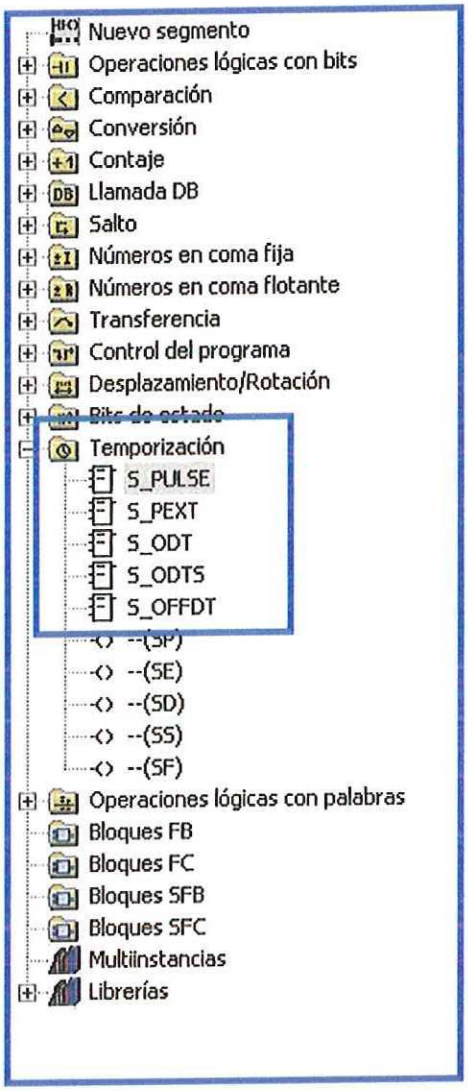
5. DESARROLLO METODOLÓGICO

Nota: *para el desarrollo de esta actividad es necesario haber elaborado la Práctica 4, Lógica Secuencial.*

Abrir ***SIMATIC/Administrador Simatic***, cree un nuevo proyecto con el nombre *Práctica 5*.

Entre al Bloque de organización No.1 (OB1). Antes de proceder con la actividad de práctica es necesario explicar el funcionamiento del los temporizadores disponibles en Step 7, el programador de PLC's Simatic.

Cerciórese que la ventana de *Elementos de Programa* se encuentre abierta, para esto vaya al menú *VER* y verifique que la opción *Vista General* esté seleccionada.

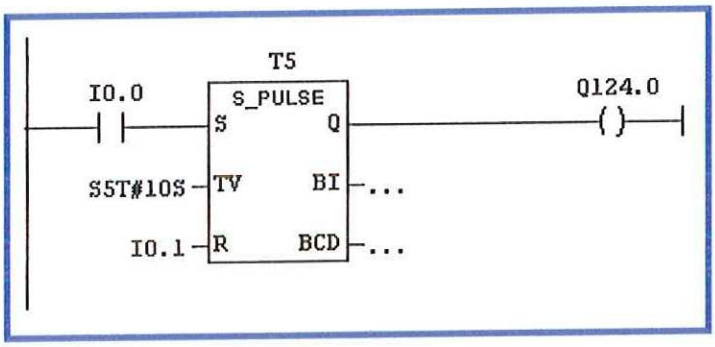


Agregue un segmento al entorno de trabajo, ahora, en la librería de funciones despliegue la carpeta *Temporización*, Step 7 tiene a su disposición una serie de temporizadores que según sus representación gráfica se dividen, más no por su funcionamiento. La primera serie representa a los temporizadores en forma integrada, como S_PULSE (en inglés) o S_OFFDDT (también en inglés), lo segundos son homólogos pero representan los componentes del temporizador por separado, para que éste sea armado según sus necesidades. La práctica se enfatizará en la implementación del primer grupo por ser más sencilla su implementación y cubre todos los objetivos.

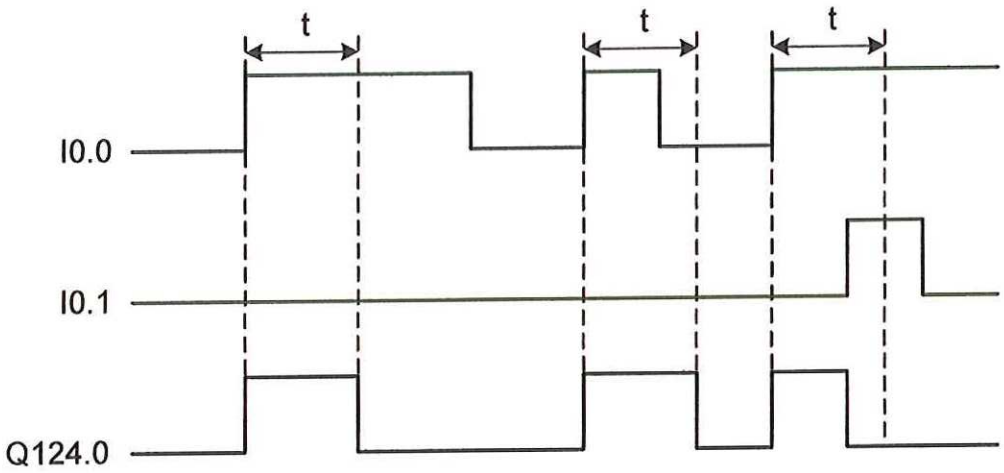
Los temporizadores son elementos que responden a cambios en sus entradas R (Reset) y S (Set), activar y desactivar en español, y según un nuevo parámetro de entrada que es el tiempo, la salida se activará o desactivará según las características del elemento. Existen cinco tipos de

temporizadores, todos se configuran de la misma forma pero difieren en su comportamiento, estos son:

1) Parametrizar y arrancar temporizador como impulso.



Esta opción denominada S_PULSE (S_IMPULS en alemán), funciona de la siguiente forma, cuando la entrada S (conectada a I0.0) recibe un cambio de 0 a 1, la salida Q (conectado a Q124.0) ***se enciende tanto tiempo como lo indica TV*** (o TW en alemán) y luego se apaga (Q=0). Si S vuelve a 0 mientras el temporizador trabaja Q se desactiva. Si R se activa, Q se desactiva y conservará este estado tanto tiempo como R se mantenga en 1. Es importante identificar el temporizador, en NºT, usualmente se escribe la letra T seguida de un número, por ejemplo para este caso T5. La siguiente gráfica describe todas las alternativas:



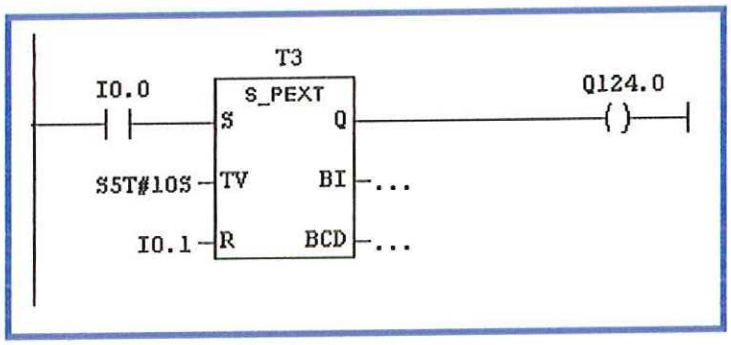
t= tiempo programado (TV)

BI y BCD llevan cuenta del tiempo que ha transcurrido, BI lo hace en hexadecimal y BCD en decimal.

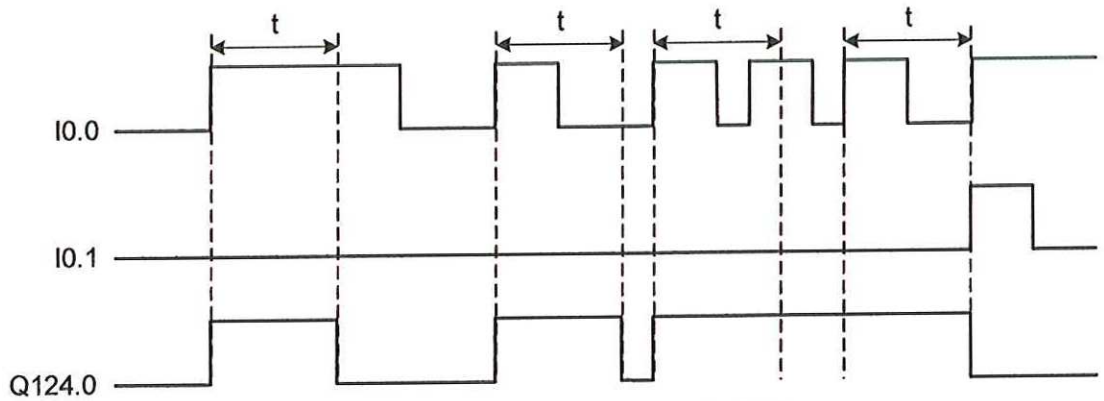
El tiempo es representado a través de una constante, la cual tiene la siguiente estructura:

S5T# es un término neutro que indica constante de tiempo, es obligatorio que se escriba siempre que se quiera referir a un período de tiempo, ***10S*** indica diez segundos, esta constante puede ser modificada según sea la necesidad del usuario, como indica segundos también pude señalar horas (***5h*** cinco horas), minutos (***1m*** un minuto), y milésimas de segundos (***500ms*** 500 milisegundos).

2) Parametrizar y arrancar temporizador como impulso prolongado.



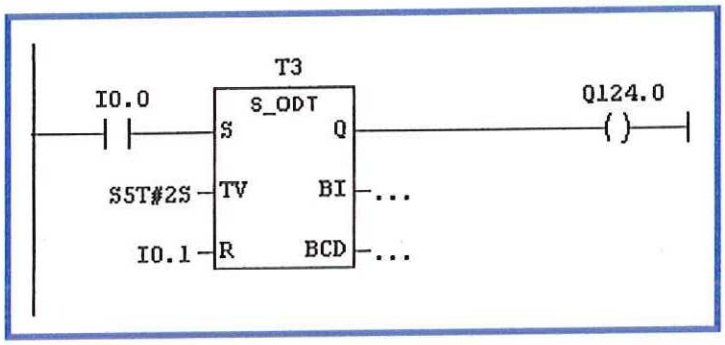
Esta opción denominada S_PEXT (S_VIMP en alemán), funciona de la siguiente forma, cuando la entrada S (conectada a I0.0) recibe un cambio de 0 a 1, la salida Q (conectado a Q124.0***) se enciende tanto tiempo como lo indica TV*** (o TW en alemán) y luego se apaga (Q=0). Si S genera otro cambio de 0 a 1 mientras el temporizador trabaja el temporizador vuelve a iniciar. Un paso de S a 0, no ocasiona cambios en Q.



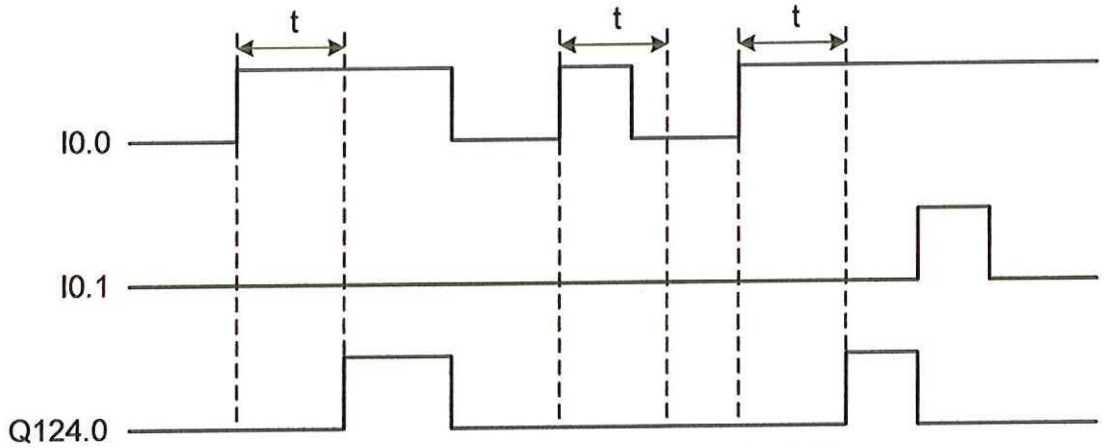
t= tiempo programado (TV)

Nota: *R, BCD y BI trabajan de igual forma.*

3) Parametrizar y arrancar temporizador como retardo a la conexión.



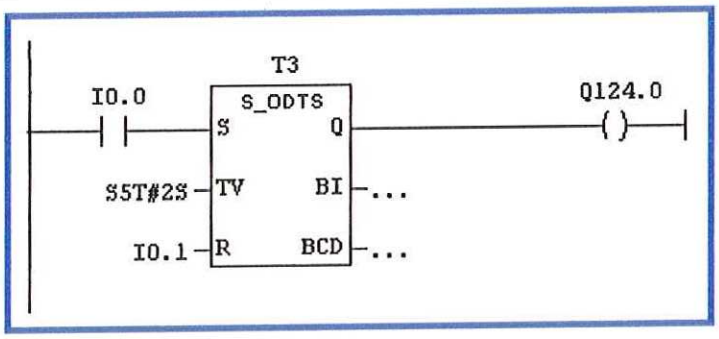
Esta opción denominada S_ODT (S_EVERZ en alemán), funciona de la siguiente forma, cuando la entrada S (conectada a I0.0) es 1 y se mantiene así, la salida Q (conectado a Q124.0) ***se tardará el tiempo que indique TV*** (o TW en alemán) ***para activarse***. Si R=1 en cualquier momento Q se apagará y el conteo se detiene.



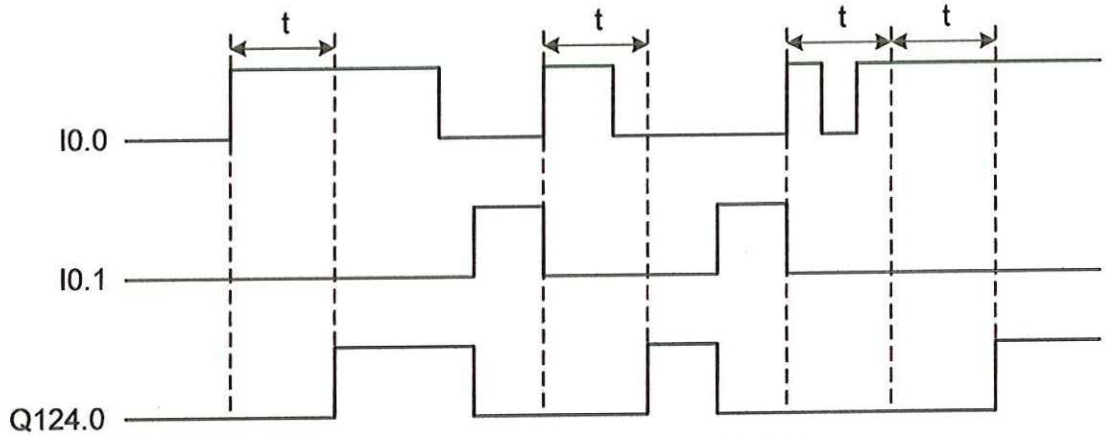
t= tiempo programado (TV)

Nota: *BCD y BI trabajan de igual forma.*

4) Parametrizar y arrancar temporizador como retardo a la conexión con memoria.

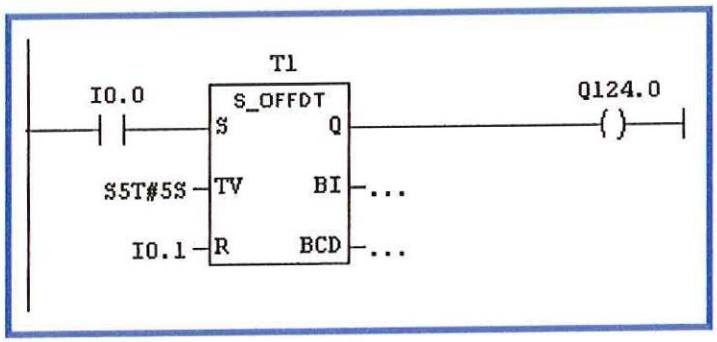


Esta opción denominada S_ODTS (S_SEVERZ en alemán), funciona de la siguiente forma, cuando la entrada S (conectada a I0.0) cambia de 0 a 1, la salida Q (conectado a Q124.0) ***se tardará el tiempo que indique TV*** (o TW en alemán) ***para activarse, si se vuelve a presentar un nuevo cambio de 0 a 1 el contador se reinicia***. Si R=1 en cualquier momento, Q se apagará y el conteo se detiene.

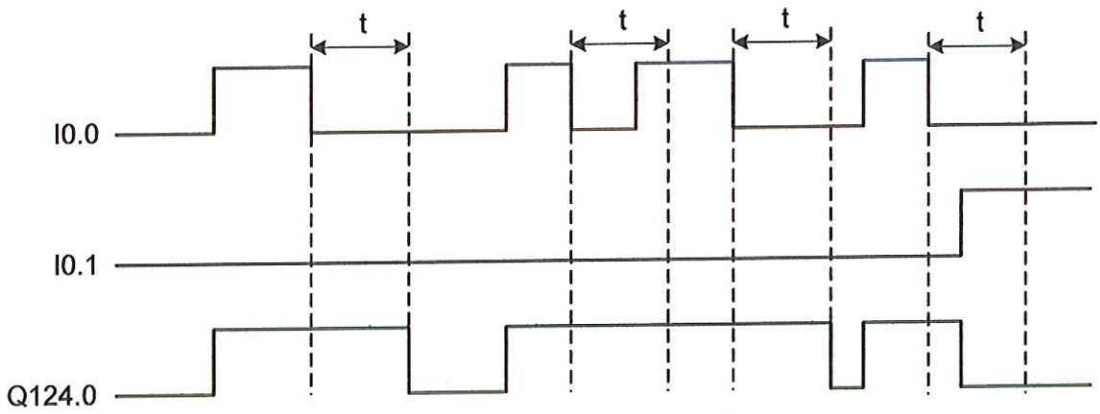


t= tiempo programado (TV)

5) Parametrizar y arrancar temporizador como retardo a la conexión.



Esta opción denominada S_OFFDT (S_AVERZ en alemán), funciona de la siguiente forma, cuando la entrada S (conectada a I0.0) cambia de 0 a 1, la salida Q (conectado a Q124.0) ***se mantiene encendido hasta el tiempo que indique TV*** (o TW en alemán), ***si S se mantiene en 1 la salida se estará siempre activada y el contador se detiene***. Si R=1 en cualquier momento Q se apagará. Un cambio de S a 0 no afecta el trabajo del temporizador.

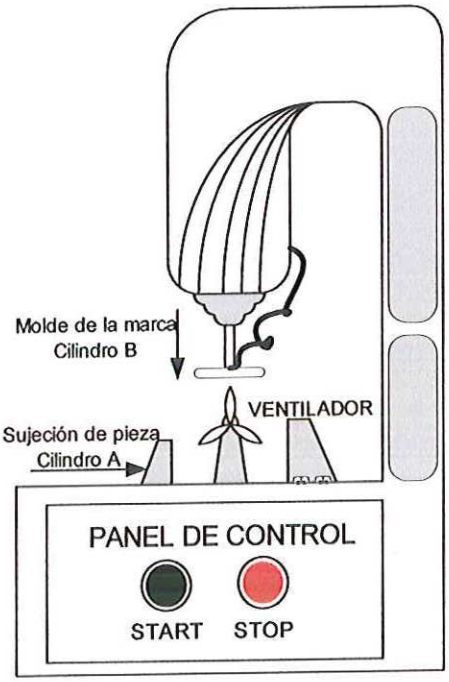


t= tiempo programado (TV)

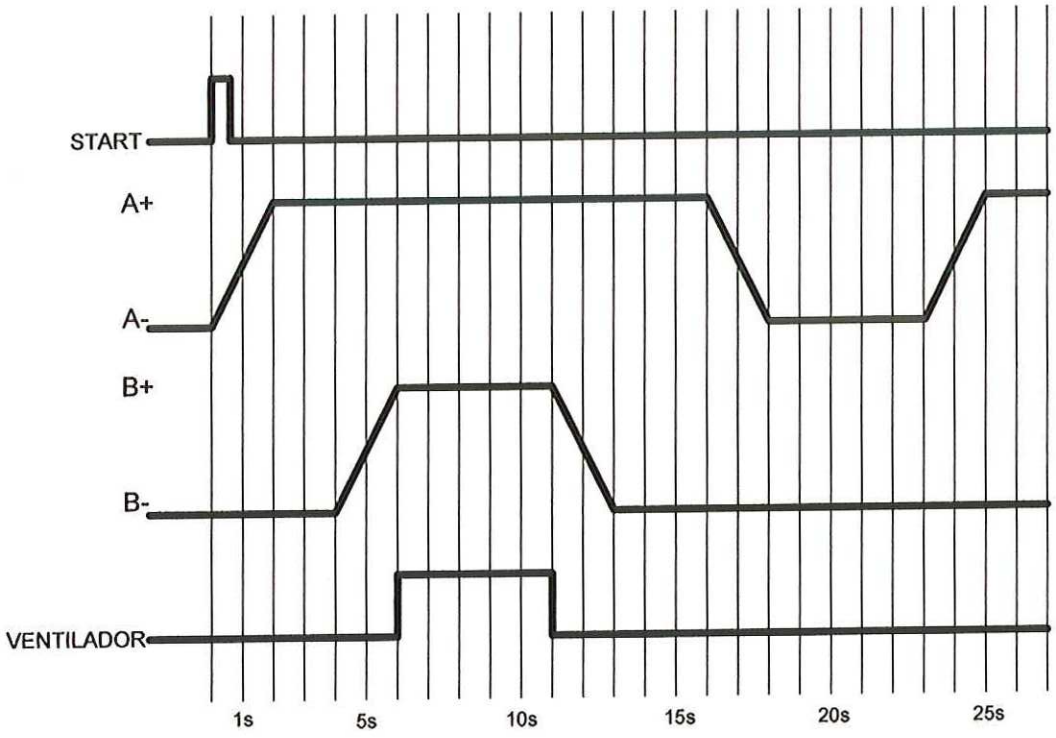
A continuación se procederá a implementar estas herramientas, pero antes de iniciar con este paso, **conecte los elementos** mencionados en el punto *4 Lista de materiales* a las direcciones de entradas y salidas que les parezcan convenientes.

En una empresa de fabricación de utensilios de aseo plásticos (recogedores, escobas, cestas de basura, cepillos, etc), utilizan una máquina encargada de poner la marquilla característica de la empresa. La máquina consta de:

- Un cilindro de doble efecto (A) que es el encargado de la sujeción de la pieza.
- Un cilindro de doble efecto (B) que es el encargado de poner la marca a cada uno de los utensilios de aseo.
- Un ventilador que se encarga de refrigerar el utensilio para que el plástico no se adhiera al molde.
- Un botón de START (normalmente abierto sin enclavamiento) el cual inicia el proceso.
- Un botón de STOP (normalmente cerrado sin enclavamiento) el cual finaliza el proceso.
- Dos sensores de presencia para el pistón A.
- Dos sensores de presencia para el pistón B.



Los sensores emiten señal cuando el pistón ha entrado (A- , B-) y salido totalmente (A+ ,B+). Ambos pistones son controlados con válvulas de doble bobina. El proceso inicia al presionar START, y desarrolla el siguiente ciclo continuamente mientras no se presione STOP.



El tiempo que se toma la máquina para volver a iniciar un ciclo es de 5 segundos, tiempo que tarda en llegar una nueva pieza. Al presionar STOP se cumple lo que haga falta para finalizar el ciclo y se apaga la máquina. Diseñe el programa que efectúa este funcionamiento. La expansión y el regreso de los pistones son inmediatos.

La solución para este proceso es la siguiente:

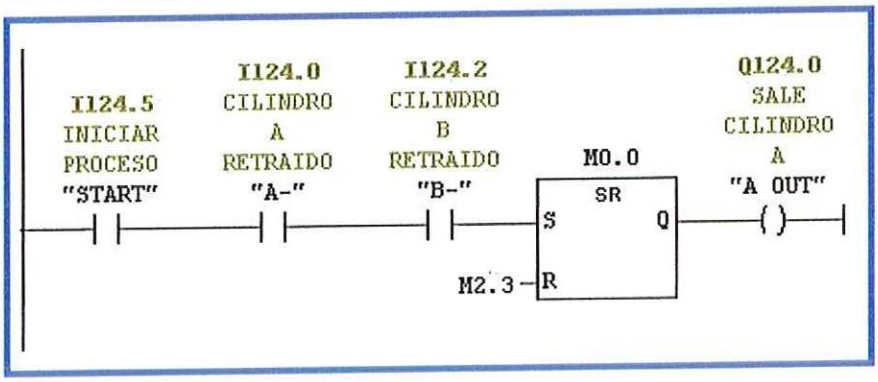
Primero se diseñará la secuencia de los cilindros sin tener en cuenta los tiempos en la ejecución.

Cada bobina de la válvula es controlada por un F-F, es necesario recordar que estas válvulas no regresan a su puesto una vez son desactivadas (no tienen retroceso por muelle) y tienen la capacidad de memorizar la última orden de posición emitida. Nunca deben estar activadas las dos bobinas al tiempo, ni dejar mucho tiempo una activa.

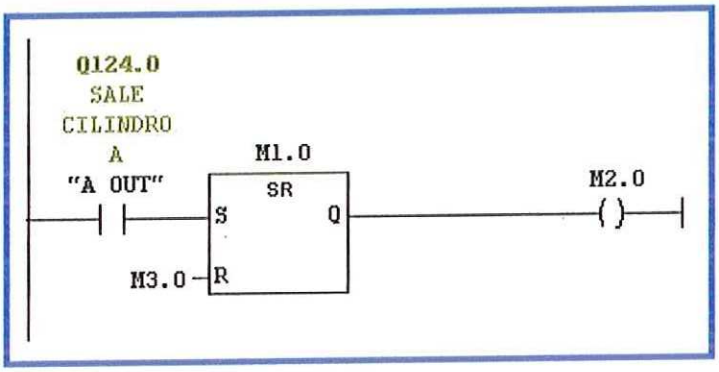
Para su mayor comodidad asígnele nombres a cada una de las direcciones antes de iniciar con la elaboración del programa, relacione la siguiente tabla de acuerdo con sus conexiones realizadas a los elementos.

	Estado	Símbolo	Dirección	Tipo de dato	Comentario
1		A-	I 124.0	BOOL	CILINDRO A RETRAIDO
2		A IN	Q 124.1	BOOL	ENTRA CILINDRO A
3		A OUT	Q 124.0	BOOL	SALE CILINDRO A
4		A+	I 124.1	BOOL	CILINDRO A EXTENDIDO
5		B-	I 124.2	BOOL	CILINDRO B RETRAIDO
6		B IN	Q 124.3	BOOL	ENTRA CILINDRO B
7		B OUT	Q 124.2	BOOL	SALE CILINDRO B
8		B+	I 124.3	BOOL	CILINDRO B EXTENDIDO
9		START	I 124.5	BOOL	INICIAR PROCESO
10		STOP	I 124.4	BOOL	APAGAR PROCESO
11		VENTILADOR	Q 124.4	BOOL	REFRIGERACION
12					

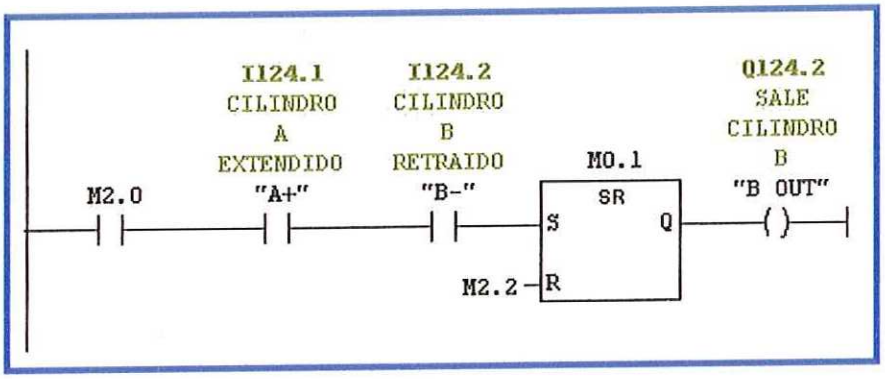
Primero se programará la salida del pistón de sujeción de la pieza, para eso se debe tener en cuenta, que condiciones (sensores activos) hay antes de que la acción se realice. Para la primera orden, se supone que tanto A como B están retraídos, por lo tanto los sensores asignados a estas posiciones deben estar activos (A- ,B-). El sistema solo debe esperar el accionamiento de START para que inicie, Un cumplimiento de todas estas órdenes activa el F-F que controla a la bobina A OUT. R se direcciona con la marca M2.3, más adelante se explicará el porqué de que esta orden desactive la bobina.



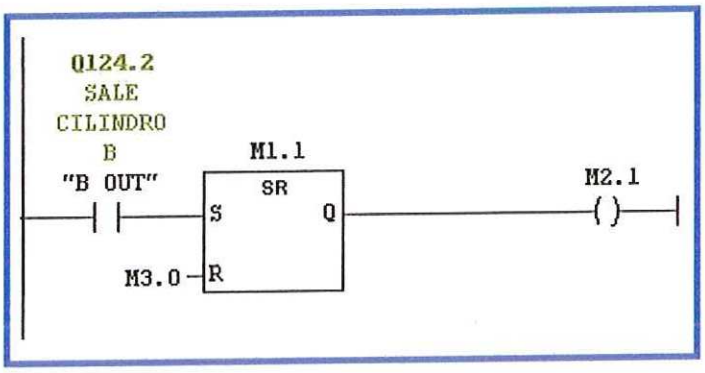
El paso anterior es memorizado en M2.0, para esto se usa otro F-F, el cual se activa una vez el pistón A salga, R se conecta con la marca M3.0 la cual es una memoria bandera que se activará en el momento de terminar el ciclo, más adelante se programará esta opción.



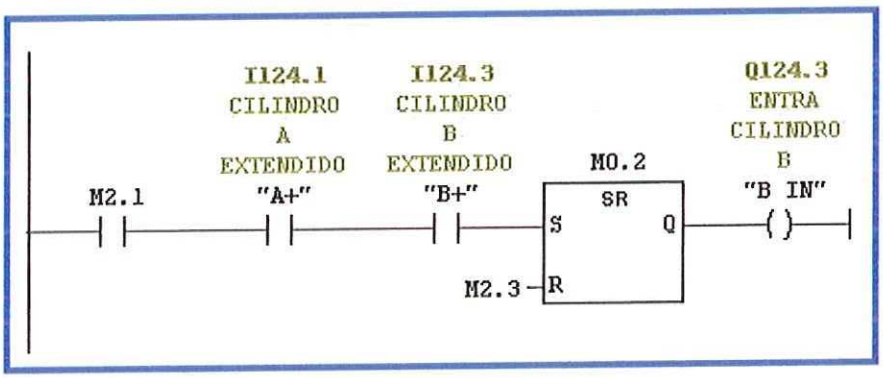
Este paso es el que se ocupa de activar la bobina que controla la salida del pistón B, las condiciones de esta etapa son A+, B- y que la orden anterior se haya cumplido (una activación de la marca M2.0). R se conecta con la marca M2.2.



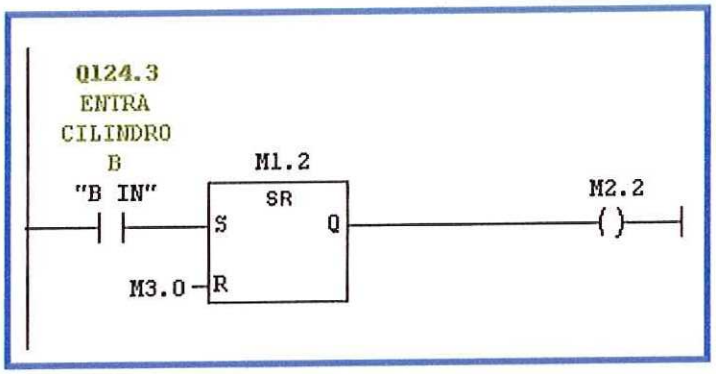
La etapa se cumple y el paso es memorizado de la misma forma que con A, para este caso se uso la marca M2.1.



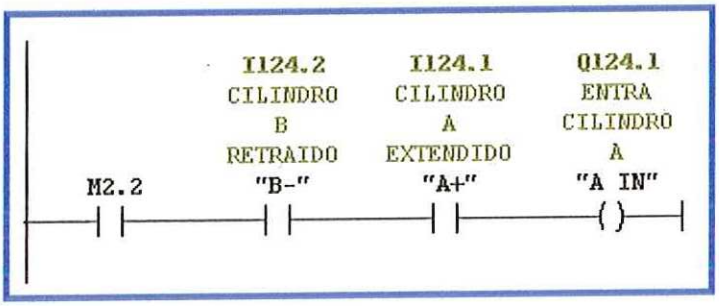
El siguiente paso es el regreso del pistón B después de haber realizado su función de marcar el utensilio. Las condiciones para que se cumpla este paso son A+, B+ y M2.1 (el paso anterior realizado). R se conecta con la marca M2.3.



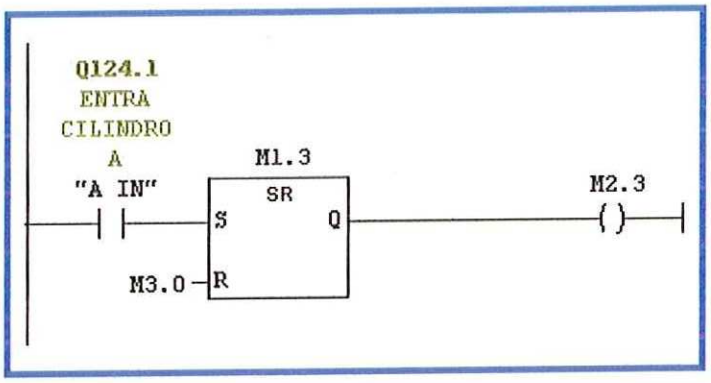
Se memoriza este paso en M2.2. Apenas se activa esta marca, se encargará de resetear el F-F que controlaba a la bobina de B OUT (paso 3). De esta forma se evita que ambas bobinas estén activas a la vez.



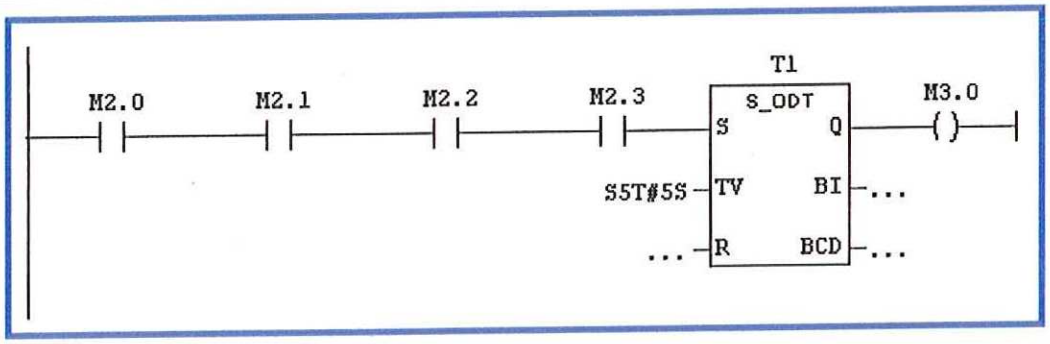
El siguiente paso, que es la liberación de la pieza no requiere de memorización por ser el último. Las condiciones para que se pueda ejecutar son B-, A+ y que el paso anterior se halla cumplido.



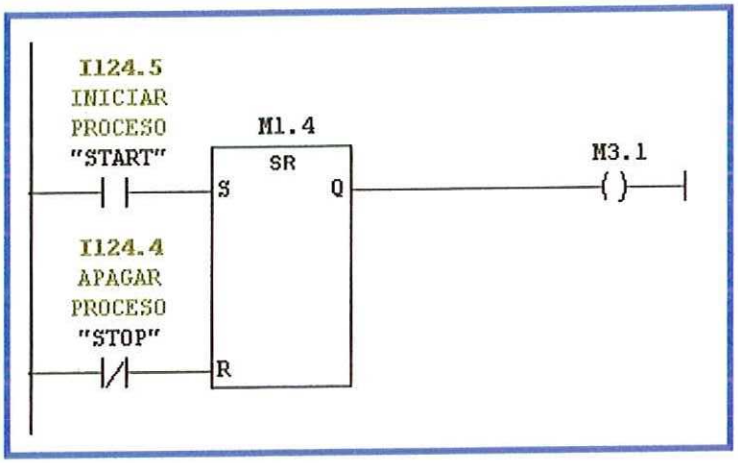
El paso se memoriza en M2.3, la activación de esta memoria resetea a la bobina que controla A OUT (paso 1).



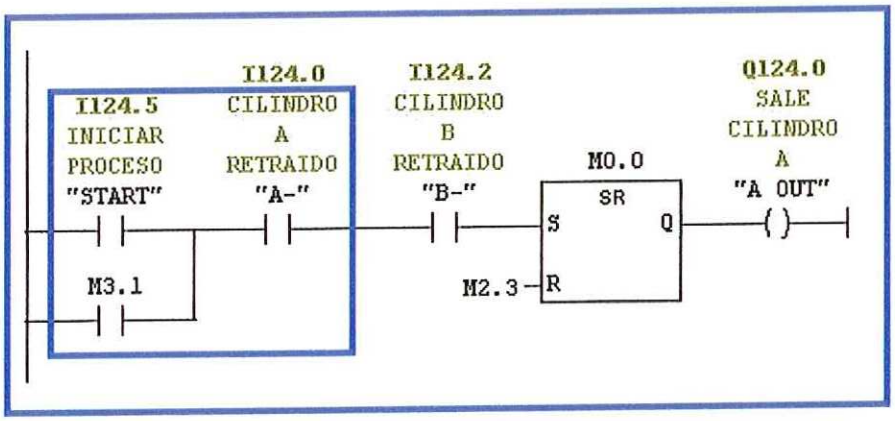
Ya una vez cumplido el ciclo es necesario que vuelva a iniciar, para que esto sea posible se deben resetear todas las memorias de los pasos cumplidos. Y para esto está M3.0, el cual se utilizó en los pasos 2, 4, 6 y 8 como elemento que desconectaba los F-F (R). Su activación se hará cuando todas las marcas se activen, es decir se hayan realizado todos los pasos o etapas. El temporizador de cinco segundos es para retrasar la activación de M3.0 (S_ODT), el reinicio del proceso y que permita hacer el paso 8 antes de que lo resetee, fíjese que en el paso anterior se utiliza M3.0 como desactivador del F-F. Si la activación de M3.0 fuera sin retardo no permitiría que M2.3 se habilitara y no se podría realizar el regreso del pistón A porque las bobinas A IN y A OUT estarían activadas a la vez.



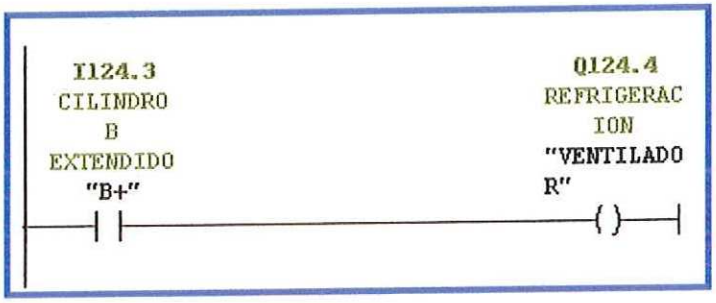
Para garantizar que el proceso se reinicie después de haber oprimido START, se memoriza su pulsación con un F-F, el dato queda almacenado en la marca M3.1. Para apagar el proceso se conecta a la entrada R el botón STOP.



Al ser activado la marca M3.1 (puesta en **OR** con START) esta mantendrá el proceso en operación hasta que se presione STOP el cual hará que se desactive y se corte el lazo de activación del F-F que controla a la bobina de A OUT.

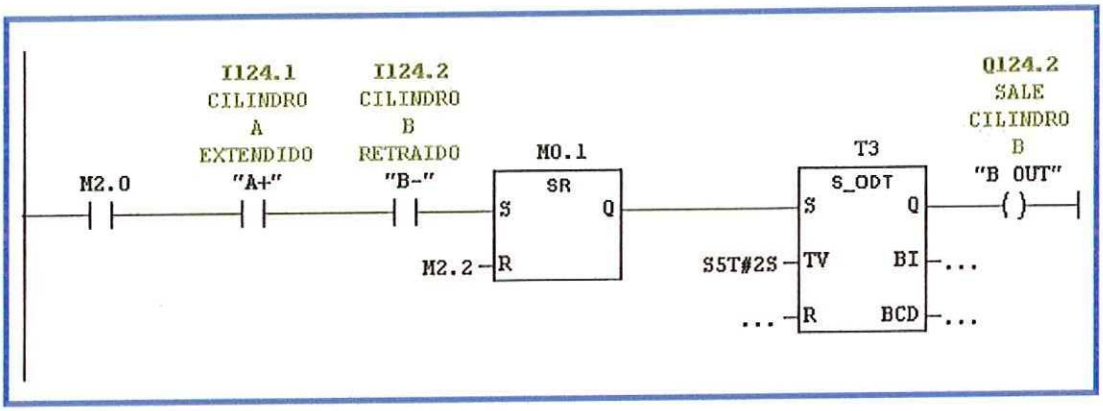


La ventilación del proceso es activada cuando el sensor de presencia de B+ detecta la salida del pistón, el ventilador se desactiva una vez el pistón regresa.

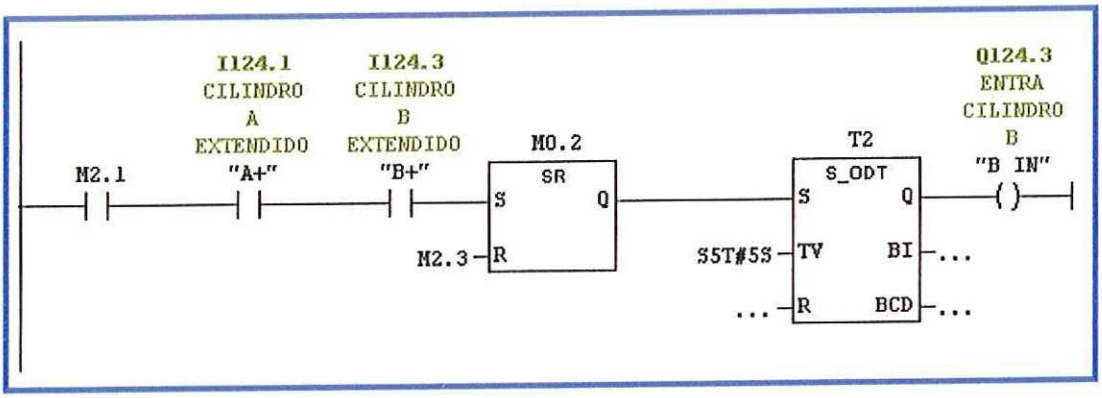


Ahora se insertarán al proceso los tiempos que se exigen para su correcto funcionamiento.

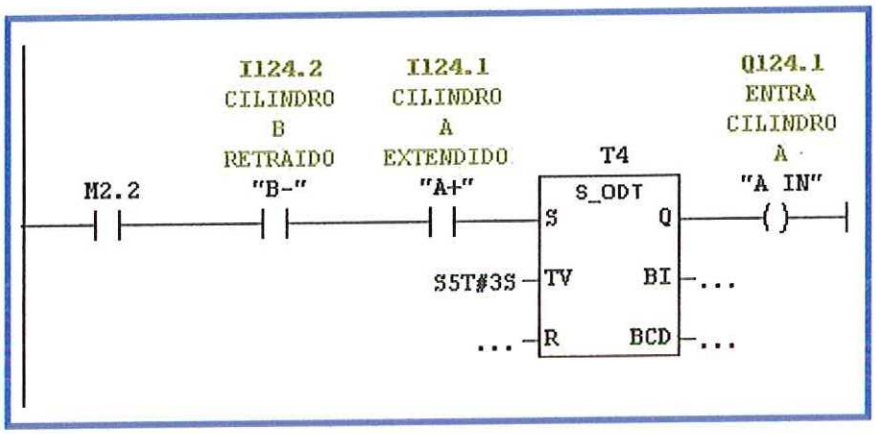
Retardo a la salida de B.



Retardo a la entrada de B.



Retardo a la entrada de A.



6. ACTIVIDAD COMPLEMENETARIA

Para la actividad complementaria de esta práctica, se dispuso la implementación de una práctica abierta, la cual exigirá la implementación de los conceptos enseñados en este documento.

Sin embargo es necesario establecer unos parámetros mínimos que la actividad debe contener para garantizar su validez ante la presentación al profesor u orientador, estos parámetros se mencionan a continuación.

- Utilizar dos pistones ya sean de simple o de doble efecto.
- Tener un seleccionador de ciclo único o continuo, similar al implementado en la práctica 4.
- Utilizar temporizadores.
- El proceso controlado debe ser real y debe mostrar algún grado de dificultad en el diseño del programa.

Existe una segunda opción dentro del marco de la práctica abierta de este documento, y es el control de un motor paso a paso con el PLC, el programa debe contener:

- Un seleccionador de dirección del movimiento del motor.
- Un iniciador de proceso.
- Un finalizador de proceso.

Para este caso, no olvide tener en cuenta si el nivel de tensión requerido por el motor se encuentra dentro de los limites permitidos por el PLC, el número de cables y la secuencia de estos.

Si encuentra algún impedimento para la realización de la actividad, detalle las causas debidamente fundamentadas y sus posibles soluciones.

PRÁCTICA 6
TRANSFERENCIA DE PAQUETES DE DATOS

PRÁCTICA 6
TRANSFERENCIA DE PAQUETES DE DATOS

1. OBJETIVOS

- ✓ Identificar los diferentes tipos de datos que se pueden trabajar con el PLC.
- ✓ Utilizar e implementar en proyectos las herramientas que proporciona el PLC para transferir datos.

2. CONCEPTOS FUNDAMENTALES

Las prácticas trabajadas hasta el momento sólo se han enfatizado en la operación de variables que tienen sólo dos estados, encendido o apagado que en el entorno digital se conocen como 1 y 0, ó BOOL el cual opera con el similar TRUE y FALSE.

Una variable tipo BOOL tiene el tamaño de un bit, existen variables mucho más grandes con las que un PLC tiene la capacidad de trabajar. Un byte es la variable que le sigue en tamaño, tiene una extensión de 8 bits, normalmente las salidas y entradas digitales se direccionan por bytes.

Un Word (palabra) son dos bytes, es decir 16 bits algunas herramientas propias del entorno de programación trabajan con paquetes de datos de 16 bits, y una Double Word es una variable cuya extensión es de dos Word's (cuatro bytes, 32 bits), una variable tan grande es útil para enviar paquetes de datos vía redes de comunicación, como Profibus o Ethernet.

Un PLC también puede trabajar con números enteros o variables tipo INT como se conocen, una variable tipo INT tiene una extensión de 16 bits, comprende números negativos desde -32768 hasta el positivo 32767. El Double INT es una variable similar del doble de tamaño (32 bits) su rango va desde -2147483648 hasta 2147483647. Los números en coma flotante (REAL) usados para las variables de tipo analógica tienen una extensión cuyo límite superior es 3.402823e+38 y se reduce hasta1.175 495e-38, un número tipo REAL tiene una extensión de 32 bits.

Existen otras variables que dan referencia al tiempo las cuales son tema de otra práctica.

3. ACTIVIDAD PREVIA

a) Consulte la nomenclatura que utiliza el PLC Siemens Simatic S7 para definir los diferentes tipos de datos.

4. LISTA DE MATERIALES

1) 4 bombillos de 12 o 24 voltios.
2) 1 electroválvula de una bobina con retroceso por muelle.
3) 1 pistón de simple efecto.
4) 1 ventilador.
5) 1 pulsador normalmente cerrado.
6) 1 pulsador normalmente abierto.

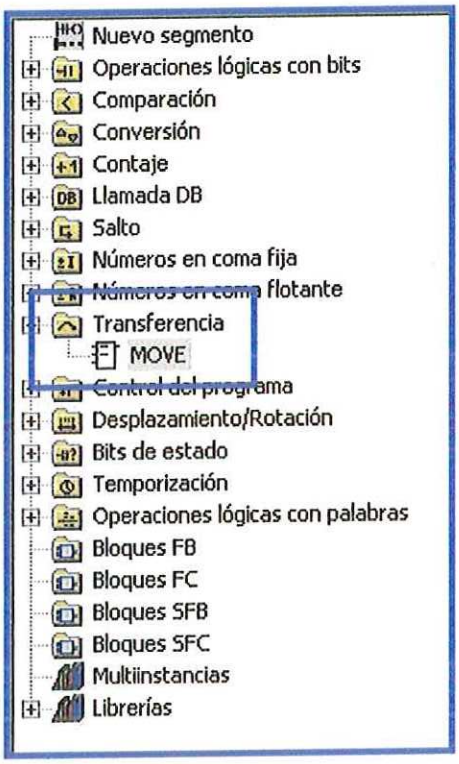
5. DESARROLLO METODOLÓGICO

Nota: *para el desarrollo de esta actividad es necesario haber elaborado la Práctica 5, Temporizadores.*

Abrir ***SIMATIC/Administrador Simatic***, cree un nuevo proyecto con el nombre *Práctica 6*.

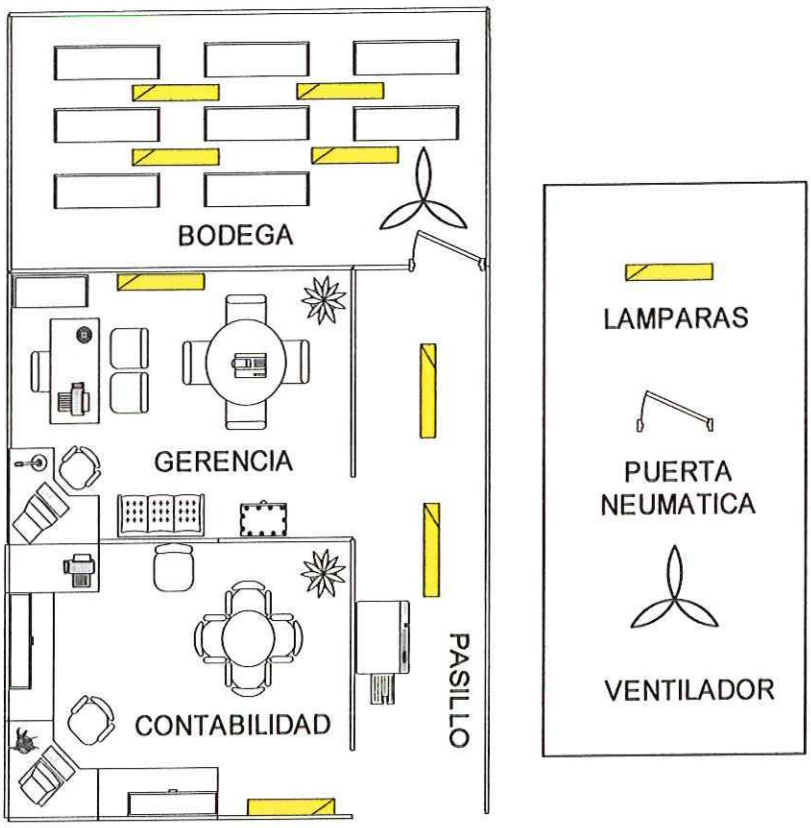
Entre al Bloque de organización No.1 (OB1). Cerciórese que la ventana de *Elementos de Programa* este abierta, para esto vaya al menú *VER* y verifique que la opción *Vista General* esté seleccionada.

En este menú encontrará una carpeta llamada *Transferencia* la cual despliega la función denominada *MOVE*. Para agregar esta función al sitio de trabajo se hace igual que con las funciones anteriormente vistas.



Para introducir este tema se irá directamente a la aplicación de esta función en un problema planteado a continuación.

En una empresa la cual labora las 24 horas del día se a implementado a través de los mismos PLC's que controlan los procesos; un panel de control para las instalaciones eléctricas de las oficinas el cual es accesible para el personal de vigilancia.



El panel consiste en una serie de interruptores ON/OFF que están enlazados mediante el PLC al sistema eléctrico de las instalaciones. La gerencia se ha trasladado al mismo lugar en donde la fábrica labora, por esto se necesita agregar un nuevo segmento en el programa el cual controle el nuevo bloque el cual está compuesto por: gerencia general, contabilidad, bodega y un pasillo común, cada uno con sus respectivas lámparas; en la bodega hay un sistema de ventilación y una puerta de seguridad que es accionada desde el panel de control. Este panel consta además de un botón de habilitación (START) para que pueda operar y uno de deshabilitación (STOP).

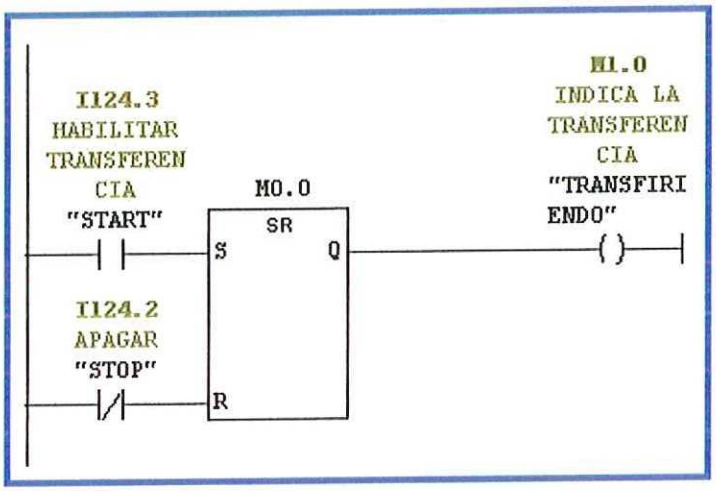
Ahora se procederá a desarrollar el programa el cual controlará el sistema eléctrico del bloque de gerencia.

Conecte los elementos mencionados en el punto 3 *Lista de Materiales*, **procure que todos queden comprendidos dentro de un mismo Byte.**

En la *tabla de símbolos* edite las direcciones correspondientes al START (pulsador normalmente abierto), STOP (pulsador normalmente cerrado) y etiquete un bit de las marcas con el nombre de *TRANSFIRIENDO* (para el ejemplo se usó el bit M1.0).

	Estado	Símbolo	Dirección	Tipo de dato	Comentario
1		START	I 124.3	BOOL	HABILITAR TRANSFEREN...
2		STOP	I 124.2	BOOL	APAGAR
3		TRANSFIRIENDO	M 1.0	BOOL	INDICA LA TRANSFEREN...
4					

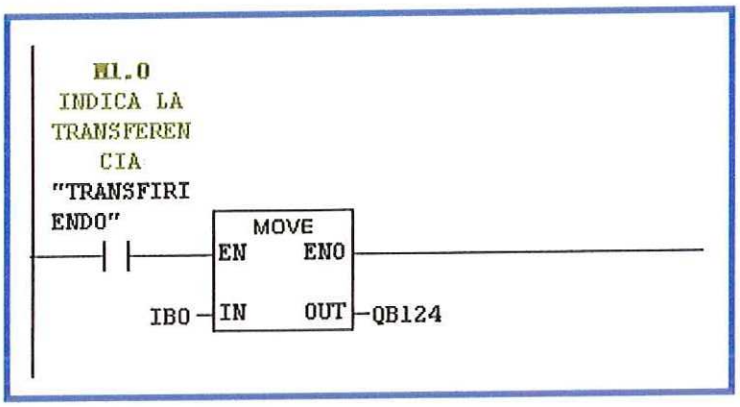
A través de un flip-flop tipo SR se almacenará el pulso correspondiente al START (conectado a S),cuando se pulse, este hará que *TRANSFIRIENDO* (conectado a Q), se conserve activado hasta que se presione STOP (conectado a R).



Arrastre el bloque de transferencia *MOVE* a un nuevo segmento, este bloque consta de una entrada EN la cual es encargada de habilitar la transferencia de datos, la entrada IN tiene la dirección de los datos origen (es decir de donde provienen) y la salida OUT tiene la dirección del destino (es decir a donde se transfieren los datos).

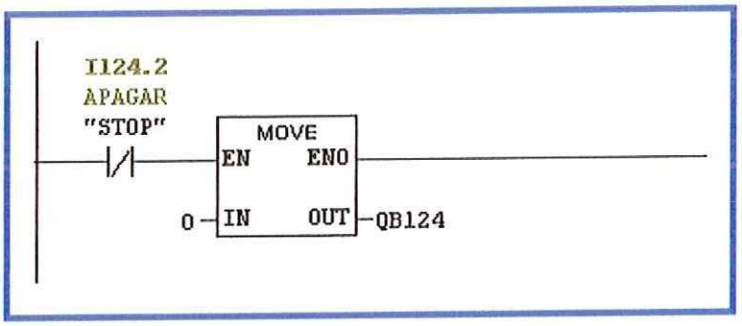
Para el caso del programa se desea transferir los datos provenientes de un panel de control (que se homologará por el módulo de entradas simuladas) a un byte de salidas digitales el cual debe corresponder al mismo que se utilizó para hacer las conexiones de los elementos.

Conectando a la entrada EN la marca *TRANSFIRIENDO,* se habilita el módulo para que este pueda realizar su función, en IN se digita la nomenclatura IB0 (Input Byte 0 "Cero"), en OUT se digita QB124 (Q "salida" Byte 124).



De esta forma mientras se mantenga habilitada la marca M1.0 , todo cambio que se presente en el Byte de entradas simuladas (Panel de Control) se verá reflejado en las salidas que controlan cada uno de los elementos (Q124).

Para detener la transferencia se presiona STOP. **Para evitar que alguna salida quede accionada al parar el proceso de transferencia**, se inserta en un nuevo segmento otro bloque de transferencia este se encargará de enviar un dato nulo (0) a la salida que controla los componentes.



Un cero en IN se interpreta como un Byte de ceros en OUT.

5. ACTIVIDAD COMPLEMENETARIA

En el bloque de gerencia del ejercicio anterior existen horarios de trabajo en cada oficina, por lo tanto es necesario desarrollar un programa que mantenga encendidas las luces solo en el momento en que se necesiten; de esta forma el ahorro de energía será aun mayor.

Los horarios de trabajo son los siguientes:

Gerencia general	10-12 am	3-5 pm
Contabilidad	8-12 am	2-6 pm

Bodega	6-12 am	----------
Ventilador	6-12 am	----------
Pasillo	----------	6-8 pm

Para fines prácticos cree una relación entre las horas y los segundos.

PRÁCTICA 7
CONTADORES Y COMPARADORES

PRÁCTICA 7
CONTADORES Y COMPARADORES

1. OBJETIVOS

- ✓ Identificar los diferentes contadores y comparadores disponibles en el entorno de programación STEP7.
- ✓ Implementar las funciones descritas a lo largo de esta práctica en el diseño de proyectos para PLC's.

2. CONCEPTOS FUNDAMENTALES

En los sistemas digitales se obtienen datos de información codificados en binario que continuamente se utilizan en alguna forma determinada. Por ejemplo, en los computadores, los comparadores son utilizados como parte de la circuitería de decodificación de direcciones, empleada en los computadores para seleccionar un dispositivo específico de entrada/salida o un área de memoria para guardar o recuperar un dato. En procesos industriales los compradores también son útiles en aplicaciones de control donde un número binario que representa a una variable física sobre la que se ejerce una acción de control (temperatura, caudal, velocidad) se compara con un valor de frecuencia. Los comparadores llevan generalmente el control de una variable que se va modificando sobre una constante que es el valor de referencia.

Cuando se quiera llevar un registro de una cantidad que va variando se utilizan los contadores, de esta forma se puede controlar un proceso para que este trabaje un número determinado de veces o para que se fabriquen una cantidad específica de existencias de un producto.

Las utilidades de los contadores y comparadores son tan variadas como el usuario las requiera, su uso va desde los circuitos electrónicos (integrados como el 74293 que realiza operaciones de conteo) hasta los algoritmos de programación, ahora se introducirán sus utilidades en el desarrollo de programas para el PLC.

3. ACTIVIDAD PREVIA

a) Repase los conceptos sobre tipos de datos que maneja el PLC analizados en prácticas anteriores.
b) Estudie los métodos que se pueden utilizar para convertir los números de decimal a binario, de binario a decimal, de BCD a decimal y de decimal a BCD.

4. LISTA DE MATERIALES

1) 2 bombillos de 12 o 24 voltios.
2) 1 electroválvula de una bobina con retroceso por muelle.
3) 1 pistón de simple efecto.
4) 2 sensores de posición.
5) 1 motor CD de 12 o 24 voltios.
6) 1 pulsador normalmente cerrado.
7) 1 pulsador normalmente abierto.

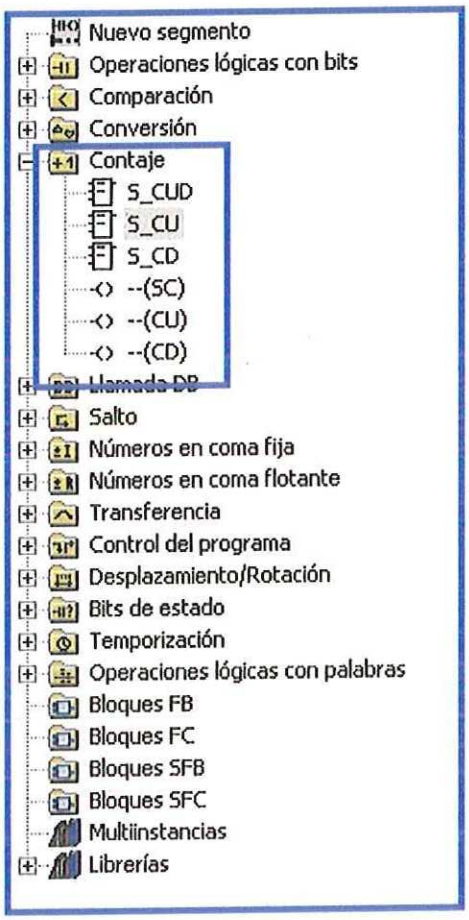
5. DESARROLLO METODOLÓGICO

Nota: *para el desarrollo de esta actividad es necesario haber elaborado la Práctica 6,Transferencia de Paquetes de Datos.*

Abrir ***SIMATIC/Administrador Simatic***, cree un nuevo proyecto con el nombre *Práctica 7*.

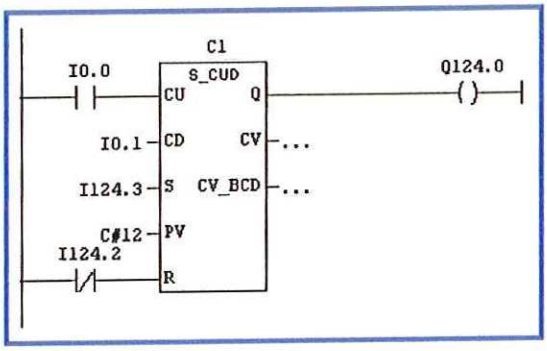
Entre al Bloque de organización No.1 (OB1). Cerciórese que la ventana de *Elementos de Programa* este abierta, para esto vaya al menú *VER* y verifique que la opción *Vista General* esté seleccionada.

En este menú encontrará una carpeta llamada *Contaje* en donde se despliegan los diferentes tipos de contadores disponibles para la programación. Para agregar cualquiera de estas funciones al proyecto se selecciona con el mouse y se arrastra hacia el segmento de trabajo.



Se comenzara por explicar la configuración y funcionamiento de los diferentes tipos de contadores.

- **CONTADOR ASCENDENTE_DESENDENTE.**



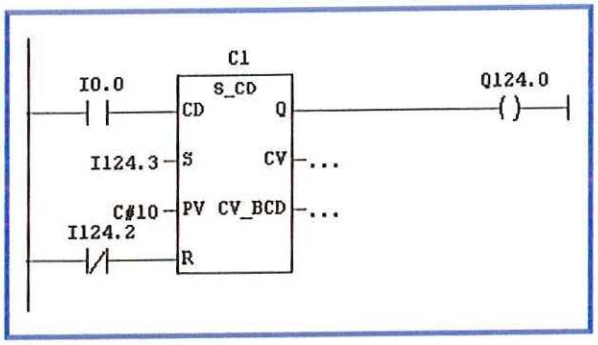
Este contador tiene la capacidad de contar pulsos de forma ascendente o descendente, el módulo consta de un selector de uso (CU,CD), un cargador de inicio de conteo (S), un inicio de conteo (PV), un reseteador (R), una

salida (Q) y un almacenador de conteo (CV, CV_BCD).

El contador S_CUD trabaja de la siguiente forma, si la entrada en S cambia de 0 a 1, el valor almacenado en PV es cargado en la función (en este caso el contador inicia desde 12, un inicio desde 0 sería C#0), un pulso (cambio de 0 a 1) proveniente de CU hace que el conteo sea ascendente de uno en uno (12 aumentaría progresivamente hasta un máximo de 999), si el pulso se hace desde CD el conteo será descendente de uno en uno (12 disminuiría hasta 0). El valor del conteo se visualiza consecutivamente en CV (valor del conteo en hexadecimal) y CV_BCD (valor del conteo en BCD); la salida Q será 1 siempre y cuando el valor del contador no sea cero.

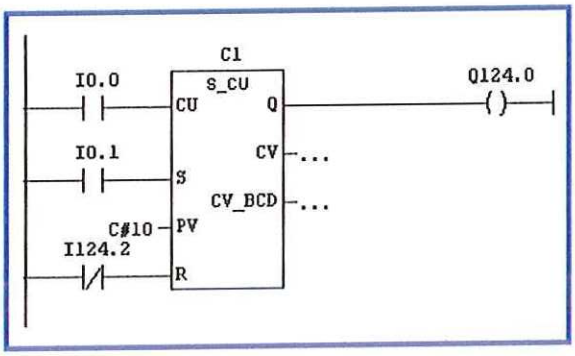
En otras palabras con el valor que toma el contador, si recibe un pulso por CU lo aumenta y si lo recibe por CD lo disminuye. Es importante asignarle un nombre al contador, el Simatic exige la letra C (en inglés) seguida de un número, no se pueden nombrar dos contadores de la misma forma.

- **CONTADOR DESCENDENTE**



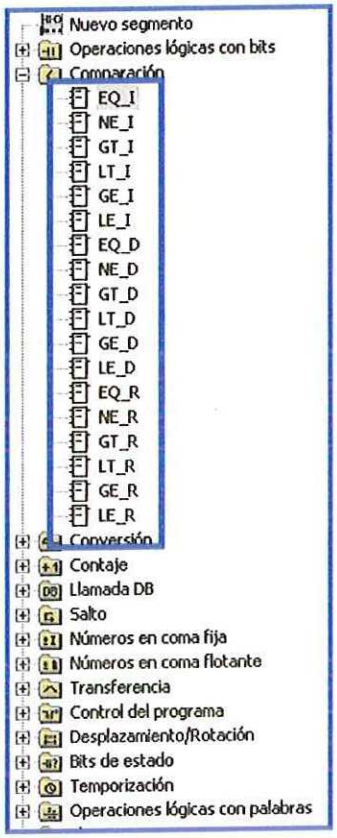
El funcionamiento de este contador es similar al anterior, sólo que este realiza un conteo descendente. El valor de inicio de conteo (PV) se carga con S, y cada vez que entra un pulso por CD el valor se decrementa de uno en uno. Cuando el valor llegue a cero Q se desactiva.

- **CONTADOR ASCENDENTE**



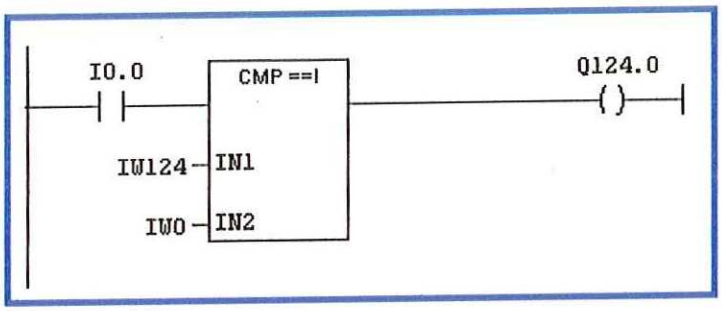
El funcionamiento de este contador es similar a los otros, esta función realiza un conteo ascendente. El valor de inicio de conteo (PV) se carga con S, y cada vez que entra un pulso por CU el valor se incrementa de a uno. El estado de Q será 1 siempre y cuando el valor del contador no sea cero.

Se explicaron anteriormente las tres primeras opciones que aparecen en la carpeta *Contaje*, las tres siguientes tiene la misma funcionalidad pero su programación es diferente y mas dispendiosa. En el menú *Elementos de Programa,* también se encuentra la carpeta *Comparación* aquí se tiene acceso a los diferentes tipos de comparadores disponibles para la programación.



A continuación se explicará la configuración y funcionamiento de los diferentes tipos de comparadores.

- **IGUAL QUE**



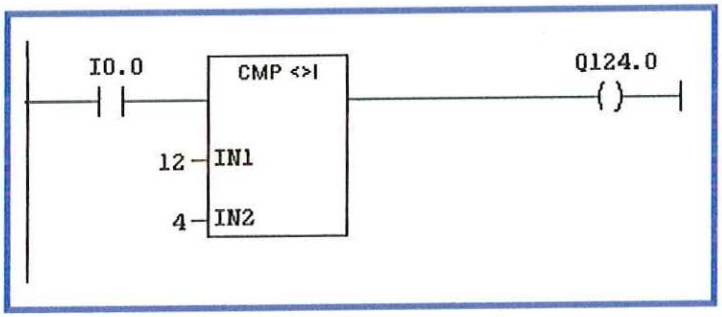
El comparador CMP==1 se encarga de determinar si los datos almacenados en las entradas IN1 e IN2 son iguales, de ser así la salida conectada a Q124.0 se activa. Esta comparación solo se lleva a cabo si el valor de la entrada conectada a I0.0 es 1.

Se pueden comparar datos que se encuentren en las salidas, entradas marcas y números enteros. Este comparador solo acepta valores de una

magnitud de 16 bits es decir una palabra, por eso la necesidad de escribir la letra W al momento de dar referencia a la dirección que se quiere comparar.

El ejemplo compara la entrada IW124 con IW0.

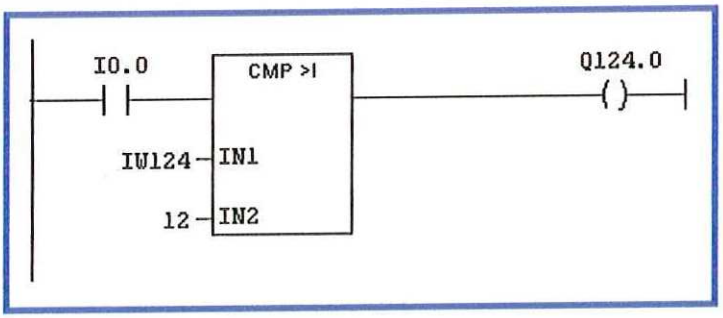
- **DIFERENTE A**



El comparador CMP<>1 se encarga de determinar si los datos almacenados en las entradas IN1 e IN2 son diferentes, de ser así la salida conectada a Q124.0 se activa. Esta comparación solo se lleva a cabo si el valor de la entrada conectada a I0.0 es 1.

El ejemplo compara a 12 y a 4 para determinar si son diferentes.

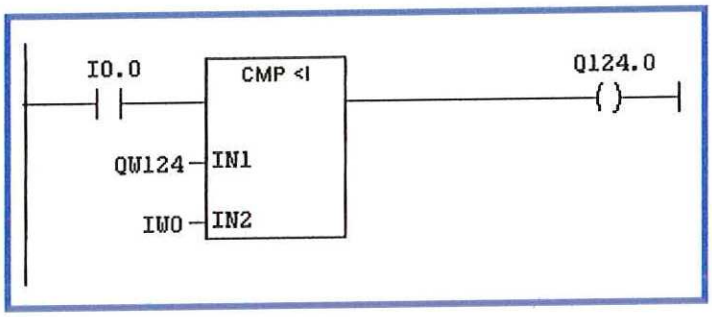
- **MAYOR QUE**



El comparador CMP>1 se encarga de determinar si IN1 es mayor que IN2, de ser así la salida conectada a Q124.0 se activa. Esta comparación solo se lleva a cabo si el valor de la entrada conectada a I0.0 es 1.

En el ejemplo se compara la entrada IW124 con el numero 12 (pasado a binario, es decir 00000000 00000110).

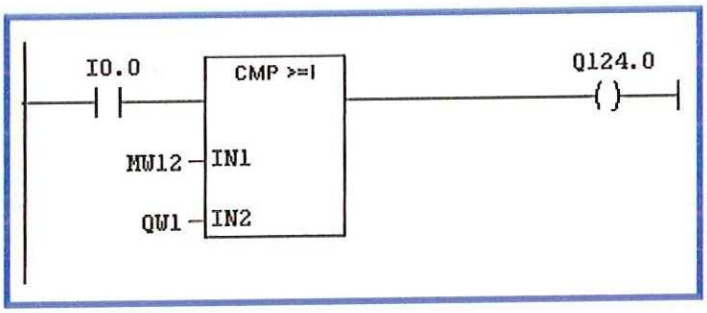
- **MENOR QUE**



El comparador CMP<1 se encarga de determinar si IN1 es menor que IN2, de ser así la salida conectada a Q124.0 se activa. Esta comparación solo se lleva a cabo si el valor de la entrada conectada a I0.0 es 1.

En el ejemplo se compara la salida QW124 con la entrada IW0.

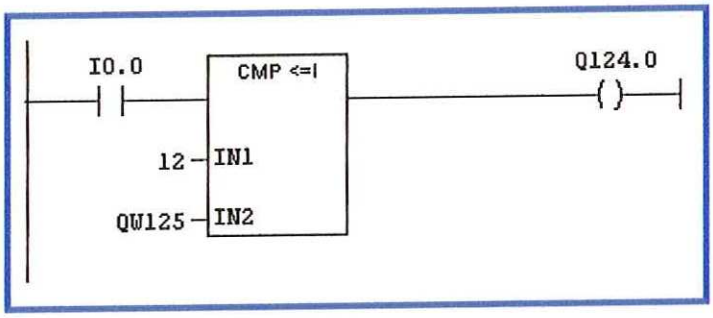
- **MAYOR O IGUAL QUE**



El comparador CMP>=1 se encarga de determinar si IN1 es mayor o igual que IN2, de ser así la salida conectada a Q124.0 se activa. Esta comparación solo se lleva a cabo si el valor de la entrada conectada a I0.0 es 1.

En el ejemplo se compara la marca MW12 con la salida QW1.

- **MENOR O IGUAL QUE**



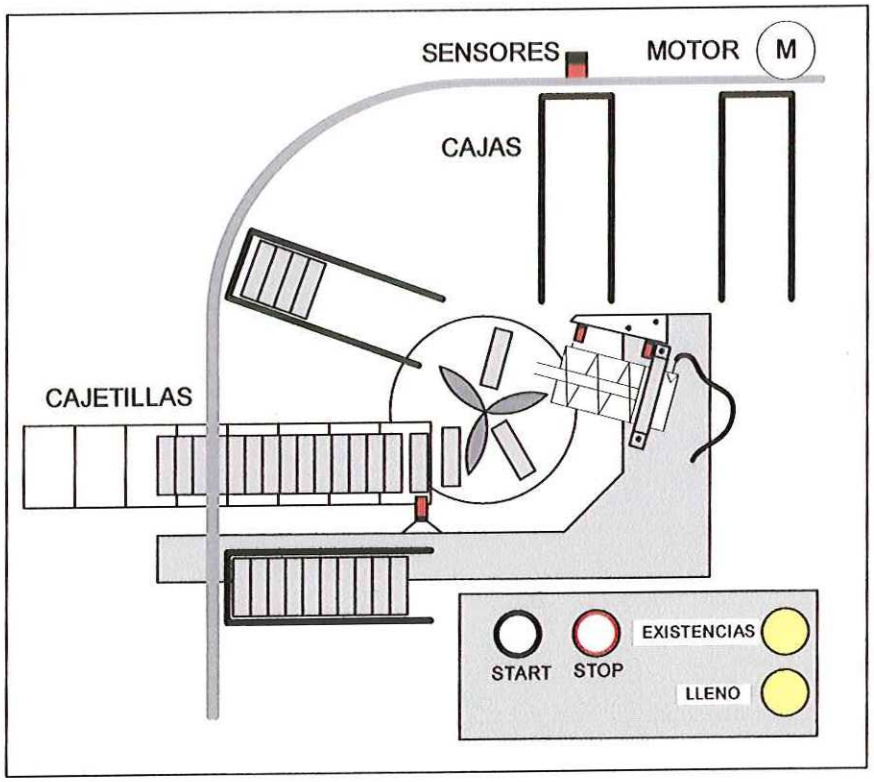
El comparador CMP<=1 se encarga de determinar si IN1 es menor o igual que IN2, de ser así la salida conectada a Q124.0 se activa. Esta comparación solo se lleva a cabo si el valor de la entrada conectada a I0.0 es 1.

En el ejemplo se compara el número entero 12 con la salida QW125.

Los comparadores que siguen en lista (EQ_D, NE_D, GT_D, LT_D, GE_D y LE_D) , trabajan con números con una extensión de 32 bits, es decir con Double Words. EQ_R, NE_R, GT_R, LT_R, GE_R y LE_R comparan números reales.

Ahora se pondrá en práctica las funciones anteriormente descritas.

Una empresa tabacalera tiene en su proceso de empaquetado una máquina que se encarga de crear paquetes de 10 unidades de cajetillas de cigarrillos. La máquina está compuesta por dos sensores de presencia encargados de notificar la existencia de empaques y cajas, una luz indicadora de la existencia tanto de cajas como de empaques, una luz que indica que la caja esta llena, un pistón de simple efecto que es el encargado de poner cada empaque en la caja, un motor el cual acciona una banda que se desplaza para dar paso a otra caja vacía; la maquina cuenta con botones de START y STOP para iniciar y parar el proceso respectivamente.



El proceso inicia al accionarse START y funcionara repetitivamente hasta que STOP sea presionado y detenga el proceso en su totalidad, o, si la existencia de cajas o cajetillas se agota la luz indicadora se apaga, por tanto el proceso se detiene.

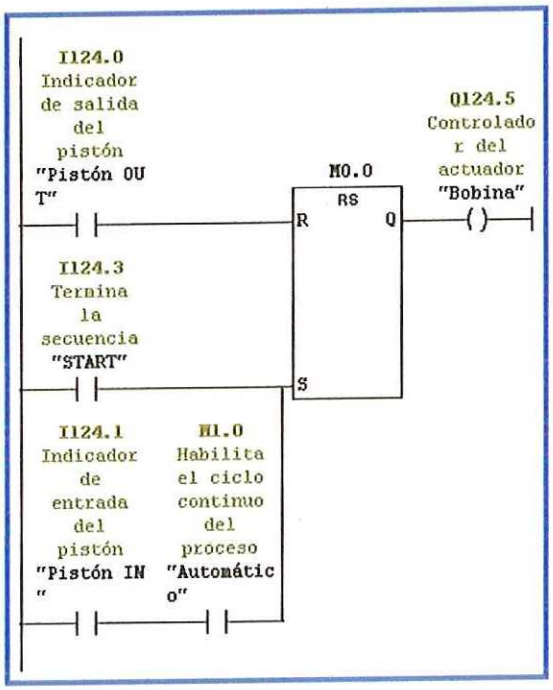
Implemente el programa a continuación el cual efectúa el proceso anteriormente descrito, conecte los componentes mencionados en la *Tabla de Elementos*.

Dada la magnitud del programa es importante desde su inicio editar la tabla de símbolos para evitar confusiones al momento de trabajar con las direcciones de los componentes sirva de referencia la siguiente tabla.

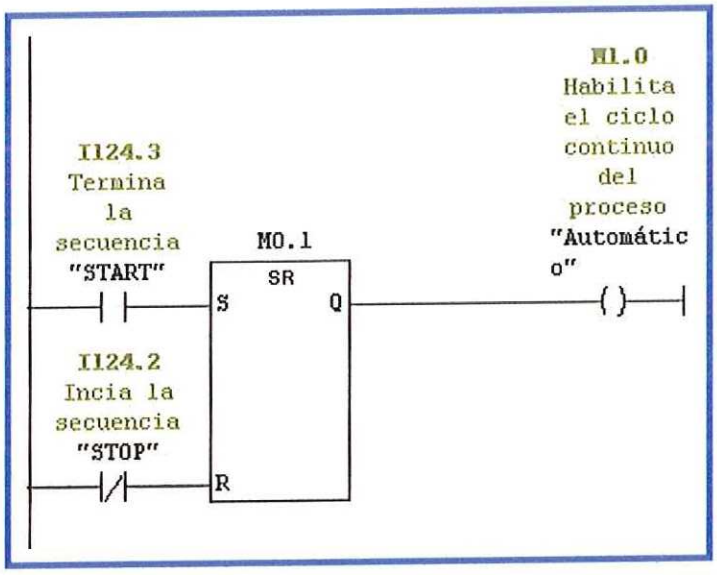
	Símbolo	Dirección	Tipo de dato	Comentario
1	Sensor de empaque	I 0.0	BOOL	Determina la existencia de empaques
2	Sensor de cajas	I 0.1	BOOL	Determina la existencia de cajas
3	Pistón OUT	I 124.0	BOOL	Indicador de salida del pistón
4	Pistón IN	I 124.1	BOOL	Indicador de entrada del pistón
5	STOP	I 124.2	BOOL	Termina la secuencia
6	START	I 124.3	BOOL	Inicia la secuencia
7	Automático	M 1.0	BOOL	Habilita el ciclo continuo del proceso
8	Parar proceso	M 3.0	BOOL	Detiene el proceso mientras el motor trabaja
9	Existencia de material	Q 124.0	BOOL	Enciende si hay empaques y cajas
10	Limite alcanzado	Q 124.1	BOOL	Enciende si se llegó al limite de cajas
11	Motor de la banda	Q 124.4	BOOL	Banda
12	Bobina	Q 124.5	BOOL	Controlador del actuador
13				

Nota: además de los componentes que están conectados, asigne dos bits de marcas con los nombres de *Automático y Parar proceso.*

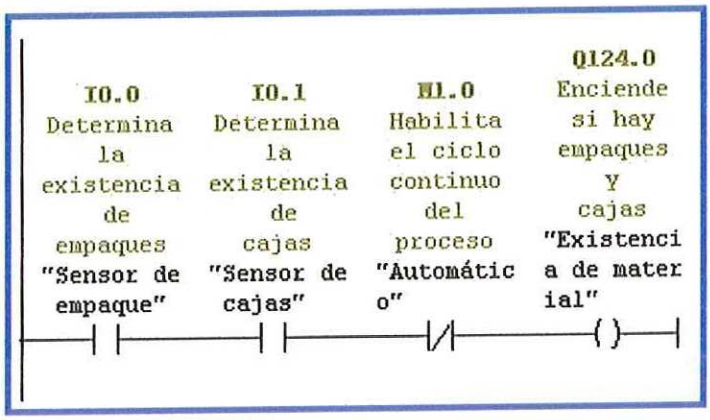
Desarrolle el esquema de control como el que se muestra en las figuras, este se encargará de hacer que el pistón salga y se devuelva indefinidamente (al presionar START) hasta que se presiona STOP (este esquema es el mismo al utilizado en al etiqueteadora de la Práctica 4).



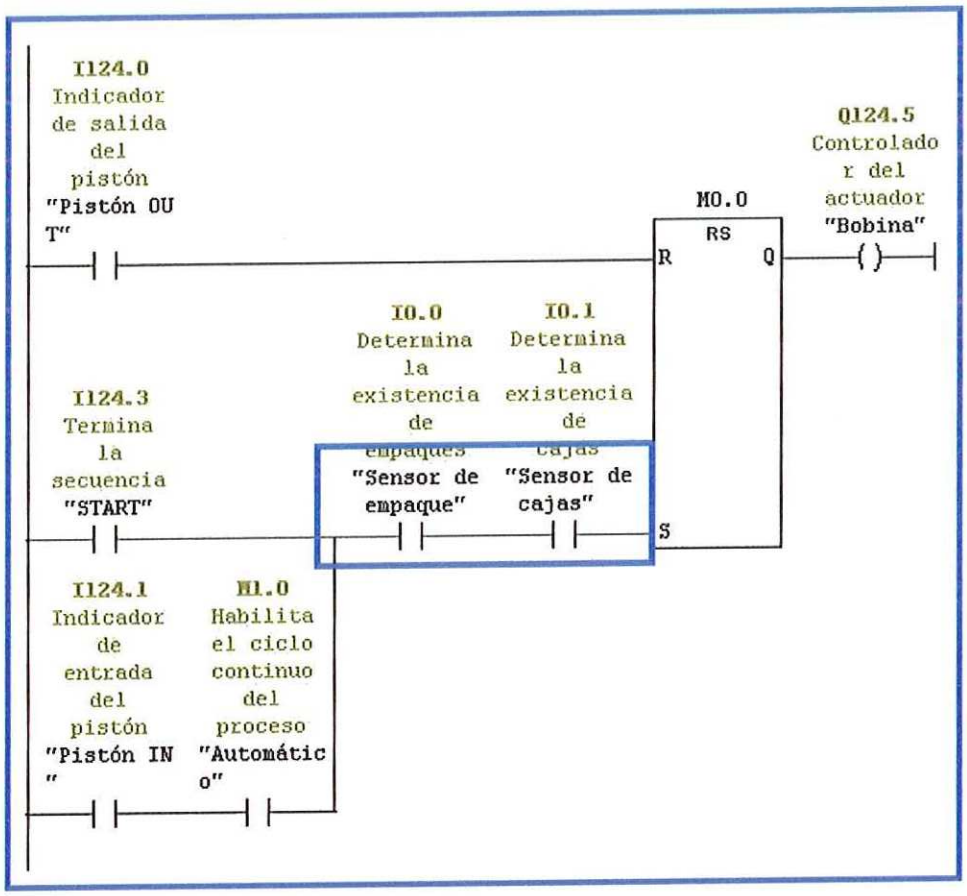
Habilitación del automático.



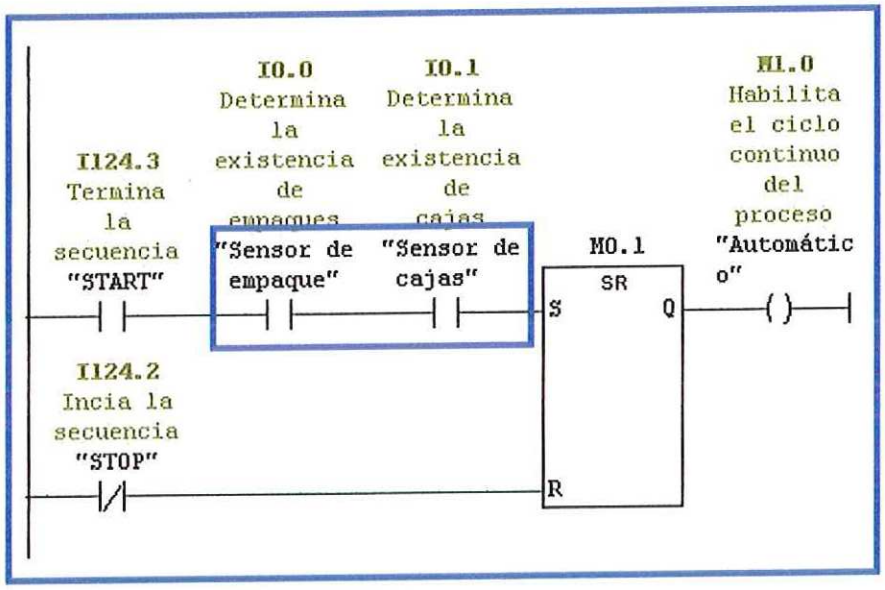
Una existencia de empaques y cajas activa la salida Q124.0 quien controla la luz indicadora de existencia de material. Automático negado en serie, desconecta la salida una vez el proceso inicia a trabajar.



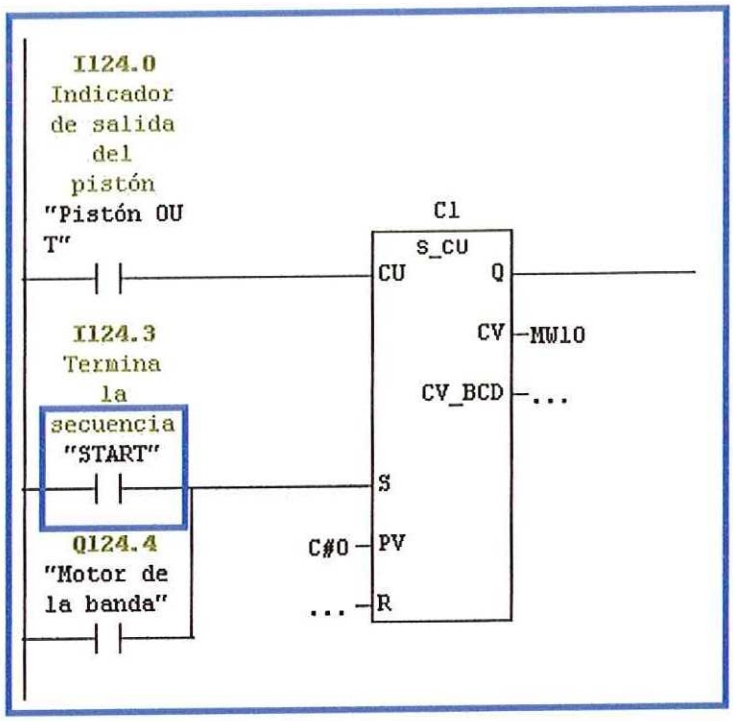
para hacer que el proceso se detenga si la existencia de empaques o de cajas se acaba, se agregan estas dos señales de forma tal como se muestra en la gráfica, de esta forma el proceso no arrancará si se da START o si esta trabajando en automático.



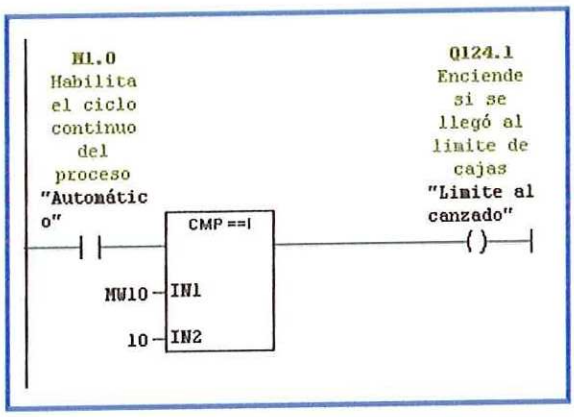
De igual forma se hace en el controlador del automático para evitar una habilitación errónea de este mismo.



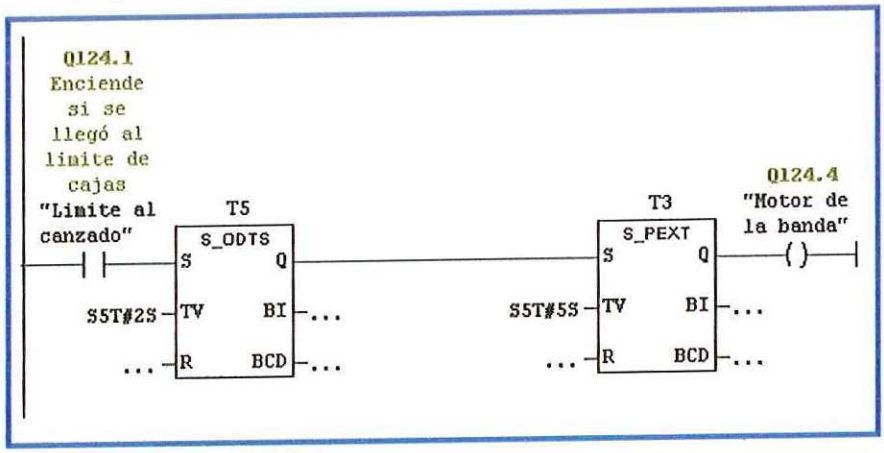
En otro segmento inserte un contador de tipo ascendente, nómbrelo C1. En PV cargue el valor de 0, de esta forma cuando presione el botón START (conectado a S), el contador iniciará desde cero y empezará a contar ascendentemente cada vez que el sensor de salida del pistón emita una señal, direccione CV a la marca MW10, aquí quedará almacenado el conteo.



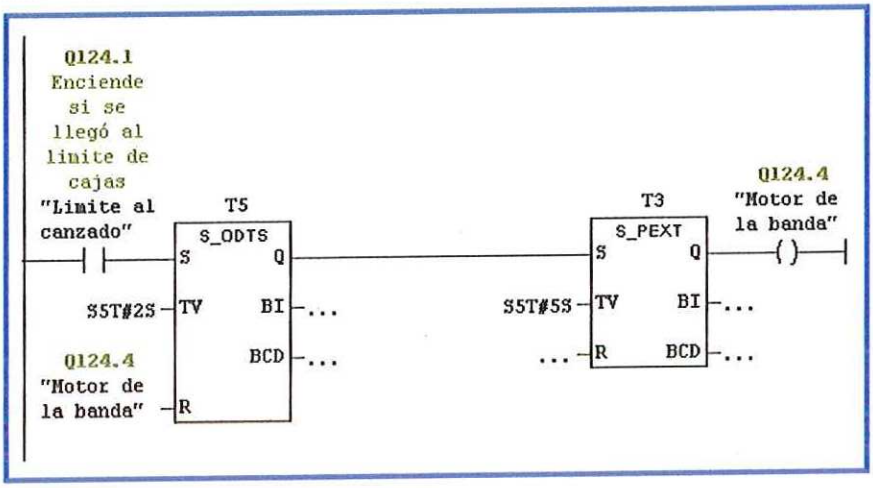
En un nuevo segmento agregue un comparador de tipo *igual a*, permita que la señal de Automático sea la encargada de habilitarlo, ya que esta es la única señal que está activa durante todo el proceso. Cuando el valor que se está almacenando en MW10 sea 10 el comparador emitirá una señal, esta señal está conectada a una lámpara (controlada por la salida Q124.1).



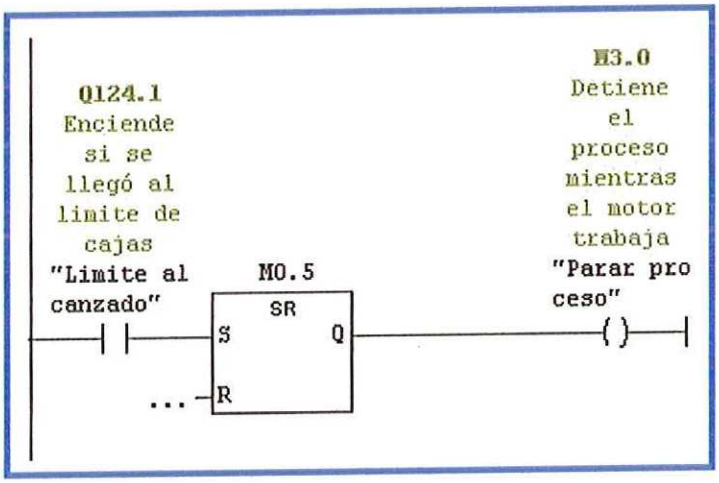
La señal de límite alcanzado activa un temporizador (T5) con retardo a la conexión (S_ODTS), espere un tiempo de 2 segundos para que retarde la activación de otro temporizador (T3) el cual controla al motor de la banda y lo mantendrá activo por 5 segundos (tiempo que tarda en retirar y poner otra caja en su lugar).



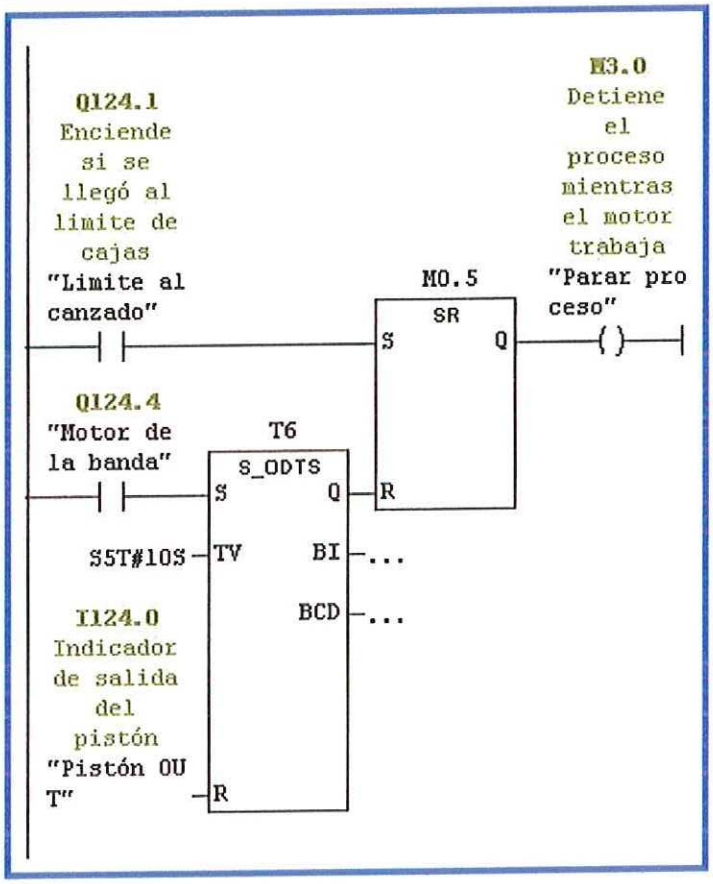
Es necesario resetear el primer contador T5, la señal de activación del motor de la banda puede hacerlo, de esta forma él estará listo ante otra secuencia.



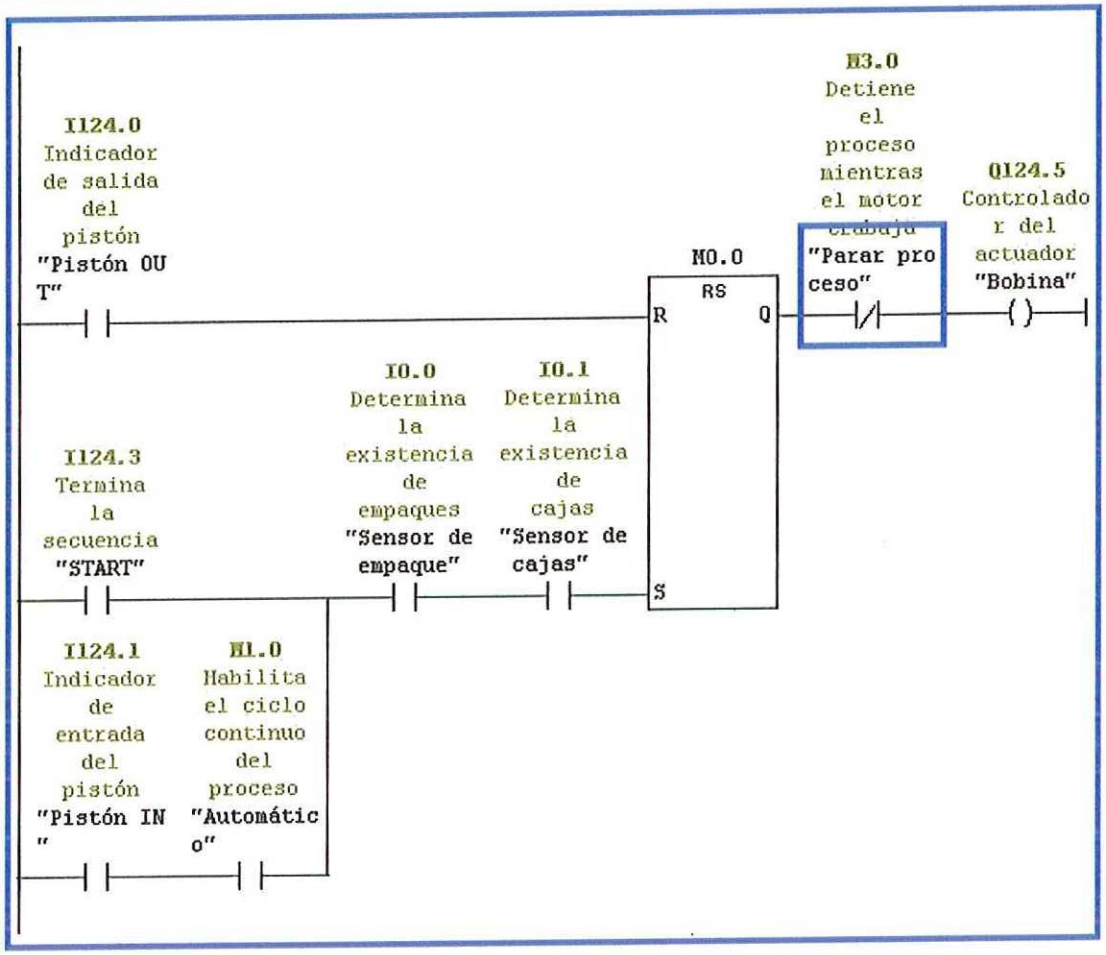
Es necesario detener el trabajo del pistón mientras el motor se encuentra en movimiento, agregue un F-F tipo SR, el cual controlará a la marca M3.0 (denominada para este caso *Parar proceso*), conecte a S la señal de salida del comparador, de esta forma cuando se llegue al límite el proceso se detiene.



Para reactivar el proceso se agrega a R un temporizador (T6) con retardo a la conexión (S_ODTS), el tiempo de este es de 10 segundos (tiempo en que se considera que todas las piezas están en su lugar para reiniciar el proceso). Este temporizador se inicia en el momento en que el motor se activa y se resetea en cuando el pistón reinicia (Pistón OUT).



Ubique *Parar proceso* en forma negada a la salida de Q, en el primer segmento de operaciones donde se controla a la válvula.

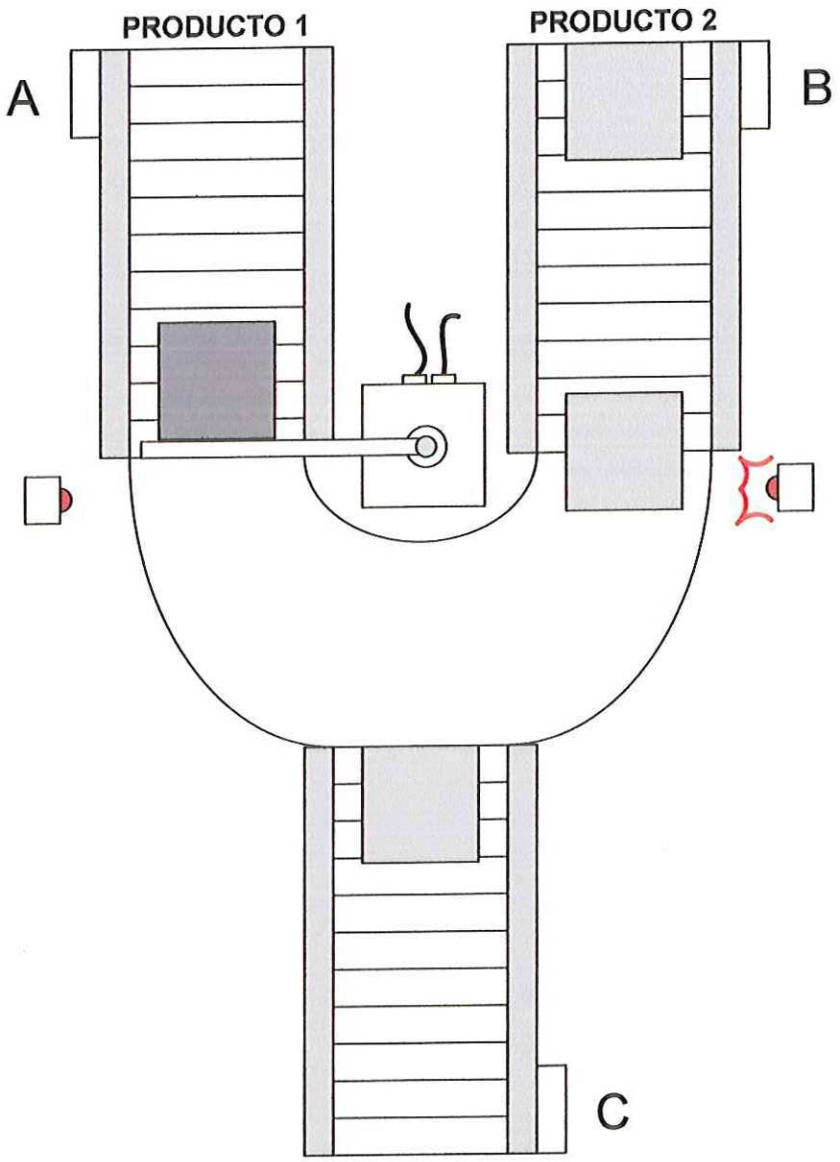


Guarde los cambios y cargue el programa al PLC.

6. ACTIVIDAD COMPLEMENETARIA

Cuando una fábrica tiene que despachar en empaque dos productos totalmente diferente al mismo tiempo, se ha ideado un mecanismo controlado mediante PLC para facilitar el almacenamiento de estos.

Cada producto llega en su respectiva banda transportadora, las cuales son accionadas por un motor (A y B) que es controlado por el PLC, las bandas al final del trazado se unen a una sola (accionada por el motor C).



Se requieren 5 existencias de un producto para el empaque, así que se adaptó una portezuela accionada hidráulicamente la cual gira sobre su eje en cualquier dirección (la cual depende del puerto por donde está entrando el flujo), para que obstruya el paso de un producto. Para evitar el represamiento, el motor que acciona la banda del producto detenido se apaga durante el tiempo en que las 5 existencias del otro producto se empacan (el motor C esta en movimiento desde el encendido de la máquina).

Existen dos sensores que se encargan del conteo de los productos que van a empaque.

Nota: *simule la portezuela con un pistón de doble efecto, el pistón no tiene sensores para conocer su posición.*

El sistema se enciende cuando se acciona el botón START y se detiene cuando se da STOP.

Nota: *Idee el control para que el sistema se apague luego de realizar cinco operaciones de empaque de cada producto.*

PRÁCTICA 8
FUNCIONES

PRÁCTICA 8
FUNCIONES

1. OBJETIVOS

- ✓ Crear e implementar funciones para el desarrollo de proyectos en PLC's.
- ✓ Visualizar la utilización de funciones como un método para facilitar la comprensión de los programas.

2. CONCEPTOS FUNDAMENTALES

Cuando los programas se hacen más grandes se vuelven evidentemente mucho más complejos y su comprensión se reduce (incluso para el mismo diseñador); y esto es muy grave cuando se requiere realizar un modificación, porque ahora se necesita de mucho más tiempo de prueba y en algunos casos, es mucho más fácil volver a diseñar el algoritmo.

En entornos de programación como visual C o Pascal, existe la posibilidad de crear subprogramas para dividir el programa principal en varias secciones, cada subprograma realiza una tarea determinada y es diferente de los otros.

Los desarrolladores de Step 7 adaptaron este mecanismo de agrupamiento en el entorno de programación y las denominó funciones. Son muy útiles, en especial en un entorno de programación tan abstracto como el de Simatic, donde muchas veces no se hace evidente la labor de los segmentos si no se ha detallado de forma específica. De ahí que es importante etiquetar las entradas, salidas y marcas y también señalar la función de cada segmento. Puede que después de desarrollar un programa, este se entienda, pero después de un largo período de tiempo la comprensión se va reduciendo, hasta el punto en que se olvida totalmente.

La siguiente práctica no sólo tiene el objetivo de enseñar, es una invitación a la realización de proyectos de forma ordenada, no olvide que muchas veces la forma de ser de una persona se ve reflejada en sus acciones, y un programa para PLC's no es la excepción.

3. ACTIVIDAD PREVIA

a) Repase el desarrollo metodológico de la *Práctica 4, Lógica Secuencial Uso de Flip-Flop's*, tenga en cuenta la metodología empleada para la solución del problema planteado.

4. LISTA DE MATERIALES

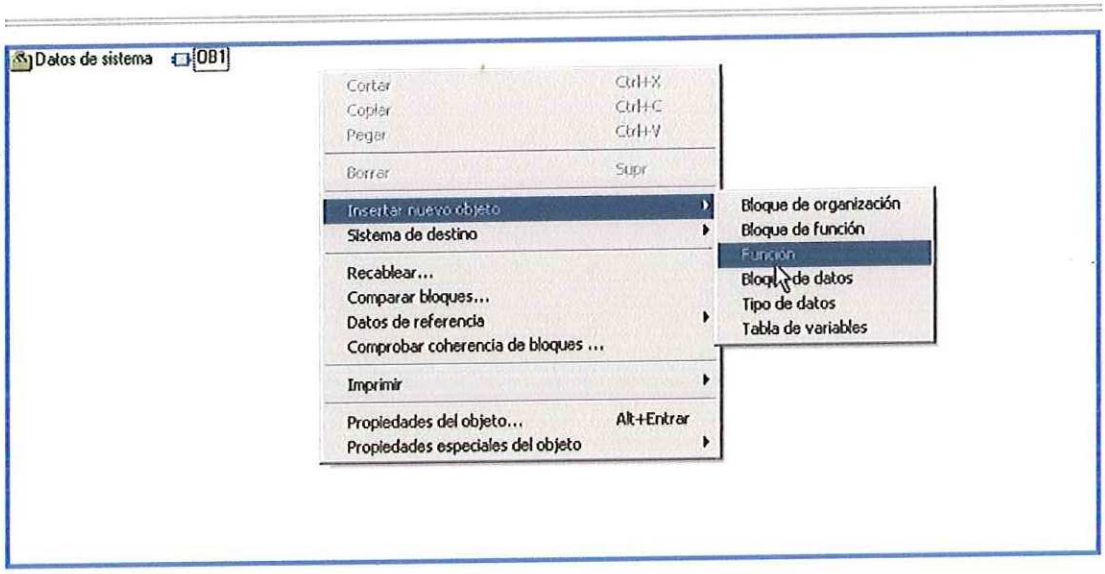
1) 2 bombillos de 12 o 24 voltios.
2) 2 electroválvulas de una bobina con retroceso por muelle.
3) 2 pistones de simple efecto.
4) 4 sensores de posición.
5) 1 interruptor con enclavamiento de 2 posiciones.
6) 1 pulsador normalmente cerrado.
7) 1 pulsador normalmente abierto.

5. DESARROLLO METODOLÓGICO

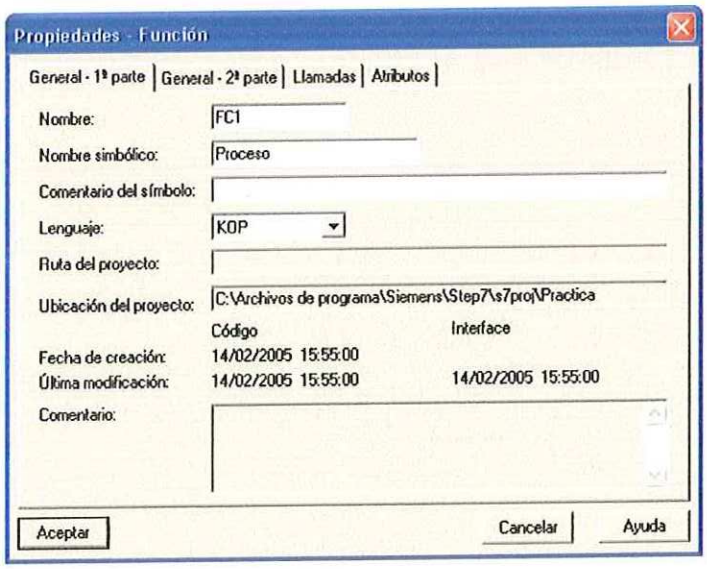
Nota: *para el desarrollo de esta actividad es necesario haber elaborado la Práctica 4 ,Lógica Secuencial Uso de Flip-Flop's.*

Abra ***SIMATIC/Administrador Simatic***, cree un nuevo proyecto con el nombre *Práctica 8.*

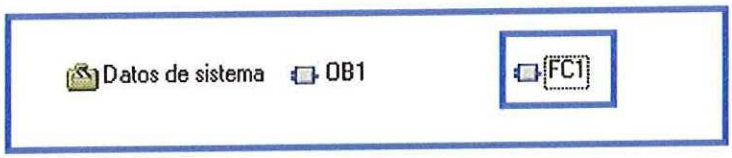
Para crear una función, en el *Simatic Manager* seleccione la carpeta *Bloques*, ahora visualizará sus datos contenidos, entre ellos el *Bloque de Organización* OB1, de clic derecho en el espacio en blanco para visualizar una ventana emergente, ahora vaya hasta la opción *Insertar nuevo objeto* y seleccione *Función.*



Aparecerá una nueva ventana llamada *Propiedades-Función*, en la casilla *nombre simbólico* escriba *Proceso*, **de clic en Aceptar**.



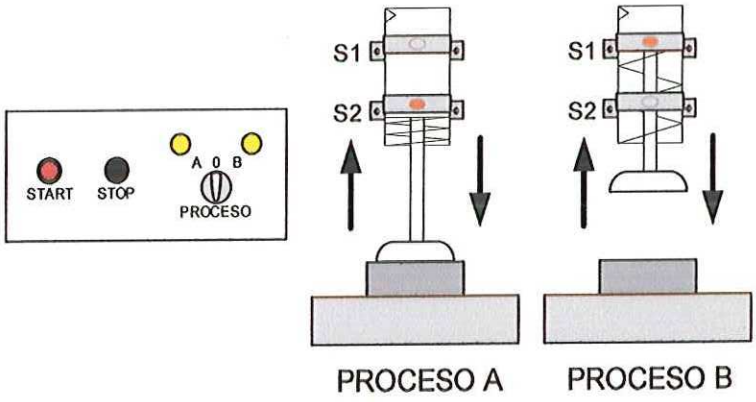
Aparecerá un nuevo icono en el espacio de trabajo llamado *FC1*, es decir, ***función número 1***. Para editarla basta dar doble clic sobre ella, y entrará en el mismo entorno de trabajo que utilizaba para editar el *OB1*, por ahora no realice ningún cambio.



Ahora se implementará lo anteriormente descrito para la solución de un problema.

Una empresa crea dos tipos diferentes de productos A y B, cuya producción no es constante, según estudios, de acuerdo a una época del año la demanda exige más existencias de A, sucediendo el mismo caso para B, esto implica que en algunas épocas solo se produzca A o B. Sin embargo la empresa maneja inventarios, así durante un período del año A y B son producidos al mismo tiempo.

Tanto A como B, tienen su propio proceso de estampado (Similar al de la *Práctica 4*), el control de estos procesos exige un selector el cual se encarga de habilitar la máquina para que solo trabaje con A o con B o ambas a la vez, también cuenta con dos luces indicadoras que señalan cuál de los procesos está trabajando, si A o B o ambos. El proceso inicia al presionar el botón START, y se detiene al presionar STOP.



Diseñe el controlador para dicho proceso.

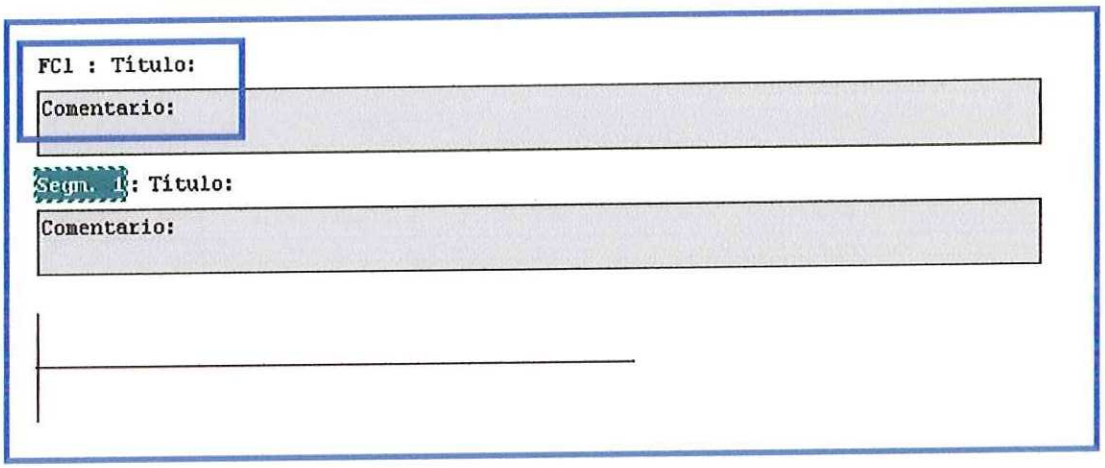
Antes de iniciar, conecte todos los elementos mencionados en el punto 4 *Lista de Materiales.*

Cada estampadora se controlará por separado, el selector se encargará de decidir la opción de trabajo, se utilizarán las funciones para separar los controladores de cada estampadora y el control de las luces indicadoras, en otras palabras se crearán 3 funciones distintas.

De doble clic, en *FC1* creado anteriormente.

Note la aparición del título FC1 en el entorno de trabajo, todo cambio que haga aquí sólo afectará a dicha función, notifique también que todas las funciones, menús son las mismas con las que ha trabajado anteriormente,

por lo que no habrá dificultad en la programación de dicha función.

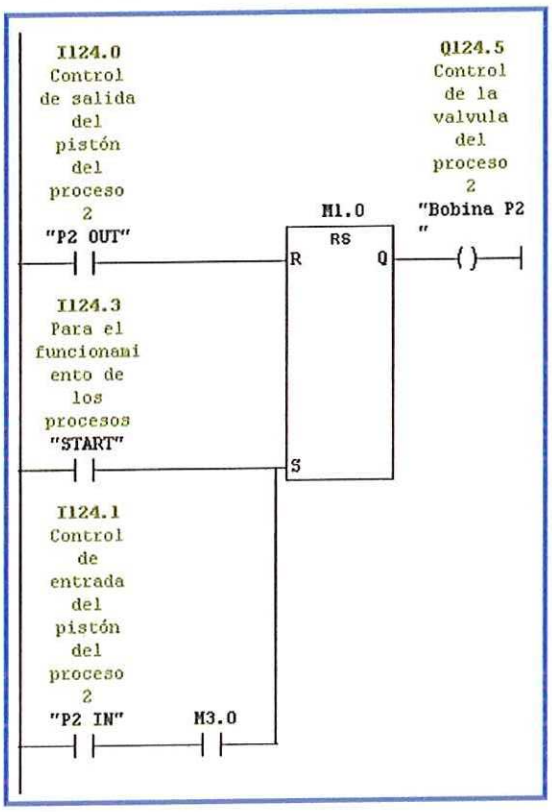


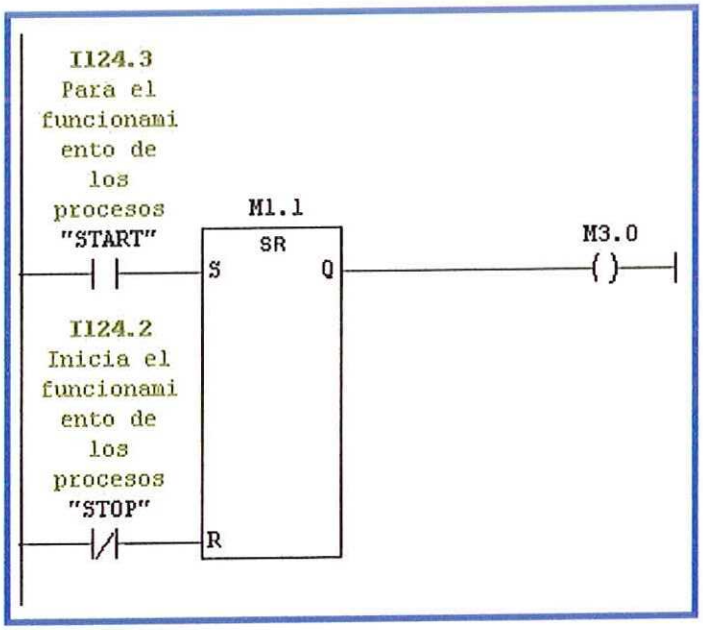
Antes de iniciar la programación, en la *Tabla de Símbolos* etiquete los componentes que se utilizaran durante la práctica. Tabla de Símbolos es una opción global y lo que se edita aquí también afectará a las otras funciones y al OB1. (No olvide conectar los elementos)

P2 OUT	I	124.0	BOOL	Control de salida del pistón del proceso 2
P2 IN	I	124.1	BOOL	Control de entrada del pistón del proceso 2
STOP	I	124.2	BOOL	Inicia el funcionamiento de los procesos
START	I	124.3	BOOL	Para el funcionamiento de los procesos
P1 OUT	I	124.4	BOOL	Control de salida del pistón del proceso 1
P1 IN	I	124.5	BOOL	Control de entrada del pistón del proceso 1
Seleccionador P1	I	124.6	BOOL	Interruptor con enclavamiento de dos estados
Seleccionador P2	I	124.7	BOOL	Interruptor con enclavamiento de dos estados
Luz Proceso 1	Q	124.0	BOOL	Indicador de funcionamiento de proceso
Luz Proceso 2	Q	124.1	BOOL	Indicador de funcionamiento de proceso
Bobina P2	Q	124.5	BOOL	Control de la valvula del proceso 2
Bobina P1	Q	124.6	BOOL	Control de la valvula del proceso 1

Simatic desarrolla dos tipos de funciones bajo el mismo entorno, una vacía usada comúnmente para agrupar varios segmentos, la cual trabaja con datos globales (como los editados en la Tabla de Símbolos). El otro tipo de función, trabaja con datos locales (datos que sólo usa la función), cuyos valores o estados dependen de la variables globales que éstos referencian. Estas funciones pueden devolver un resultado. Un ejemplo sería un Flip-Flop o un bloque de transferencia.

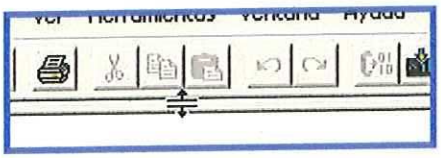
En la función FC1 se creará el control para los procesos de estampado, se implementará el mismo que se usó en la práctica 4.



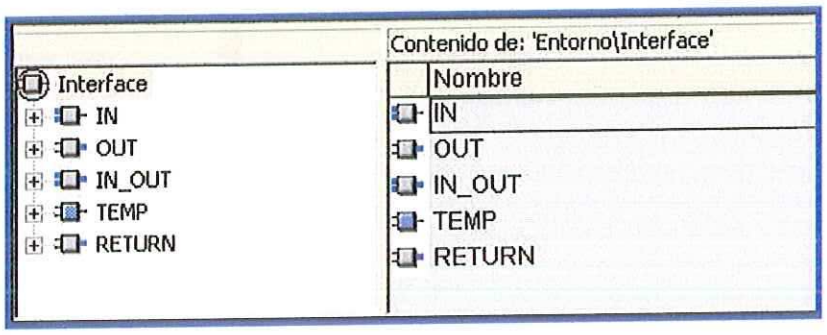


Usando como referencia el diagrama extraído de la práctica 4, se toma nota de las variables de entrada y de salida que la función implementa. La marca que memoriza el START no será tenida en cuenta. Estas variables determinarán la cantidad de datos locales que se necesitan.

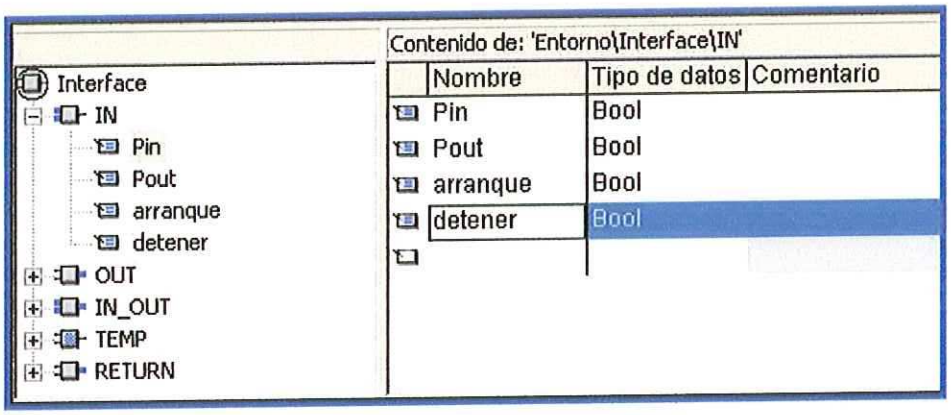
Para abrir el panel de datos ubique el cursor del Mouse debajo en la parte superior de la ventana de trabajo, justo debajo de los íconos del programa.



El cursor se transformará en una herramienta que expande las ventanas, desplácelo hacia abajo y un nuevo marco aparecerá sobre el espacio de trabajo.



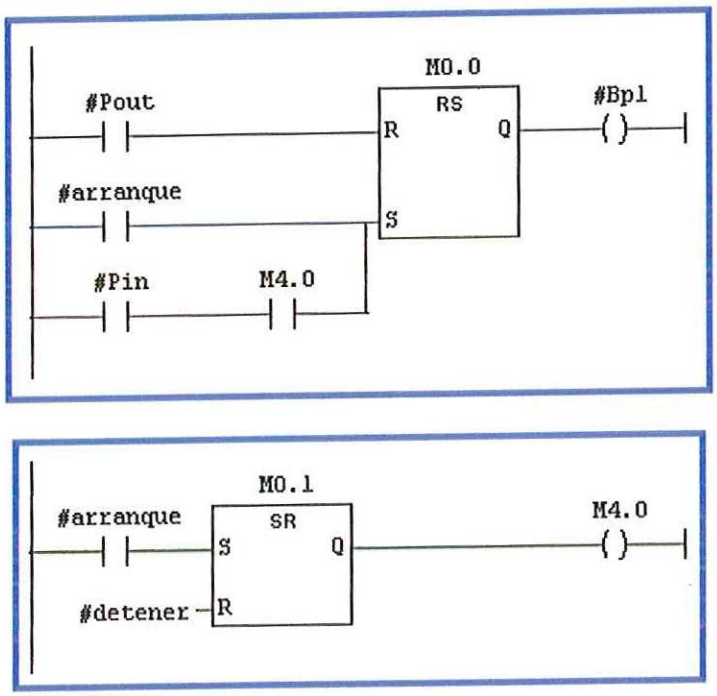
Estos iconos hacen referencia a las variables con las que el sistema trabajará, de clic sobre *IN* para declarar los datos locales que son entradas. Tome como referencia la siguiente imagen,



Pin, es la señal del sensor de posición cuando el pistón entra, *Pout* cuando este sale, *arranque* es la señal de START y *detener* STOP, se usan estas referencias genéricas porque la misma función controlará ambos pistones.

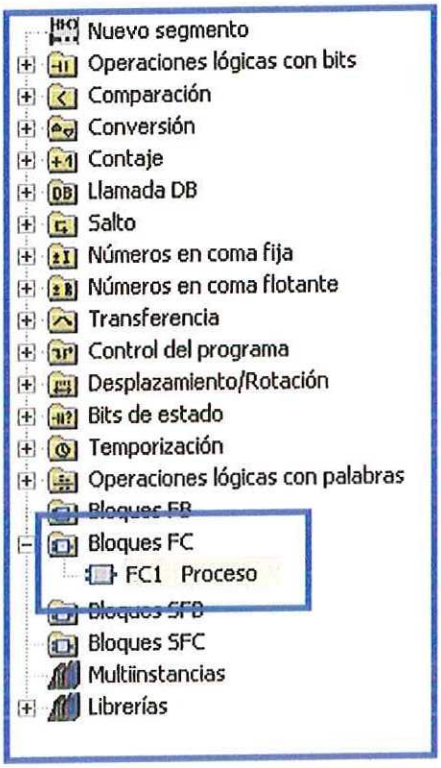
De clic sobre el icono *OUT* para declarar las variables de salida. Declare a *BP1* como la salida de la función, esta será la que controlará a la bobina del pistón.

Ahora declare estas variables como lo muestra la figura, el signo #, hace entender al programa que estas variables son de referencia y sus estados dependerán de los valores que se asocien a ellas.

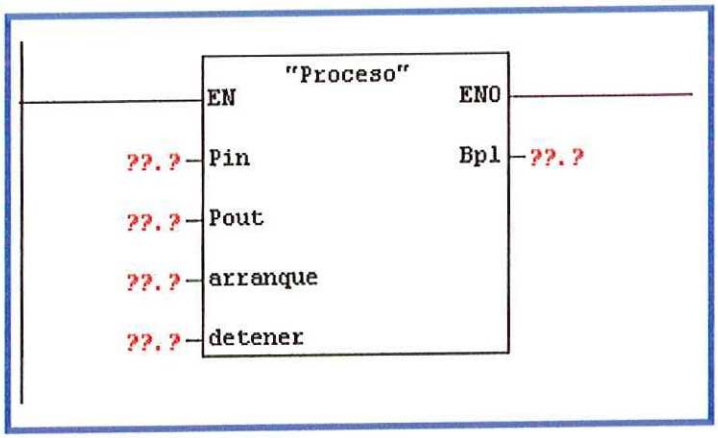


Guarde la función y cárguela al PLC.

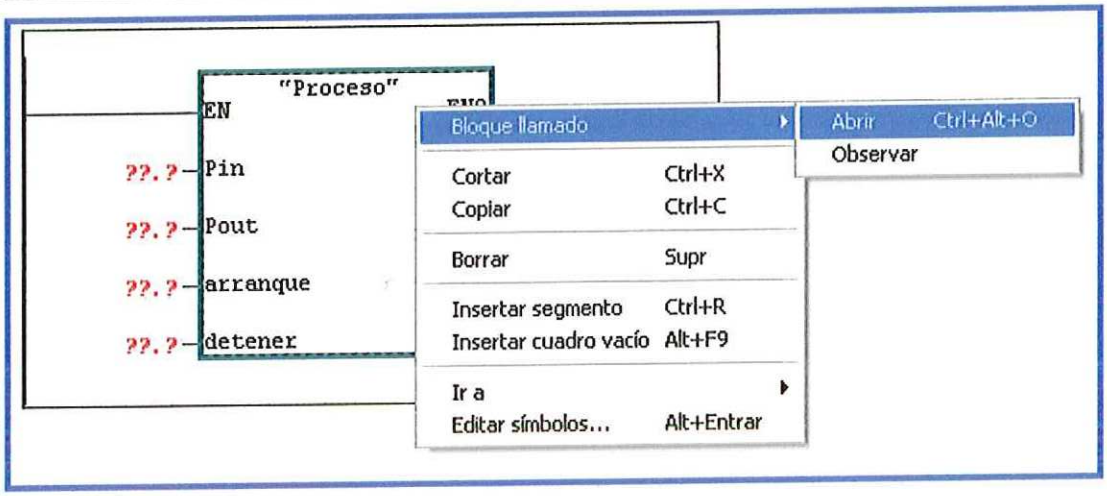
Ahora abra el *OB1*, el objetivo de este punto es insertar la función anteriormente creada al proyecto, para esto en Elementos de Programa de clic sobre Bloques FC, aparecerá entonces *FC1 Proceso*, función que fue creada en el paso anterior, selecciónela y arrástrela hacia un segmento.



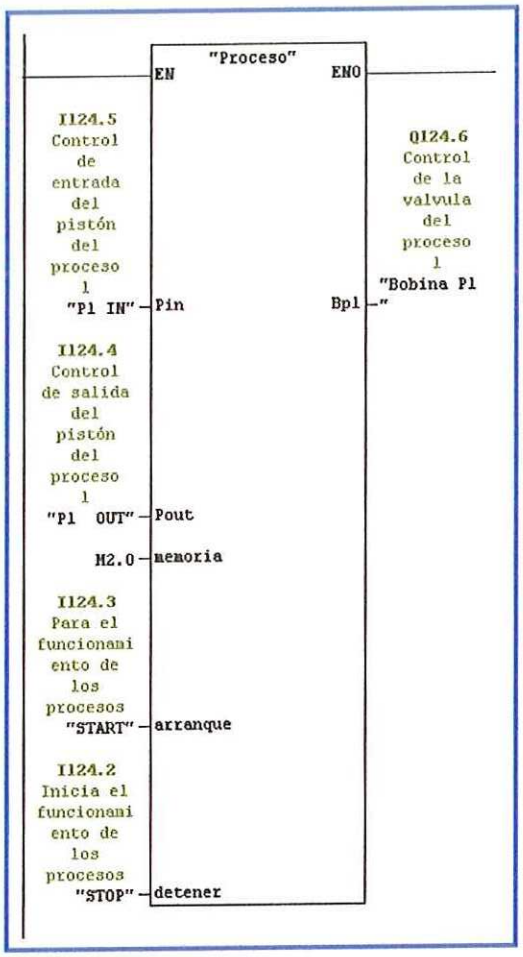
La función aparecerá sobre el segmento como se muestra a continuación.



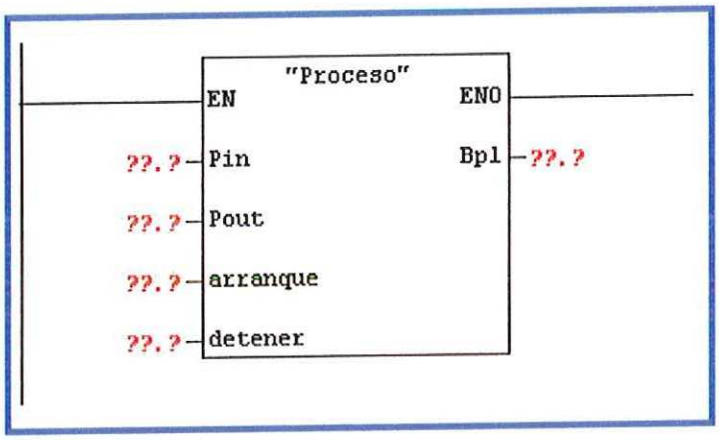
Si desea editarla para crear algún cambio, de clic derecho sobre la función y seleccione *Bloque llamado*, ahora seleccione Abrir, esto lo llevará nuevamente a los esquemas que conforman esta función, puede trabajar sobre ellos, pero recuerde guardar y cargar estos cambios al PLC.



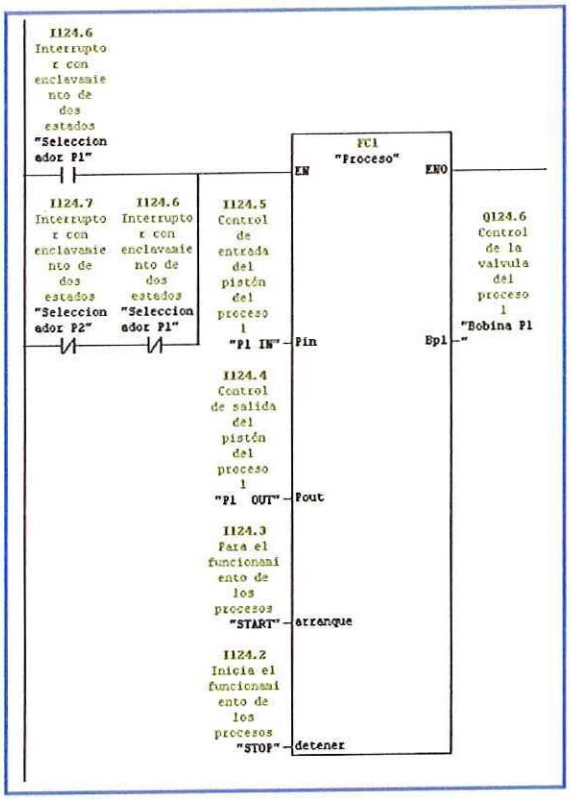
Esta función controlará una de las estampadoras, asocie a sus entradas los elementos que conectó previamente. Sólo debe tener en cuenta los elementos de una estampadora. Tome nota de la siguiente gráfica.

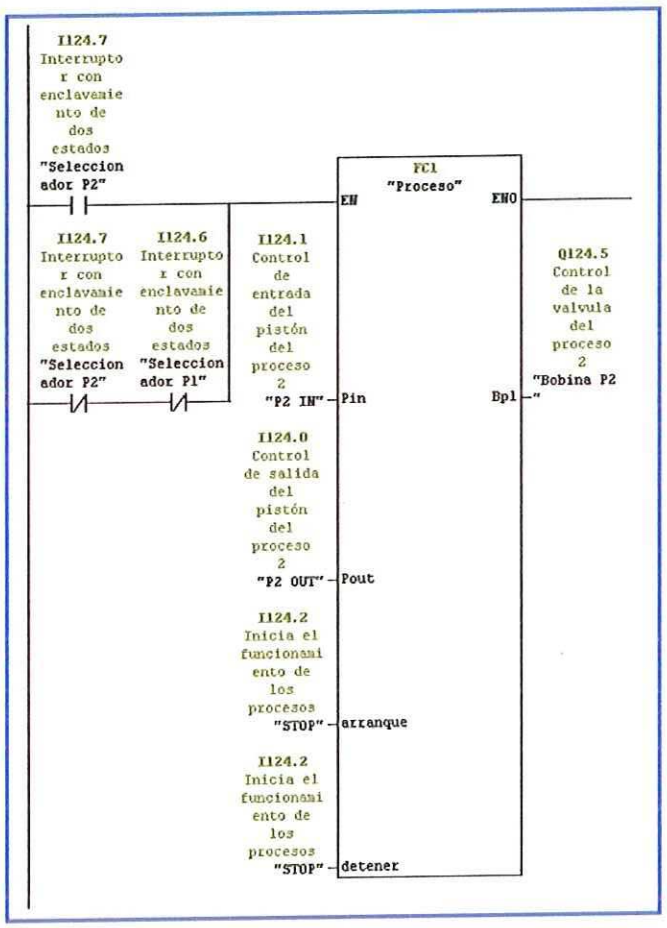


Ahora inserte otro segmento y asocie a él una nueva función FC1. Y configúrela de igual forma. Esta vez se debe tener en cuenta los elementos asociados a la otra estampadora.



El interruptor de dos posiciones se encargará de seleccionar los procesos, en una posición seleccionará a A, en la otra a B, y en la posición central estarán habilitados ambos procesos. A cada posición se le ha asignado una entrada del PLC. Teniendo en cuenta estos parámetros se modifican los segmentos donde se encuentran las funciones.



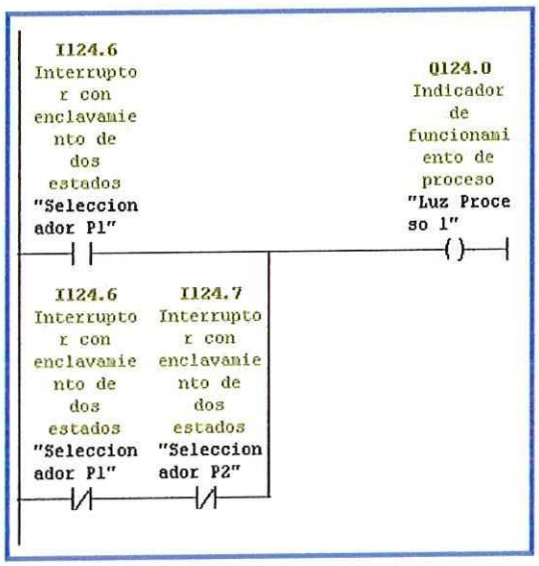


Si el seleccionador está en una posición definida activará uno de los procesos, si esta en el centro, ambos procesos operarán una vez se presiones START y no se detendrán hasta que se presione STOP.

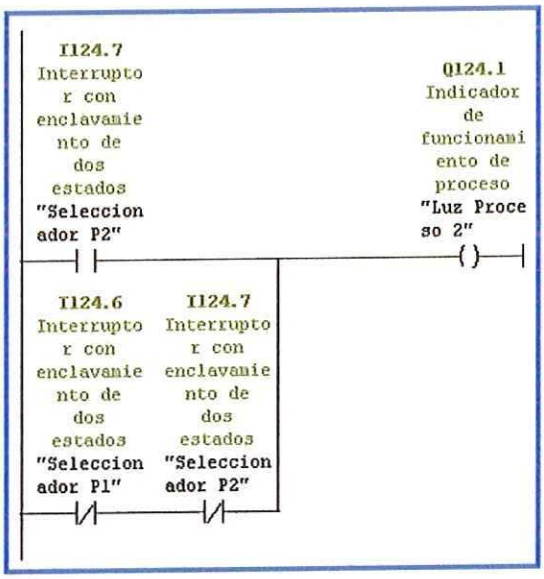
Guarde y cargue el OB1.

Lista esta parte, ahora se procederá a crear la función que se encargará de controlar las luces indicadoras, desde el *SIMATIC Manager,* cree el FC3 y asigne el nombre de *Luces Indicadoras*.

Estas funciones tendrán el segundo formato, el de tipo vacío, en este caso no habrán variables locales que referenciar, su única función es la de agrupar una secuencia de líneas de comando.

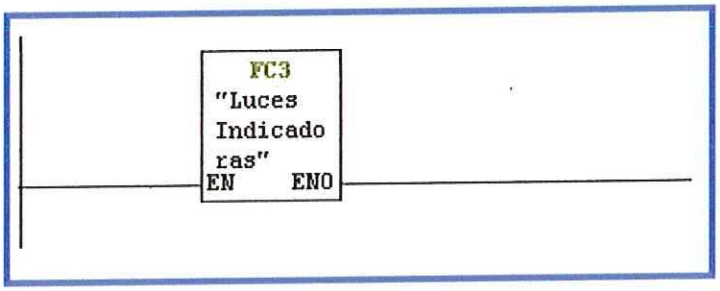


Con un control similar al usado para la selección del los procesos, las luces se encenderán según la posición de la perilla.



Guarde la función y cárguela al PLC.

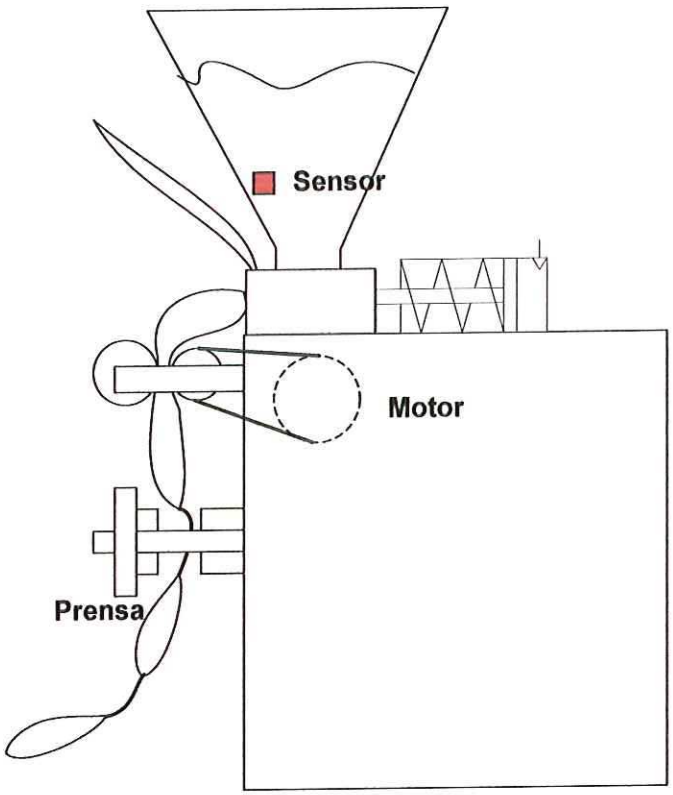
En el OB1 inserte esta nueva función en otro segmento, ahora guarde los cambios hechos y cargue el *bloque de organización* al PLC.



Pruebe el proyecto ya creado.

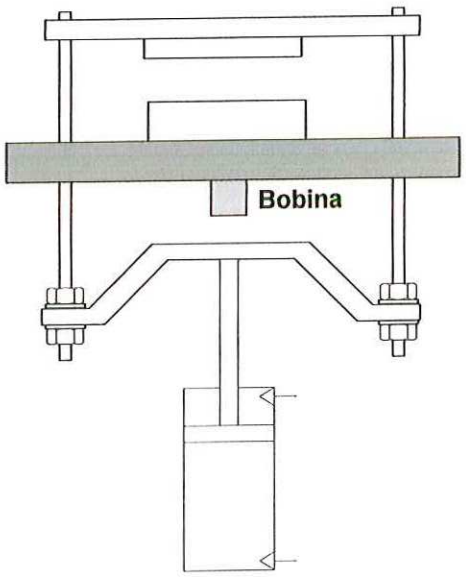
6. ACTIVIDAD COMPLEMENETARIA

Una empresa de alimentos local ha adquirido un máquina empacadora de alimentos. La máquina consta de una tolva que almacena el producto y distribuye en cantidades iguales, un sistema de accionamiento que se encarga del movimiento del empaque y una prensa que se encarga de sellarlo.



Mediante un mecanismo neumático, la tolva se encarga de administrar una cierta cantidad de producto dentro de un empaque, cuando esta listo, un motor se encarga de llevar esta parte del empaque hacia la prensa.

Prensa



La prensa es un mecanismo neumático que sella la bolsa a través de un impulso eléctrico que la calienta (el impulso dura 3 segundos) provocando la fusión.

El proceso debe operar de forma continua hasta que se acabe el producto que esta almacenado en la tolva (se adaptó un sensor para este propósito) o se acabe el empaque (el cual tiene un alcance de 20 sobres).

Hay tres luces indicadoras para las tres fases, llenado, transporte y sellado, y una cuarta para indicar que la máquina esta trabajando. El motor, las válvulas de los pistones (las cuales son de una bobina con retroceso por muelle) y el impulso eléctrico son controlados por un PLC. Además a el proceso se le ha adaptado un START (normalmente abierto sin enclavamiento) y un botón de paro de emergencia (normalmente abierto con enclavamiento) para detener el proceso en cualquier instante.

El pistón de la tolva es de simple efecto y el de la prensa es de doble efecto.

Nota: *utilice un adecuado manejo de tiempos para el accionamiento de los actuadores del sistema. Implemente al sistema ambos tipos de funciones, considere cual debe ser el adecuado para aplicar a cada parte del proceso.*

PRÁCTICA 9
MÉTODO DE PROGRAMACIÓN ESTRUCTURADA

PRÁCTICA 9
MÉTODO DE PROGRAMACIÓN ESTRUCTURADA

1. OBJETIVOS

- ✓ Diseñar programas utilizando la metodología de programación estructurada.
- ✓ Implementar la programación estructurada en el diseño de problemas para el PLC.

2. CONCEPTOS FUNDAMENTALES

La programación estructurada es una solución a los múltiples inconvenientes que se presentan cuando se quiere resolver un problema neumático, esta consiste en implementar una serie de pasos lógicos que permiten la solución de un problema en base de un diagrama de tiempos.

Introduciendo al desarrollo de programas mediante este método surge la programación multitarea, que es simplemente el desarrollo modular de un programa, es decir la aplicación de funciones como un método de organización.

Es necesario aclarar que este método es sólo una herramienta, y es solo una de las posibles soluciones que un problema puede presentar, su ventaja radica en la facilidad de la lectura del programa, al igual que su rápida implementación.

3. ACTIVIDAD PREVIA

Esta práctica no tiene una actividad previa asignada ya que aquí se consideran algunas de las actividades realizadas en prácticas anteriores. Sin embargo se tiene como requisito indispensable haber hecho las actividades anteriores a ésta.

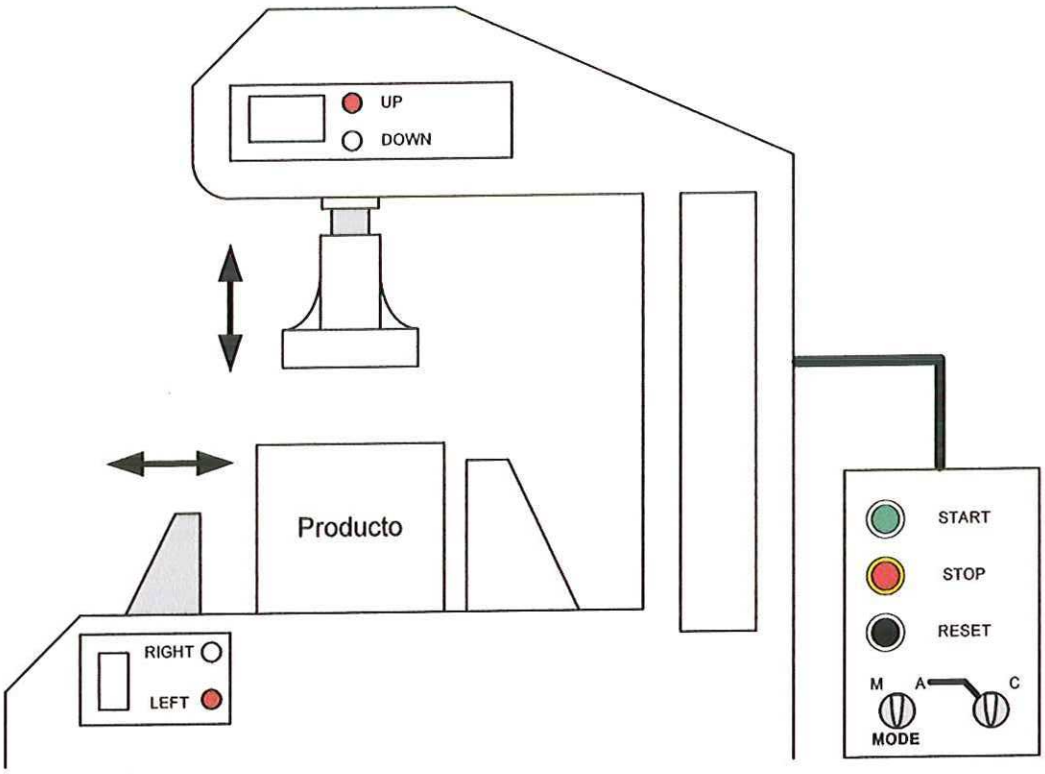
4. LISTA DE MATERIALES

1) Paro de Emergencia.
2) 2 pulsadores normalmente abiertos.
3) 2 interruptores con enclavamiento de 2 posiciones.
4) 1 pistón de doble efecto
5) 1 pistón de simple efecto.
6) 2 electroválvulas de una bobina con retroceso por muelle.
7) 4 sensores de presencia

5. DESARROLLO METODOLÓGICO

Nota: *para el desarrollo de esta actividad es necesario haber elaborado todas las prácticas de programación en PLC.*

Abra ***SIMATIC/Administrador Simatic***, cree un nuevo proyecto con el nombre *Práctica 9*.



Una empresa cuenta con un proceso de estampado, el cual está conformado por un pistón de doble efecto el cual realiza la operación de estampado (A) y

un cilindro de simple efecto, el cual sujeta la pieza (B). Existe un panel selector el cual permite al proceso seleccionar el modo de operación, *Automático* o *Manual.*

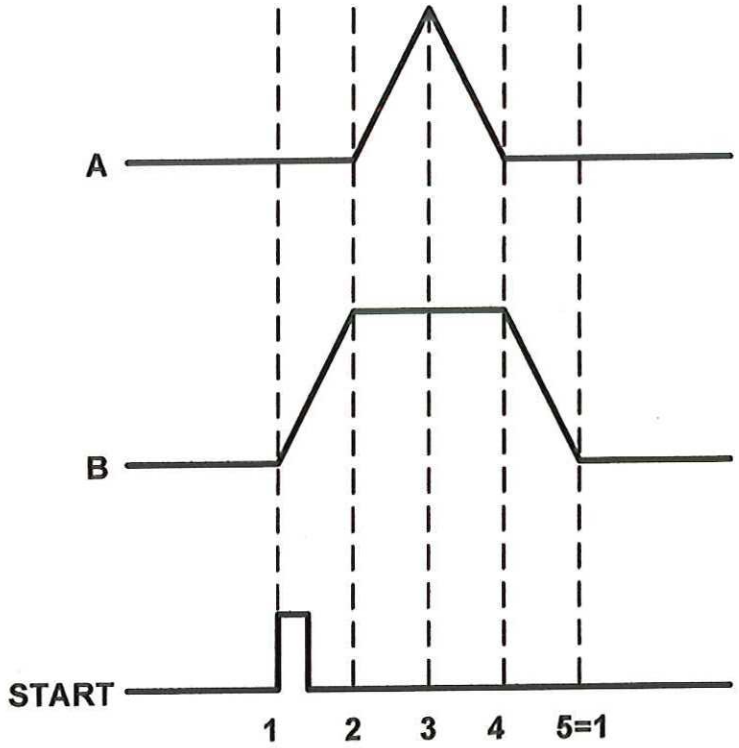
El primer modo, a su vez, puede trabajar de dos formas *Continuo* y *Único.* El modo *Continuo* ejecuta el proceso indefinidamente y no se detiene hasta que se presione STOP. *Único* ejecuta el ciclo sólo una vez y se queda en espera que START se vuelva a oprimir para ejecutarse del mismo modo.

El modo *Manual* ejecuta el ciclo paso por paso cada vez que se oprime el botón START, una vez que termina, queda de igual forma en espera que se vuelva a oprimir el mismo botón para reiniciar y operar de la misma forma.

Cuando se presiona STOP el proceso se para indefinidamente (los pistones se detienen y conservan la posición) y es necesario dar RESET para que vuelva a operar (al dar RESET los pistones vuelven a la posición inicial).

Presionando nuevamente START el proceso inicia según las opciones estén indicadas.

El diagrama de tiempos que describe el proceso es el siguiente:



Diseñe un programa que ejecute las acciones anteriores.

Para poder desarrollar este programa con el método estructurado es necesario aplicar funciones, memorias y marcas de forma intensiva, las funciones permiten el ordenamiento del programa, las memorias facilitan su ejecución y las marcas son indicadores que notifican el desarrollo de un paso o fase.

Conecte todos los sensores, actuadores e interruptores, tome nota de las direcciones que le son correspondidas, notifique su correcto funcionamiento y **etiquételas en la *Tabla de Símbolos***.

- Para los sensores de presencia y el botón START se utilizó la siguiente nomenclatura.

Símbolo	Dirección	Tipo de dato	Comentario
A+	I 124.0	BOOL	Cilindro A fuera
A-	I 124.1	BOOL	Cilindro A entra
B+	I 124.2	BOOL	Cilindro B fuera
B-	I 124.3	BOOL	Cilindro B entra
START	I 124.4	BOOL	Incia el proceso

Como siguiente paso **proceda a crear las funciones** (en el OB1), en total son cinco, una para el **Ciclo Automático (FC1)**, otra para el **Ciclo Manual (FC2)**, otra para la etapa de **Potencia (FC3)**, una más se encarga del paso de **ciclo único a ciclo continuo (FC4)** y la última se encarga del **Paro y el Reset (FC5)**.

Se procede entonces a editar la primera función (FC1) **Ciclo Automático.** Esta función es la encargada de administrar el desarrollo del ciclo automático ya sea en su modo continuo o único, no trabaja directamente sobre el control de las bobinas, sólo se encarga de notificar los pasos de la operación que se han cumplido.

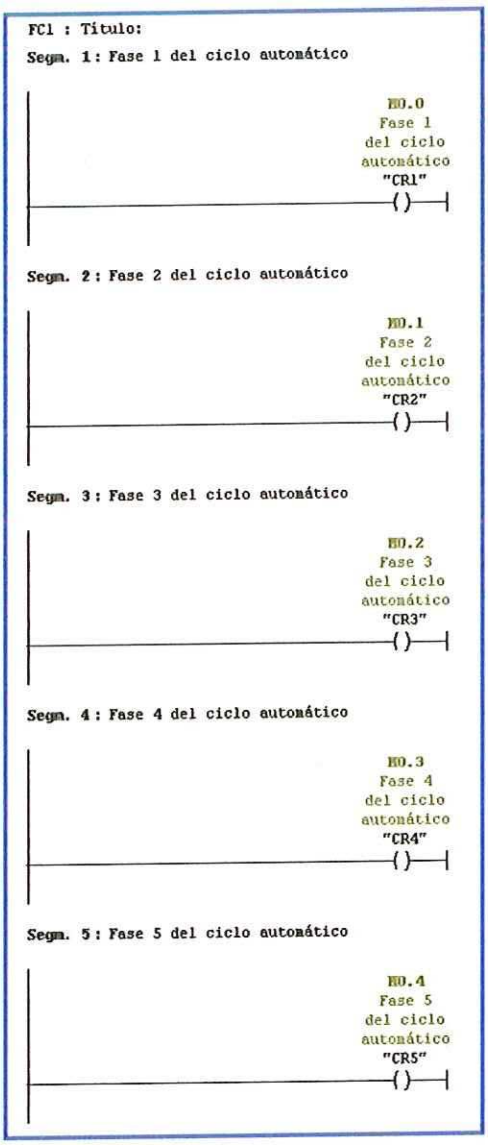
Como primera medida en la *Tabla de Símbolos* aparte un Byte de marcas (en el ejemplo se utilizó la MB0) y configure cinco de sus bits tal como lo muestra la siguiente gráfica.

Símbolo	Dirección	Tipo de dato	Comentario
CR1	M 0.0	BOOL	Fase 1 del ciclo automático
CR2	M 0.1	BOOL	Fase 2 del ciclo automático
CR3	M 0.2	BOOL	Fase 3 del ciclo automático
CR4	M 0.3	BOOL	Fase 4 del ciclo automático
CR5	M 0.4	BOOL	Fase 5 del ciclo automático

El diagrama de tiempos mostrado previamente describe un proceso que se

ejecuta en cinco pasos o fases (de ahí que se etiqueten cinco bits de una marca), en cada fase el estado del proceso es diferente, en la fase 1 (inicial) tanto el pistón A como el B están en su posición inicial en la cual ambos están retraídos. En la fase 2, B ya se ha expandido totalmente, A conserva la posición, en la fase 3 tanto A como B están fuera, en la fase 4 A regresa a su posición y en la 5 (que es similar a la 1) A y B están retraídos nuevamente.

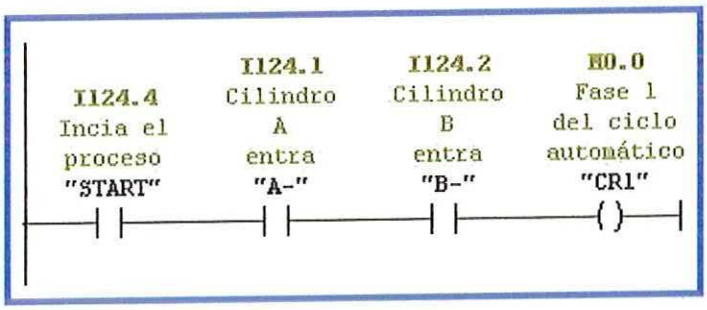
En el entorno de programación inserte cinco segmentos, uno para cada fase. Inserte las marcas etiquetadas previamente como se muestra en la siguiente figura.



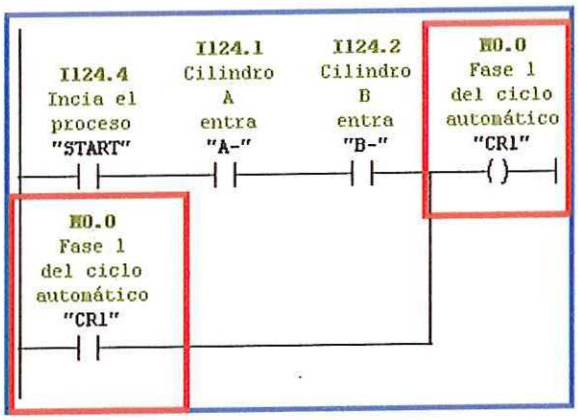
Como se había mencionado cada fase es diferente y eso depende del estado

en que se encuentren los pistones. Los sensores de presencia son los encargados de determinar la posición en que se encuentran, así que para cada fase habrá un grupo de sensores que estarán activos y estos son la referencia que se usa.

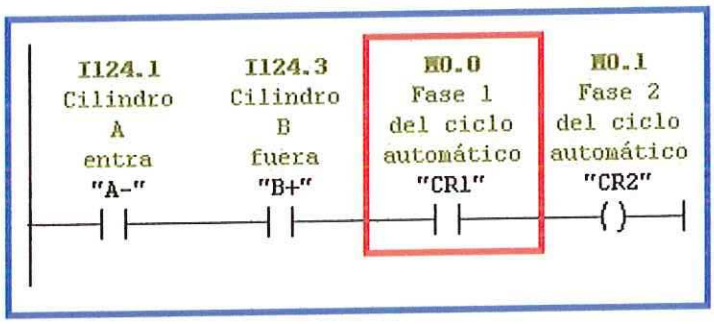
Para la primera fase (denominada CR1) se espera el accionamiento de START (conectado a la entrada I124.4) para que inicie el proceso, en esta fase los pistones A y B están retraídos, por tanto los sensores de presencia A- y B- están activos, estos son los que se insertan al segmento.



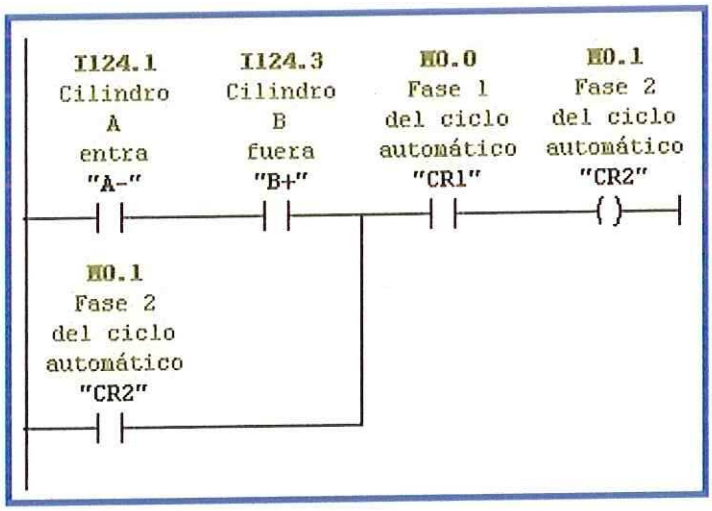
Tome nota de lo siguiente, START es un pulsador normalmente abierto sin enclavamiento, por lo tanto una vez se suelta, CR1 se desactiva. El método estructurado exige que cada fase cumplida debe quedar notificada, es decir, activa hasta que el ciclo completo de trabajo termine. Anteriormente se usaban los F-F para almacenar los estados de los interruptores normalmente abiertos, el proceso a continuación realiza la misma labor, y trabaja mediante la retroalimentación de la señal; cuando se presiona (y suelta) START el efecto inmediato es la salida del pistón B, por lo que no sólo START cambia de estado sino también los sensores de presencia (B- se desactiva), así que existe tan sólo un breve periodo de tiempo en que CR1 está activo. La retroalimentación es la misma marca conectada en OR como se muestra a continuación, esta conexión permite que CR1 conserve el estado de activo una vez el pistón cambie de sitio y START se suelte.



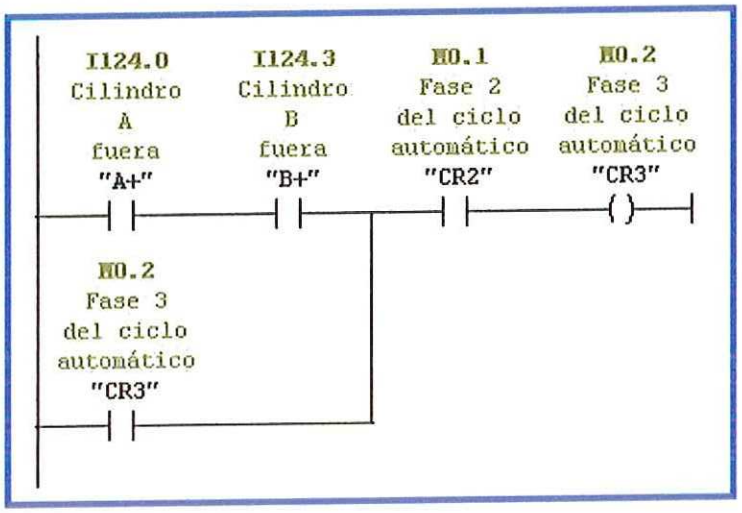
La siguiente fase ya no requiere de START, por lo que las condiciones de activación se centran en el estado de los sensores. En esta fase sólo el cilindro B esta expandido. Sin embargo ahora exige otra condición y es que la fase anterior se halla realizado, es decir CR2 no se realizará hasta que no se haya ejecutado CR1, por lo que este se agrega como una entrada justo al frente de los sensores tal como lo muestra en la siguiente figura.

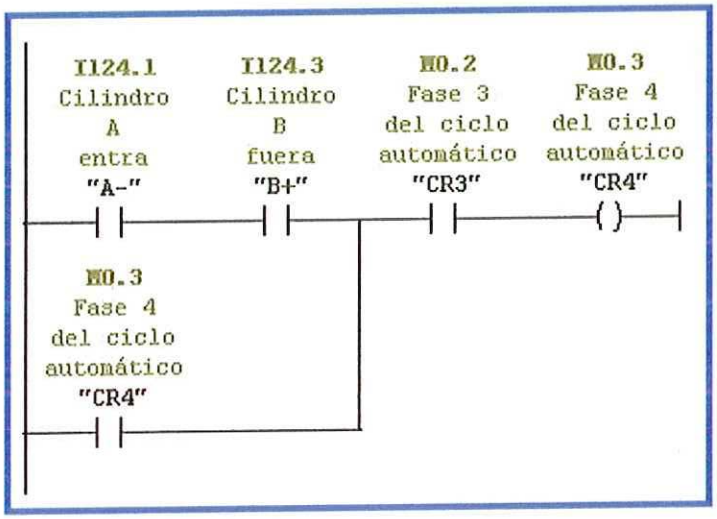


De igual manera que se hizo con la fase 1, la fase 2 también requiere de una memoria, la conexión en OR se hace justo después de los sensores ya que estos son los elementos que cambian de estado rápidamente.

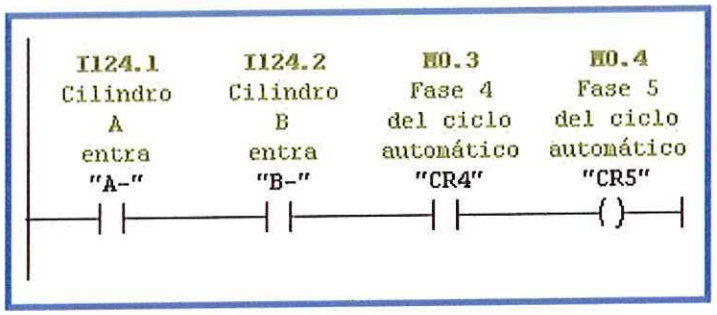


Se realiza el mismo análisis para la fases 3 (CR3) y 4 (CR4).

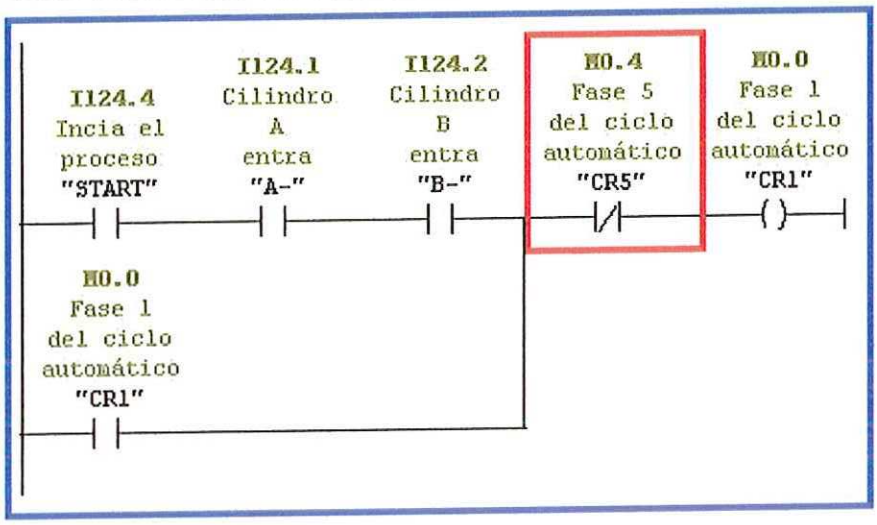




La fase 5 (CR5) no requiere de memorización por ser la última.



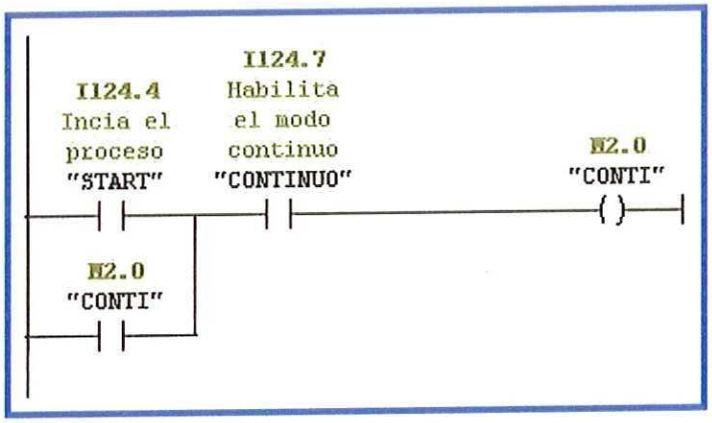
Cuando todas las fases se hayan ejecutado es necesario desactivarlas para que el programa pueda iniciar de nuevo. Para esto se usa CR5.



Mientras no se cumpla la última fase CR5 negado se mantendrá activo, cuando se cumpla esta condición, CR1 se desactivará haciendo que CR2 se desconecte y así consecutivamente. De esta forma el programa quedará nuevamente listo y volverá a trabajar una vez se vuelva a presionar START.

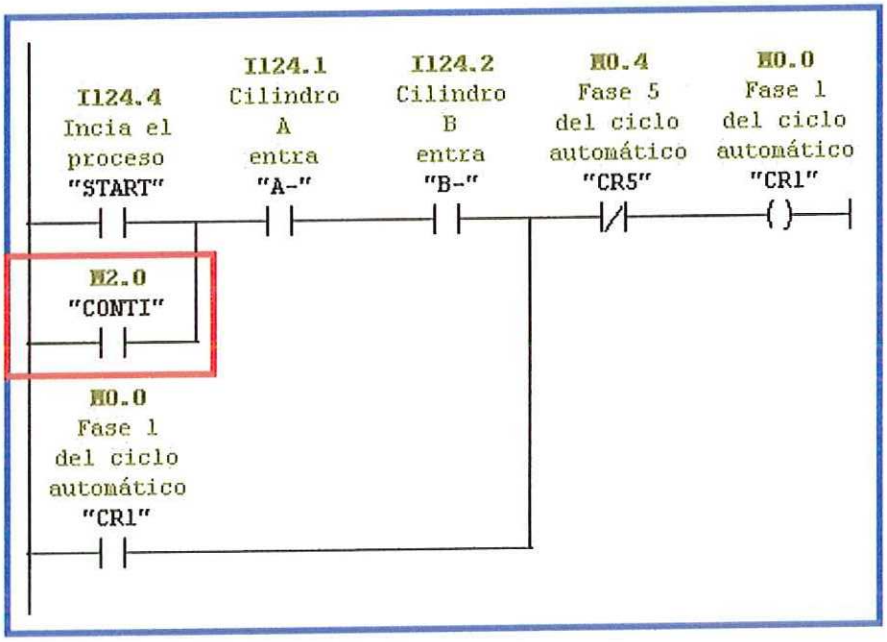
Guarde los cambios y cárguelos al PLC.

Para crear el modo continuo edite ahora la función FC4. El selector de continuo es uno de los interruptores con enclavamiento (conectado a I124.7), así que no necesita ser memorizado. **Etiquete un bit de las marcas como *Conti*** y en un nuevo segmento realice la operación que se muestra a continuación.



Cuando se presiona START, y *CONTINUO,* está seleccionado se activa la marca M2.0 *Conti*, esta se conservará así, mientras *CONTINUO* no se desconecte.

Volviendo a FC1 se modifica el primer segmento de la siguiente manera.

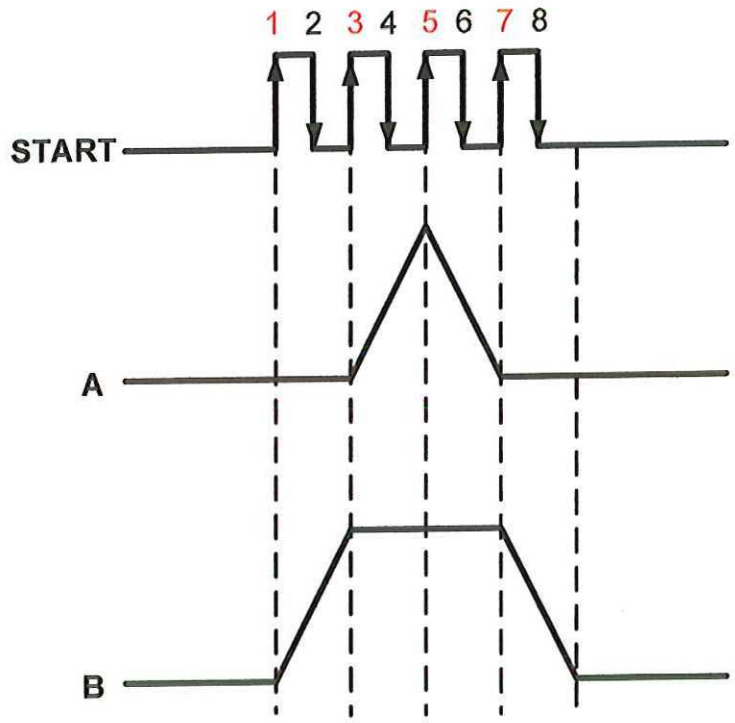


De esta forma el sistema adaptado a *Continuo* espera a que se oprima START para que funcione de forma indefinida ya que el sistema no se queda esperando una orden de inicio. **Guarde la función y cárguela al PLC**.

Ahora se configurará la segunda función FC2 para que pueda controlar el sistema en modo Manual.

Manual es un método que ejecuta cada una de las fases al oprimir el botón START, una vez el ciclo se ha completado, queda listo para volver a iniciar. **El número de fases que comprende el ciclo manual, depende del número de veces que se necesite oprimir START para que el proyecto vuelva al estado inicial multiplicado por dos**. En este caso (como lo muestra la figura a continuación) se necesita oprimir START 4 veces seguidas para que el proyecto recupere los estados que tenía en la fase 1.

El hecho que se tenga que doblar el número de fases, obedece al estado de botón suelto que precede cada una de las veces que se oprime START. El estado de botón suelto es el tiempo de espera que existe entre cada accionamiento de START, el cual es indefinido y depende del usuario.

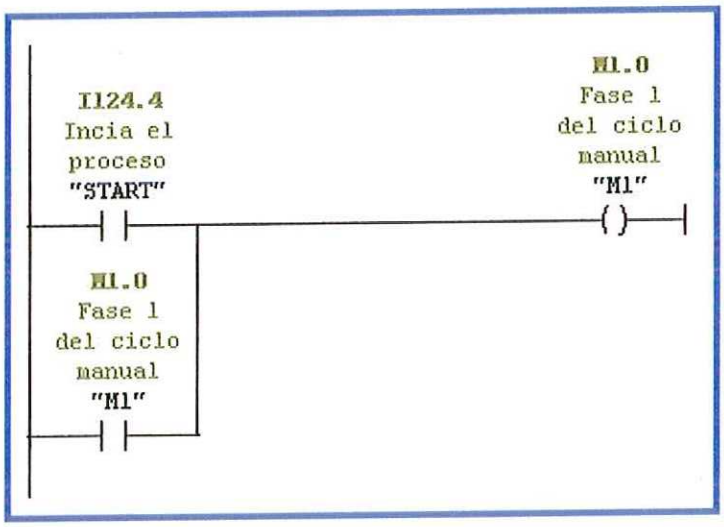


El modo manual de este proceso requiere de 8 fases en total, 4 donde se acciona START (las impares) y 4 donde no se presiona START (las pares). En *Tabla de símbolos* configure un byte de las marcas (en el ejemplo se usó la MB1) como se muestra a continuación.

M1	M	1.0	BOOL	Fase 1 del ciclo manual
M2	M	1.1	BOOL	Fase 2 del ciclo manual
M3	M	1.2	BOOL	Fase 3 del ciclo manual
M4	M	1.3	BOOL	Fase 4 del ciclo manual
M5	M	1.4	BOOL	Fase 5 del ciclo manual
M6	M	1.5	BOOL	Fase 6 del ciclo manual
M7	M	1.6	BOOL	Fase 7 del ciclo manual
M8	M	1.7	BOOL	Fase 8 del ciclo manual

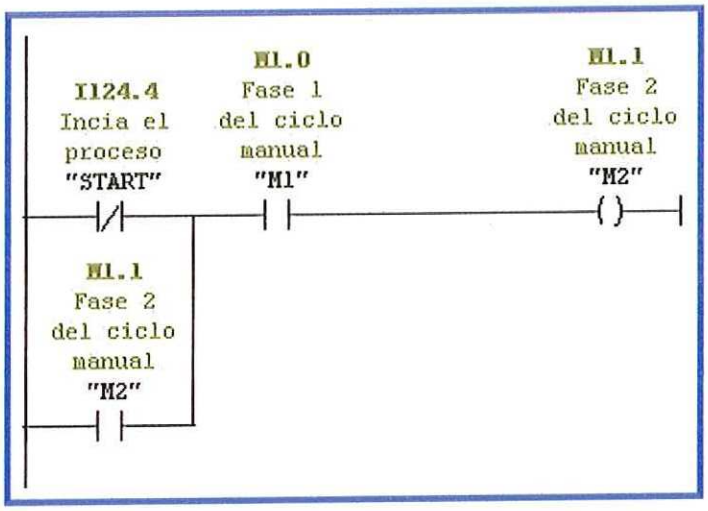
En FC2 inserte 8 segmentos (uno para cada fase) y configúrelos de igual forma a como se hizo con Automático.

Procediendo de la misma forma a como se hizo en Automático, el primer segmento (M1) espera el accionamiento de START para que inicie la operación, como se necesita memorizar este estado se usa el mismo recurso de la retroalimentación.



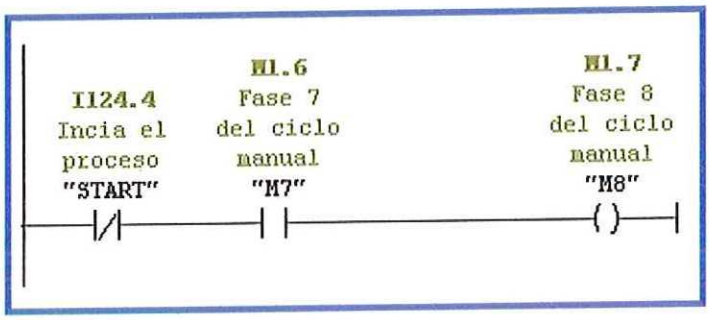
Nota: *el ciclo manual no requiere del uso de la señal de los sensores.*

La siguiente fase (M2) es el momento en que se suelta START, este segmento se comanda por START negado y requiere además que su predecesor M1 haya operado.

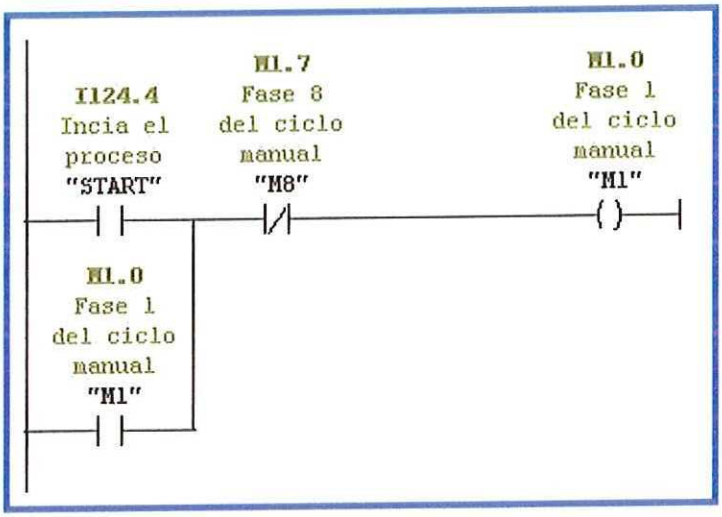


M3, M4, M5, M6 y M7 se configuran de la misma forma, recuerde que según la gráfica en M3, M5 y M7 se oprime START y los estados restantes son de espera.

M8 por ser el último ciclo no se memoriza esta fase también es de tiempo de espera y es la que culmina todo el ciclo de operación.



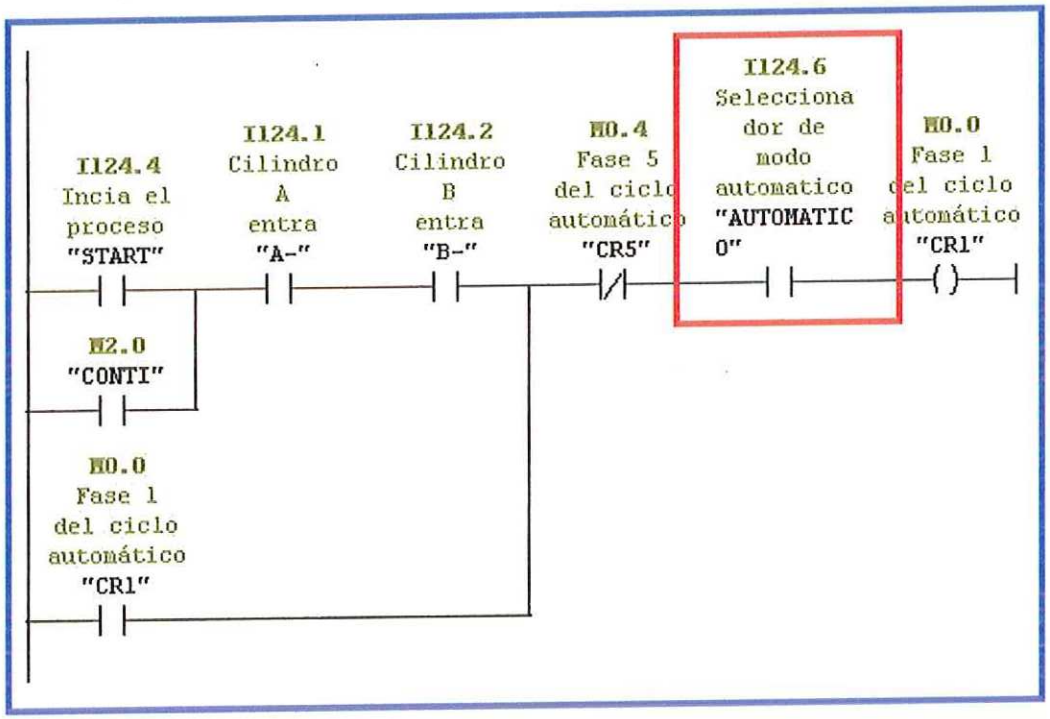
La ultima fase desactiva la primera, así que esta fase negada se agrega al segmento de M1. Al desconectarse M1 las demás fases se desconectan automáticamente.



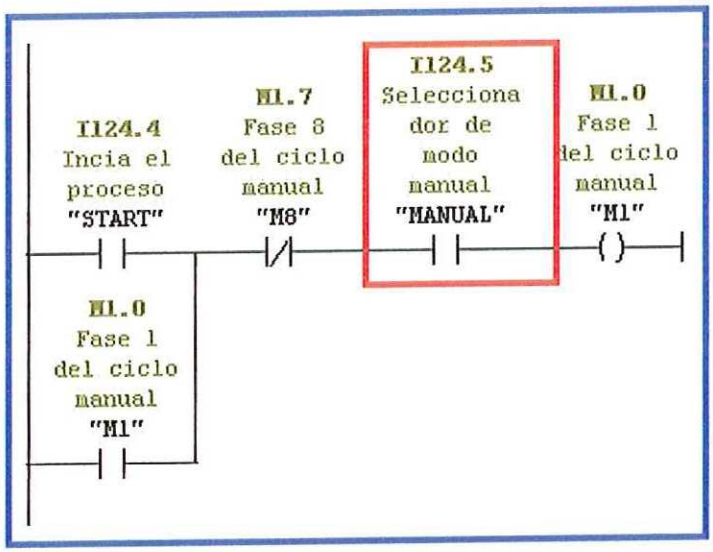
Guarde los cambios y cárguelos al PLC.

Usando el segundo interruptor con enclavamiento, se usan sus dos posiciones para convertirlo en un selector de modo *Manual* a *Automático.* Este selector se agrega al primer segmento ambos modos.

Seleccionador de el modo automático.



Seleccionador del modo manual.

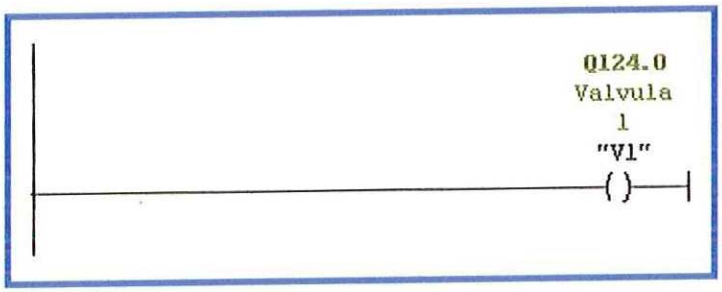


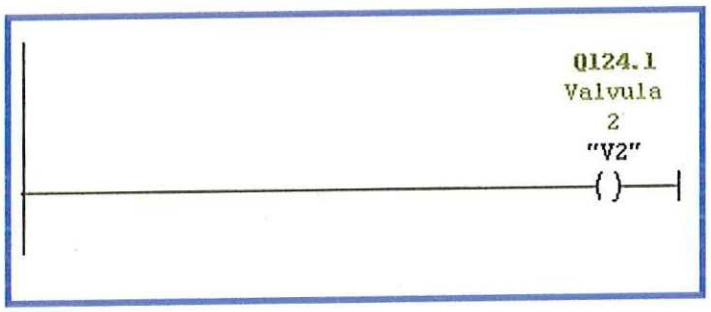
Ya listo los controles **es necesario implementarlos en el accionamiento** directo de las bobinas de la válvulas las cuáles aún no se han mencionado. **Para esto edite la función FC3.**

Para este ejemplo se configuraron las salidas Q124.0 para controlar la

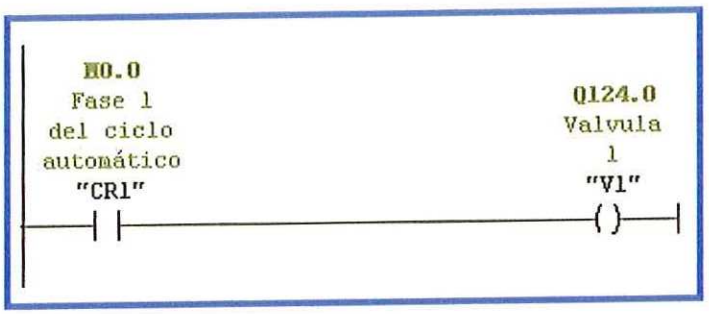
válvula que gobierna al pistón B (llamada V1), y Q124.1 para el pistón A (llamada V2).

Asigne a cada válvula un segmento.

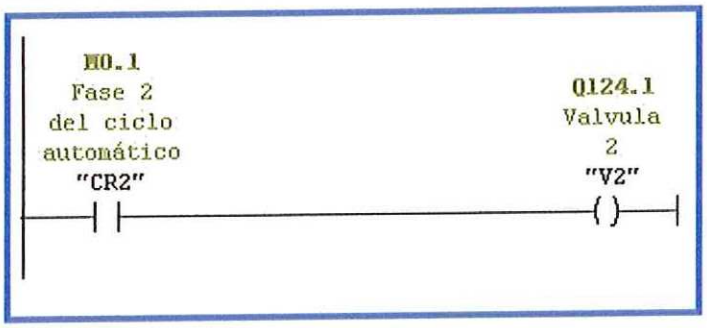




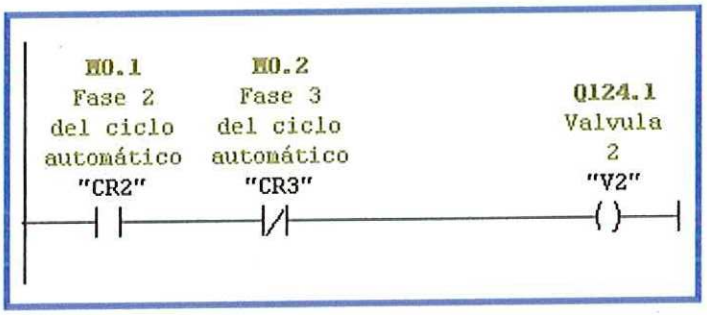
Las válvulas son accionadas en dos modos el *Manual* y el *Automático,* en el modo *Automático* el pistón B (observando la gráfica de tiempos) es el primero en salir, es decir se expande en la fase 1 (CR1), así que cuando CR1 (M0.0) se active su estado accionará directamente a V1 la cual controla a la válvula del pistón B.



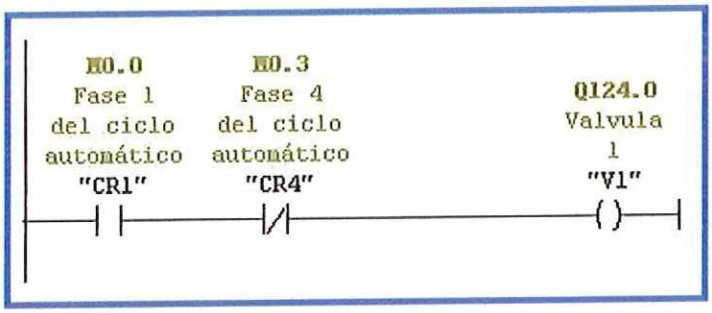
De igual forma CR2 activa al pistón A el cual es controlado por V2.



Para desactivar a A se agrega la fase 3 (CR3) negada, de esta forma cuando se cumple esta fase V2 se desactiva.



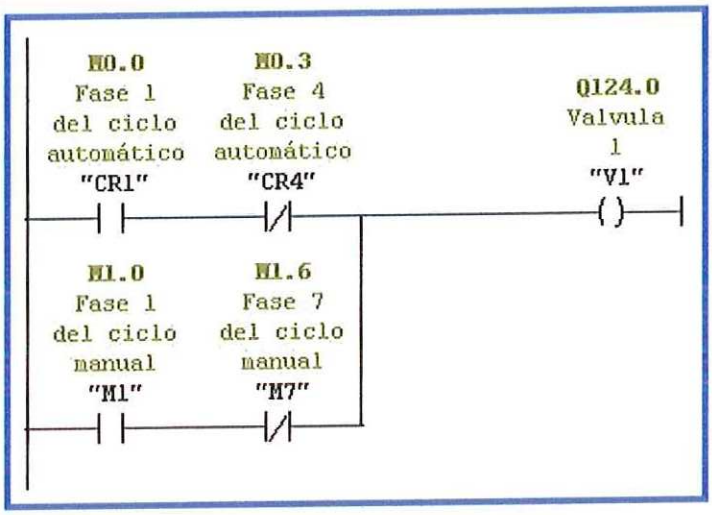
El pistón B se desactiva en la fase 4.



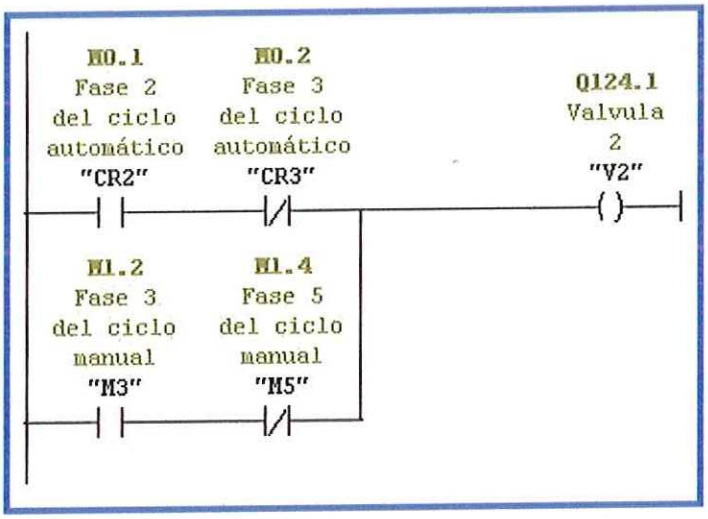
De esta forma el sistema ya puede operar en modo *Automático* ya sea de manera *Continua* o de manera *Única*.

Ahora se adaptará el control Manual, para esto se hace una conexión tipo OR con Automático y se hace el mismo análisis de ver en que fase se activa o se desactiva cada pistón.

Para B el control quedó de la siguiente forma.



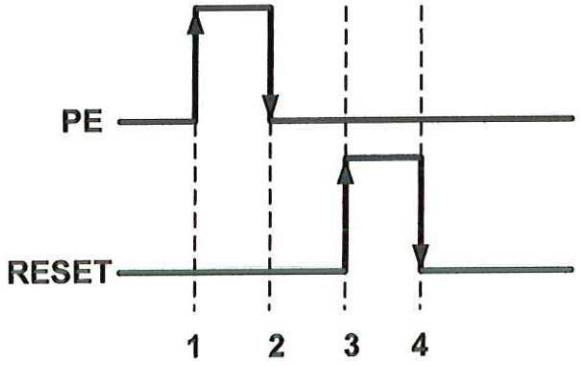
Mientras que para A se puede observar que quedó así.



Guarde los cambios y cárguelos al PLC.

Ahora sólo queda configurar el *STOP* y el *RESET*, el efecto del sistema al presionar el botón STOP es el de parar y terminar el proceso, cuando se presiona STOP los cilindros quedan en una posición fija que la define la fase en la que aconteció el momento en que se decidió detener el proceso. Volverán a la etapa inicial cuando se presiona RESET, mientras tanto el sistema no aceptará las órdenes que se puedan dar manteniéndose de esta forma inerte.

El STOP y el RESET son dos botones que se tienen que usar consecutivamente su proceso dura cuatro fases e inicia cuando se presiona STOP. El siguiente diagrama describe la etapa.

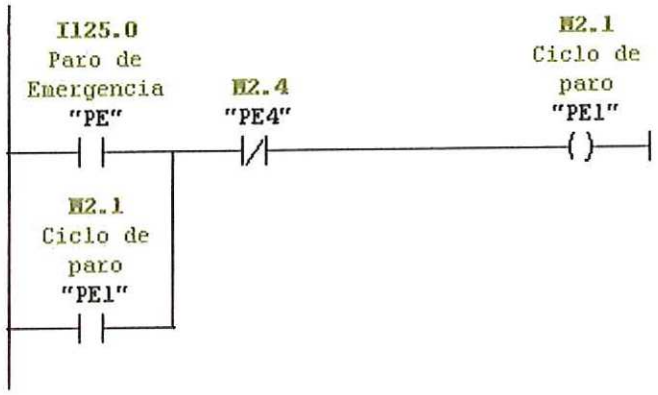


Editando a FC4 se creará un programa que ejecute los pasos de detención, edite en la tabla de símbolos cuatro bits de las marcas para esto, como se muestra a continuación.

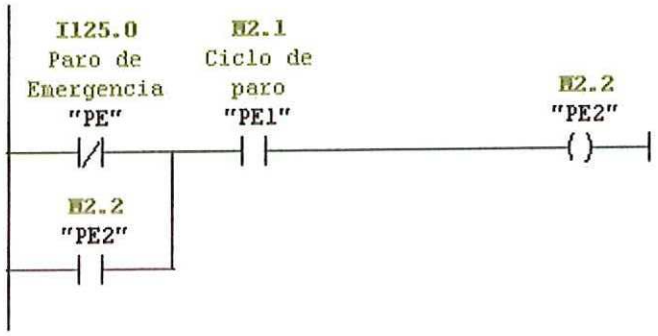
PE1	M	2.1	BOOL	Ciclo de paro
PE2	M	2.2	BOOL	
PE3	M	2.3	BOOL	
PE4	M	2.4	BOOL	

Ahora se deben insertar cuatro segmentos para cada fase, recordando que STOP y RESET son accionamientos normalmente abiertos sin enclavamiento, se realiza el procedimiento de retroalimentación implementado anteriormente en los modos *Manual* y *Automático*.

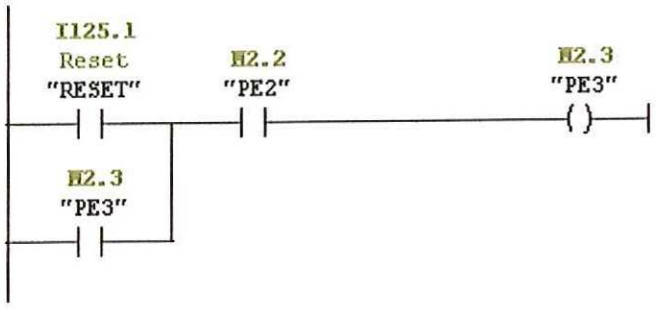
Segm. 1 : Ciclo de paro



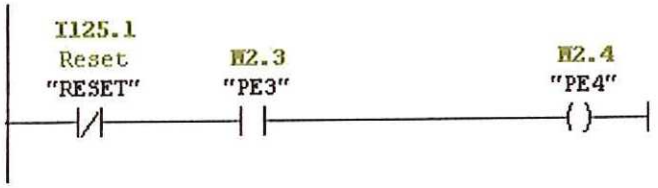
Segm. 2 : Título:



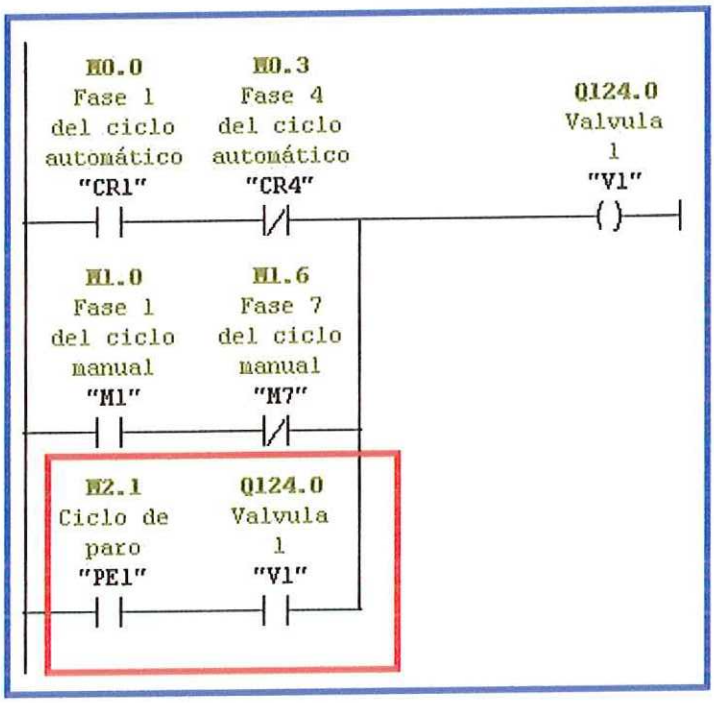
Segm. 3 : Título:

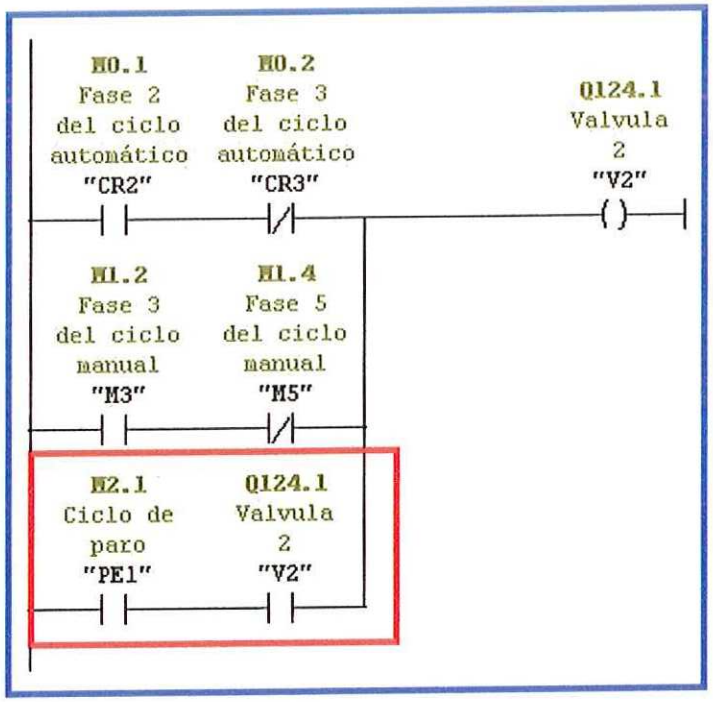


Segm. 4 : Título:



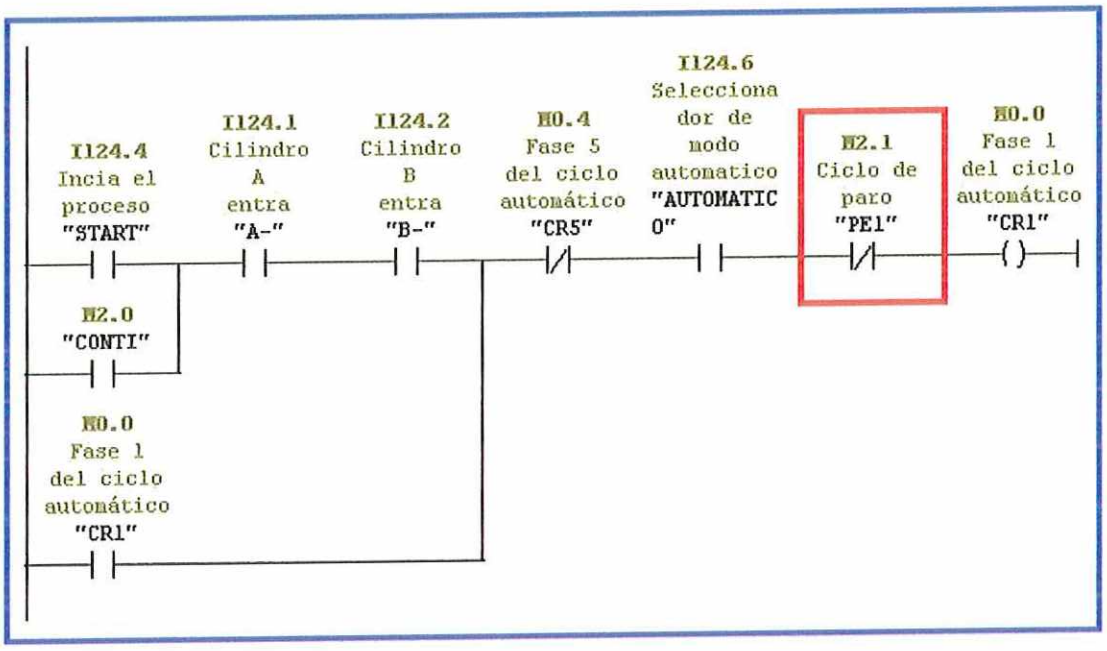
Es necesario generar el efecto de detención en los cilindros cuando se da STOP. Para esto se realiza la siguiente modificación a la etapa de potencia.

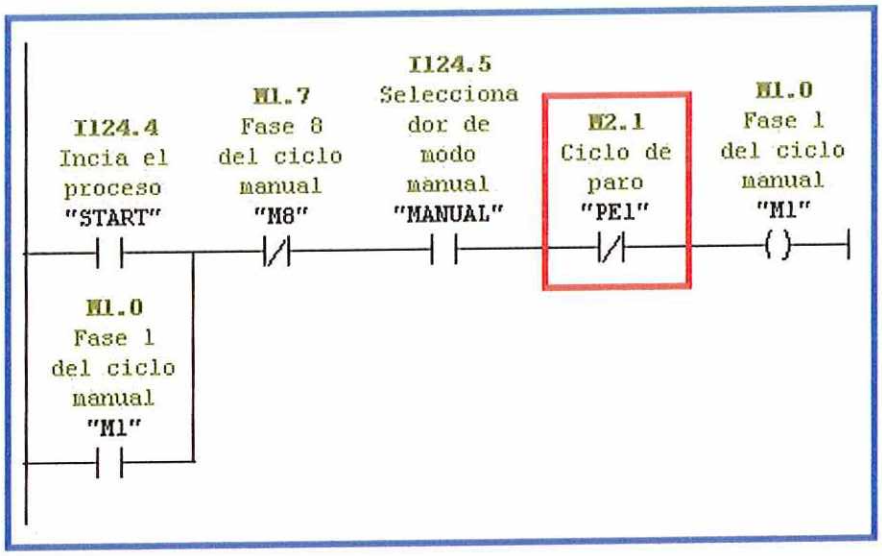




Cuando se acciona STOP, PE1 se activa inmediatamente, y de acuerdo al estado de las válvulas (V1 y V2) estas conservarán la última posición que tengan.

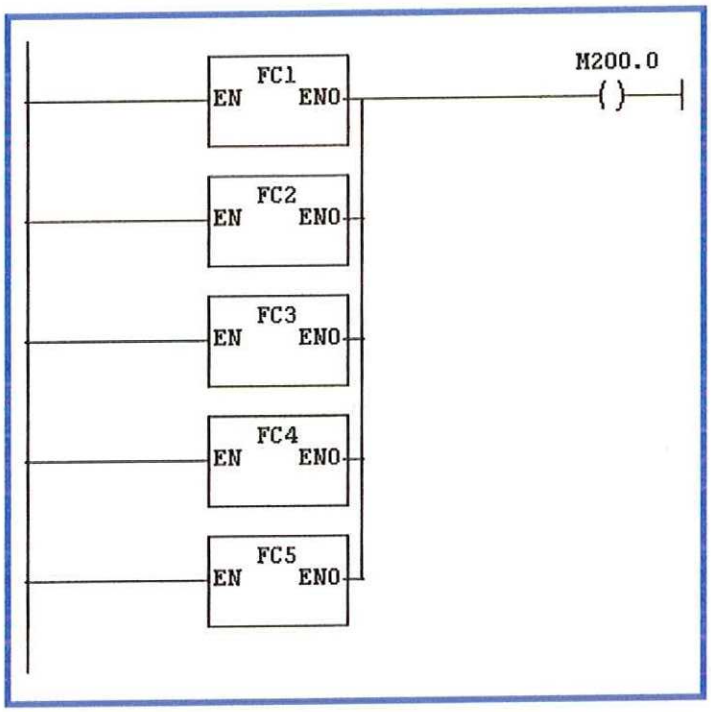
Sin embargo es necesario resetear el sistema para que vuelva a la posición inicial, para esto se agrega al primer ciclo de segmento (tanto en *Manual* como en *Automático*) PE1 negado el cual, al activarse desconecta a CR1 y así consecutivamente al resto de fases.





De esta forma ya están listas las funciones que controlarán el proceso. Ahora solo falta implementarlas en el OB1 para que puedan operar.

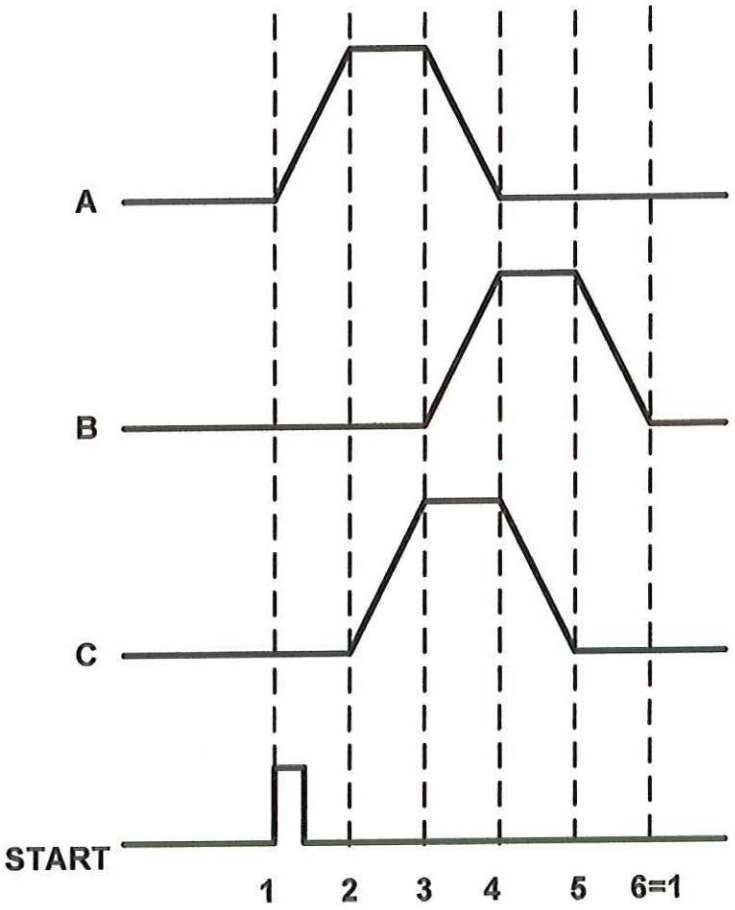
En el OB1 ingrese todas las funciones en un solo segmento en conexión tipo OR, cierre el segmento con una marca, esta debe tener una dirección que no coincida con las usadas durante la práctica.



Guarde los cambios y cárguelos al PLC. Ahora observe su funcionamiento.

6. ACTIVIDAD COMPLEMENETARIA

Dos pistones de doble efecto (A y C) y uno de simple efecto ejecutan el proceso que se describe en el siguiente diagrama de tiempos:



Aplique al problema anterior todos los conceptos implementados en el desarrollo de problemas a través del método estructurado, es decir Modo Manual, Automático (Continuo, único), etapa de potencia, Paro y Reset.

El programa requiere de luces indicadoras para señalar que se está ejecutando en ciclo Manual o en Automático. Además cuando se accione STOP debe quedar una luz intermitente que señale este estado. **La luz se debe apagar**

***Nota**: tome previa indicación del número de sensores que necesita para poder realizar esta práctica.*

PRÁCTICA 10
ENTRADAS Y SALIDAS ANÁLOGAS

PRÁCTICA 10
ENTRADAS Y SALIDAS ANÁLOGAS

1. OBJETIVOS

- ✓ Configurar el módulo de entradas y salidas análogas para cualquier tipo de proyecto.
- ✓ Utilizar las herramientas disponibles en Step 7 y Simatic para el tratamiento de señales analógicas.
- ✓ Implementar los conocimientos adquiridos en la solución de problemas.

2. CONCEPTOS FUNDAMENTALES

Hasta el momento todas las prácticas se han enfocado en la solución de problemas a través de variables digitales. En este entorno el comportamiento de las señales se basaba en sólo dos estados 1 y 0. Cuando una variable tiene un comportamiento cuyo estado puede tomar infinitos valores los cuales están comprendidos entre un margen mínimo y uno máximo, se dice que es una variable de tipo analógica.

Las señales analógicas representan magnitudes cambiantes de comportamientos físicos reales, como lo son presión, temperatura, velocidad, cambio de longitud, etc. La tecnología ha permitido la creación de sensores y transductores que puede percibir estos cambios y convertirlos en señales proporcionales de tipo eléctrico (ya sean voltios o amperes).

Este tipo de señales tienen un comportamiento errático que puede ser afectado por el mismo entorno, señales como el ruido electromagnético, o efectos físicos como una mal puesta a tierra, generan cambios en al magnitud de la señal eléctrica proveniente del transductor que pueden afectar un posterior control de la misma.

Uno de los puntos más importantes del tratamiento de señales análogas es la linealidad, la cual se describe como la respuesta ante los cambios en la magnitud de una forma rectilínea (una curva que se puede describir con la

formula matemática $y= mx + b$), no hay una señal ciento por ciento lineal, y una linealidad absoluta es en teoría un parámetro ideal. La linealidad depende de los componentes del transductor como los condensadores, resistencias y bobinas. Muchas veces se hace imposible describir una magnitud a través de una línea recta y se recurre a otras gráficas como la curva, o la parábola ($y=ax^2$). Para conocer la respuesta de un sensor es necesario recurrir al data sheet del fabricante, si no se encuentra, se recurre a la toma de datos.

El otro punto a considerar es la conversión de la señal, a pesar de la naturaleza analógica de esta, el PLC tiene que convertirla en una variable digital para poder trabajar con ella. Para esto cuenta en su hardware con un dispositivo llamado conversor análogo digital, este hace un muestreo sobre la señal a transformar y va arrojando valores digitales que son proporcionales, la capacidad de leer el más mínimo cambio de la magnitud se llama resolución y entre más resolución tenga un conversor más eficiente es.

3. ACTIVIDAD PREVIA

Consulte el data sheet del sensor de presión ***7MF1563*** de Siemens, tenga en cuenta para esta consulta, la alimentación, los rangos de medición del equipo, la capacidad máxima, los rangos y tipos de señal que pude generar como respuesta, el estado del entorno para un óptimo trabajo y las conexiones, para esta última parte consulte el material extra que está al final de la práctica.

4. LISTA DE MATERIALES

1) 1 sensor de presión 7MF1563 de Siemens
2) 1 multímetro

5. DESARROLLO METODOLÓGICO

Nota: *para el desarrollo de esta actividad es necesario haber elaborado la practicas 0 , Configuración del PLC Siemens Simatic y 6 Transferencia de paquetes de datos.*

Abra ***SIMATIC/Administrador Simatic***, cree un nuevo proyecto con el nombre *Práctica 10.*

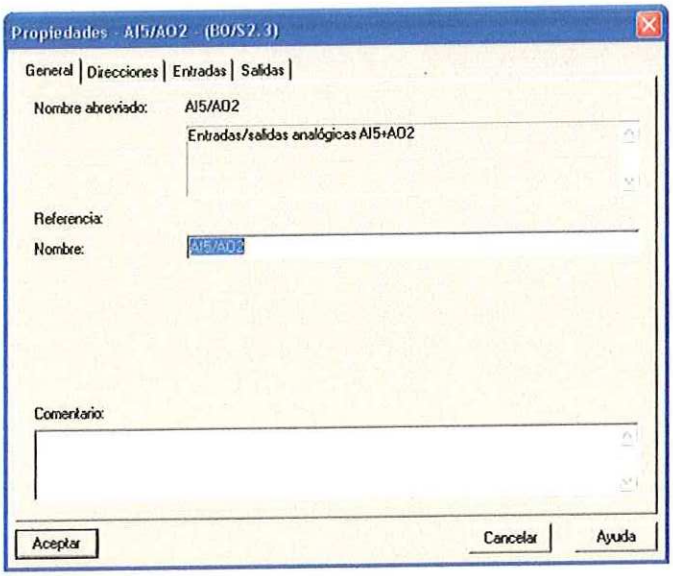
En el ***HW-Config,*** tome nota especial de la siguiente indicación, **para poder configurar el módulo de entradas y salidas análogas** es necesario dar doble clic sobre este ítem, lo puede identificar por tener el nombre de *AI5/A02.*

1	PS 307 2A	6ES7 307-1BA00-0AA0				
2	**CPU 314C-2 DP**	**6ES7 314-6CF00-0AB0**	**V1.0**	2		
X2	DP				1023*	
2.2	DI24/DQ16				124...126	124...125
2.3	AI5/AO2				752...761	752...755
2.4	Conteje				768...783	768...783
2.5	Posicionamiento				784...799	784...799
3						
4	DI8/DO8xDC24V/0.5	6ES7 323-1BH80-0AA0			0	0
5						

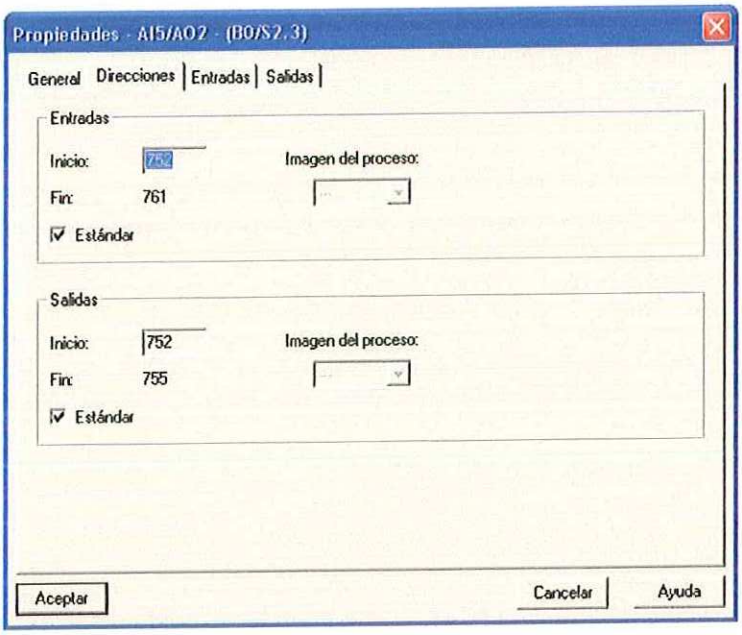
El PLC Simatic 314C 2-DP cuenta con cinco entradas análogas y dos salidas, cada entrada tiene una tamaño de una Word (2 bytes), empezando desde el 752-753 para la entrada 0 (siguiendo 1, 2 y 3), hasta la 760-761 para la PT (esta es una conexión especial para potenciómetros) . Las salidas tienen el mismo tamaño 752-753 para la salida 0 y 754-755 para la salida 1.

2.2	DI24/DQ16				124...126	124...125	
2.3	AI5/AO2				752...761	752...755	
2.4	Conteje				768...783	768...783	

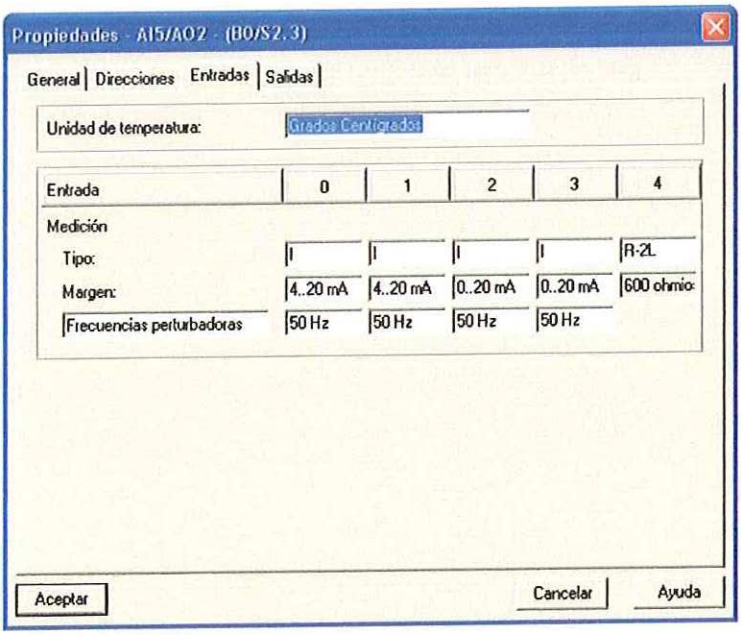
Cuando se da doble clic sobre *AI5/AO2*, se despliega una nueva ventana con el nombre de *Propiedades - AI5/AO2 – (B0/s2.3),* esta se abrirá automáticamente sobre la pestaña *General,* aquí se puede cambiar el nombre e insertar cualquier comentario sobre el módulo.



La siguiente pestaña *Direcciones*, especifica la ubicación del módulo, las direcciones iniciales (byte 752 tanto para las salidas como para las entradas) son dadas de forma predeterminada por el equipo, esta se pueden modificar con deseleccionar la casilla *Estándar,* una vez hecho esto basta con modificar la ubicación del primer byte para que los demás se adapten automáticamente a este cambio.



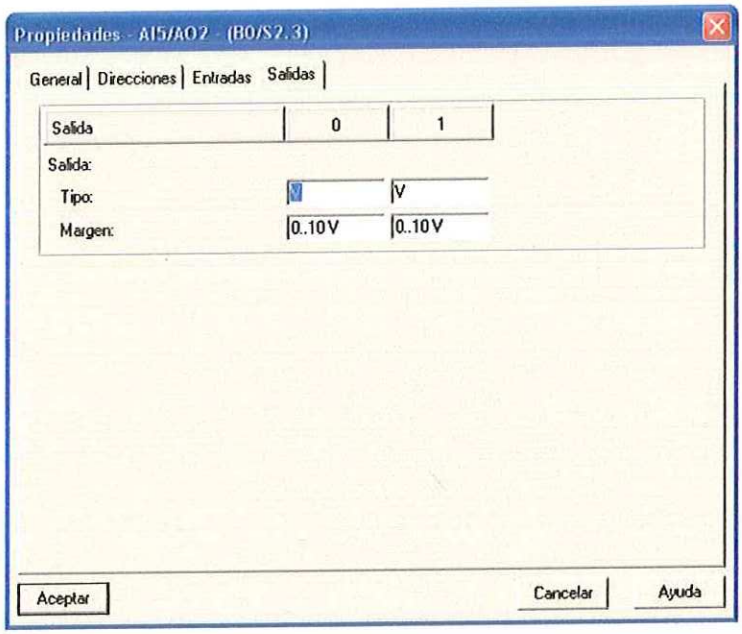
La siguiente pestaña es *Entradas*.



Esta pestaña puede modificar la configuración del módulo de Entradas análogas, cada una de las cinco (0, 1, 2, 3, 4, y 5) puede ser configurada de forma independiente si afectar a las otras. En el espacio identificado como medición, se puede modificar el Tipo (Voltaje o Corriente, de acuerdo a la señal que genere el sensor), el Margen (que es el rango de medición del sensor) que puede variar de acuerdo a la señal (0..10V ó –10...10V para señales de voltaje, y, 0..20mA ó 4...20 mA para señales de corriente). El tercer rango específica la magnitud de cualquier frecuencia que puede afectar al aparato.

Configure la entrada 1, para que lea una señal de corriente con un rango de 4 a 20 mA.

La siguiente pestaña *Salidas*, se encarga de la configuración de las salidas análogas. Bajo la dirección de esta, se puede especificar el tipo de señal que se desea generar con el PLC (ya sea de voltaje o de corriente) y su margen.

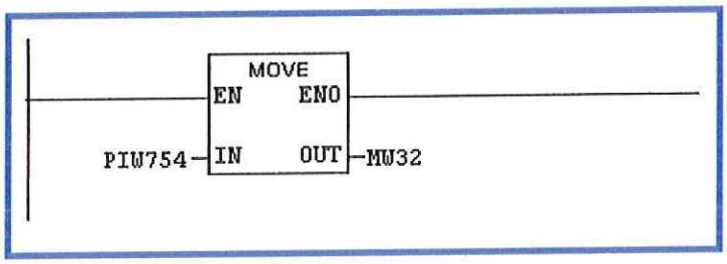


Configure la salida 0 para que genere una señal de voltaje con un rango de 0 a 10 voltios.

Guarde estos cambios y cargue la configuración al PLC.

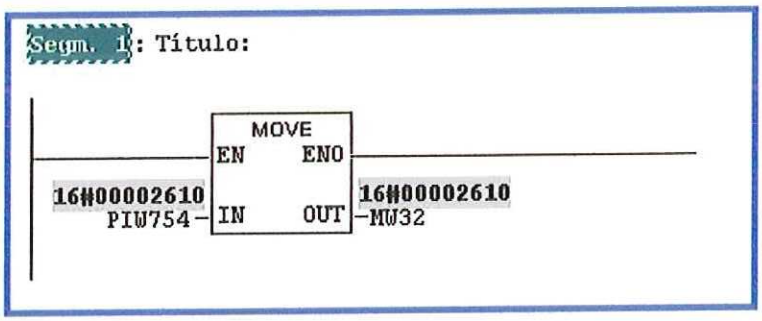
Ahora conecte el sensor de presión a al entrada análoga 1 (Ver *Material Extra N° 4*).

Desde el *OB1* se controlará la señal entrante del sensor. Como primera medida es necesario transferir estos datos a un espacio de las marcas. Así que inserte un módulo de transferencia (MOVE) al primer segmento.



Se transferirán datos de la entrada análoga 1 (identificada con la dirección *PIW 754*) a la marca 32 (MW32), las entradas análogas tienen una magnitud de un Word por eso es necesario especificar el tamaño de la marca con la letra W.

Guarde los cambios y cárguelos al PLC. Ponga este en RUN y en el computador de clic en observar. Verá una serie de datos sobre la dirección PIW754, modifique la presión que el sensor esta detectando y verifique que estos datos cambian a la par con el sensor. **Si no es así, verifique la conexión del sensor o la configuración de la entrada análoga.**



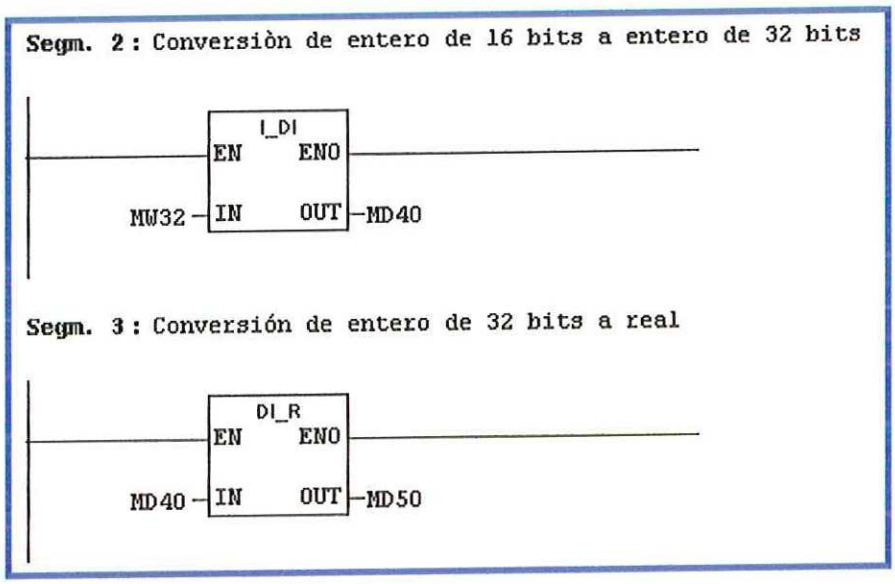
La señal mostrada aquí es de tipo hexadecimal, para visualizar su magnitud es necesario convertir estos valores a datos que sean más fáciles de entender por el usuario, por esto es necesario hacer una conversión de hexadecimal a real.

Para esto existe una serie de funciones muy útiles para realizar trabajos de conversión, esta se despliegan en una carpeta con el mismo nombre *Conversión.* Aquí existe funciones como transferir de BCD a enteros (BCD_I), de enteros a reales (DI_R) y opciones de complemento a 2 (el negativo) para valores (NEG_I, NEG_DI, NEG_R), etc.

Para efectos prácticos sólo se explicarán las funciones que se implementarán en el proyecto, las demás quedan para investigación del estudiante.

Sólo existe una función para convertir números enteros a reales DI_R, sin embargo, esta **sólo convierte enteros con una magnitud de 32 bits a reales**, la señal proveniente del sensor tiene una magnitud de 16 bits la cual es totalmente incompatible con DI_R.

El módulo I-DI, convierte un dato tipo Word a uno Double-Word, la señal almacenada en MW32 pasará primero por este tratamiento para luego ser convertida a real. Inserte dos nuevos segmentos y desplace estas funciones. Cada cambio que haga sobre la variable debe guardarse en una marca diferente para evitar errores.



La marca MD40 (D por Double Word) almacenará el número convertido a 32 bits, mientras la marca MD50 almacena el valor convertido en real.

Ahora se puede conocer el valor que toma el sensor cuando varía la magnitud de la presión, sin embargo el interés es crear proyectos que operen con valores con los cuales una persona esta acostumbrada a trabajar, es decir, una valor expresado en Pascales, psi o Bares.

Teniendo en cuenta el data sheet del sensor de presión, se sabe que este tiene una respuesta de tipo lineal, es decir que su comportamiento se puede describir con la ecuación:

$$\textbf{(1)} \qquad y = mx + b$$

En esta ecuación es importante identificar dos constantes la pendiente (m) y el término que corta con el eje de las y (b).

Para esto es necesario conocer varios puntos de la recta y graficarla. Para esto disminuya la presión del sistema hasta 0 y visualice desde el computador el valor que toma *MD50* (valor en formato Real). Observará como se reduce, es necesario que espere un tiempo mientras el dato que da el computador se estabiliza, cuando esto sucede tome nota.

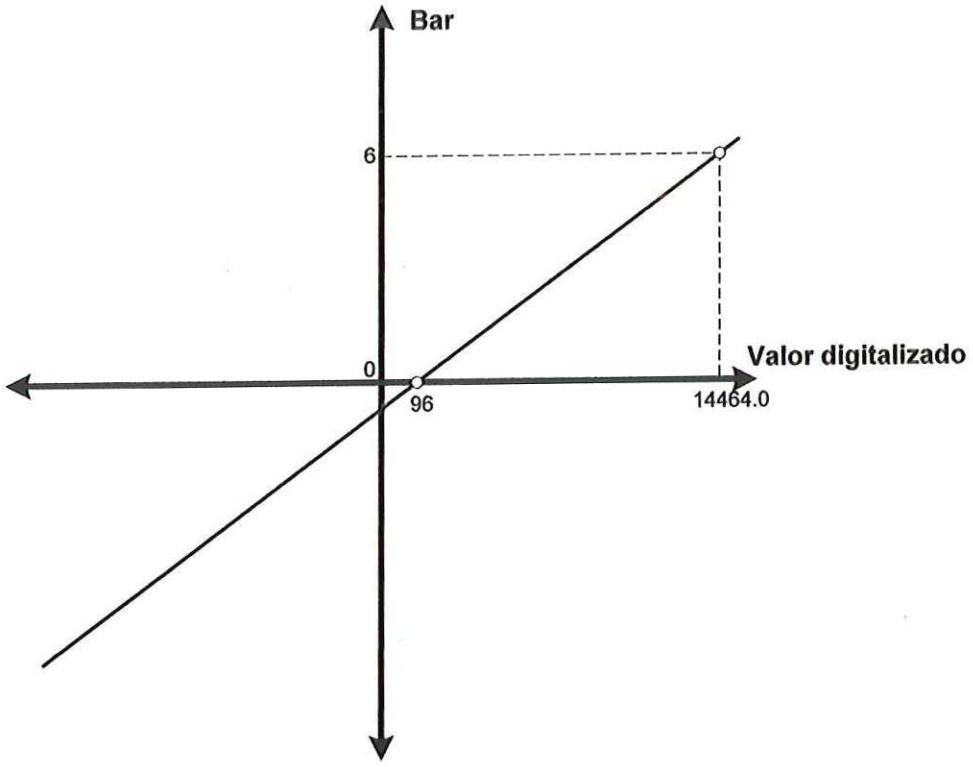
VALOR OBSERVADO POR EL PLC	BARES
96,0	0

***Nota**: el valor observado por el PLC puede variar entre equipo y equipo.*

Ahora aumente la presión hasta el valor máximo (6 bares en nuestro caso), es importante que no supere el valor máximo que el sensor puede soportar.

VALOR OBSERVADO POR EL PLC	BARES
14464,0	6

A con los datos obtenidos se pude trazar la siguiente gráfica:



Para esta gráfica la ecuación de la recta queda así:

(2) $$Bares = m * (valor\_digitalizado) + b$$

La pendiente de la recta es entonces:

$$\textbf{(3)} \quad m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{6 - 0}{1464.0 - 96.0} = \frac{6}{14368} = 4.17e - 4$$

Reemplazando m en la ecuación **(2)**:

$$\textbf{(4)} \quad Bares = 4.14e - 6 * (valor\_digitalizado) + b$$

Para hallar el valor de b, basta con reemplazar a x,y con valores ya conocidos, para esto se recurre a los datos de la primera tabla:

VALOR OBSERVADO POR EL PLC	BARES
96,0	0

Reemplazando en la ecuación se tiene:

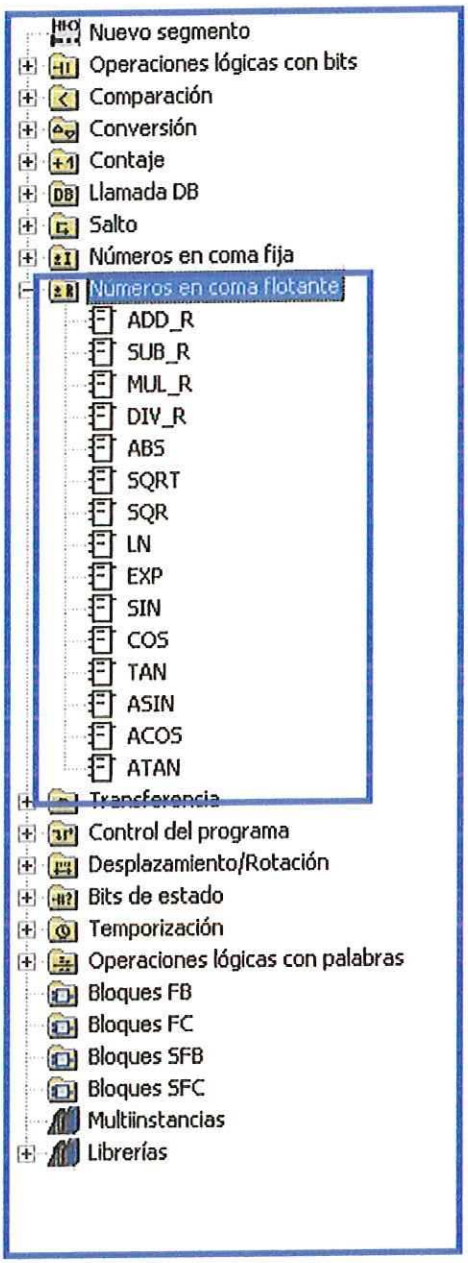
$$0 = 4.14e - 4 * 96.0 + b$$
$$0 = 0.04001 + b$$
$$b = -0.04001$$

Reemplazando a b en **(4)** se obtiene finalmente:

$$\textbf{(5)} \quad Bares = 4.14e - 6 * (valor\_digitalizado) - 0.04001$$

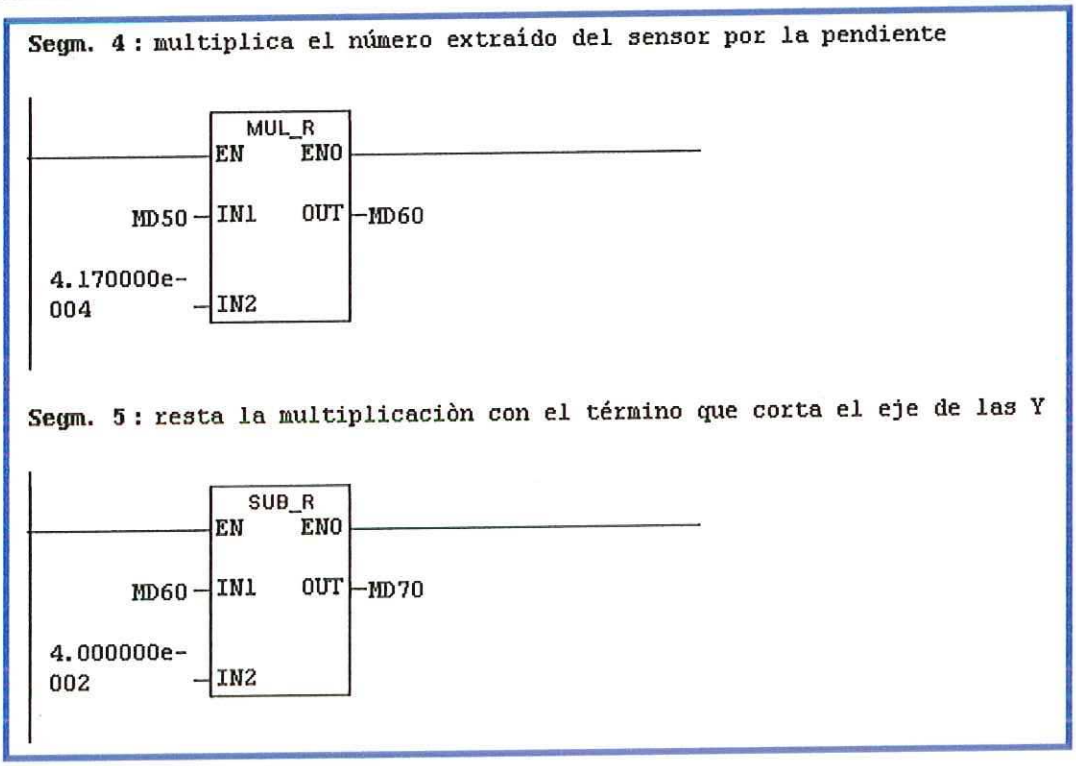
En otras palabras, el valor que da el PLC a través de la marca MD50 debe ser multiplicado por la pendiente y restado por el término que corta con el eje de las y.

Para poder realizar estos cálculos matemáticos, Step 7 cuenta con herramientas que permiten realizar procedimientos matemáticos sobre números reales, o como los llama números en coma flotante. Para esto despliegue la carpeta *Números en coma flotante*, podrá observar una gran cantidad de operaciones matemáticas tales como multiplicar, dividir, restar, sumar, hallar el valor absoluto, formulas trigonométricas, entre otras.



Para efectos prácticos sólo se explicarán las funciones que se implementarán en el proyecto, las demás quedan para investigación del estudiante.

Primero se multiplicará el valor obtenido de MD50 (valor digitalizado) por la pendiente de la gráfica, para esto se inserta a un nuevo segmento la función MUL_R, la cual multiplica dos números reales cualesquiera (IN1, IN2), el resultado es expuesto en la casilla OUT. A este valor obtenido se le resta la pendiente, para esto en un nuevo segmento se inserta la función SUB_R. Observe la gráfica que se muestra a continuación.



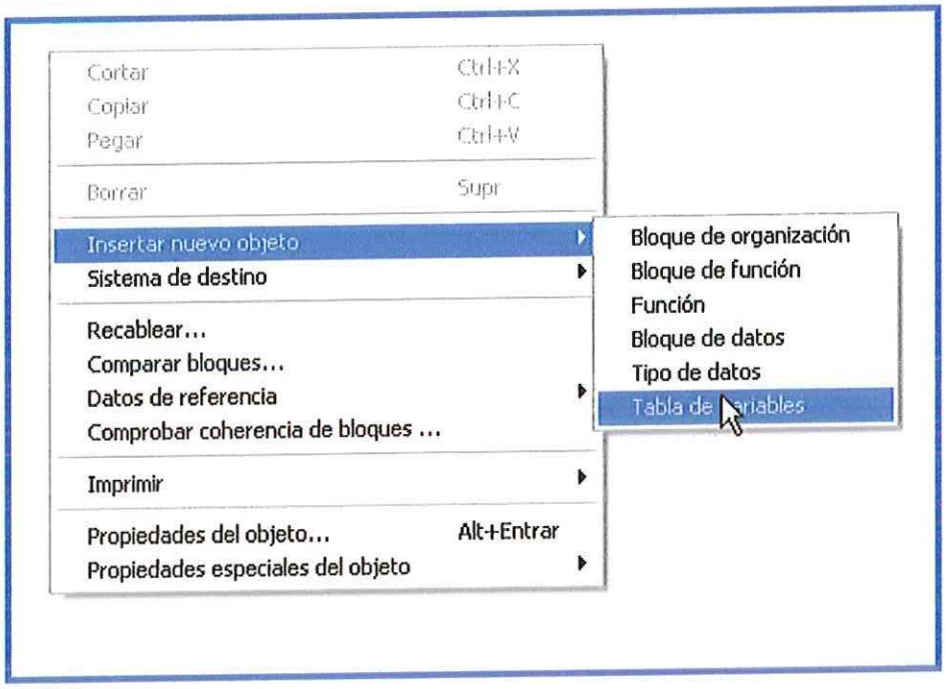
El valor que multiplica a MD50 en MUL_R se inserta en IN2, escriba en ella 4.17e-4 (sin dejar espacio), luego a la casilla OUT enlácela con un marca (MD60, por ejemplo) para que almacene el dato multiplicado.

En SUB_R, escriba MD60 (término que gurda el resultado de la multiplicación) en IN1 y 0.04 en IN2 (note como se reconfigura el valor después de dar enter). Ahora almacene el valor de la resta en otra marca (MD70, por ejemplo).

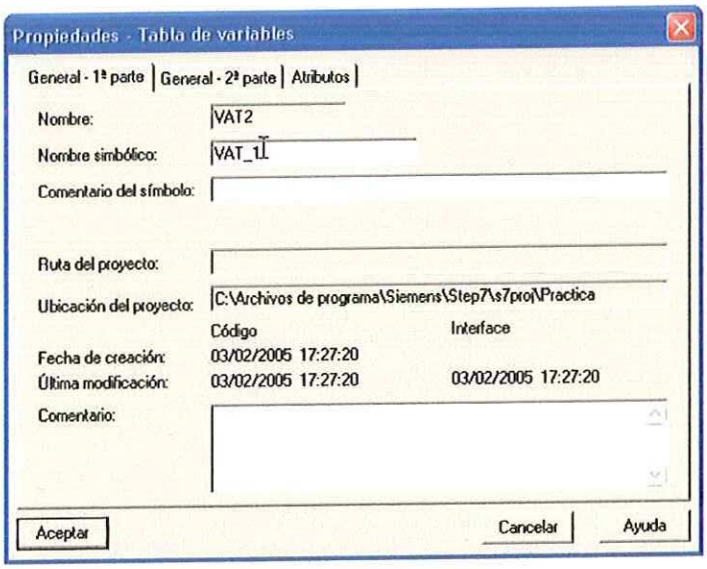
Ahora cada vez que varíe la presión MD70 dará el valor real en Bares.

Guarde los cambios y cárguelos al PLC.

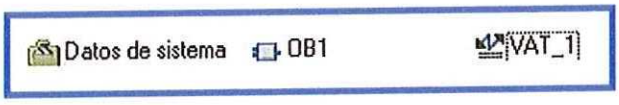
Para poder visualizar los datos reales vaya hasta el *Administrador Simatic*, aquí en la carpeta *Bloques* (la misma donde está almacenado el OB1), de clic derecho sobre el espacio en blanco, en la ventana que aparece, busque *Insertar Nuevo Objeto*, luego *Tabla de variables*.



En la ventana emergente puede modificar le nombre simbólico, de clic en *Aceptar.*

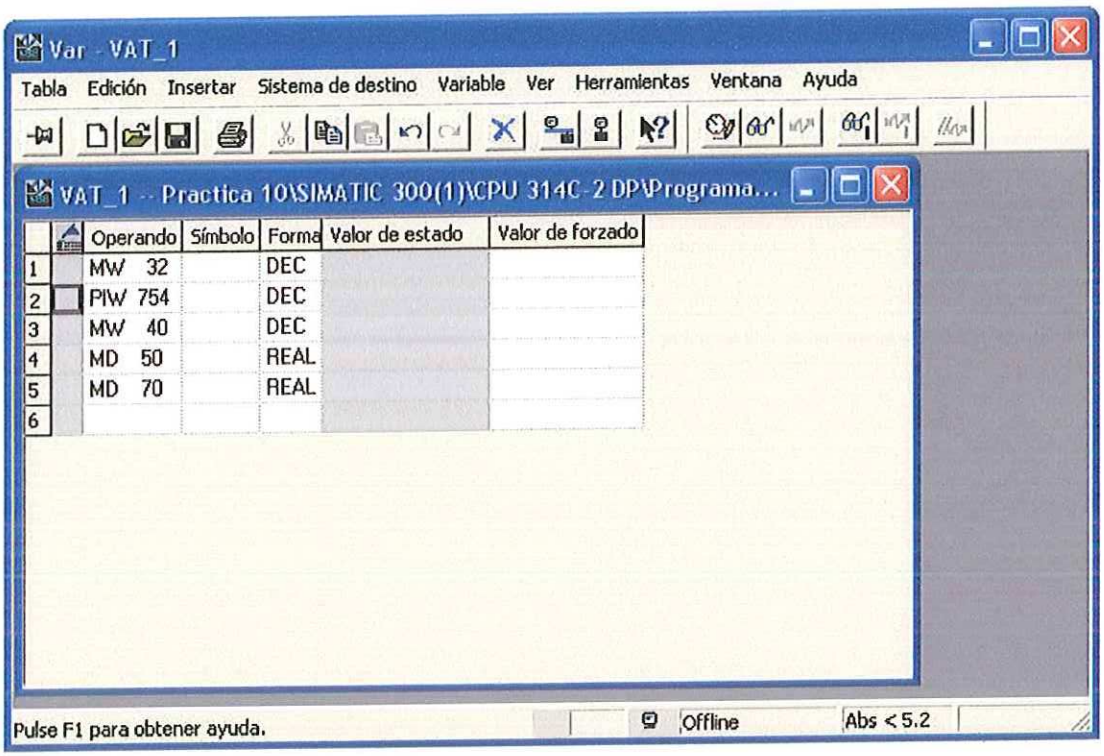


Verá un nuevo icono con el nombre de *VAT_1* en *Bloques.*



De doble clic sobre este icono.

Se abrirá una ventana emergente con el nombre de *Var – VAT_1*, en esta ventana podrá visualizar cualquier cambio que presente una variable, ya sea esta un bit, un byte, word o double word.



En la columna *Operando* digite las marcas donde se almacenan los datos importantes, como lo muestra la gráfica anterior, en la columna *Formato* puede determinar la forma en que quiere ver la variable, para MD70 asigne un formato de tipo *REAL*. Guarde estos cambios y de clic en el icono *observar*.

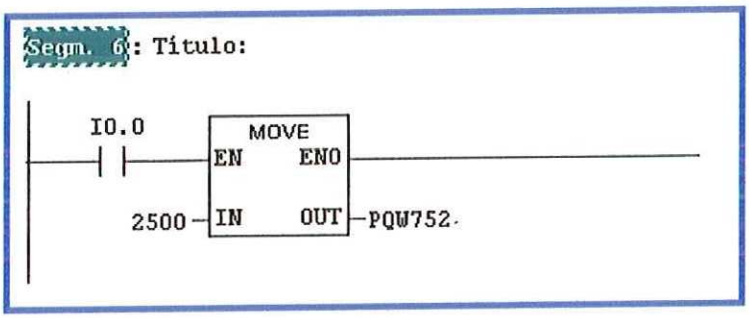
Observe la columna *Valor de estado*, modifique la presión a 4 bares y observe si MD70 muestra el valor de 4 (o algo muy cercano). Si no es así, revise la ecuación de la recta y realice los cambios que considere necesarios.

Ya habiendo confirmado el correcto funcionamiento de las entradas análogas, se procederá a trabajar con las salidas de este mismo tipo.

El bloque de salidas análogas emite una cantidad infinita de valores entre el rango que se ha configurado, en este caso de 0 a 10 V. El objetivo ahora es configurar las salidas para que trabajen a la orden del usuario, de esta forma cuando le demos un número dentro del rango establecido (por ejemplo 4.5) la salida emitirá un voltaje del mismo tipo.

Para este ejercicio se requiere de un multímetro para la toma de datos.

En el *OB1*, inserte un nuevo segmento y agregue un bloque de transferencia, conecte a OUT con el canal 0 de las salidas análogas (PQW752), en la entrada EN, inserte un contacto normalmente abierto y direcciónelo con el bit 0.0 de las entradas digitales simuladas. En IN asigne cualquier número como por ejemplo 2500. Guarde los cambios y cárguelos al PLC.



Habilite la entrada 0.0 y con el multímetro observe el valor del voltaje que arroja la salida análoga.

Para realizar el ejercicio propuesto, es necesario proceder de la misma forma que se hizo con el sensor de presión. Es decir, realizar una gráfica a partir de los valores tomados. Se supone que las salidas análogas del sensor son de respuesta lineal.

Con el mismo bloque de transferencia modifique el valor de IN y tome nota del voltaje arrojado, con esos datos construya una tabla.

Valor digital	Voltaje
2500	0.9
4000	1.45
13661	5
0	0

La salida análoga del PLC trabaja según lo muestra la siguiente ecuación:

(6) $$valor\_digital = m * (voltaje\_deseado) + b$$

Determinando la pendiente de esta gráfica se obtiene:

$$\textbf{(7)} \qquad y = \frac{y_2 - y_1}{x_2 - x_1} = \frac{4000 - 2500}{1.45 - 0.9} = \frac{1500}{0.56} = 2732$$

Para hallar b, se reemplaza a voltaje y valor_digital por magnitudes conocidas al igual que la pendiente.

$$0 = 0 * 2732 + b$$
$$0 = 0 + b$$
$$b = 0$$

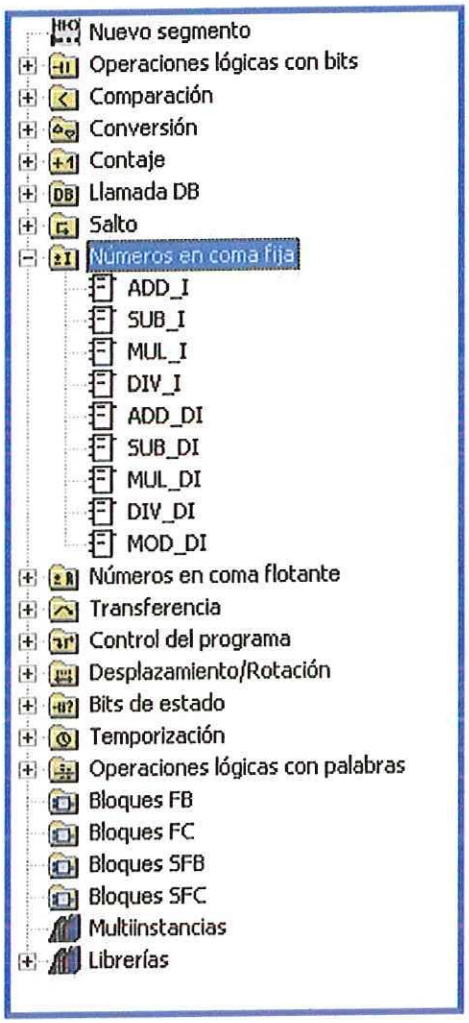
Reemplazando el valor de m y b en la ecuación (6).

$$\textbf{(8)} \qquad valor\_digital = 2732 * (voltaje\_deseado)$$

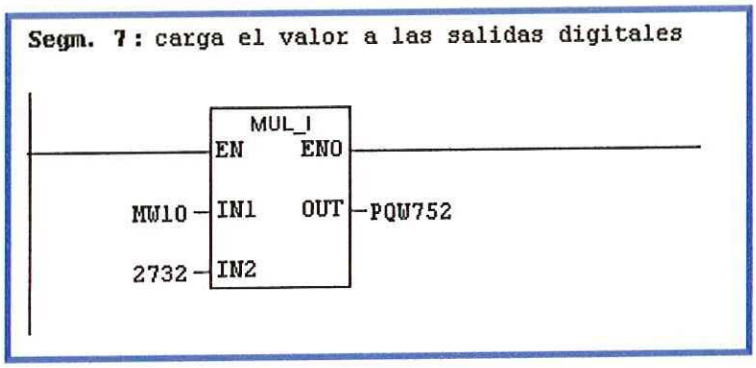
Ahora se modifica el bloque de transferencia, reemplazando a PQW72 por la marca MW10.

Números como 2500, o los que se encuentran dentro del rango de 0 a 10 V, son magnitudes enteras no reales, conocidas como *Integer*, (cuyo rango va desde –32733 hasta 32733), las cuales tienen un tamaño de 1 Word (2 bytes). Este tipo de magnitudes son compatibles con las utilizadas en las salidas análogas.

Para este formato de número también existe un módulo con operaciones matemáticas muy similar al empleado con los números reales, estas operaciones están almacenados en la carpeta *Números en coma fija*. El cual no sólo tienen las operaciones compatibles con Integer, sino también para números que van por encima del rango de este (<-32733,>32733) denominados *Double Integer*. Las funciones para variables tipo integer se identifican por terminar por la letra I (ADD_I, SUB_I), mintras para las del otro formato terminan por la letra DI (ADD_DI, SUB_DI).



En un nuevo segmento insertamos la función MUL_I, y multiplicamos los almacenado en la marca MW10 por la pendiente de la gráfica. El resultado se verá en la salida análoga PQW752.



Al multiplicar el valor del voltaje deseado por la pendiente, se dará como resultado un número digital que la salida análoga interpretará dentro de su lógica como 5V. Compruebe este resultado con el multímetro.

6. ACTIVIDAD COMPLEMENETARIA

La actividad complementaria de esta práctica se divide en dos fases:

A) Desarrolle un programa en PLC que interprete la señal proveniente del sensor de presión 7MF1563 (0 a 10 bar) el programa debe contener además un software que permita al usuario visualizar desde la tabla de variables al presión en bares, pascales y psi. Cree además una señal de alarma que indique un elevamiento de la presión por encima de los 5 bar. (La alarma se debe desactivar una vez la presión se normaliza).

B) Cree un multímetro que se pueda controlar desde el panel de entradas y salidas digitales simuladas. Para este caso el módulo de salidas análogas debe estar configurado en el rango de –10 a 10 V y los 16 bits del módulo de simulación deben estar configurados como entradas digitales (Ver práctica 0).

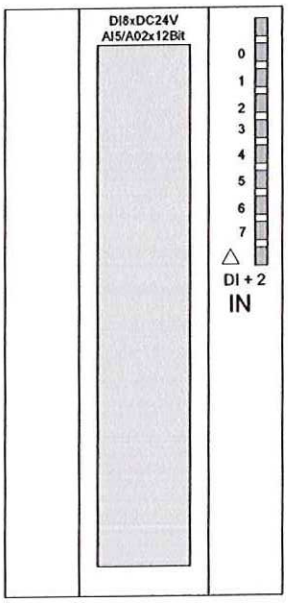
Las salidas deben responder en forma proporcional a partir de los datos que estipula la siguiente tabla:

Valor en binario	Señal de Voltaje
0000000000000000	-10
0000000011111111	0
1111111111111111	10

Verifique la repuesta con el multímetro.

MATERIAL EXTRA No. 4

El módulo de entradas y salidas análogas cuenta con cinco canales para entradas y dos para salidas.



Cada canal de entradas análogas está compuesto por tres conexiones estipuladas de la siguiente forma V, A y C. De esta forma en la entrada V se pueden conectar señales de voltaje variable, a la entrada A señales de corriente y C es el común. Así, cuando se desea conectar un sensor cuya señal de salida sea una magnitud de corriente variable se procede a trabajar con la conexión A de la entrada análoga. La entrada C, es la referencia del módulo de salida análogo y es necesario conectarla a la referencia digital del PLC.

Las salidas digitales tienen una conformación similar, **cada una tiene un puerto que emite señales de intensidad y otro que emite señales de voltaje variable.** Existe una sola conexión común para ambas salidas y es necesario referenciarla a tierra cuando se van a utilizar.

Existe una tercera terminal denominada PT y se utiliza para elementos que varían su resistencia para medir una variable física (como las galgas extensométricas).

PRÁCTICA 11
TRABAJO FINAL

PRÁCTICA 11
TRABAJO FINAL

1. OBJETIVO

- ✓ Implementar e integrar los conceptos adquiridos durante el transcurso de las prácticas para la solución de un problema.

2. CONCEPTOS FUNDAMENTALES

A través del transcurso de estas prácticas se han implementado múltiples herramientas para la solución de problemas a través de un PLC. Se inició desde temas tan básicos como el desarrollo de programas a través de esquemas de contactos, pasando por herramientas de memorización, la implementación en los proyectos de los diversos tipos de datos existentes, terminando con el trabajo de variables de tipo analógicas.

Ahora es el turno del estudiante para que aplique todos estas herramientas en la solución de un solo problema, el cual es libre, y sólo se exige que cumpla con los estándares que requiere una práctica como esta. Existen muchos procesos que se pueden imitar, como el método de doblado de láminas de aluminio, o el proceso de generación de adoquines de cemento, una embotelladora, el control de temperatura de un tanque, etc. Si no encuentra un proyecto que satisfaga sus necesidades puede crearlo. Recuerde que el desarrollo de estos procesos requiere de la aplicación de muchas de las funciones y métodos estudiados durante las prácticas.

3. DESARROLLO METODOLÓGICO

Nota: *para el desarrollo de esta actividad es necesario haber elaborado todas las prácticas.*

Parámetros a incluir en le desarrollo de la actividad:

- Para el desarrollo de esta práctica es importante que el proceso a desarrollar posea:

1) Un ciclo de trabajo, es decir, un diagrama de tiempos donde interactúen como mínimo tres actuadores (pistones, motores). El diagrama de tiempos de poseer más de cuatro etapas.

2) Debe tener un seleccionador de ciclo Manual y Automático (único y contínuo).

3) Debe poseer un panel de luces indicadoras que especifiquen el modo de trabajo del proceso y la fase en la que se encuentra (en el caso de que el proceso conste de varias fases).

4) Debe tener un botón de Paro de emergencia y un Reset.

5) Finalmente, debe incluir mínimo tres de las funciones desarrolladas a lo largo de la actividad.

6) La inclusión del módulo de entradas y salidas análogas es opcional.

7) En ningún momento se puede usar el módulo de simulación de entradas y salidas digitales.

8) El proyecto se debe construir sobre un ambiente físico que simule el entorno.

Anexo al programa, se debe incluir un trabajo escrito donde se describa el proceso y se incluya programa y la metodología aplicada para el desarrollo de este. Sí como diagramas o fotografías del prototipo.

Éxitos en su proyecto.

CAPÍTULO 2
NATIONAL INSTRUMENTS LABVIEW

- Introducción al entorno LabVIEW
- Creación y depuración de un VI
- Estructuras
- Arrays
- Creación de Sub-VI's
- Gráficas

PRÁCTICA 12
INTRODUCCIÓN AL ENTORNO LABVIEW

PRÁCTICA 12
INTRODUCCIÓN AL ENTORNO LABVIEW

1. OBJETIVO

Familiarizar al estudiante con el entorno de programación de LabView.

2. CONCEPTOS FUNDAMENTALES

Los sistemas SCADA (Supervisory Control And Data Aquisition, control y adquisición de datos de supervisión) están dentro del grupo de los elementos de control, proporcionando no sólo control y supervisión de un gran número de procesos (los cuales pueden estar a grandes distancias entre sí), sino que también presentan información relacionada de forma amigable, tal que un usuario la pueda entender.

Los primeros sistemas SCADA fueron sistemas que se modificaban según su propósito, los fabricantes atendían las peticiones de los compradores que requerían de un sistema de control para su industria, de acuerdo a estas pautas, modificaban un SCADA antiguo y le agregaban los nuevos requerimientos, en algunos casos omitían funciones que no se requerían y vendía el nuevo producto al fabricante. Estos eran sistemas que generaban reportes periódicos a partir de señales que representaban medidas y condiciones de estado, muchas de estas provenientes de lugares remotos, el monitoreo y control eran muy simples, la muestra de los datos se daba con contadores y lámparas, tiempo después los computadores se encargaron de recolectar los datos, y también de presentarlos sobre una pantalla, realizando actualmente, funciones de control mucho más complejas.

Los actuales proveedores de sistemas SCADA crean diferentes módulos, para procesos específicos, se puede encontrar entonces, módulos para procesamiento de papel y celulosa, industria de aceite y gas, hidroeléctricas, plantas de agua, control de fluidos, etc. Además, estos sistemas tienen la capacidad de integrarse a la parte gerencial del proceso, suministrando información que es necesaria a la hora de tomar decisiones económicas cotidianas.

Los sistemas SCADA agrupan puntos de un proceso, los cuales proporcionan información tanto analógica como digital o de estado (por ejemplo, abierto o cerrado). A su vez pueden enviar datos y controlar el secuenciamiento de las operaciones, aplicando las reglas de funcionamiento que el proceso requiere.

Los datos de lectura, pueden ser de tres tipos, digitales (on/off) asociados a alarmas, análogos representados a través de gráficos, y de pulso (conteo de revoluciones de un medidor). La muestra de estos datos se hace a través de una pantalla, generalmente, en una representación gráfica de la planta o proceso a controlar, los datos que cambian se muestran como dibujos de primer plano denominados "foreground ", y la planta es un fondo estático (background), el "foreground " es actualizado a medida que los datos de lectura van cambiando, otros valores de tipo analógico representan sus cambios a través de una gráfica. Estos datos pueden ser mostrados continuamente, o por disposición del operario, cuando este considere relevante revisarlo.

3. ACTIVIDAD PREVIA

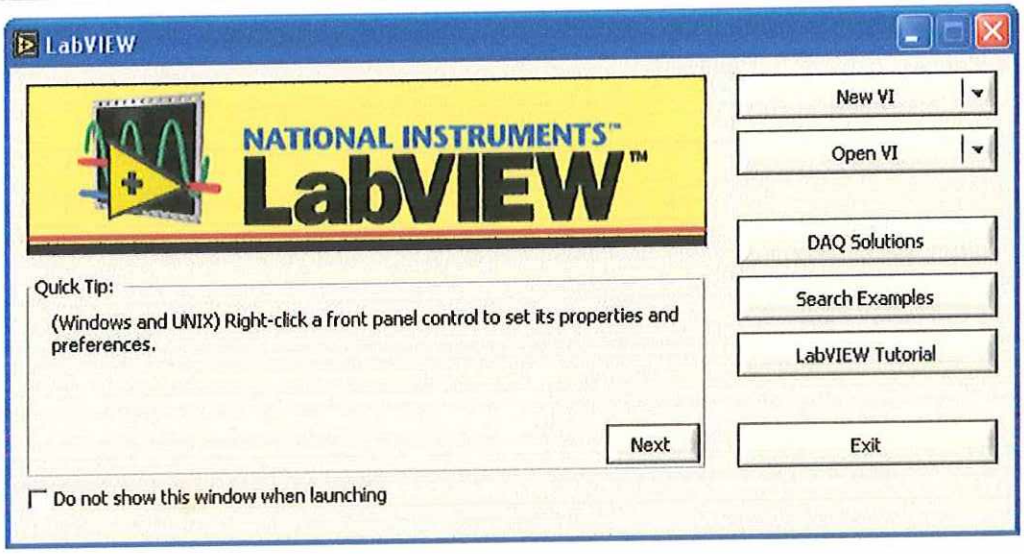
Antes de proceder al desarrollo de esta práctica es necesario que elabore el siguiente cuestionario:

1) ¿Porqué LabVIEW se considera como un sistema SCADA?
2) ¿Existen otros sistemas SCADA ?. Nómbrelos, detalle fabricante y bondades del sistema.
3) ¿Con que hardware (dícese de sistemas de adquisición de datos) es compatible LabVIEW?

4. DESARROLLO METODOLÓGICO

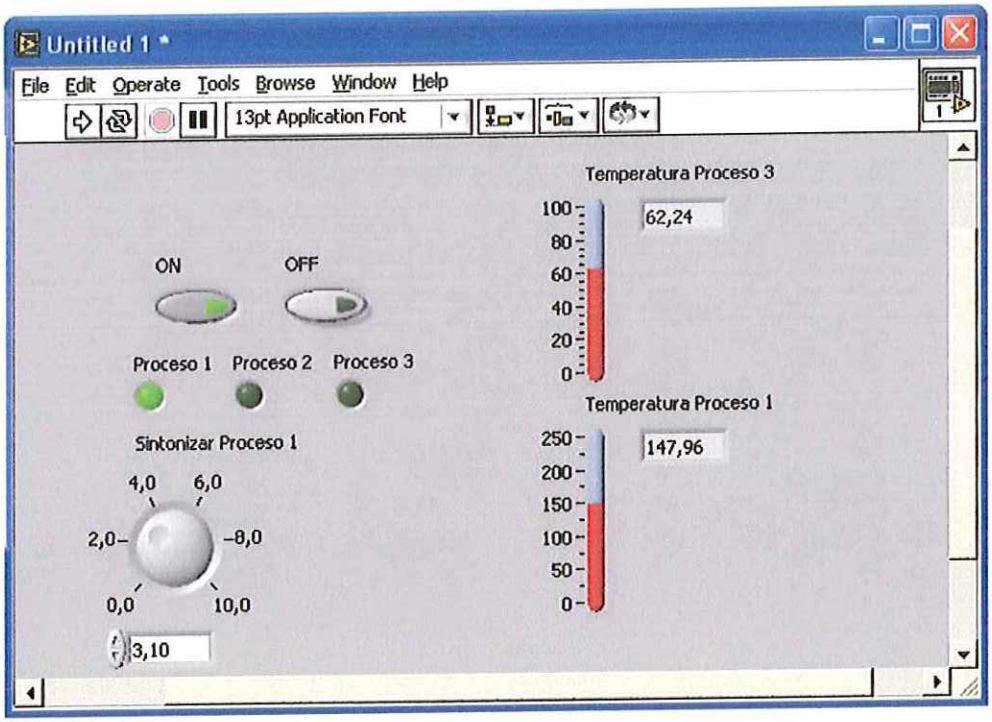
Abrir ***National Instruments LabVIEW,*** desplegará entonces una ventana de selección.

Nota: *este compendio de prácticas se desarrolló a través de* ***National Instruments LabVIEW versión 6i.***



De clic sobre ***New VI (Virtual Instrument)***, con esto se abrirá el entorno de programación LabVIEW el cual consiste en dos ventanas, una de fondo gris denominada ***Panel*** y otra de fondo blanco llamada ***Diagram***.

- **PANEL**

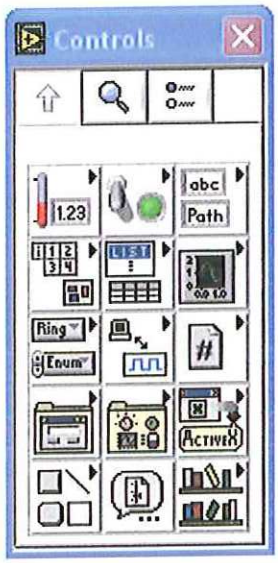


La ventana Panel suministra la parte visual del proceso, los indicadores, interruptores, gráficas y sintonizadores. El usuario puede interactuar con estos objetos, y a partir de la información que suministran, toma de decisiones sobre las actividades del sistema que se está controlando.

Como todas las ventanas, Panel ofrece un Menú de opciones, ***File,*** suministra las opciones básicas como Nuevo programa (New VI), guardar (Save), Exit (Salir). ***Edit***, ofrece las herramientas para la edición de los programas, tales como cortar (Cut), Pegar (Paste), Copiar (Copy). ***Operate***, tiene las funciones para poner en marcha el sistema creado. ***Tools***, tiene un conjunto de opciones para la parte de interactividad del programa con elementos de adquisición de datos tales como sensores. ***Browse***, ejecuta opciones de búsqueda sobre el programa que esta trabajando. ***Window***, despliega paneles muy importantes para el trabajo sobre los VI. ***Help***, tiene todas las opciones de ayuda de LabVIEW, como también un acceso a artículos del tutorial del sistema.

Todas las imágenes que representan los interruptores y diagramas se pueden obtener de un submenú denominado ***Controls Palette,*** para visualizarlo en el menú ***Window***, seleccione la opción ***Show Controls Palette***.

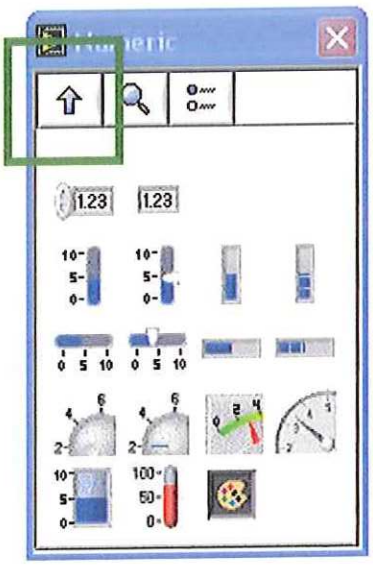
Aparece una pequeña ventana ***Controls*** sobre el entorno de trabajo.



En esta ventana están agrupados por tipo todos los controles y visualizadores del sistema. Con el Mouse se puede desplazar sobre cada una de las casillas para visualizar su nombre y de ahí prever las herramientas que tiene disponibles en su contenido.

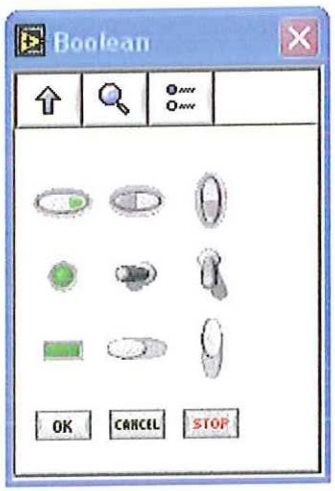
En orden de arriba a abajo y de izquierda a derecha algunas de las casillas son:

a) **Numeric**: todos los indicadores y controles de tipo numérico se agrupan en esta casilla.

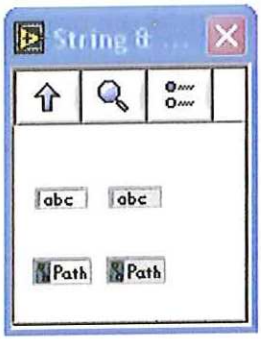


***Nota**: para volver al menú de casillas de clic sobre el **icono en forma de flecha** de la parte superior de la ventana.*

b) **Boolean**: las variables de tipo bolean (1,0) son dispuestas al programa a través de esta casilla. Vea la diversas interpretaciones de interruptores y luces indicadoras.

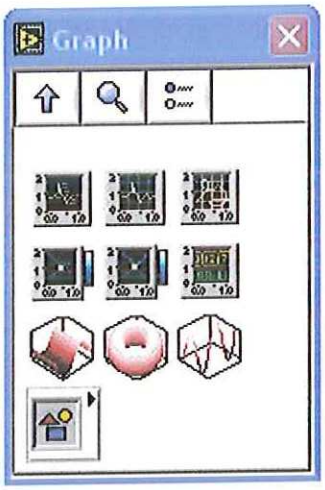


c) **String & Path:** los indicadores y entradas para cadena de caracteres están comprendidos dentro de esta casilla.

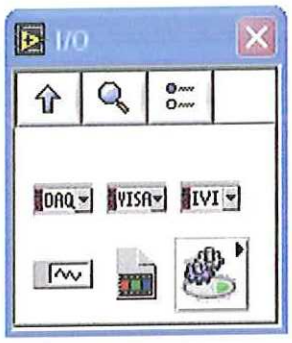


d) **Arrays & Clusters:** las variables de tipo array son dispuestas en esta sección.

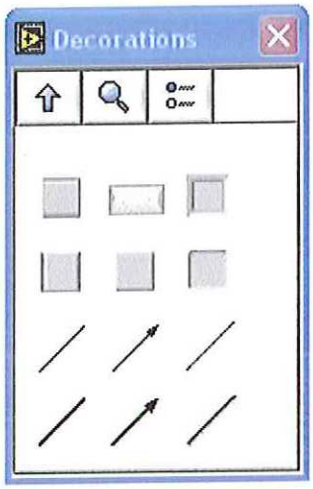
e) **Graph**: en esta opción se visualizan los diferentes tipos de gráficas que se pueden trazar a través de LabVIEW, las gráficas son uno de los elementos más indispensables para conocer el estado de un proceso.



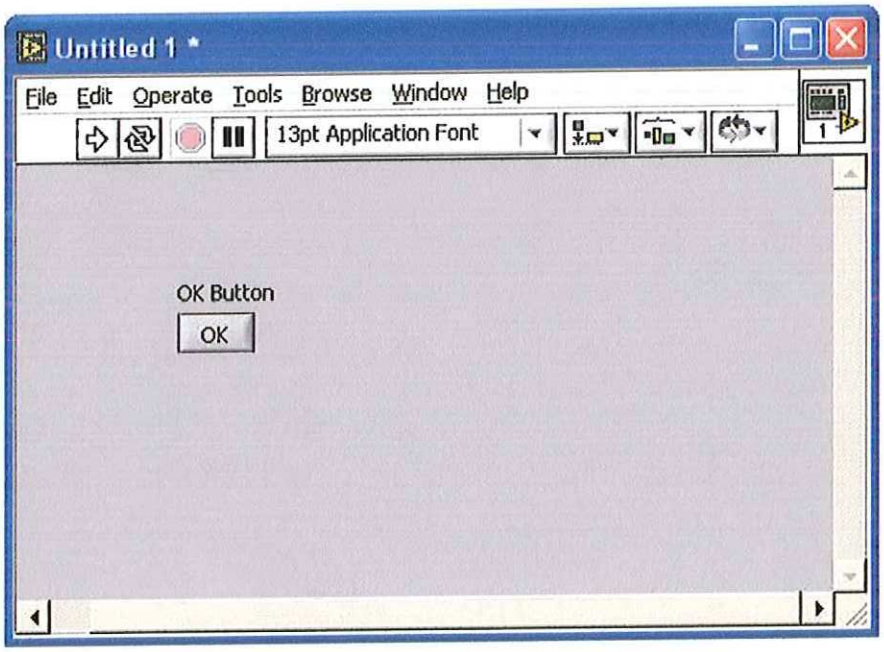
f) **I/O**: interfaz hacia los elementos de adquisición de datos y los actuadores conectados al sistema.



g) **Decorations**: agrega imágenes visuales para decorar las ventanas y el entorno de visualización del sistema de control. No generan ningún efecto sobre el sistema de control.



Insertar y editar objetos en la ventana Panel.



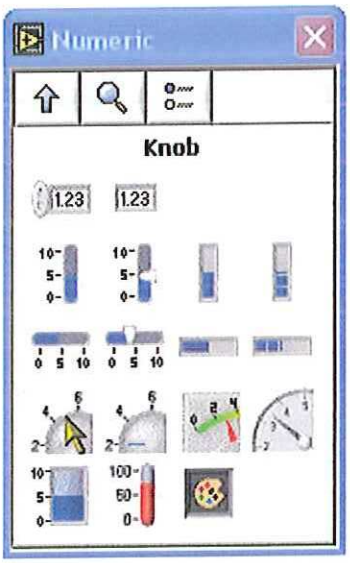
Como se había mencionado anteriormente Panel genera la parte visual de un programa en LabVIEW, su función es la de suministrar al usuario los diferentes objetos que se consideran indispensables para el desarrollo de un entorno amigable y funcional.

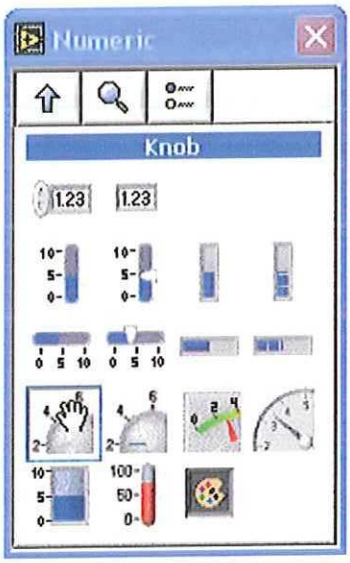
Los objetos pueden ser editados y modificados completamente a gusto del usuario, para esto Panel dispone de un submenú donde están almacenadas las herramientas necesarias. Para visualizar este sub-menú en ***Window*** seleccione ***Show Tools Palette.***

Esto desplegará una ventana sobre el entorno de trabajo llamada ***Tools***.

Ahora se insertará un objeto y a través de la ventana Tools será editado.

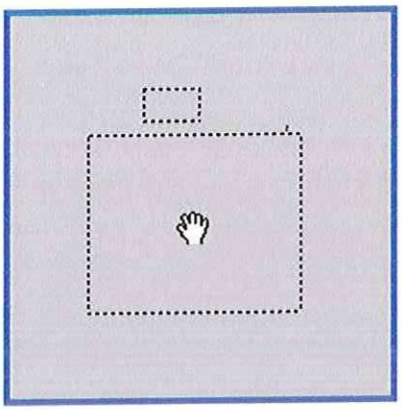
En la ventana o submenú ***Controls,*** de clic sobre la casilla ***Numeric***, seleccione el objeto denominado k*nob*, para seleccionarlo basta dar un clic sobre él, el Mouse cambiará su forma de puntero tradicional al de una mano.

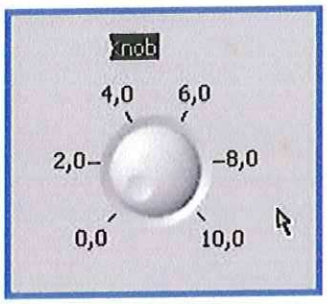




Ahora desplácese hacia el entorno de trabajo de color gris, observará que el puntero del Mouse esta rodeado por un cuadro de contorno punteado, de clic sobre el área. El objeto aparecerá.

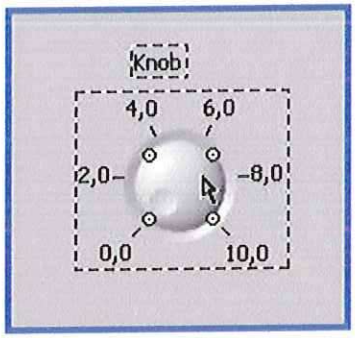
Tal y como se ve el objeto, este toma el tamaño, color y nombre predefinido por el sistema. En algunos caso como este, el rango de selección esta también predeterminado.





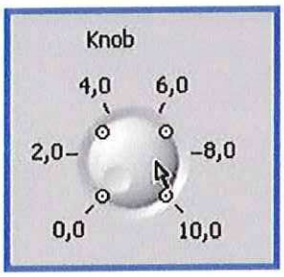
El Mouse cambiará su forma inmediatamente a un puntero de color gris. Este puntero lleva el nombre de *Postion/Size/Selected* que de acuerdo a LabVIEW son sus funciones.

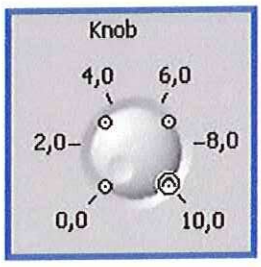
Cuando el puntero toma esta forma puede seleccionar un objeto, para esto basta con dar clic sobre él.

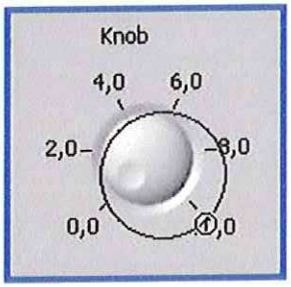


Una cuadrícula intermitente rodea al objeto, **con clic sostenido puede transportarlo hacia cualquier punto del entorno gris de Panel**.

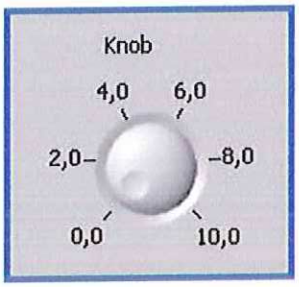
Para cambiar el tamaño de un objeto, ubique el puntero sobre él y espere a que aparezcan cuatro nodos de selección, ubíquese sobre uno de estos nodos y verá que el puntero vuelve a cambiar su forma. Ahora con clic sostenido arrastre el nodo para modificar las dimensiones del objeto.

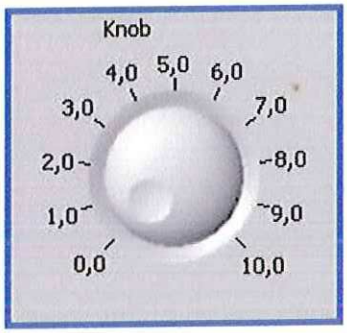




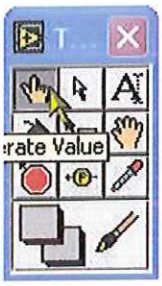


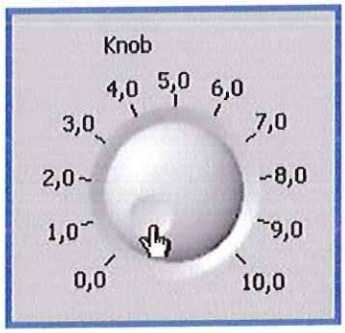
Verá que no sólo se modificó el tamaño del objeto, sino que también se amplió el número de posiciones dentro del mismo rango.

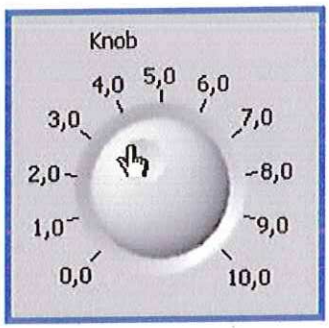




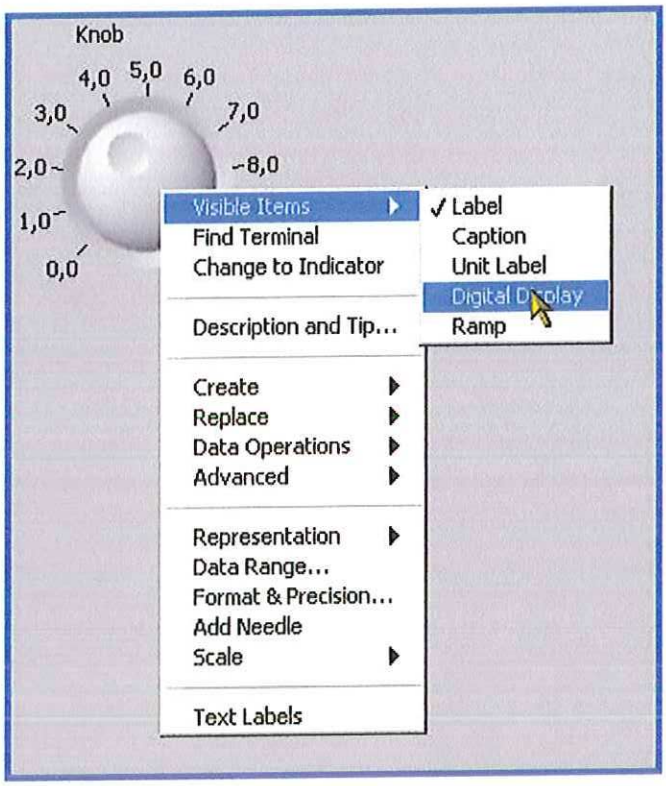
Para poder desplazar la perilla basta con dar clic sobre el icono en forma de mano con el dedo índice levantado (***Operate Value***) para transformar el cursor en una figura similar. Ahora desplace el Mouse hacia el objeto y ubíquese sobre la perilla, con clic sostenido arrastre el indicador hacia cualquier otro número.



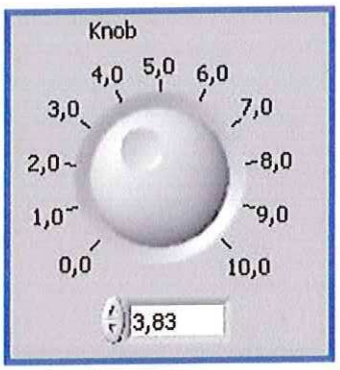




Es importante entender que el objeto insertado al sistema no es tan solo eso, es también una variable de tipo numérico cuyo valor cambia de acuerdo a las opciones del usuario. En el estado en que se encuentra, es muy difícil saber cual es el valor exacto que se da cuando se desplaza la perilla del knob, **para conocer el valor específico en el que se encuentra la variable**, es necesario agregar un visor o display que muestra de forma exacta el valor del objeto. Para esto basta con dar clic derecho sobre el knob, seleccionar ***Visible Item*** y luego ***Digital Display***.

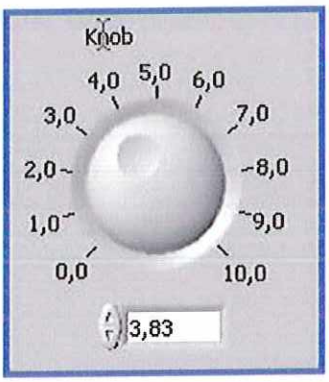


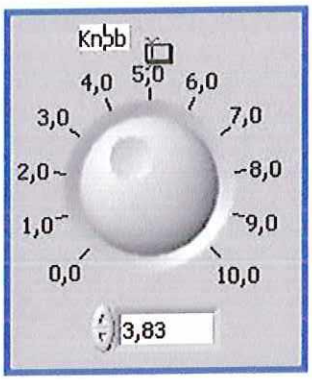
Observará que se ubica un recuadro con el valor del objeto al lado de él. Ahora desplace de nuevo la perilla del knob y visualice su estado de forma exacta con el display.



Supongamos ahora que el knob es uno de varios sintonizadores que tiene un sistema. Es necesario etiquetarlo para poderlo diferenciar de los otros y de esta forma evitar errores de operación. Para esto el submenú ***Tools***, seleccione la opción ***Edit Text***. El Mouse volverá a cambiar su forma.

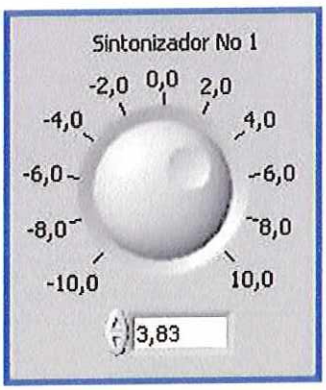
Ahora desplace este Mouse sobre el texto denominado knob y de clic sobre él, el fondo del texto cambiará a un color blanco, ahora con el teclado modifique el texto.





Para aprobar el texto escrito basta dar clic sobre el espacio en gris que rodea al seleccionador, o dar clic sobre el icono ***Enter Text*** (el que tiene forma de visto bueno) que esta en la parte superior del la ventana Panel.

Con Edit Text se pude también modificar el rango de selección del sintonizador, de igual forma que se hizo con el texto de encabezado del objeto, de clic sobre el límite inferior (0,0) y escriba –10.



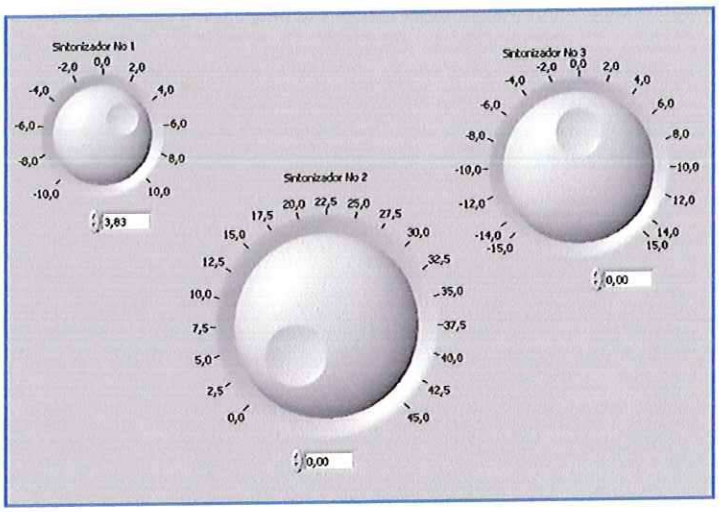
Obsevará que el rango de selección del Sintonizador ha cambiado. También puede modificar el rango superior con el mismo método.

Actividad: inserte dos nuevos knob al programa y modifíquelos para que cumplan los parámetros que se especifican en la siguiente tabla .

	Título	Rango
Knob 2	Sintonizador No 2	[0..45]
Knob 3	Sintonizador No 3	[-15...15]

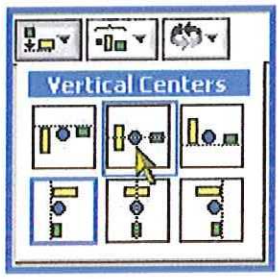
***Nota**: Modifique el tamaño de los selectores par que se pueda apreciar los números contenidos dentro del rango. No olvide insertar un display.*

Ahora se procederá a ordenar los objetos creados.

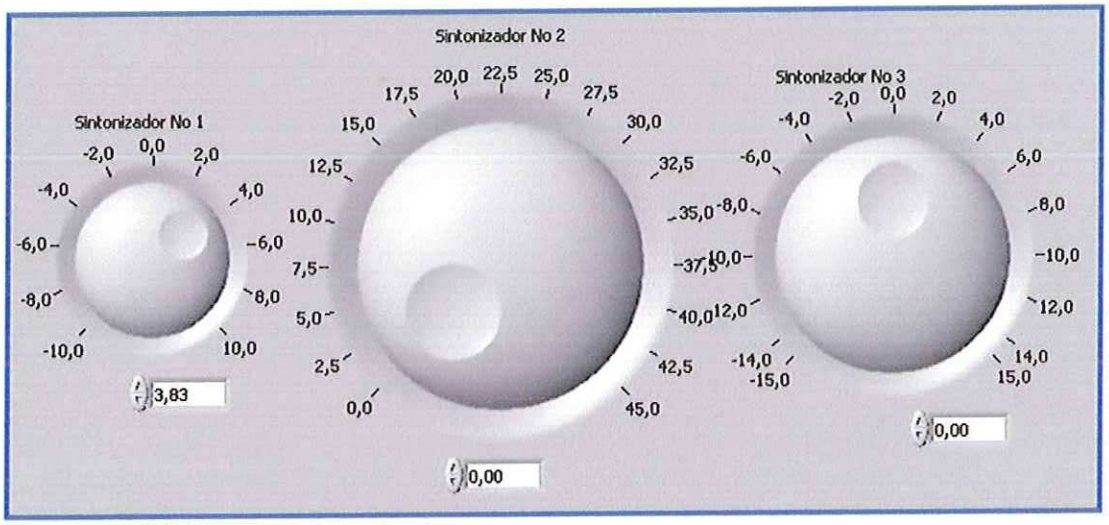


Se pueden **reordenar los objetos** insertado dentro de Panel para que tengan una apariencia ordenada. Lo primero, es alinearlos a un mismo margen horizontal, para hacer esto se recurre al submenú ***Align Objects*** ubicado en la parte superior de la ventana Panel.

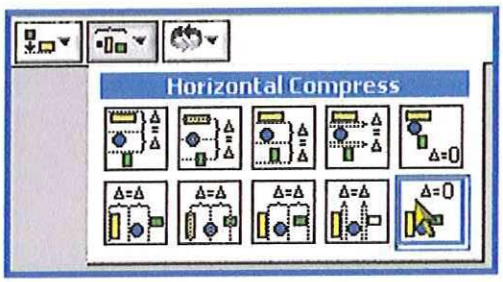
Con Shift sostenido seleccione los tres objetos, luego de clic en Align Objects, se despliegan un número de opciones, tome nota de las gráficas que tienen para identificar su funcionalidad. Ahora seleccione ***Vertical Centers.***



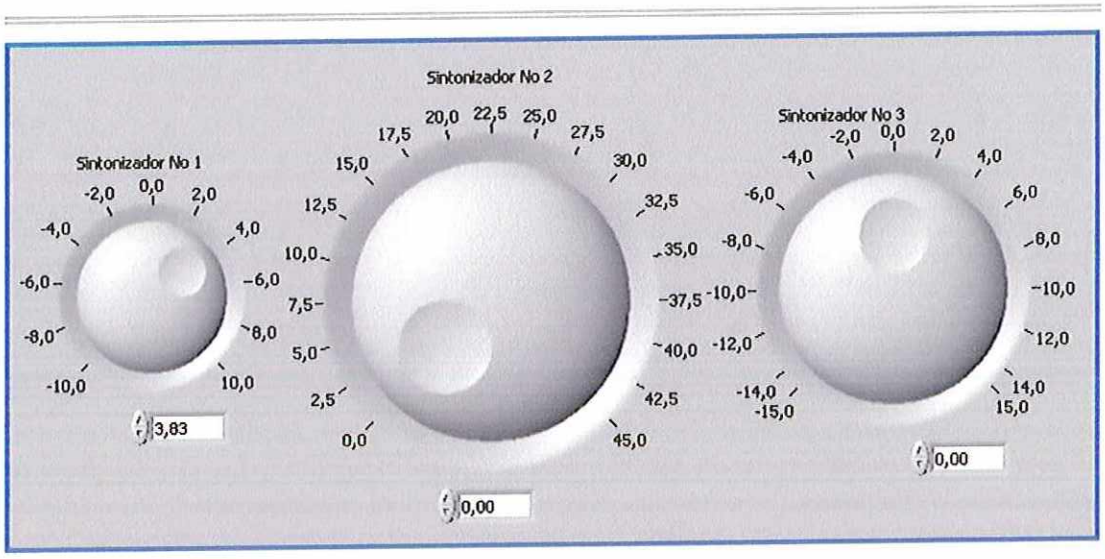
Los objetos seleccionados se alinean en el centro.



Ahora es indispensable separarlos entre sí, para esto selecciónelos nuevamente y seleccione el submenú ***Distribuye Objects***, ubicado justo al lado de Align Objects, ahora elija la opción **Horizontal Compress.**



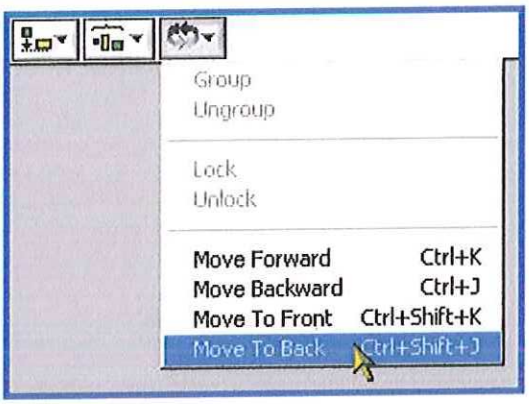
Los objetos se separarán entre sí justo al margen.



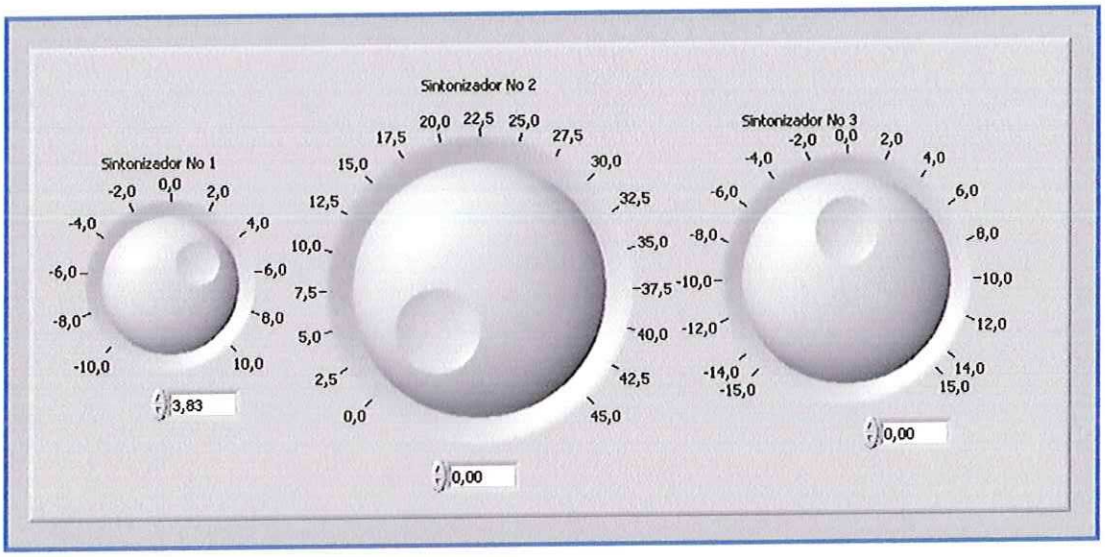
Ahora se insertará al Panel una cuadrícula con relieve para separar los sintonizadores del espacio gris. Para esto en el submenú Controls, seleccione la casilla Decorations, ahora seleccione ***Raised Box,*** e insértelo al programa.

Un recuadro con relieve es agregado.

Con el Mouse convertido en puntero, modifique el recuadro para que cubra a los tres sintonizadores. Ahora estos están ubicados justo debajo de él. Para desplazarlo hacia el fondo, selecciónelo y busque el submenú ***Reorder***, ahora seleccione la opción ***Move to Back***.



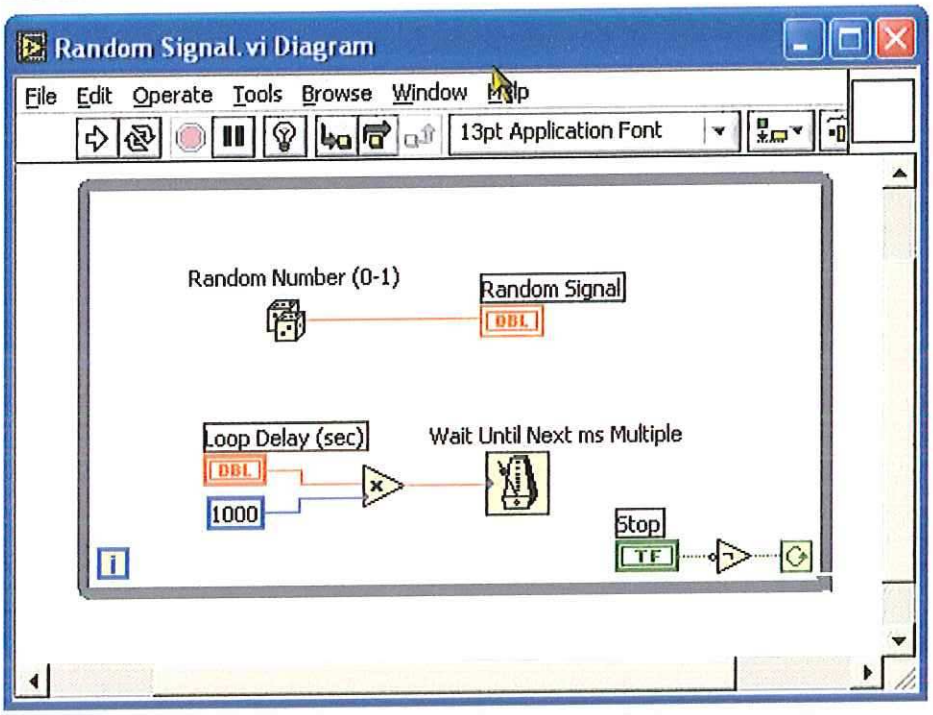
El control del sistema quedará como se muestra en la siguiente figura.



Lo anterior es tan sólo un esbozo de las opciones que la ventana Panel puede ofrecer, ahora se cambiará de ambiente para proceder a la segunda parte de esta práctica.

- **DIAGRAM**

Mientras la ventana Panel se encarga de crear el entorno para el usuario, en ***Diagram*** se desarrolla el medio de programación con el cual se dispone a procesar toda la información proveniente de Panel. Al final, Panel exhibirá los resultados del proceso de datos a través de Diagram.

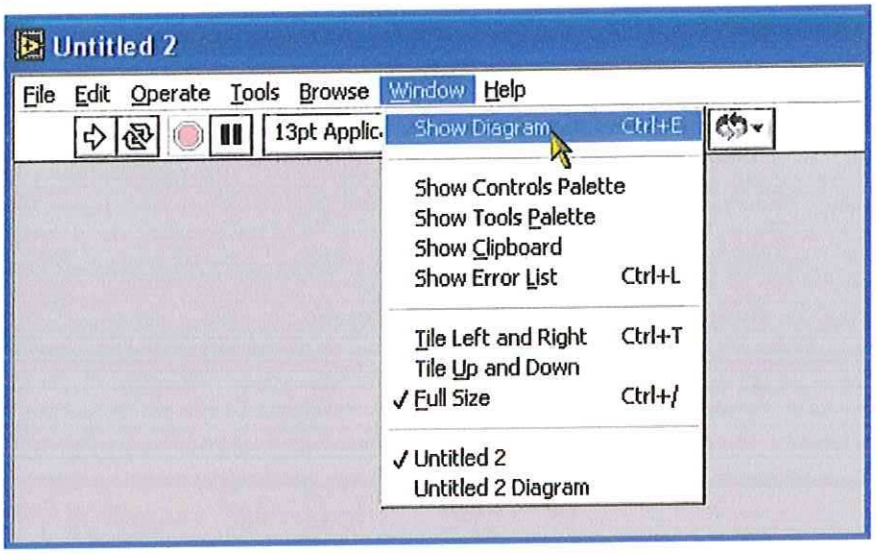


Este entorno de programación es enteramente de tipo gráfico, las condiciones, variables, entradas y salidas, todo es representado mediante imágenes interconectadas entre sí. Para aquellos estudiantes que tienen habilidades para la programación en lenguajes como C o Pascal, verán en LabVIEW como una nueva alternativa para desarrollar programas ya tengan estos o no interactividad con el medio.

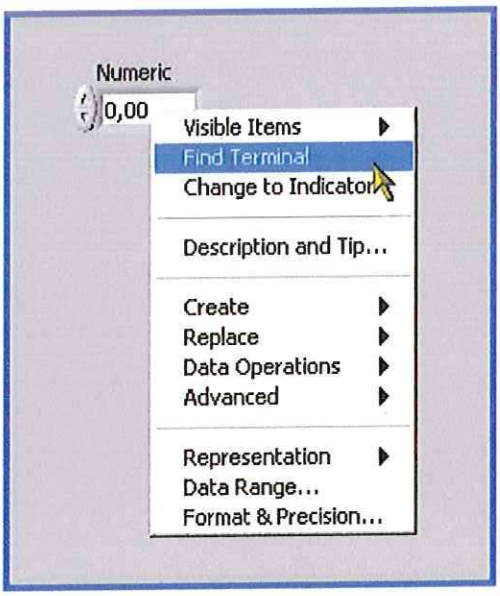
Se puede acceder a Diagram de cinco formas:

- Desde la ventana de Windows seleccionando la ventana con el nombre del proyecto + Diagram.

- Desde la ventana Panel seleccionando el menú Window, ***Show Diagram.*** *Si se está en Diagram se selecciona* ***Show Panel*** *para volver a Panel.*

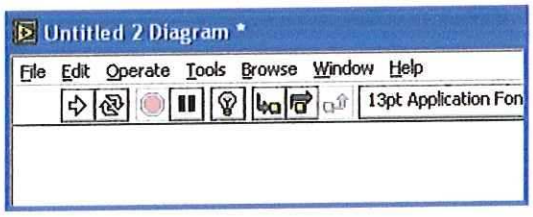


- Desde Panel, dando la combinación de teclas CTRL+E.
- Desde Panel, dando doble clic sobre cualquier elemento insertado.
- Desde Panel, con clic derecho sobre cualquier objeto insertado y seleccionando en el menú emergente la opción ***Find Terminal.*** *Si se está en Diagram se selecciona* ***Find Control*** *para volver a Panel.*



Nota*: estas opciones también son válidas para regresar de Diagram a Panel.*

Diagram también cuenta con opciones de menú, las cuales son exactamente las mismas que fueron descritas en Panel.



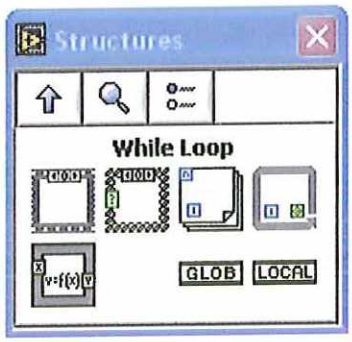
Las gráficas utilizadas para desarrollar los programas se obtienen del submenú ***Functions***, para activarlo en el menú Window seleccione ***Show Functions Palette .***

Una ventana aparece sobre el entorno de trabajo.

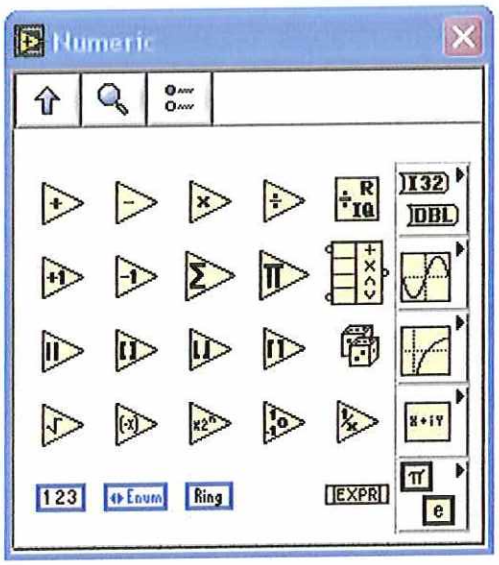
Todas las acciones para modificar y controlar una variable, para crear espacios de almacenamiento y para la toma de decisiones están comprendidas dentro de esta ventana.

En orden de arriba a abajo y de izquierda a derecha algunas de las casillas son:

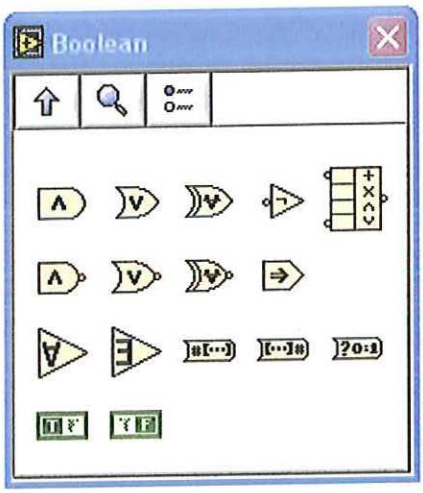
a) **Structures**: las funciones básicas para la toma de decisiones y la operación de los programas están acá, estructuras como for o while, qu existen en cualquier lenguaje de programación son representadas de forma gráfica.



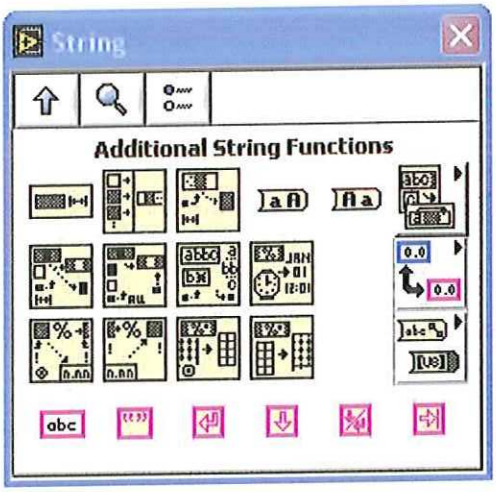
b) **Numeric**: todo el tratamiento matemático que se puede realizar sobre un término esta en esta casilla, aquí se encuentran operaciones matemáticas básicas, identidades trigonométricas, conversiones y constantes, entre otras.



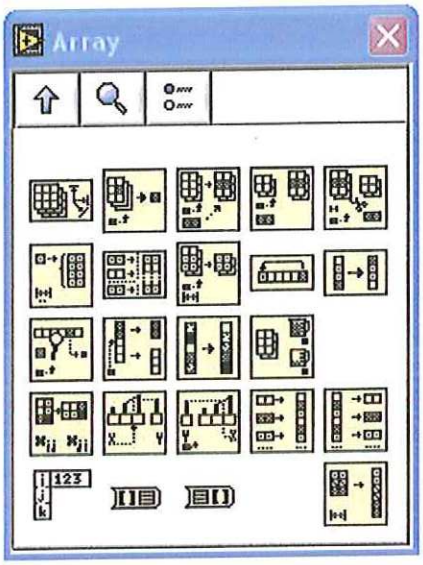
c) **Boolean**: aquí esta resguardado todo el tratamiento matemático en el entorno digital (1 y 0), como las operaciones booleanas OR, XOR, AND, etc.



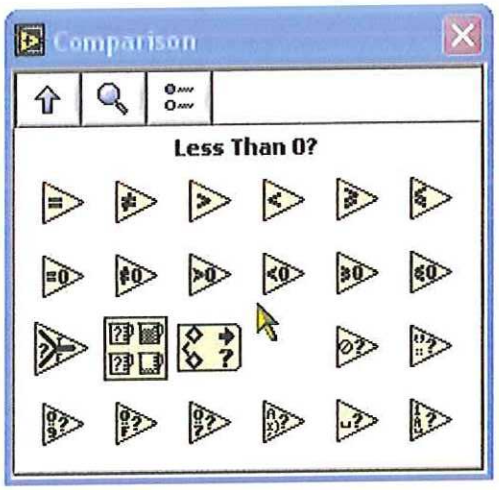
d) **String**: tratamiento de variables tipo cadena de caracteres.



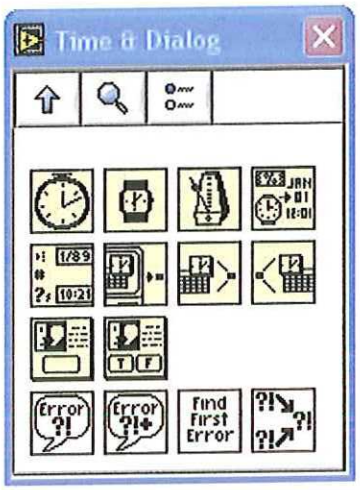
e) **Array**: tratamiento de variables de más de una dimensión como vectores y matrices. Contiene funciones para insertar y borrar elementos de un vector o matriz, y rotar los elementos que la componen.



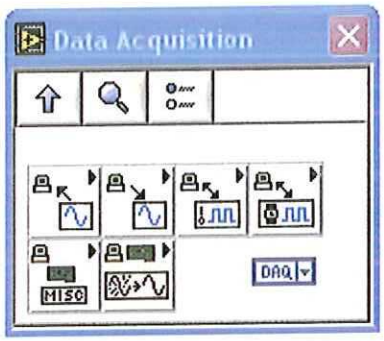
f) **Comparision**: en esta casilla están contenidos los módulos de comparación de variables y elementos.



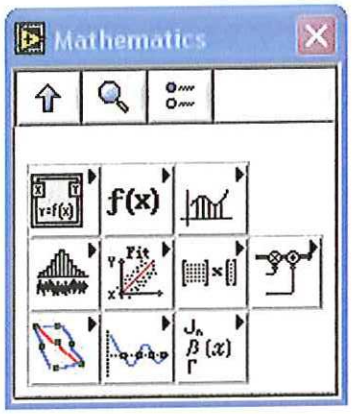
g) **Time & dialog**: los retardos, tratamiento del tiempo en general como obtener el tiempo están dentro de esta casilla. Además hay opciones de diálogo para informar de la aparición de errores que podrían acontecer durante el paso de un programa.



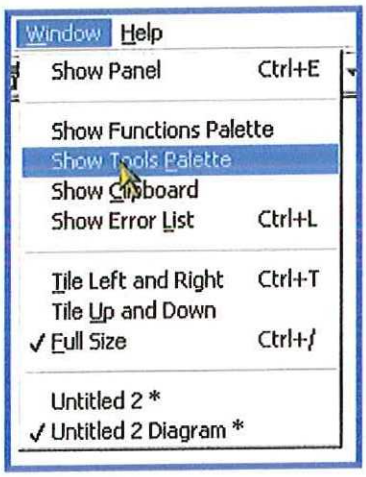
h) **Data acquisition**: adquisición de datos provenientes de sensores u otros sistemas.



i) **Mathematics**: los tratamientos matemáticos más complejos fueron traducidos en pequeñas funciones, análisis de probabilidad, encontrar las raíces de una función son realizados automáticamente.



Los elementos aquí descritos se insertan al espacio de trabajo de la misma forma que en Panel. La edición de los mismos se hace a través del submenú ***Tools***, para activarlo es necesario ir al menú Window y seleccionar ***Show Tools Palette.***



Aparece la siguiente ventana emergente sobre el área de trabajo:

Note que es exactamente la misma que se utiliza en Panel, es más, muchas de las herramientas insertadas en Panel se repiten y tienen la misma función en ***Diagram***. Esto facilita mucho la comprensión y el desempeño en el trabajo de LabVIEW, en especial cuando los programas se vuelven tediosos.

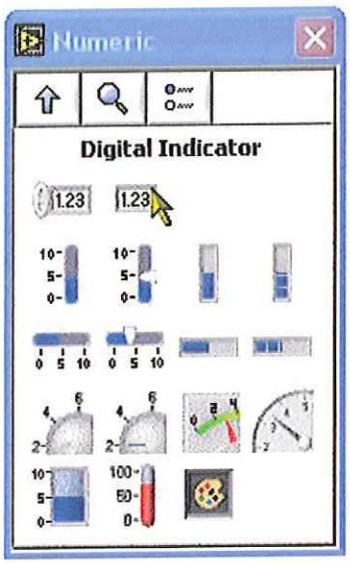
Cuando se inserta un objeto en Panel, en Diagram se visualiza una imagen que lo representa, esto es lógico, porque el interés de Diagram es recibir los datos para poderlos procesar y arrojar una respuesta. Por ejemplo el sintonizador que fue insertado en la actividad pasada tiene las siguientes apariencias:

Sintonizador Visto desde Panel	Sintonizador Visto desde Diagram
	

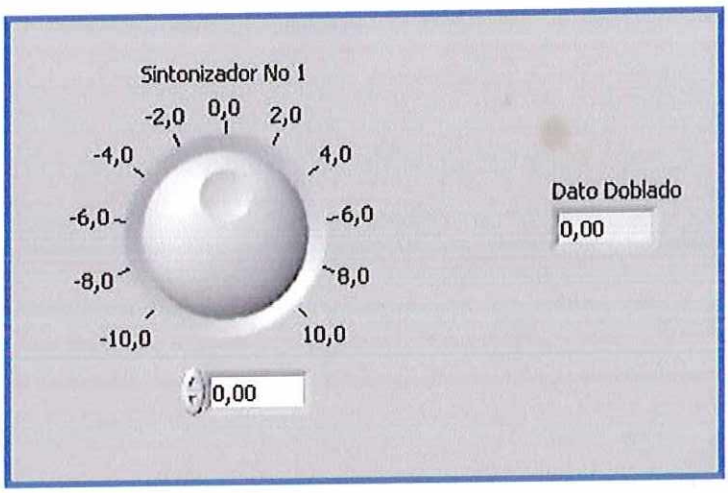
Diagram interpreta al sintonizador como una variable, sólo que esta es modificada por el usuario, el origen de los datos es transparente para Diagram, el interés en esta ventana es del procesamiento y generación de una señal.

Por ejemplo, **se desea doblar la información suministrada por el sintonizador y generar esta respuesta en un visor**. Para poder hacer esto es necesario sincronizar el trabajo entre Diagram y Panel.

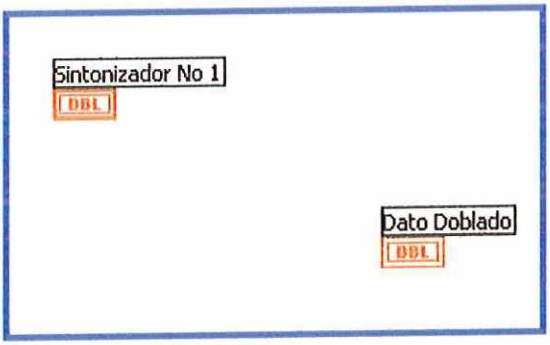
Primero que todo, se desea observar la repuesta en un visor. Para esto, es necesario insertar uno desde la ventana Panel. Para esto, en el submenú Controls, en la casilla Numeric seleccione e inserte al espacio de trabajo el objeto ***Digital Indicador***.



Modifique el texto que lo identifica por un título que lo identifique, por ejemplo *dato doblado.*

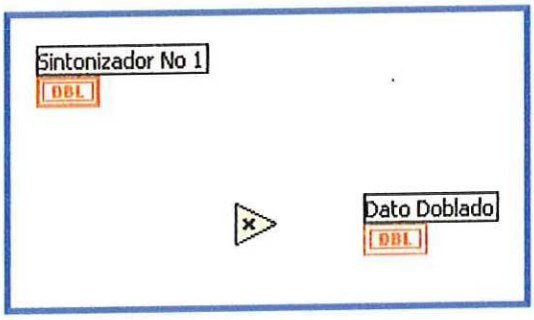


Ahora es necesario conectar estos dos elementos, para esto ingrese a la ventana Diagram.



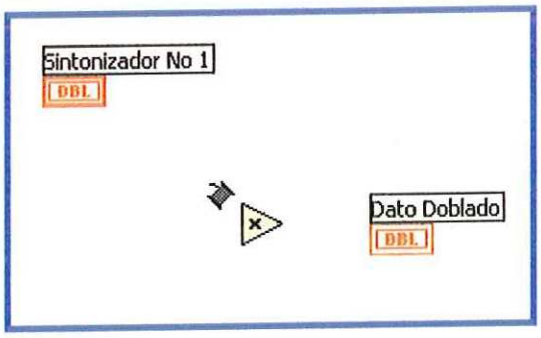
Aquí se puede diferenciar totalmente un objeto del otro, gracias al nombre que se les dio desde Panel, note además que el objeto *Sintonizador No 1* tiene un marco mas grueso que el objeto *Dato Doblado,* esto se debe a que el primero se trata de una variable tipo entrada, mientras que el segundo es una salida sobre la cual no se pueden realizar cambios a la fuerza, sólo a través de algún proceso

Se pide doblar el valor del sintonizador y visualizar el resultado en el visor, para poder hacer esto en Diagram, en el submenú Functions seleccione la casilla Numeric, luego inserte el objeto ***Multiply*** (multiplicar).

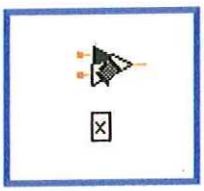


Multiply toma dos términos y los múltiplica, esta función consta de tres bahías o puertos de conexión, dos entradas y una salida. **Para enlazar los objetos con la función**, en le submenú Tools seleccione con el Mouse el icono ***Connect Wire***.

El puntero tomará la forma de un carrete de hilo.

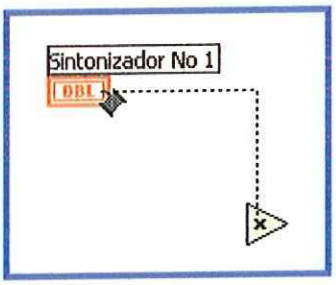


Ahora ubíquese justo encima de la función Multiply, note como aparecen los puertos de conexión sobre esta.

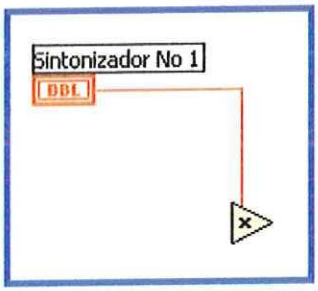


Podrá identificar los puertos por la aparición de un mensaje emergente de color amarillo sobre el área de trabajo, Multiply consta de dos entradas X y Y ubicadas en la parte posterior de la función, su salida X*Y esta ubicada en la parte delantera.

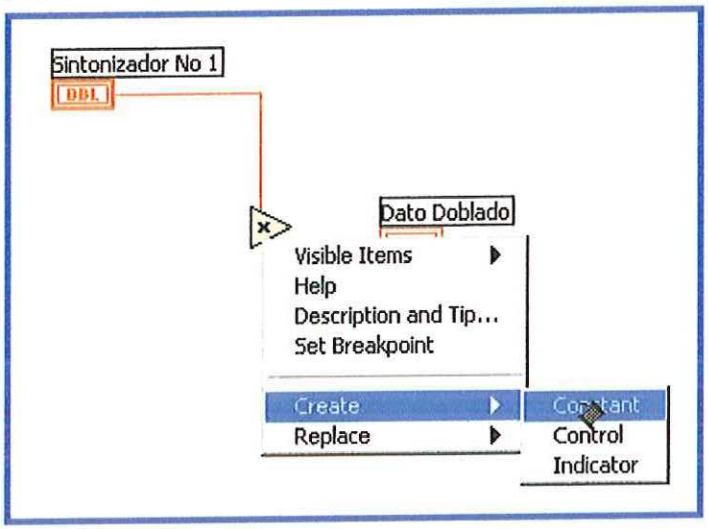
Ahora de clic sobre la casilla X, notará que por donde desplace el puntero le seguirá una línea intermitente. Busque el sintonizador y ubíquese sobre él, este objeto empezará a parpadear para indicar que la conexión entre los dos se realizará. Ahora de clic sobre él.



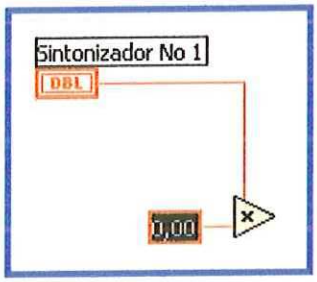
Si el objeto y la función están unidas mediante una línea de color, significa que la conexión es correcta.

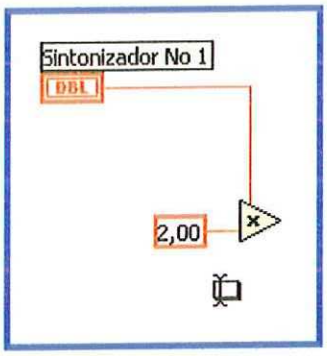


El valor dentro de sintonizador se debe multiplicar por 2, dando clic derecho sobre el puerto Y de la función Multiply aprece un menú emergente. Seleccione la opción ***Create*** y posteriormente ***Constant***.

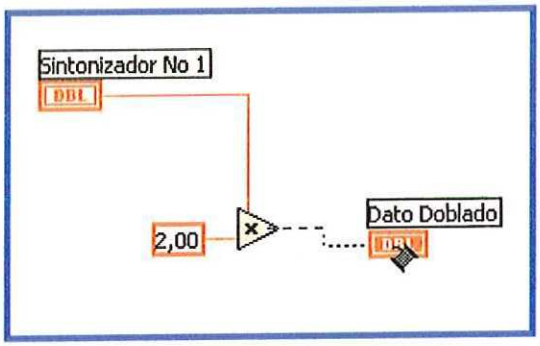


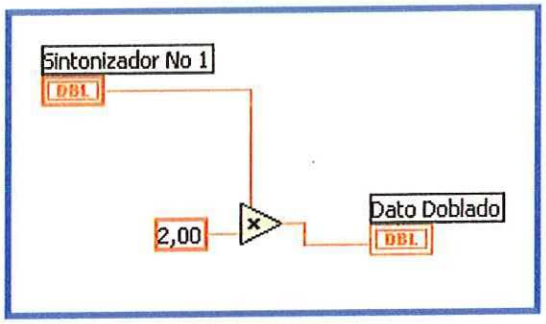
Un valor constante es ahora enlazado al puerto Y, para editarlo de clic en la opción ***Edit Text,*** del submenú Tools y modifique este valor por 2.



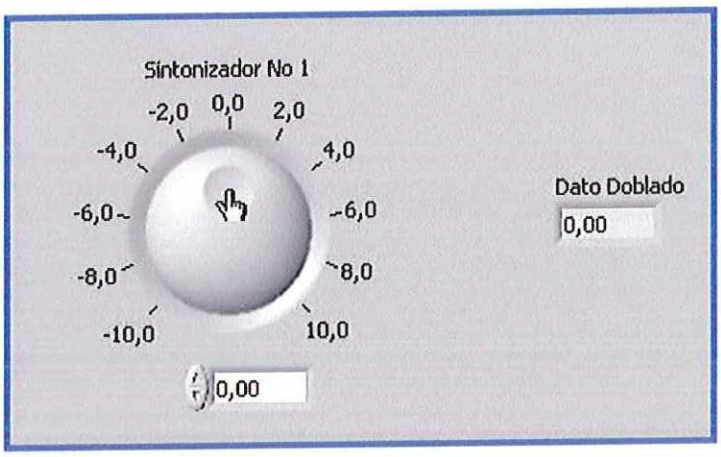


Ahora sólo basta conectar la salida de la función Multiply con el visor.



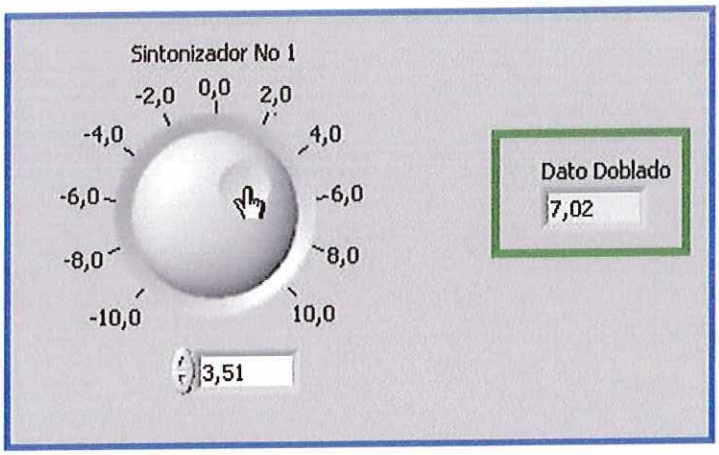


El programa ya esta listo basta sólo probarlo para verificar su correcto funcionamiento. Para esto vaya a la ventana Panel y con el cursor modifique la entrada del sintonizador aun valor diferente de 0.



Ahora de clic sobre el ícono ***Run*** ubicado en la parte superior de la ventana Panel.

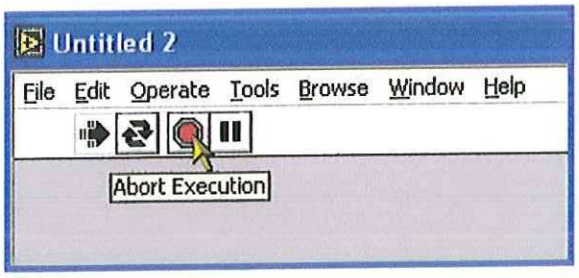
Observe el valor de dato doblado.



Cuando se da clic sobre Run, el proceso sólo se ejecuta una vez. Para ver un desempeño continuo de clic sobre el icono ***Run Continuosly***.

Ahora modifique el valor del sintonizador, verá como el dato del visor cambia a la par que se hace la sintonización.

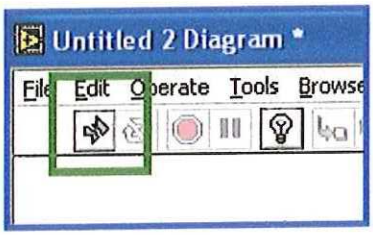
Para detener el proceso de clic en el icono Abort Execution.



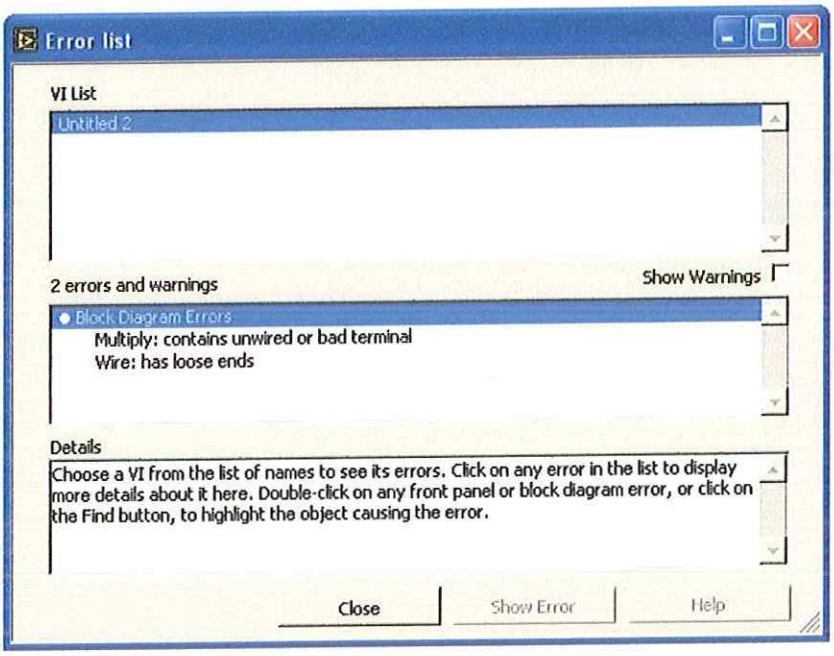
Actividad: modifique los otros dos selectores para que realicen las siguientes funciones

	Función
Sintonizador No 2	El valor obtenido de este se debe dividir por 10.
Sintonizador No 3	El valor obtenido de este se debe sumar con 50 y luego multiplicar por 3.

***Nota**:. No olvide insertar un display para visualizar la respuesta. Si se presenta algún error en la conexión de los elementos, la flecha Run tendrá una imagen como la que se muestra a continuación:*



Mientras el error no se haya corregido el programa no se ejecutará. Para identificar plenamente el error que se está generando, basta con dar clic sobre la flecha para desplegar la ventana ***Error List****.*



El error se describe en la ventana ***Details****. Si los errores son múltiples (lista de errores se almacena en la ventana del medio) puede seleccionarlos con el Mouse para conocer los detalles de cada uno.*

Para salir de la ventana de errores de clic sobre ***Close****.*

Una vez listo y comprobado el funcionamiento del programa cierre las ventanas ***Panel*** y ***Diagram*** (no guarde los cambios).

Con esto se finaliza al introducción al entorno de programación de LabVIEW.

PRÁCTICA 13
CREACIÓN Y DEPURACIÓN DE UN VI

PRÁCTICA 13
CREACIÓN Y DEPURACIÓN DE UN VI

1. OBJETIVOS

- ✓ Crear y desarrollar un VI (Virtual Instrument) en el entorno LabVIEW.
- ✓ Implementar los conceptos aprendidos en la solución de problemas a través de LabVIEW.

2. CONCEPTOS FUNDAMENTALES

En al práctica anterior se dio una breve referencia a los sistemas SCADA, describiendo sus ventajas y alternativas de control. Ahora nos referiremos sólo a uno de ellos, LabVIEW.

LabVIEW es un acróstico de *Laboratory Virtual Instrument Engineering Workbench,* su función es la de desarrollar programas de alto desempeño científico e ingenieril. Bajo su entorno se han desarrollado tesis de grado, maestría y doctorado, todos trabajos de alto desempeño, los cuales utilizan al máximo las herramientas disponibles por el programa.

Un VI es una herramienta práctica para la solución en el tratamiento de señales, como un buen SCADA, LabVIEW no sólo recibe información proveniente de los sistemas de adquisición de datos, es también, capaz de procesarla y de acuerdo a su valor tomar acciones de control, es decir tomar decisiones. En otros campos, puede ser tan sólo una herramienta para la visualización de datos de una planta a través de un PC, donde las acciones de control son desarrolladas por el usuario a través de un panel de decisiones que ofrece el programa.

Para el desarrollo de un VI se deben tener en cuenta las disposiciones interfaz (desarrollado por Panel), sistema de control (desarrollado por Diagram), muchas veces el ir y venir entre cada ventana genera cansancio y en otras, un mal desempeño de el software de control hace perder el trabajo de un buen diseño de interfaz con el usuario.

Mas sin embargo, LabVIEW es una gran herramienta, las opciones y el gran número de funciones permite que en su software se desarrolle cualquier tipo de proyecto, dado a esto existe un gran número de usuarios y actualmente (Febrero del 2005) ya va en su séptima edición, tiene en su haber más de 10 años de experiencia en el desarrollo de software para personal profesional, lo que lo convierte en una herramienta muy eficaz y confiable.

3. ACTIVIDAD PREVIA

Antes de proceder al desarrollo de esta práctica es necesario que elabore el siguiente cuestionario:

1) ¿Cuáles son los diferente entornos de programación existentes en LabVIEW para la elaboración de proyectos? Nómbrelos y de una explicación de cada uno.
2) National Instruments es la empresa desarrolladora del programa LabVIEW, y líder en la creación de equipos para empresas y universidades, investige y redacte una breve descripción acerca de esta empresa y de sus actividades.

4. DESARROLLO METODOLÓGICO

Nota: *para el desarrollo de esta actividad es necesario haber elaborado la Práctica 0 .*

Un VI es un programa estructurado que bajo un ambiente gráfico y de fácil entendimiento desarrolla una labor determinada. El programa no tiene estructura, es una secuencia de gráficos que trabajan en un orden determinado según la disposición del programador.

El VI se desarrolla entre los dos ambientes de programación, ***Panel***, que desarrolla toda la interfaz humano máquina y ***Diagram*** que desarrolla el algoritmo de trabajo. Sólo es posible desarrollar una metodología de programación a partir de la práctica constante, a medida que el usuario se familiariza y conoce mejor las herramientas y atajos que LabVIEW dispone, mejor será su desempeño y efectividad en la creación de programas.

- **Creación de un VI**

Se ha pedido la creación de un programa a través de LabVIEW el cual toma la temperatura de un indicador, y la convierta a grados Fahrenheit y Kelvin.

La temperatura inicialmente está en grados centígrados.

1) Primero es necesario tomar nota de los elementos que se requieren para desarrollar el programa. Para esto es importante identificar las entradas y las salidas.

Para el caso del problema propuesto:

2) Una vez identificado el programa y las variables que intervienen, se puede plantear la estrategia de solución.

Para convertir la temperatura de grados centígrados a Fahrenheit:

$$T°F = T°C * \frac{9}{5} + 32$$

Para convertir de grados centígrados a Kelvin:

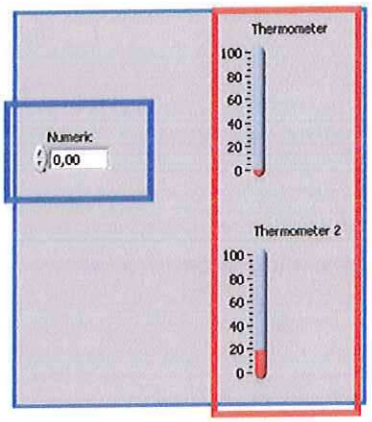
$$T°K = T°C + 273$$

LabVIEW ofrece las herramientas para realizar los cálculos matemáticos necesarios, el programa requiere de la utilización de las funciones de suma y multiplicación para poder realizar las conversiones.

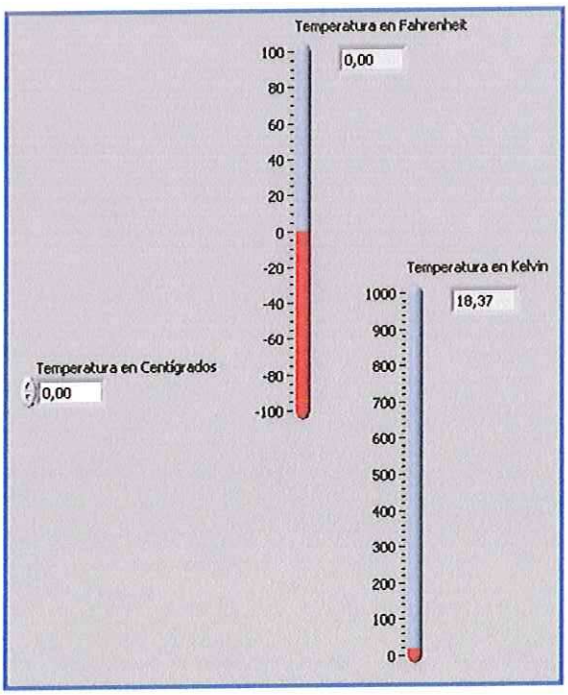
3) El programa consta de una entrada y dos salidas. En Panel están

disponibles los elementos necesarios.

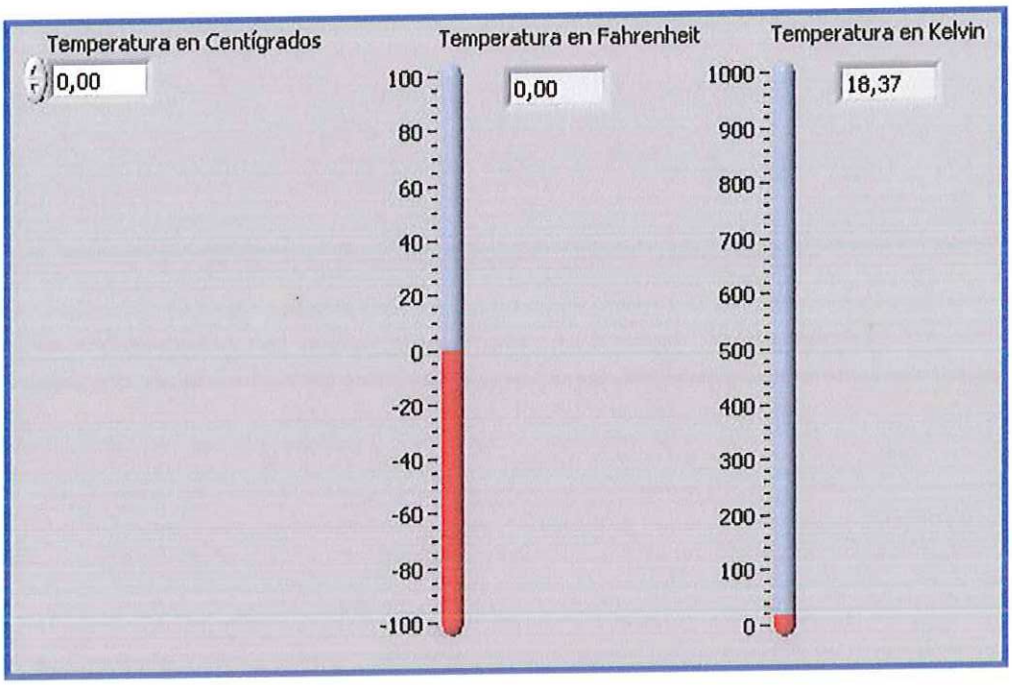
Desde Panel, con la casilla ***Numeric*** del submenú ***Controls*** inserte un ***Digital Control***, el cual será el elemento de entrada y dos termómetros (***Thermometer***), los cuales serán los elementos de salida.



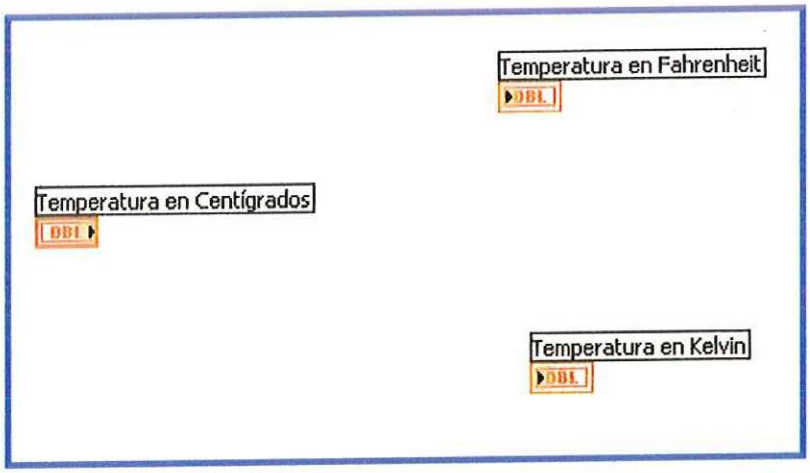
Ya insertados los objetos dentro del espacio de trabajo, es necesario editarlos para tener una buena visualización y entendimiento durante el procedimiento del programa. Para esto se etiquetan la entrada como *Temperatura en Centígrados* y la salidas como *Temperatura en Fahrenheit* y *Temperatura en Kelvin*. Además se insertan unos visores (***Digital Display***) a los termómetros para tener una mejor lectura .



También se modifica el rango de visualización de los termómetros y su tamaño. Por último se ordenan los objetos.



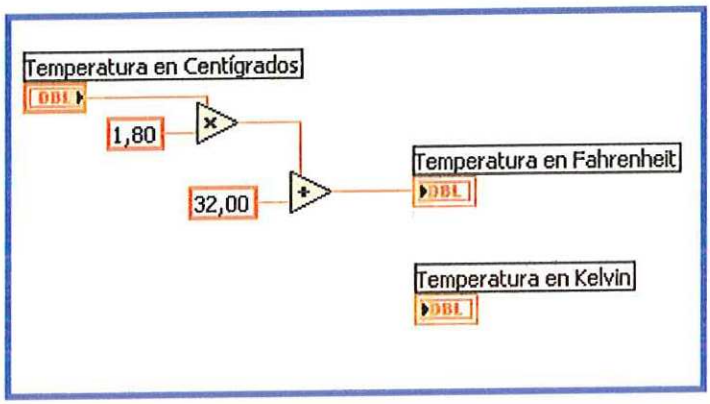
4) Con el trabajo realizado de Panel, se procede ahora a editar el programa de control en Diagram.



Se pueden identificar los objetos insertados por la etiqueta en la parte superior de cada imagen. A través de la casilla ***Numeric*** de la ventana ***Functions*** se insertan las funciones de multiplicación y suma necesarios.

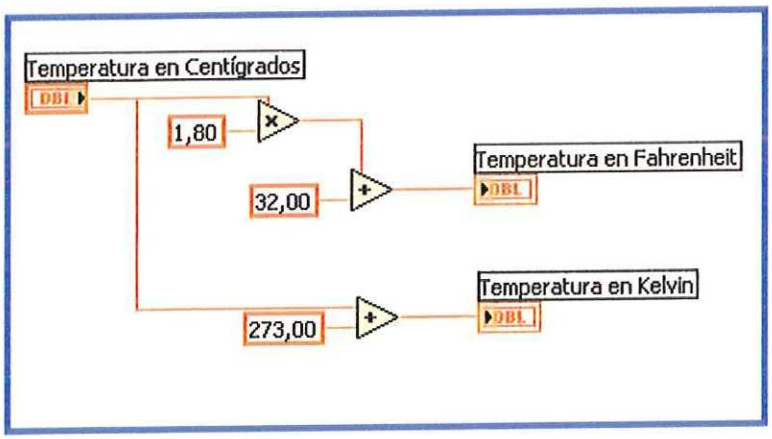
Para pasar de centígrados a Fahrenheit se toma el dato proveniente de la entrada, se multiplica por **1,8** y se suma con **32**.

***Nota**: puede cambiar la posición de las casillas asociadas a los objetos de Panel, esto no afectará el orden que ya se estableció.*



El mismo dato proveniente de la entrada se puede utilizar para hacer la conversión a grados Kelvin. Sólo se necesita sumar el valor de la entrada con **273**.

***Nota**: puede conectar el terminal X de la función Suma directamente en la línea de datos de la entrada Temperatura en Centígrados.*



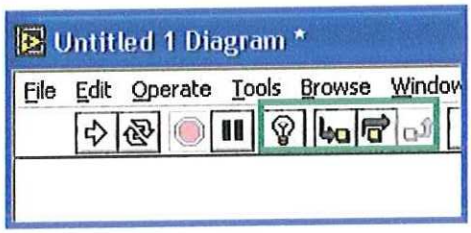
- **Depuración de un VI**

Una vez hecho el proceso se puede ejecutar el programa, para esto existen varias alternativas.

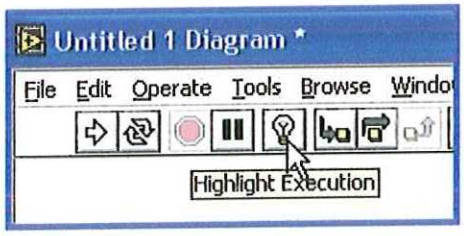
a) Correr el proceso paso a paso.

A través de una serie de íconos que se encuentran disponibles en Diagram

se puede ejecutar el programa paso a paso, de esta forma se puede verificar el trayecto de las variables, el cambio en su valor y así determinar las causas de un posible problema.

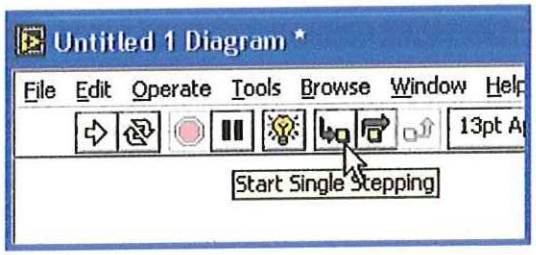


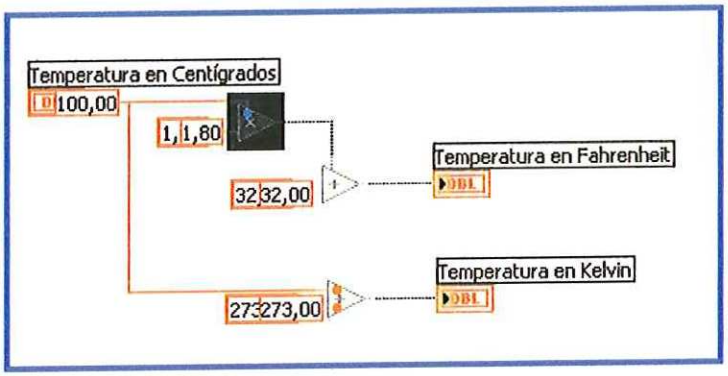
Para activar el paso a paso, de clic sobre el icono ***Highlight Execution***.



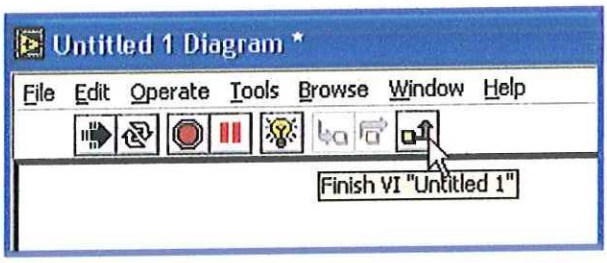
El icono cambia de imagen, adquiriendo la apariencia de un bombillo encendido.

Ahora es necesario dar clic en el icono ***Start Single Stepping*** para ejecutar el proceso paso por caso. Podrá ver mediante indicaciones luminosas la trayectoria del dato y los valores que va tomando a medida que avanza.



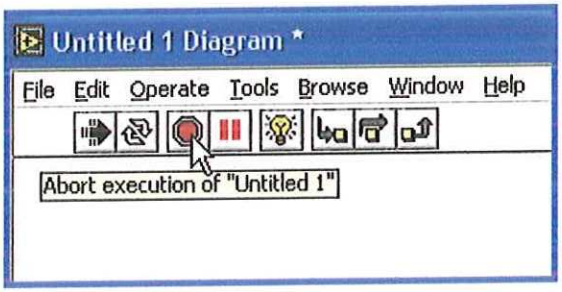


Cuando el programa finaliza los iconos Start Single Stepping se desactivan y dan paso a otro icono ***Finish File***. De clic sobre él para finalizar el proceso.



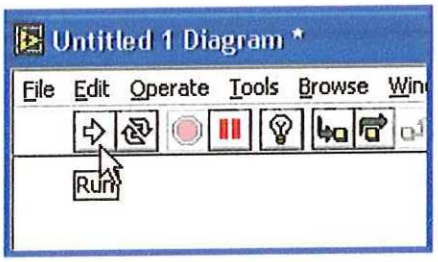
Para desactivar el modo paso a paso de clic nuevamente sobre el icono Highlight Execution, ahora denominado ***Do not highlight execution***. La bombilla se apaga.

Nota: *puede abortar la exploración paso a paso dando clic sobre el icono* ***Abort Execution****. El programa vuelve a la etapa inicial.*



b) Correr el proceso sólo una vez

Para esto es necesario dar clic sobre el icono ***Run,*** esta opción está tanto en Diagram como en Panel.



c) Correr el proceso de forma indefinida

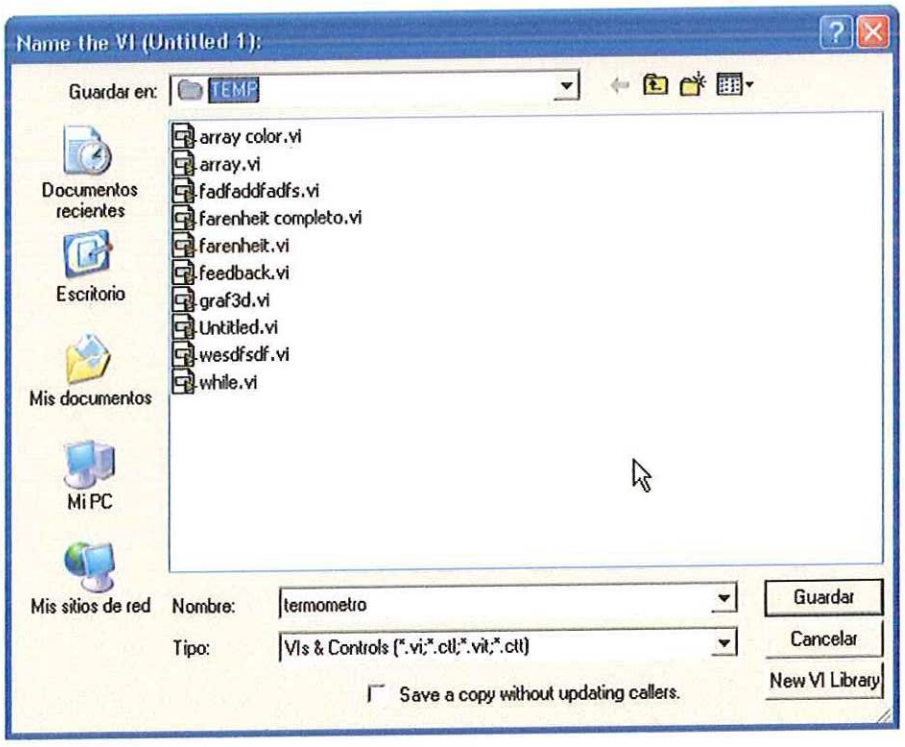
Para esto es necesario en Diagram o en Panel dar clic sobre el icono ***Run Continuosly***.

***Nota**: para detener el proceso se da clic sobre el icono **Abort Execution**.*

- **Guardar el VI**

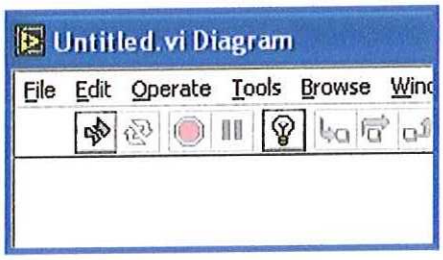
El programa se puede guardar tanto en Diagram como en Panel. Para esto en menú File, seleccione Save, o desde la ventana CTRL. + S.

Una nueva ventana aparece.

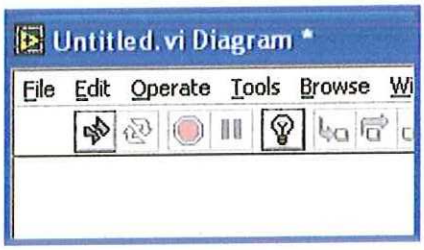


En la casilla ***Nombre*** puede asignar un nombre al programa, se le asignará la extensión VI. De clic sobre ***Guardar***.

***Nota**: Una vez se halla guardado el proceso, el nombre en el encabezado de la ventana toma la siguiente apariencia.*



Cuando se realiza una modificación sobre él, el encabezado agrega un asterisco al nombre.



El asterisco desaparece una vez se halla guardado nuevamente el proyecto.

5. ACTIVIDAD COMPLEMENTARIA

- Cree un programa que a partir de un voltaje y una resitencia conocida calcule el valor de la corriente.

- Cree un programa mediante LabVIEW para convertir la presión que se toma de un medidor en magnitud de Bar, a Psi, Pascales y atmosféra. ***Nota:*** *use una expresión matemática para realizar la conversión.*

PRÁCTICA 14
ESTRUCTURAS

PRÁCTICA 14
ESTRUCTURAS

1. OBJETIVOS

- ✓ Identificar los diferentes tipos de estructuras de programación disponibles en la ventana de trabajo Diagram.
- ✓ Identificar los diferentes tipos de datos que se pueden usar durante el desarrollo de un VI.
- ✓ Implementar los conceptos aprendidos en la solución de problemas a través de LabVIEW.

2. CONCEPTOS FUNDAMENTALES

Existen elementos que son totalmente esenciales en la elaboración de un programa. Hasta ahora su desarrollo de ha sido completamente lineal, sin embargo existen un gran número de situaciones que no se pueden resolver mediante este método, muchas de las cuales, requieren de la comparación de valores, de toma de decisiones, de ciclos repetitivos.

Cuando se habla de estructuras se hace referencia a algoritmos prediseñados que ejecutan funciones determinadas. Los bucles o ciclos repetitivos como los for y while, la selección de acuerdo al estado (switch) de una variable y las secuencias, son elementos que por su utilidad se requieren de forma indispensable en la solución de problemas mediante algoritmos de programación.

3. ACTIVIDAD PREVIA

Antes de proceder al desarrollo de esta práctica es necesario que elabore el siguiente cuestionario:

1) Estudie los diferentes tipos de estructras de programación que existen en los software existentes. Analice su estructura y funcionamiento.

2) Repase el material extra disponible al final de esta práctica: ***Tipos de datos***.

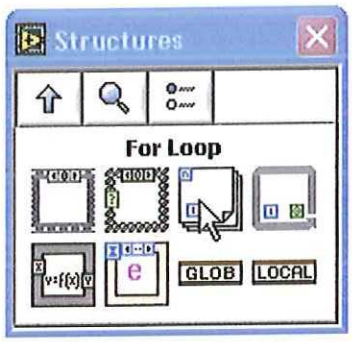
4. DESARROLLO METODOLÓGICO

Nota: *para el desarrollo de esta actividad es necesario haber elaborado la Práctica 1 .*

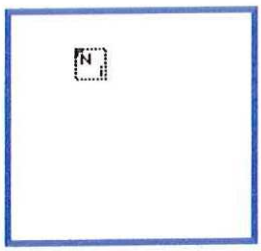
- **Estructuras de programación**

Las estructuras de programación están contenidas dentro de la casilla ***Structures***, en el submenú ***Controls*** de la ventana ***Diagram***.

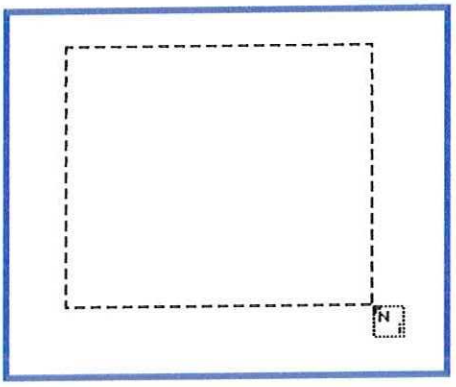
Para insertar cualquiera de los diagramas, basta con dar clic sobre una de las imágenes a seleccionar.



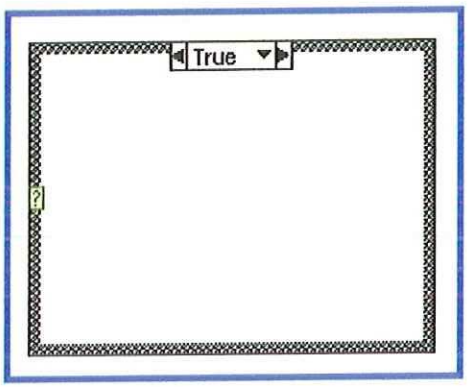
Cuando se desplace el cursor hacia el espacio de trabajo se transformará en un cuadro.



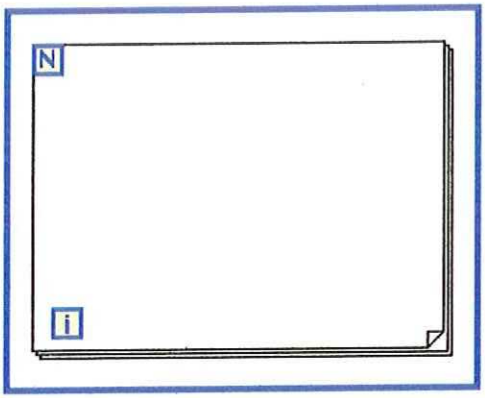
Dibuje un cuadricula con clic sostenido, el tamaño de la función es libre, arrastre el cursor y suéltelo cuando considere que el tamaño es justo.



La función queda sobre el espacio de trabajo.



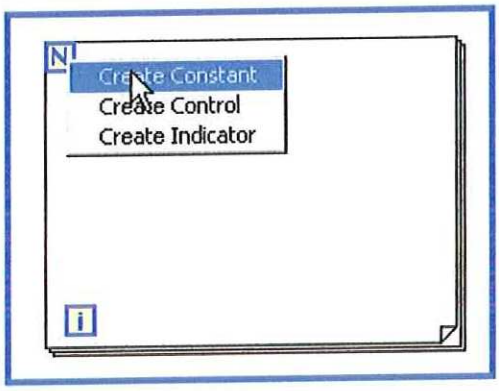
a) For

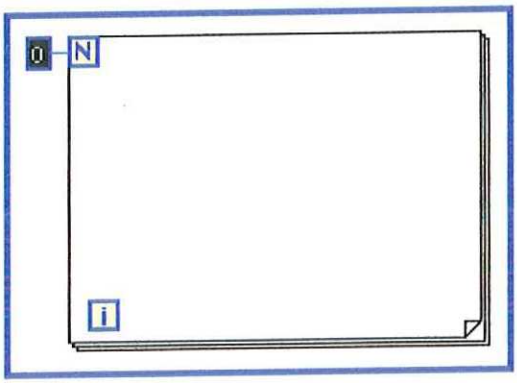


El ciclo repetitivo ***for*** ejecuta la estructura de programación que está contenida dentro de su espacio. El ciclo se repite N veces, y el conteo se almacena en la casilla i (el conteo se efectúa desde 0 asta el valor n-1).

Configuración:

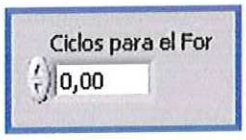
Para determinar el número de ciclos se da clic derecho sobre el icono N y se selecciona ***Create Constant***.

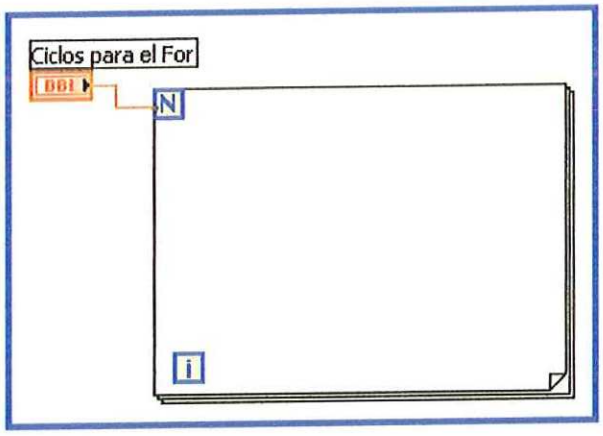




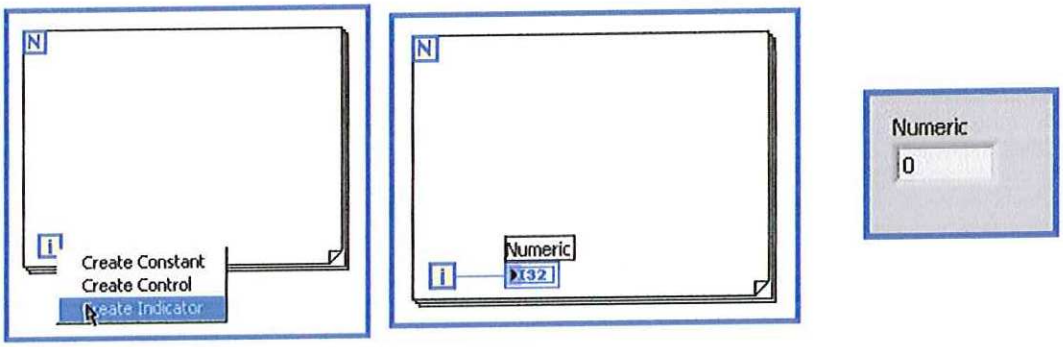
Con la herramienta de texto se puede determinar el valor numérico. Esta constante sólo se puede modificar cuando el programa no se está ejecutando.

También se puede adaptar una entrada desde Panel, para que esta controle el número de ciclos que For realizará, con esta opción se puede modificar la función mientras el programa se está ejecutando.



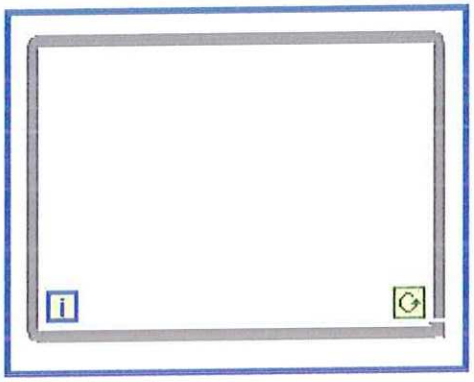


Si se quiere conocer el ciclo en el que está trabajando la función For, se puede adaptar un visor al icono i, para esto de clic derecho sobre él y seleccione ***Create indicator.***



Sobre Panel se ubicará un visor que mostrará el ciclo actual en el que se encuentra el For.

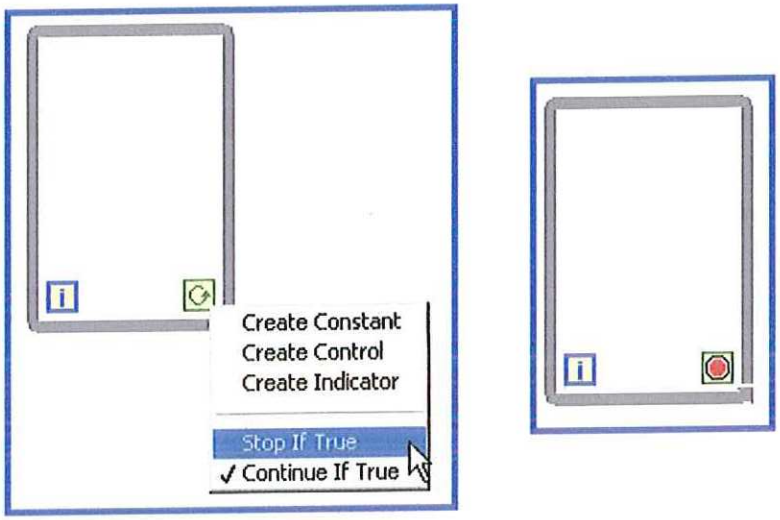
b) While



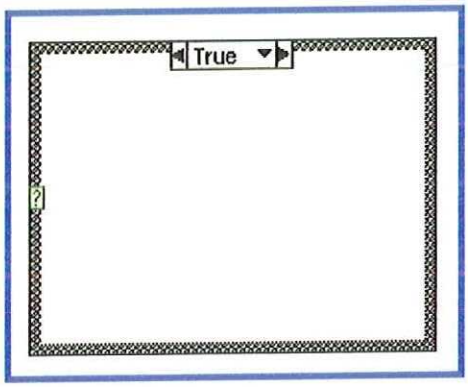
While ejecuta el diagrama que esta contenido dentro de su figura hasta que se cumpla la condición de salida. La condición es una variable tipo Boolean (TRUE o FALSE).

Un while puede trabajar de dos formas; Continue si es cierto (***continue if true***), Deténgase si es cierto (***stop if true***). El primer método establece que el ciclo repetitivo del while continuará mientras la condición de permanencia se siga cumpliendo. La segunda detiene el ciclo repetitivo del while cuando se cumple la condición de salida.

Para cambiar el modo de trabajo basta con dar clic derecho sobre el icono ***continuation behavior*** y seleccionar entre las dos opciones (Ver siguiente imagen).

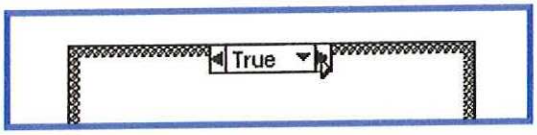


c) Case

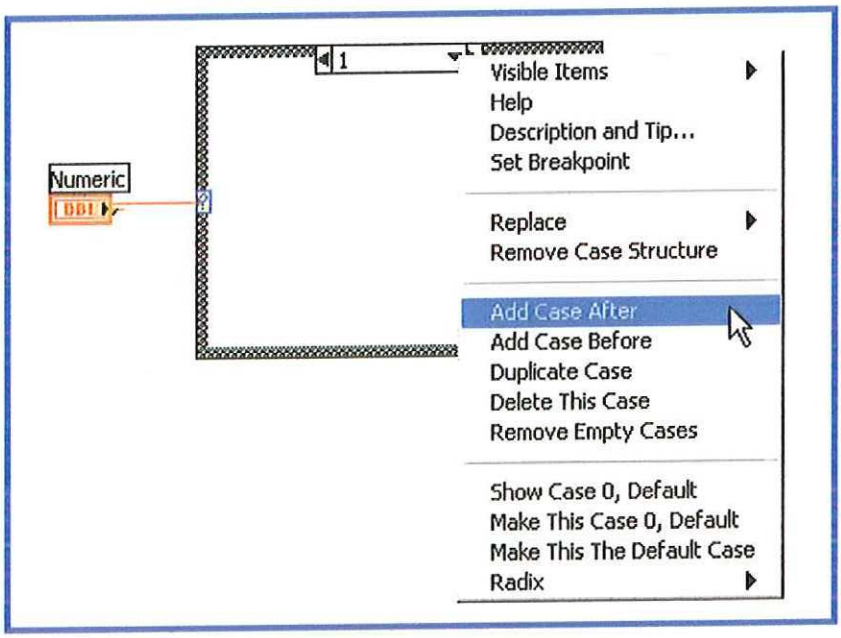


Seleccionar caso de, ejecuta una acción de acuerdo a una variable de entrada, esta se conecta casilla verde con signo de interrogación (?). Case trabaja de forma predeterminada con condiciones tipo boolean, mas si se conecta una variable tipo numeric a la casilla mencionada, Case modifica el rango de condiciones a variables numéricas.

Para cada condición Case despliega un diagrama diferente, existen tantos diagramas como condiciones se indiquen. Para desplazarse entre ellas basta dar clic sobre las flechas de selección ubicadas en la parte superior de la función.

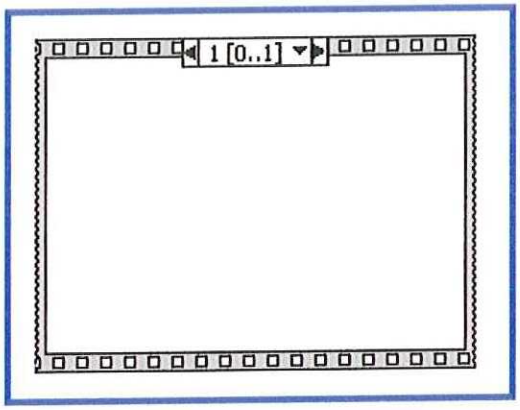


Para agregar una condición (en el caso que se quiera trabajar una variable numérica), basta dar clic derecho sobre la casilla de selección, y seleccionar ***Add Case After*** o ***Add Case Before***, (agregar un caso antes o después del actual), esto va de acuerdo a lo requerido por el usuario.



Para eliminar una condición se selecciona ***Delete this case***.

d) Sequence

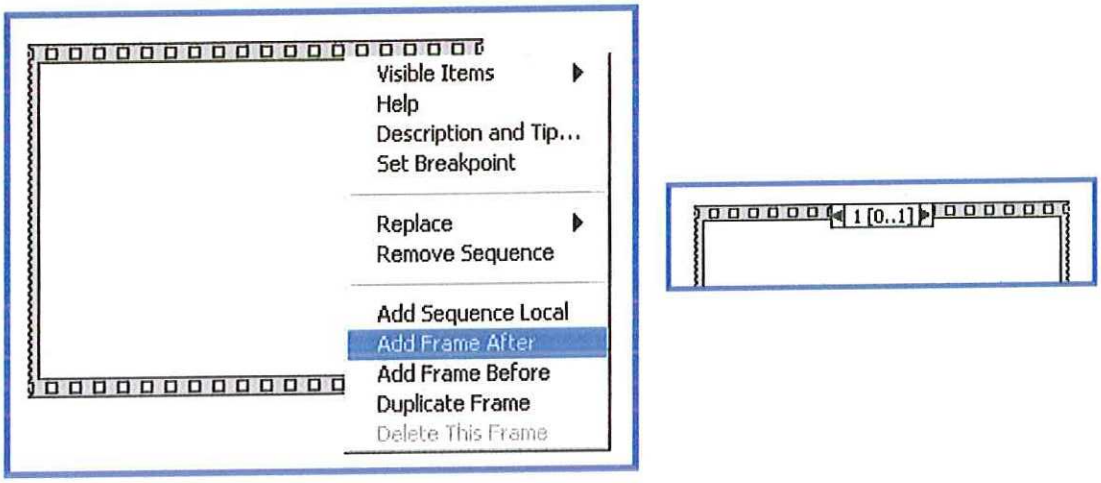


Esta función no existe en los programas de programación tradicionales, Sequence se configura como una serie de frames que se ejecutan consecutivamente, cada frame alberga en su interior una tarea, cuando la tarea se cumple, se da paso al próximo frame y así continúa el ciclo hasta

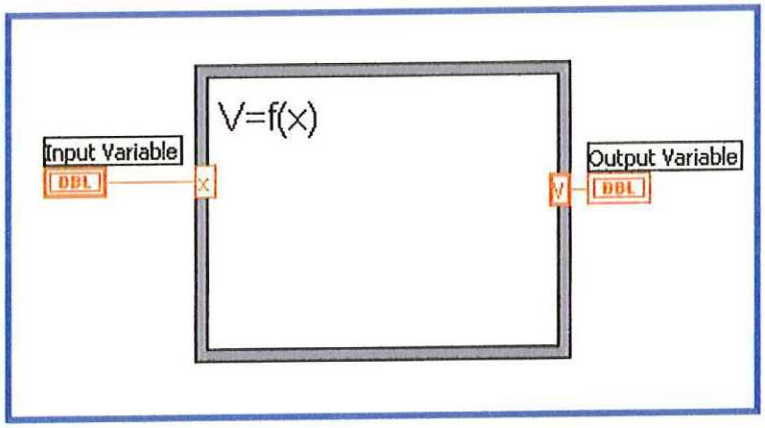
que se acaban.

Las tareas programadas para cada frame pueden ser independientes entre sí, es decir, no se comparte información. Por lo que se puede usar a sequence para ejecutar una serie de diagramas de forma consecutiva sin que exista un flujo de información entre ellos.

Para agregar un frame, basta con dar clic derecho sobre el borde y seleccionar ***Add Frame After*** o ***Add Frame Before***.

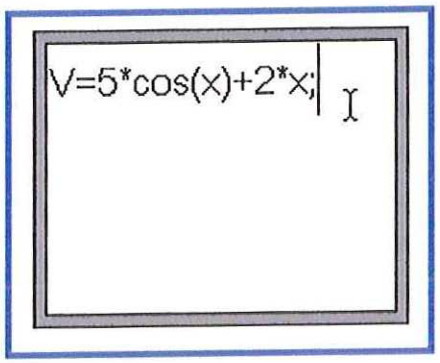


e) **Formula node**



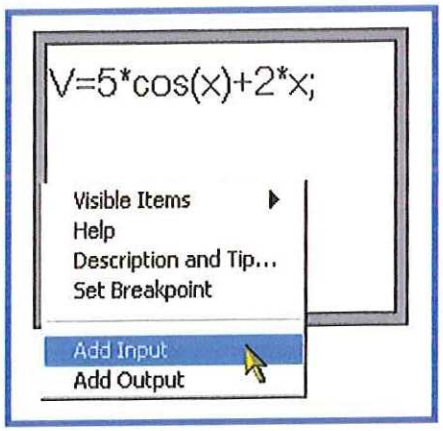
Formula Node ejecuta la función matemática que está comprendida dentro de su espacio de trabajo, mediante letras se determinan las variables y usando unos enlaces se conectan esas variables a los datos de entrada y de salida.

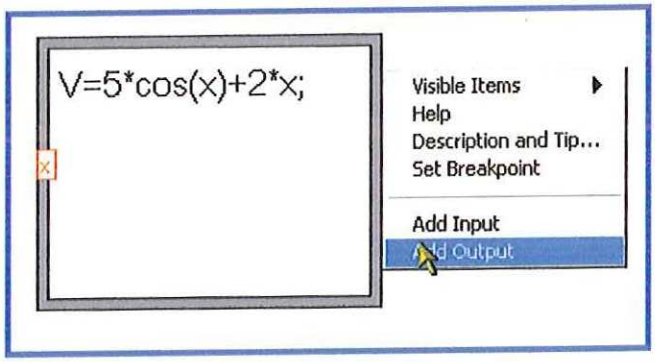
Se usa la herramienta de texto para insertar una ecuación.



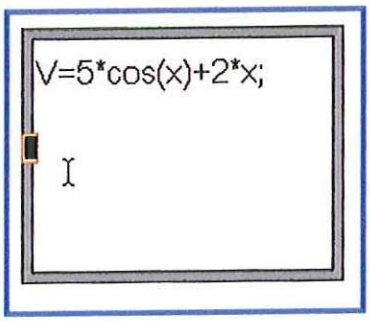
Las variables de entrada se definen mediante letras del alfabeto, es importante separar las variables de las constantes con las operaciones matemáticas correspondientes, al finalizar la ecuación se debe cerrar con un punto y coma (;).

Para enlazar las variables, y así, estas puedan tomar un valor numérico, se da clic derecho y se selecciona ***Add Input*** y ***Add Output***.

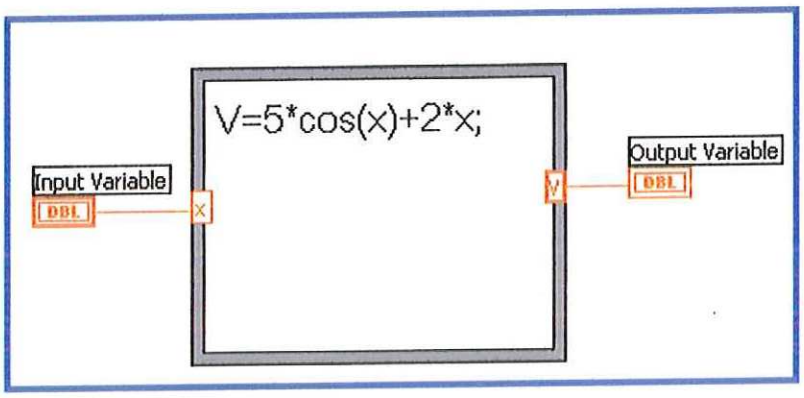




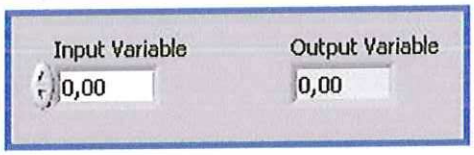
Con la herramienta de texto se configura el objeto insertado.



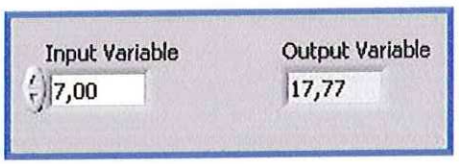
La ecuación tiene una entrada (x) y una salida (V). Las casilla deben contener el texto exacto sobre la variable que están haciendo referencia.



Al final se agrega una objeto entrada (***Input variable***) a la variable de entrada y un ***Ouput variable*** a la salida de la función. En Panel se verán estos indicadores.



Cambiando el valor de entrada se puede observar su respectiva respuesta.

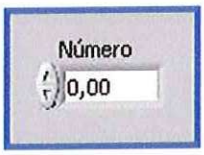


- **Trabajar con las funciones For y While**

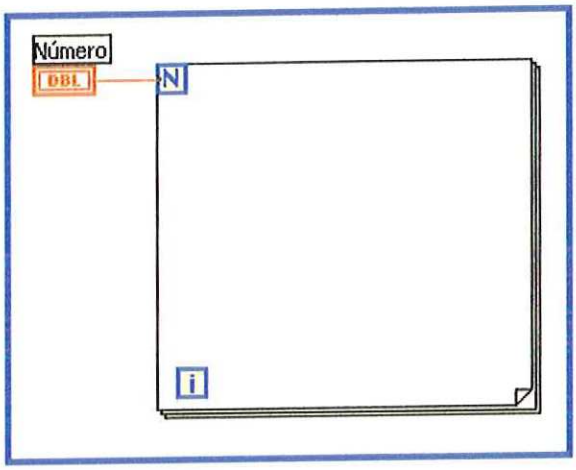
Realizar un programa que dado un número n, se calcule su factorial.

Ej: Factorial de 6= 1*2*3*4*5*6=720.

1) Desde Panel se inserta un Digital control y se etiqueta como *Número.*

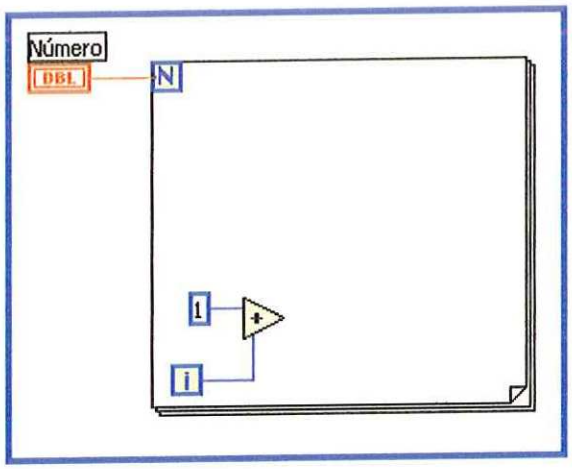


2) En Diagram se inserta una función For y se conecta Número con la casilla N.

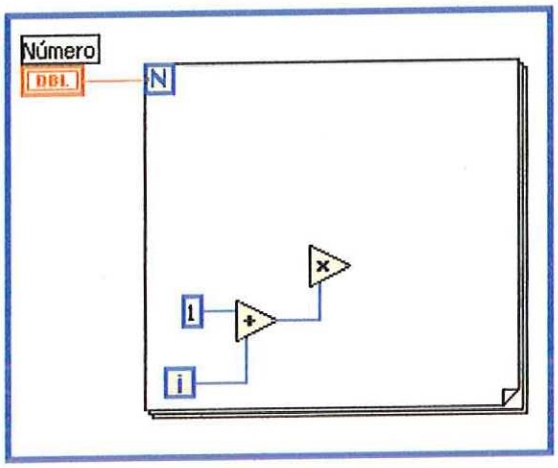


De esta forma el número de ciclos con que trabaje For dependerá de la entrada que se dé.

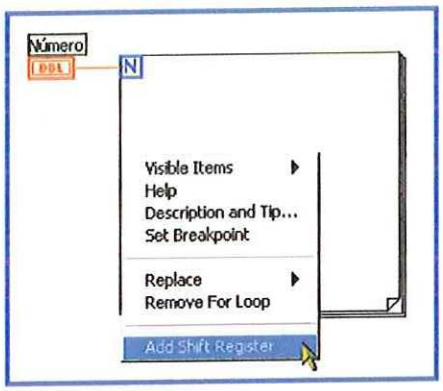
3) Ahora se inserta dentro de la función un bloque de suma y se conecta una de sus entradas con la casilla i. Conecte una constante con el valor de 1 a la otra entrada. **Recuerde que i lleva el conteo de ciclos desde 0 hasta n-1.**



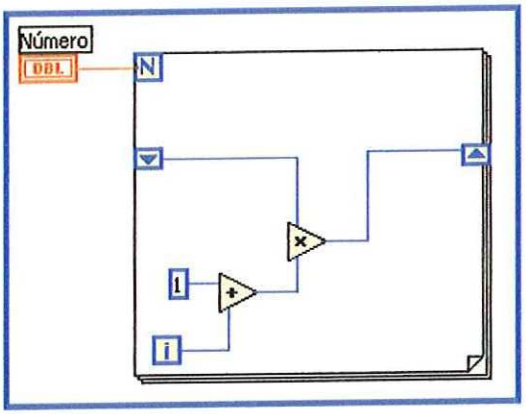
Ahora inserte un función de multiplicación y conecte la salida de suma con una de sus entradas. Como muestra la siguiente figura.



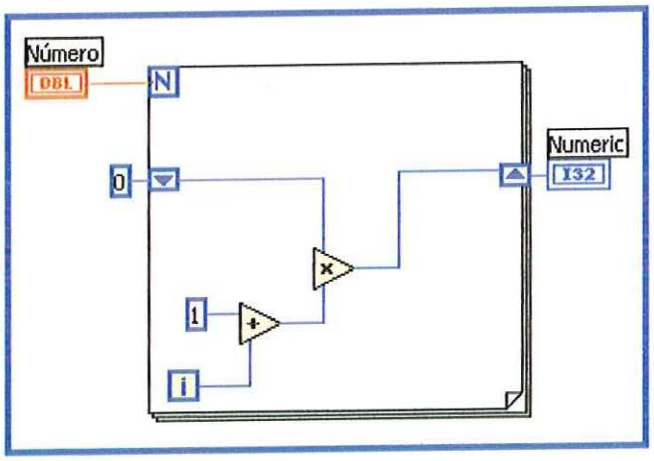
4) Es necesario tener almacenado el resultado del factorial, para poder transportar este dato entre ciclo y ciclo, se inserta un icono llamado ***Shift Registrer***, para esto de clic derecho sobre el borde de la función y seleccione ***Add Shift Registrer***.



Dos flechas aparecen entre borde y borde de la función. Conecte los puertos de multiplicación a esta flechas tal como muestra la siguiente figura. Tome nota de la dirección que estas tienen.



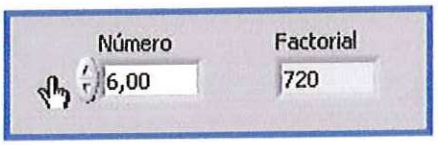
5) Conecte una constante (Clic derecho, Create/ ***Constant***)a la flecha con indicación hacia abajo (entrada de dato), y un indicador (Clic derecho, Create/***Indicator***) a la flecha con indicación hacia arriba (salida del dato).



Asigne un valor de 1 a la constante y etiquete el indicador con el nombre de *Factorial*.

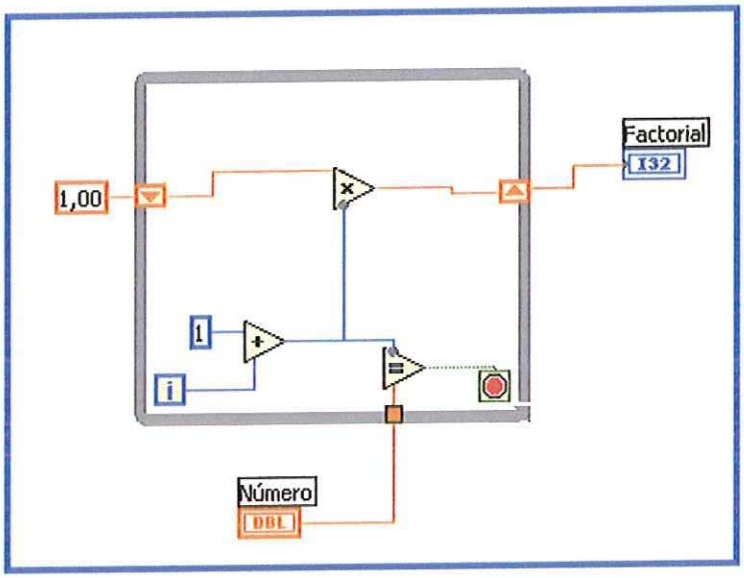
6) Guarde el trabajo con el nombre de Factorial.

7) Desde panel digite un valor en la casilla Número, de clic en Run y observe el resultado.



***Nota**: verifique el funcionamiento del programa, ejecutando el programa en modo **Paso a Paso**. (Ver Práctica 1).*

El siguiente diagrama realiza el mismo trabajo pero usando una función tipo While. Analice su funcionamiento.



Para este ejemplo se requirió de la utilización de una función de comparación igual que (***Equal?***), quien compara dos números, si estos dos son iguales genera un TRUE, de lo contrario su respuesta será siempre FALSE.

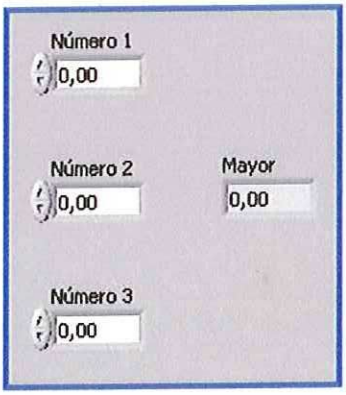
Las funciones de comparación se encuentran en la casilla ***Comparison*** del submenú ***Functions***.

- **Trabajar con funciones Case y Secuence**

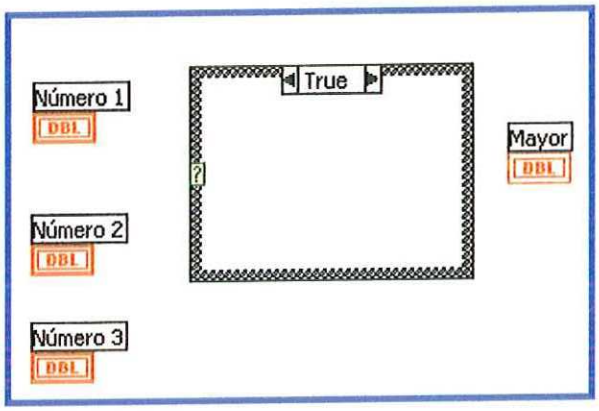
Compare tres números y determine cual es mayor.

Para realizar esta actividad es necesario recurrir a la función case.

1) En Panel inserte tres Digital Control y un Digital Display, estas serán las entradas y la salida, etiquete los objetos como muestra la figura a continuación.

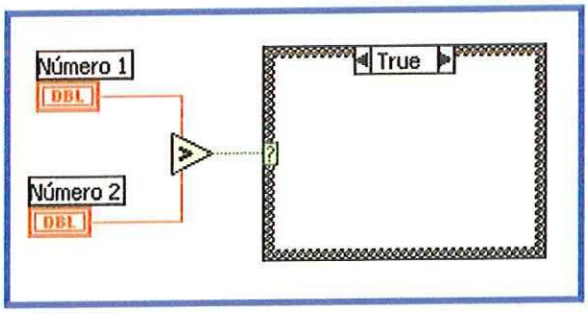


2) Ahora en diagrama inserte una estructura tipo case.

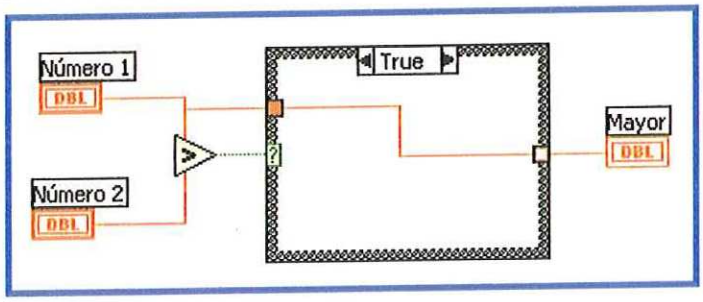


El programa se realizará de la siguiente forma, primero se compararán 2 números y se determinará cual de ellos es el mayor. Este número se comparará con el tercero y haciendo el mismo análisis se determina la respuesta.

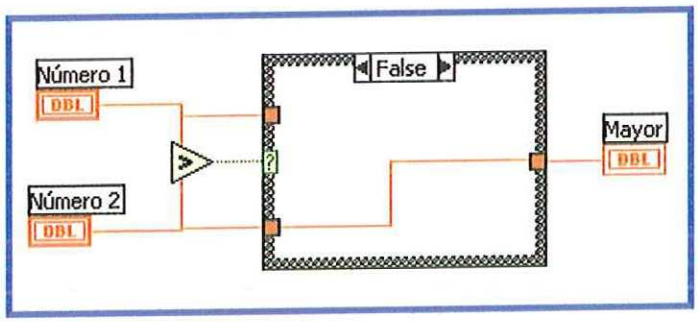
3) Primero se compararán N1 y N2, para esto de la casilla Comparasion se inserta la función Mayor (***Greater?***). Sus dos entradas X y Y se conectan con las N1 y N2, su salida (que es de tipo Boolean) con el icono ?.



En el caso que N1 sea mayor que N2, la respuesta de la función Mayor será True, el caso contrario será False. Se usará el visor para comprobar si esta primera comparación se hizo de forma correcta. Primero, con la función case en True se conecta N1 con el visor, procurando que la línea que los une pase a través de la función.

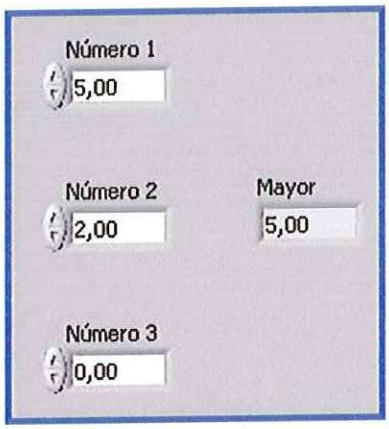


Ahora con el puntero del Mouse cambie la función Case de True a False. Notará que la conexión desaparece, ahora conecte N2 con el visor.

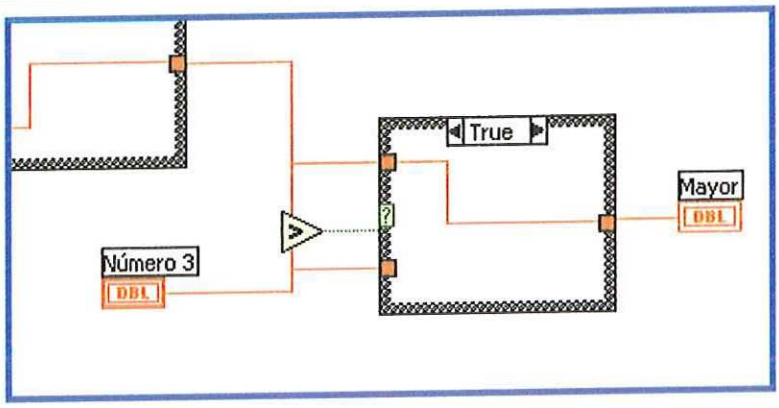


Case crea un subdiagrama para cada condición; se puede desplazar entre ellas usando el puntero del Mouse.

4) Visualice el funcionamiento del programa en Panel.

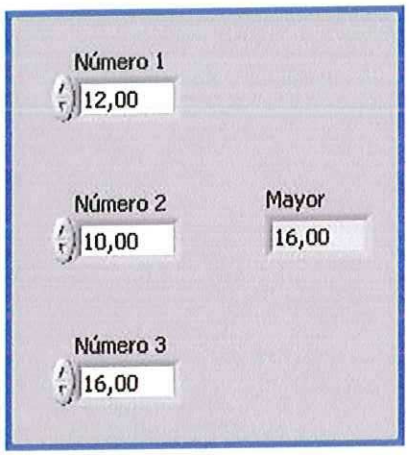


5) Ahora se comparará el mayor entre N1 y N2 con N3, haciendo el mismo análisis se obtiene el siguiente diagrama.



***Nota**: el visor se desplazó hacia la salida del segundo case, para poder de esta forma con el mismo objeto visualizar el mayor global.*

6) En Panel visualice el funcionamiento del programa.



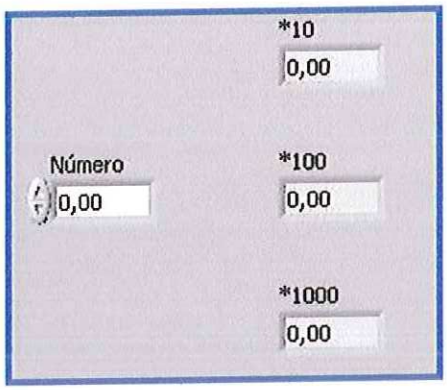
7) Guarde el programa como comparación.

Ahora el siguiente ejemplo aclarará el funcionamiento de Sequence.

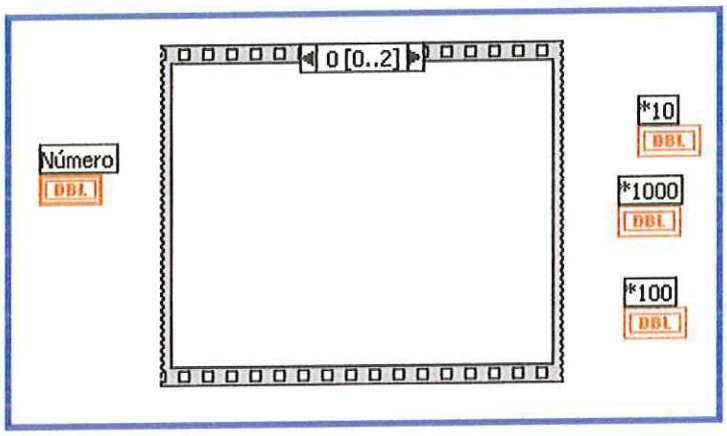
Cree un programa que de acuerdo a una entrada la multiplique por 10, 100 y 1000.

Se puede crear una secuencia de eventos que realicen estas operaciones.

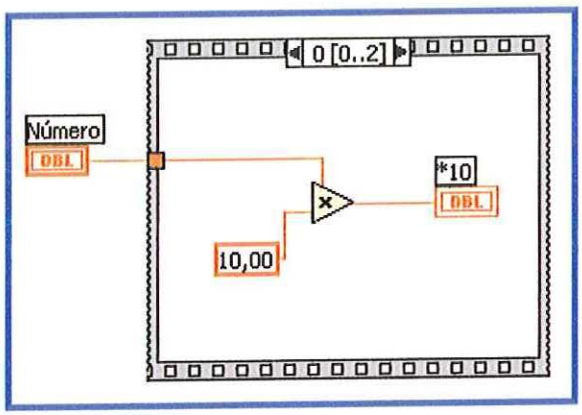
1) En Panel inserte un ***Digital Control*** y tres ***Digital Indicador.*** Etiquételos como se muestra a continuación.



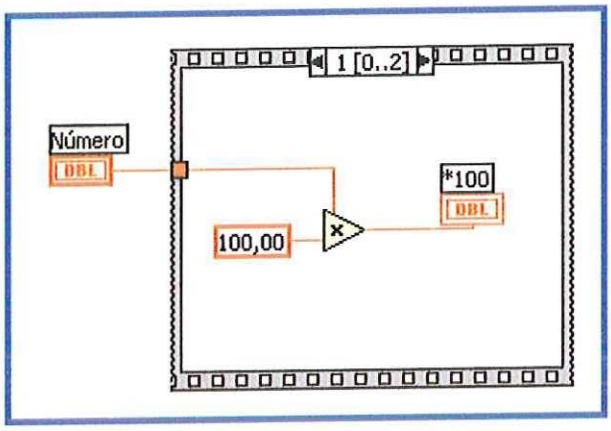
2) En Diagram inserte una estructura ***Sequence***. Cree un estructura con tres frames (de 0 a 2).



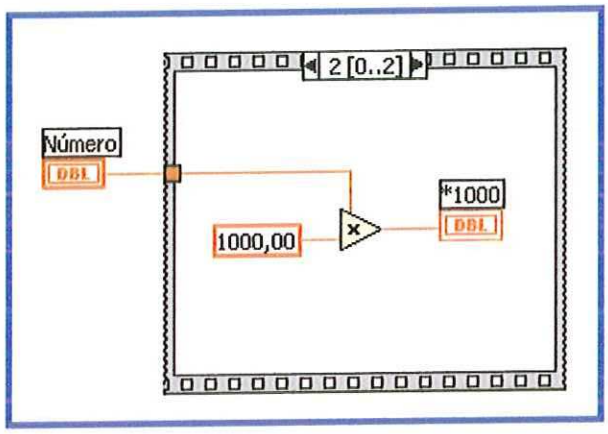
3) Con el puntero, desplácese entre los frames hasta el primero (0), sobre este inserte el siguiente diagrama.



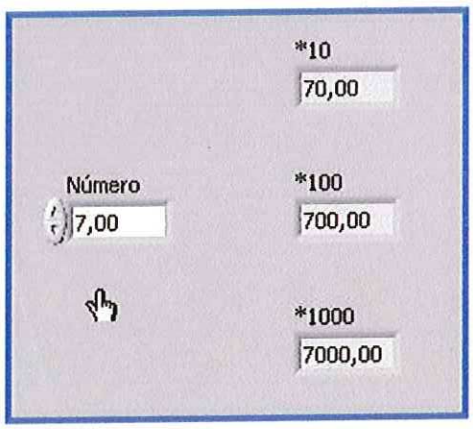
4) En el siguiente frame (1).



5) Y en el último frame (2).

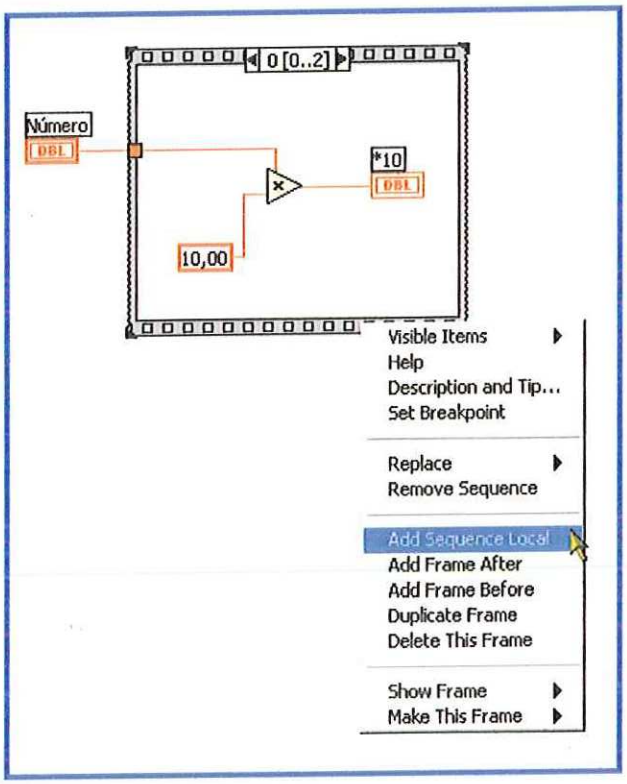


6) En Panel Puede probar el funcionamiento de este diagrama.

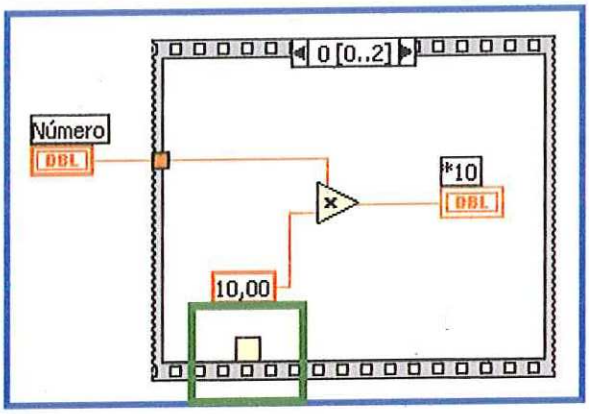


Sequence ejecuta los tres frames en secuencia, uno después del otro. Como pudo observar en ningún momento hubo un flujo de datos entre ellos.

Sin embargo, sequence puede compartir datos entre frames para esto, se utiliza un ***Sequence Local***, para insertarlo es necesario dar clic derecho en el borde de la función y seleccionar ***Add Sequence Local***.

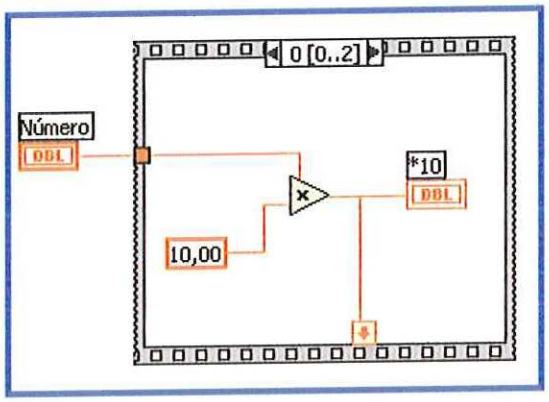


Aparece un **icono en forma de cuadrado** en el borde de la función.

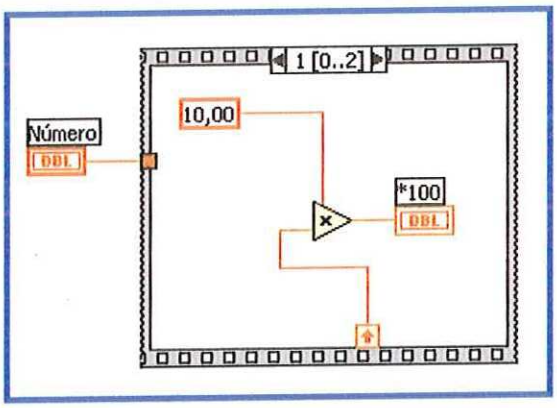


Ahora se puede modificar el programa para que realice el mismo trabajo, pero ahora multiplicará de 10 en 10 y ya no se usará la cifra aumentativa 10, 100 y 1000. Para poder hacer esto es necesario compartir datos entre cada frame.

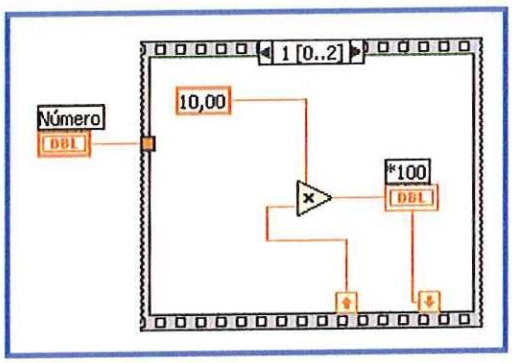
1) En el frame inicial conecte mediante la herramienta ***Connect Wire,*** la salida de la función de multiplicación con el Sequence Local, notará que la figura cambia de forma y ahora tiene una flecha en su interior.



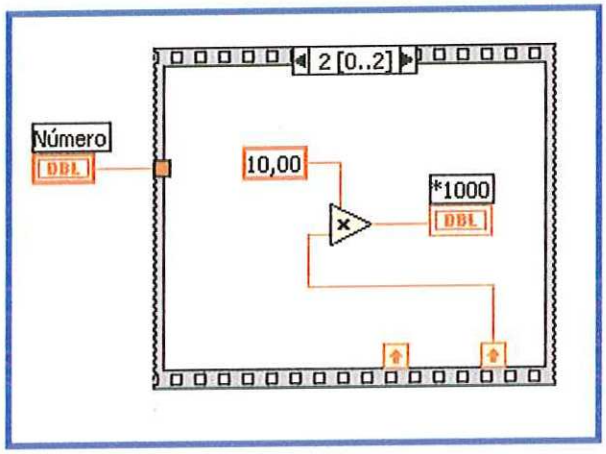
2) Desplácese hacia el siguiente frame y modifique la función de multiplicación. Ahora las entradas serán el Sequence Local y el número 10.



3) Inserte otro Sequence Local y conecte este con la salida de la función de multiplicación.



4) Modifique el último frame como lo muestra la siguiente figura.



5) Ahora visualice los cambios en Panel.

5. ACTIVIDAD COMPLEMENTARIA

- Usando la función Formula Node, edite un programa que de respuesta a las siguientes ecuaciones:

$$y = 6x^2 - 12x + 4$$
$$y = \sin(x) + \cos(x)\tan(x)$$
$$y = |4\cos(x) - 5sen(x)|$$
$$z = \left(\sqrt{x+y}\right) - \frac{x}{y}$$

Verifique la ayuda de LabVIEW para conocer la nomenclatura de algunas funciones.

- Cree un programa que calcule las raíces de una ecuación cuadrática. ***Nota:*** *mediante indicadores tipo boolean notifique las siguiente situaciones.*

  a) *Error división por cero.*
  b) *Error raíz cuadrada de un número negativo.*

- Calcule la sumatoria de un número N.

- Cree un programa que calcule a un número la sumatoria, el factorial y determine si es par o impar. ***Nota:*** *utilice visores tipo boolean para este último punto.*

MATERIAL EXTRA No. 1

Tipos de datos

LabVIEW ofrece una gran variedad de tipos de datos con los que se puede trabajar respondiendo a las necesidades reales con las que nos encontraremos. Uno de los aspectos más significativos de LabVIEW es la diferenciación que efectúa en la ventana Diagram entre los diferentes tipos de controles o indicadores, basada en que cada uno de ellos tiene un color propio. La siguiente lista muestra una descripción de su tipo y la magnitud que presentan.

Boolean (verde claro): Los tipos de datos booleanos son enteros de 16 bits. El bit más significativo contiene el valor Booleano. Si el bit 15 se pone a 1, entonces el valor del control o indicador es **true** (verdadero); por el contrario, si este bit 15 vale 0, el valor de la variable booleana será **false** (falso).

Numéricos: Hay diferentes tipos.

Extended (naranja): (Reales) Según el modelo de ordenador que se está utilizando los números de coma flotante con precisión extendida presentan el siguiente formato: Macintosh,96 bits. Windows:80 bits. Sun: 128 bits.

Double (naranja): Los números en coma flotante de doble precisión tienen un magnitud de 64 bits. Es el valor por defecto de LabVIEW.

Single (naranja): Los números en coma flotante de precisión simple tienen una magnitud de 32 bits.

Long Integer (azul): Números naturales, los números enteros largos tienen un formato de 32 bits, con o sin signo.

Word Integer (azul): Estos números tienen un formato de 16 bits, con o sin signo.

Byte Integer (azul): Tienen un formato de 8 bits, con o sin signo.

Unsigned Long (azul): Entero largo sin signo.

Unsigned Word (azul): Palabra sin signo.

Unsigned Byte (azul): Byte sin signo.

Complex Extended (naranja): Número complejo con precisión extendida.

Complex Double (naranja): Complejo con precisión doble.

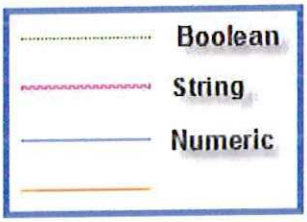
Complex Single (naranja): Complejo con precisión simple.

Arrays (el color depende del tipo de datos que contenga): Un array es una agrupación de valores, tales como un vector o una matriz.

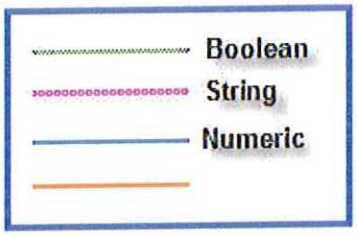
Strings (rosa): caracteres, LabVIEW almacena los strings como si fueran un array uni-dimensional de bytes enteros (caracteres de 8 bits).

No sólo los tipos de datos tienen sus propios colores, los conectores que se asocian a ellos también adoptan las mismas características. De esta forma es mucho mas fácil seguir el recorrido de una variable en un diagrama.

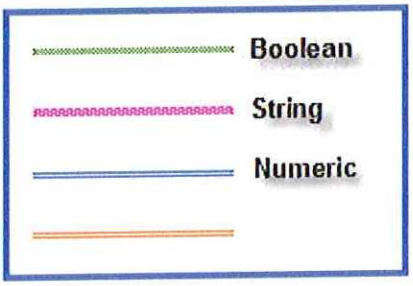
Datos Sencillos:



Arrays de 1 dimensión (vectores):



Arrays de 2 dimensiones (matrices):



PRÁCTICA 15
ARRAYS

PRÁCTICA 15
ARRAYS

1. OBJETIVOS

- ✓ Crear estructuras de datos mediante las herramientas proporcionadas por LabVIEW.
- ✓ Editar las estructuras de datos según las necesidades del proyecto.
- ✓ Implementar los conceptos aprendidos en la solución de problemas a través de LabVIEW.

2. CONCEPTOS FUNDAMENTALES

Una estructura de datos es una colección de datos que se caracteriza por su organización. Un array es una colección de datos del mismo tipo (numeric, boolean, real), cuya característica es que se almacena en posiciones consecutivas de memoria y reciben un nombre común. Cada ítem del array se denomina elemento. Para referirse a un determinado elemento de un array, se tiene un índice (index) que especifica su posición relativa en el array, Los arrays pueden ser:

- Unidimensionales: llamados vectores.
- Bidimensionales: denominados matrices, es un vector de vectores.
- Multidimensionales: de tres o más dimensiones.

La estructura de un array unidimensional es la siguiente:

index	0	1	2	3	4	5	6
	0.26	10	8	1.3	9.1	11	0

Array unidimensional de 7 elementos

Cada elemento es direccionado mediante un índice (index). El conteo inicia desde 0 y va hasta n-1.

Un array bidimensional o matriz:

	0	1	2	3	4	5	6	7	columnas
filas 0	0.26	10	8	1.3	9.1	11	0	36	
1	5	79	0.11	2	10	3.2	2	4.6	
3	7	9.36	8	3	1.1	2.6	4	0	

Array bidimensional de 21 elementos

Una matriz esta compuesta por n filas y n columnas, cada elemento se direcciona de acuerdo con el número de fila y de columna que le corresponde.

3. ACTIVIDAD PREVIA

Antes de proceder al desarrollo de esta práctica es necesario que elabore el siguiente cuestionario:

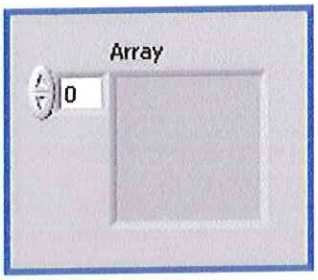
1) Consulte sobre el tratamiento que se da a las estructuras de datos en otros softwares de programación. Verifique nomenclaturas, creación y su edición.

4. DESARROLLO METODOLÓGICO

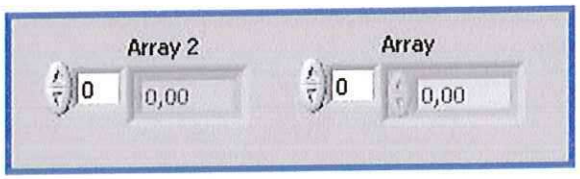
Nota: *para el desarrollo de esta actividad es necesario haber elaborado la Práctica 0, 1 y 2.*

- **Creación de un array unidimensional desde Panel**

1) En el submenú *Control,* seleccione ***Array*** de la casilla ***Arrays & Clusters***. Inserte este objeto en el espacio de trabajo.



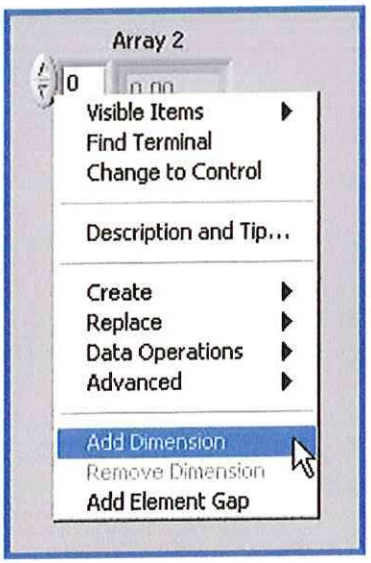
2) El objeto insertado es un array vacío. Se puede editar para que se convierta en un array de salidas (para ser editado desde Diagram), o un array de entradas, el cual se edita manualmente. Para crear el primero coloque dentro de la figura un ***Digital Indicator***. Para el segundo un ***Digital Control.***

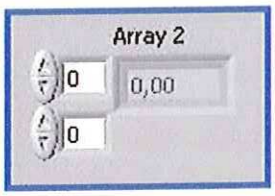


3) Se pueden crear arrays de diferentes tipos de objetos, como de booleans y de strings.

- **Creación de un array de n-dimensiones desde Panel.**

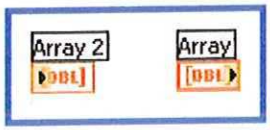
1) Para crear un array de n-dimensiones, basta dar clic derecho sobre él y seleccionar Add dimension. El array aumentará sus dimensiones.



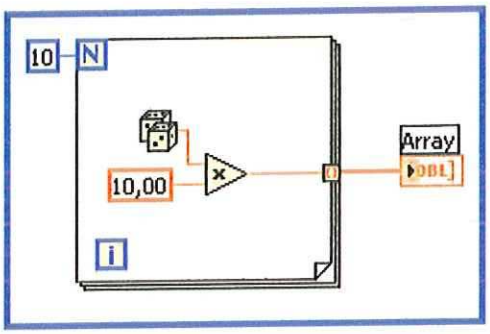


- **Edición de un array unidimensional**

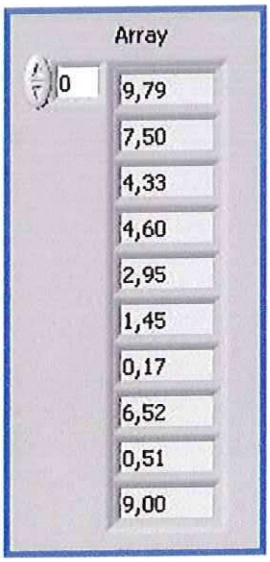
1) Un array unidimensional creado en el Panel se puede editar desde la ventana Diagram. En Diagram el array toma la forma de un bloque, el color depende del tipo de datos que contenga almacenados.



2) A parir de una función for, se puede crear un vector de dimensión N.



***Nota**: el objeto utilizado se llama **Random Number**, el cual genera un número randómico entre 0 y 1. El diagrama anterior crea un vector de 10 casillas.*

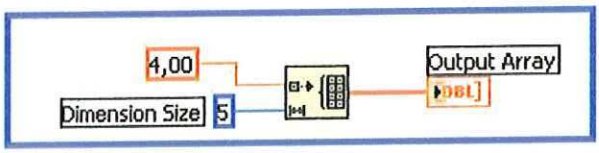


***Nota**: con el cursor del mouse puede modificar el tamaño del array para poder visualizarlo totalmente.*

- **Herramientas especiales**

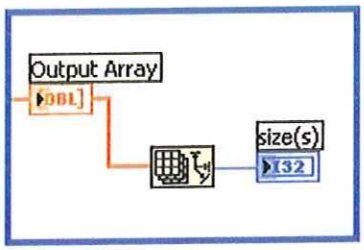
La siguiente colección de herramientas son funciones muy útiles para la edición de vectores, matrices o cualquier array de n-dimensiones. Todas hacen parte de la casilla ***Array*** del submenú ***Functions***.

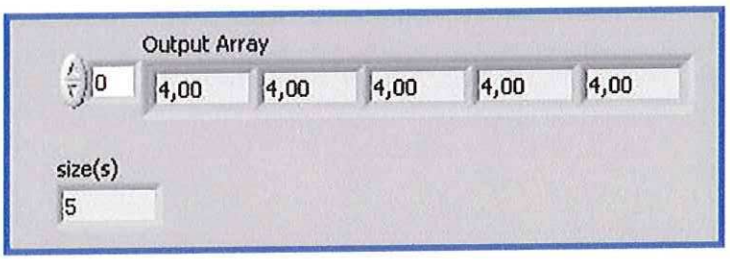
Initialize Array: crea un array de constantes con una magnitud (***Dimension Size***) la cual es también definida por el usuario.



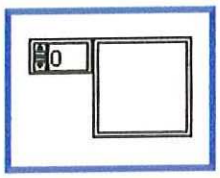
***Nota**: LabVIEW inicia el conteo desde 0, por lo que la numeración de las casillas irá hasta n-1.*

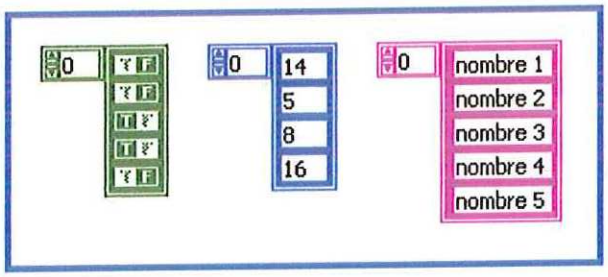
Array size: determina el tamaño de un array ya sea un vector o una matriz.





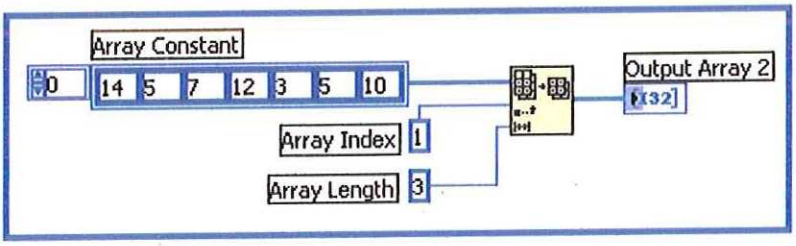
Array constant: crea un array el cual sólo es visible desde Diagram. Puede ser editado para trabajar con variables tipo numeric, boolean, string, etc.

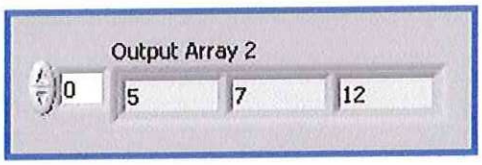




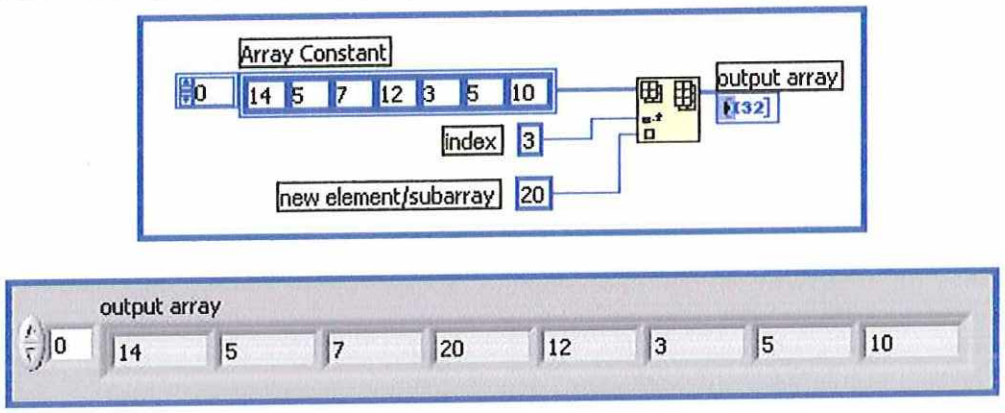
***Nota**: se editan de igual forma que los arrays creados desde Panel.*

Array Subset: recorta un número determinado de elementos (***Array Length***) que hacen parte de un array. La posición donde inicia el recorte es determinada por el usuario (***Array Index***). El resultado de la función es un nuevo array.

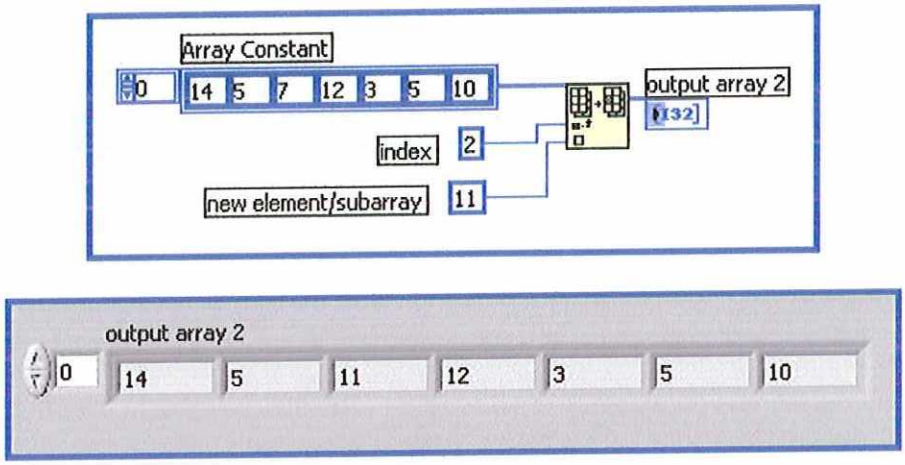




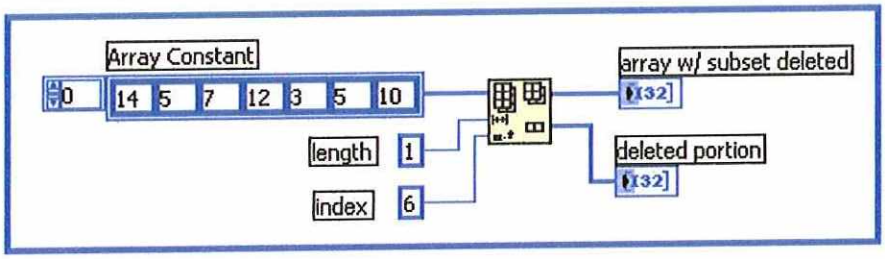
Insert Into a Array: Agrega un elemento dentro de un array. El elemento (***new element/subarray***) se ubicará en la posición determinada (***index***) y los elementos posteriores se desplazarán una posición.

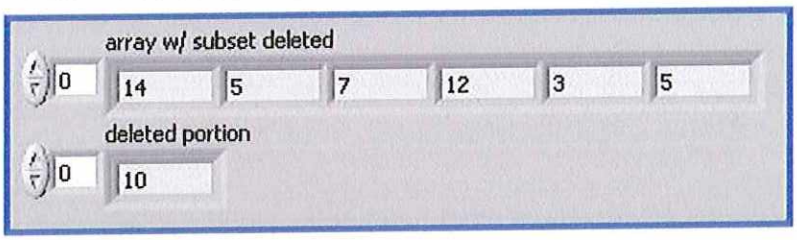


Replace Array Subset: reemplaza un elemento del array (***index***) por otro (***new element/subarray***).

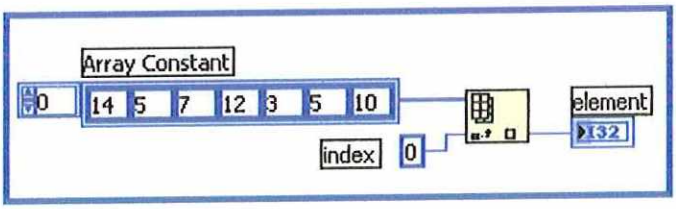


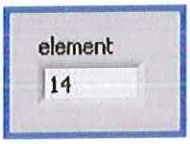
Delete from array: elimina uno o varios elementos (***length***) de un array, el recorte inicia desde la posición indicada (***index***). La función crea un nuevo array con los elementos eliminados (***deleted portion***)



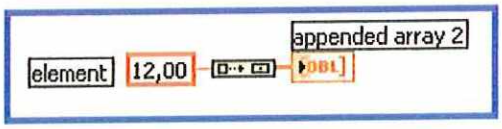


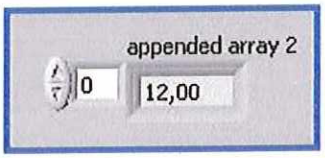
Index Array: extrae un elemento (***index***) de una array, el elemento queda a disposición para cualquier otra actividad.

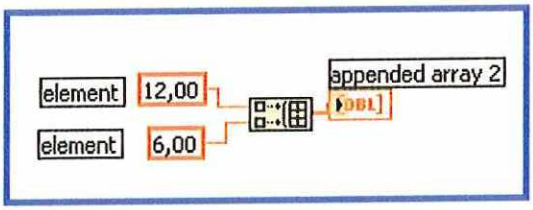


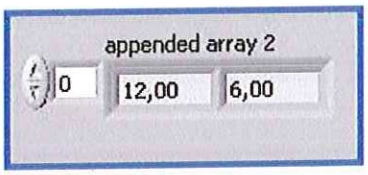


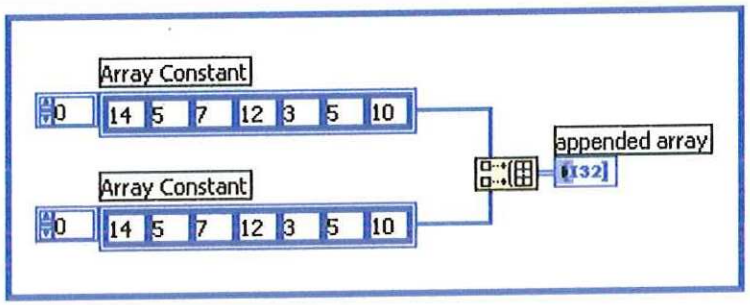
Build array: construye un array a partir de elementos sencillos, o vectores, en el caso de vectores crea una matriz.

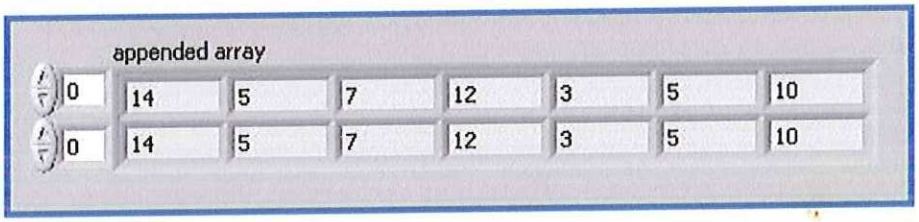


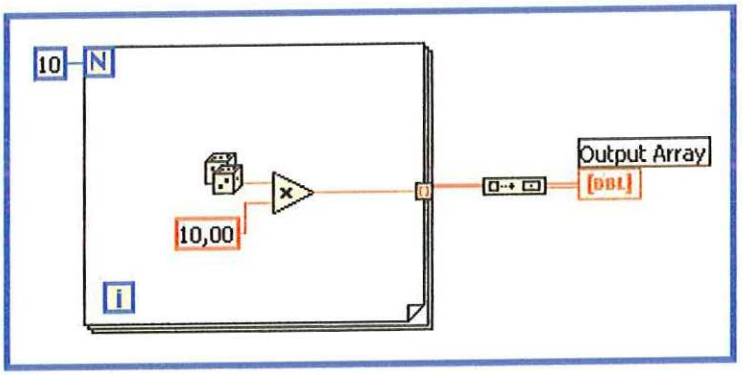


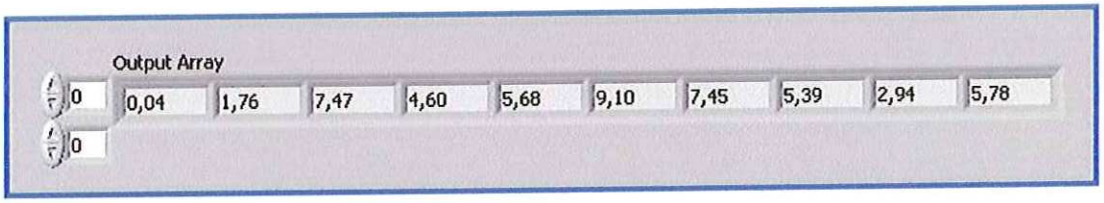








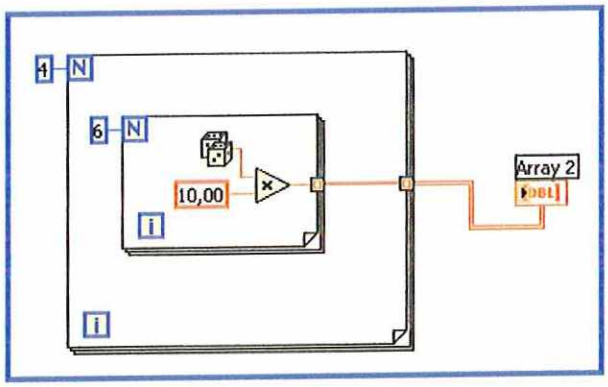




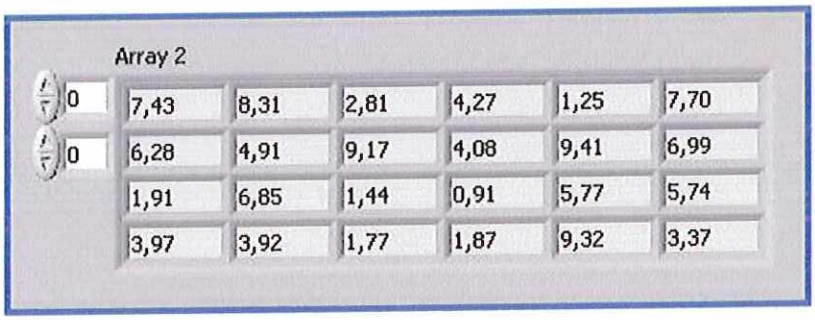
***Nota**: con el cursor modifique el tamaño de la función para poder agregar más elementos.*

- **Edición de un array n-dimensional**

Un array n dimensional se puede editar como lo muestra el siguiente diagrama:

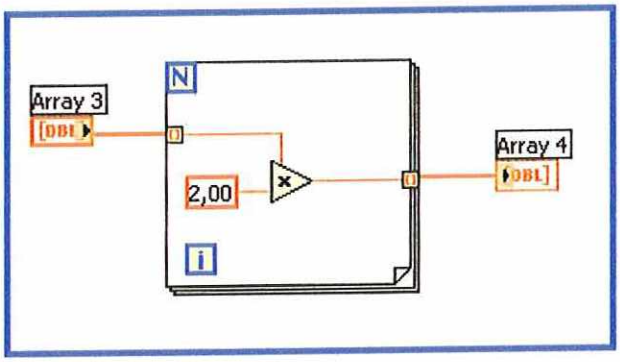


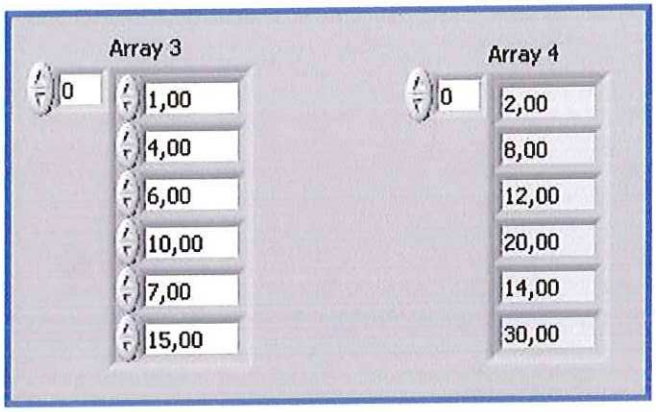
De esta forma se crea una matriz de 4 filas por 6 columnas. El for externo determina el número de filas, mientras el for interno el número de columnas.



- **Operaciones matemáticas con arrays**

Cuando se quiere realizar la misma acción matemática sobre un array, por ejemplo, multiplicar todos sus elementos por dos, se hace los siguiente:

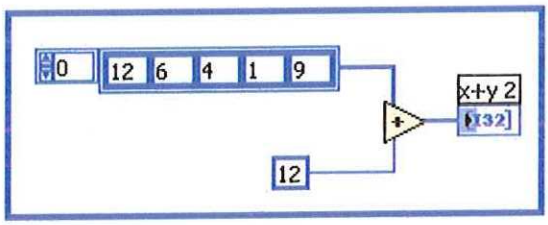


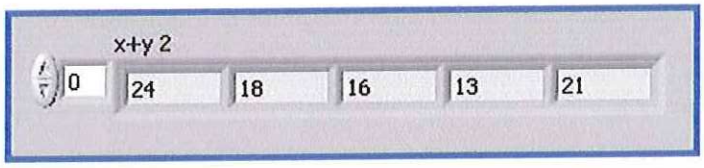


Array 3 es un vector de entradas, por lo que su edición se hizo de forma manual desde Panel. La función for asume automáticamente la dimensión del array por lo que no es necesario definir el numero de pasos en **N.**

Este mismo procedimiento se puede realizar con las demás funciones matemáticas.

Una respuesta similar se obtiene sumando un array con una constante.

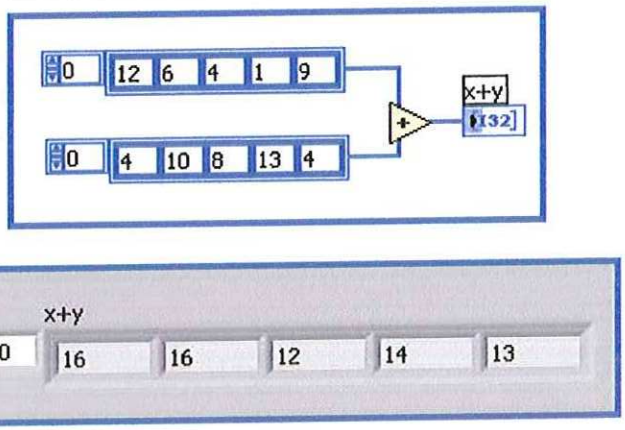




Las demás funciones matemáticas tienen el mismo comportamiento.

- **Operaciones matemáticas entre arrays.**

Usando las funciones matemáticas comunes, se pueden realizar operaciones entre arrays. El resultado de la operación será otro array.



5. ACTIVIDAD COMPLEMENTARIA

- Dada la siguiente ecuación:

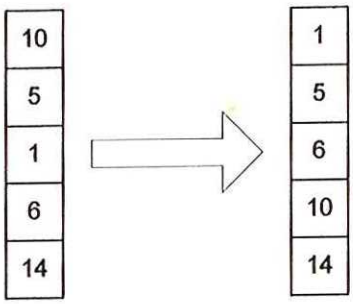
$$y = 5x^2 + 6x - 3$$

  Guarde los resultados obtenidos en una tabla. El rango de valores de x va de –10 hasta 10. La tabla debe contener el valor de x y su respectiva respuesta en la casilla de al frente.

- Diseñe un programa que sume la totalidad de los elementos que componen una matriz.

- Diseñe un programa que dado un vector calcule la sumatoria de cada elemento. La respuesta debe quedar en un nuevo vector.

- Dado un vector de 10 elementos determinados al azar, diseñe un programa que los organice. Para esto utilice el método de organización ***burbuja***. Ejemplo:



PRÁCTICA 16
CREACIÓN DE SUB-VI'S

PRÁCTICA 16
CREACIÓN DE SUB-VI'S

1. OBJETIVOS

- ✓ Desarrollar programas estructurados mediante la creación y utilización de SubVI's.
- ✓ Implementar los conceptos aprendidos en la solución de problemas a través de LabVIEW.

2. CONCEPTOS FUNDAMENTALES

Un SubVI es un programa que ejecuta una función particular, la característica más sobresaliente de esta herramienta, es que es totalmente diseñada por el usuario. Un SubVI convierte el código de un VI en una pequeña función que puede ser insertada en cualquier otro programa creado sobre LabVIEW.

La primera condición que se debe cumplir para que un VI se transforme en función es que funcione correctamente, luego, mediante una serie de herramientas se puede configurar un programa de acuerdo a sus entradas y salidas para que se convierta en una función que pueda ser implementada en cualquier otra aplicación.

Se pueden insertar tantos SubVI's como lo requiera el programa, del mismo tipo o de diferentes aplicaciones. Además un programa que contenga funciones de este tipo se puede convertir en una función mucho más grande. Así, se puede ir encapsulando un programa hasta que se convierta en una caja negra, la cual, a partir de un número reducido de entradas puede desarrollar múltiples tareas.

3. ACTIVIDAD PREVIA

Antes de proceder al desarrollo de esta práctica es necesario que elabore el siguiente cuestionario:

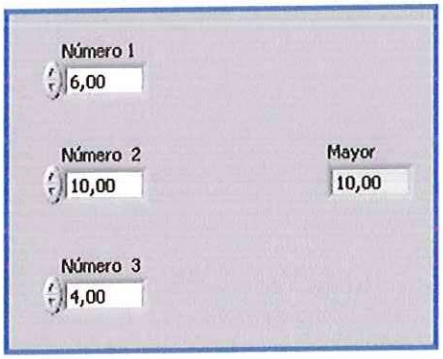
1) El tratamiento, o condensación de varias líneas de programación contenidas dentro de un solo parámetro no es nuevo. Programas antiguos como Turbo C vienen aplicando este método. ¿Cómo se crea una función en Turbo C? ¿Qué condiciones se deben tener en cuenta?

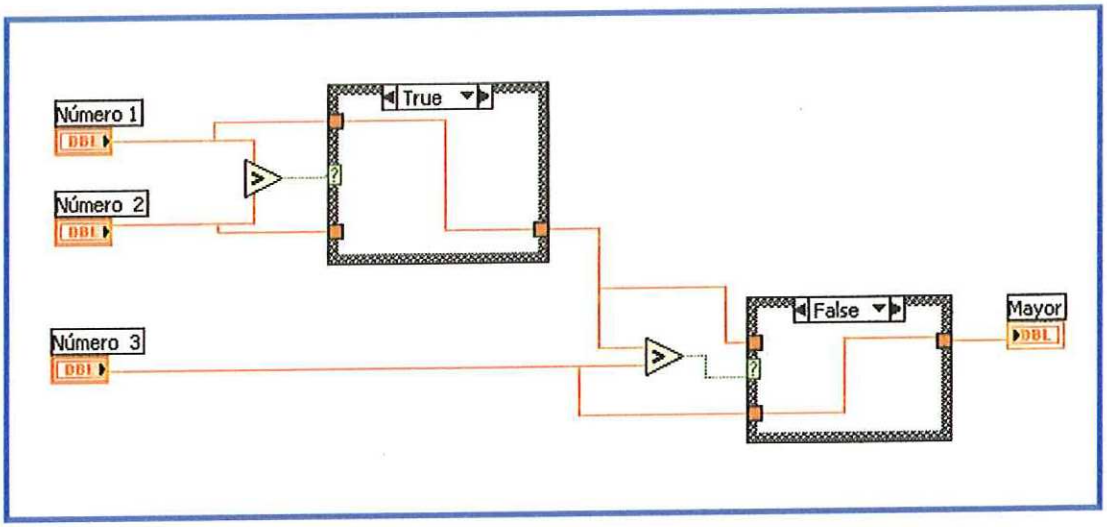
4. DESARROLLO METODOLÓGICO

Nota: *para el desarrollo de esta actividad es necesario haber elaborado la Práctica 0, 1 y 2.*

Como primera medida para poder crear un SubVI, es necesario tener un VI desarrollado. Por ejemplo, el programa realizado en la práctica 3, el cual compara tres números y define cual de ellos es el mayor, es un buen ejercicio que se puede convertir en un SubVI.

Las siguientes imágenes describen el programa. Para conocer su funcionamiento, favor refiérase a la *Práctica 3, Estructuras.*





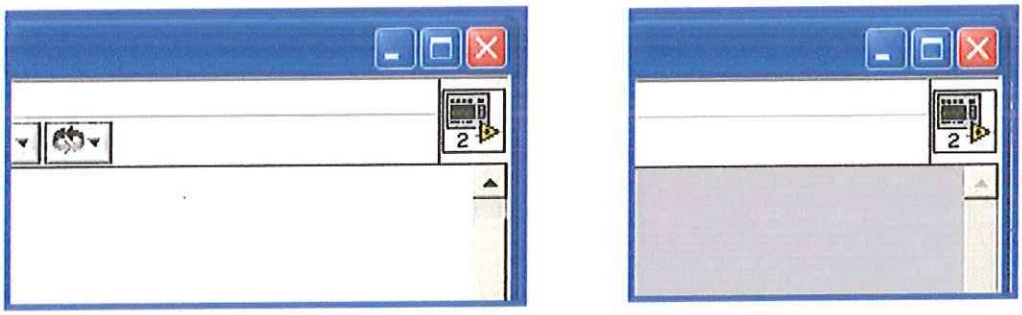
- **Parámetros que se deben tener en cuenta para usar un VI como SubVI.**

1) Debe definir cuales son las entradas y salidas del sistema.
2) Todas las entradas y salidas del sistema deben estar etiquetadas, de esta forma es más sencillo identificar los parámetros que lo componen.
3) El proyecto no debe tener errores de programación. En otras palabras se debe ejecutar correctamente.
4) El proyecto se debe guardar en una carpeta de acceso permanente y nunca se debe cambiar de puesto. Un SubVI en ningún momento reemplaza el funcionamiento de un programa, sólo es un acceso a él.

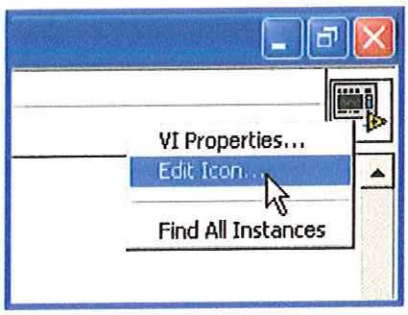
- **Pasos para configurar un SubVI.**

1) Imagen.

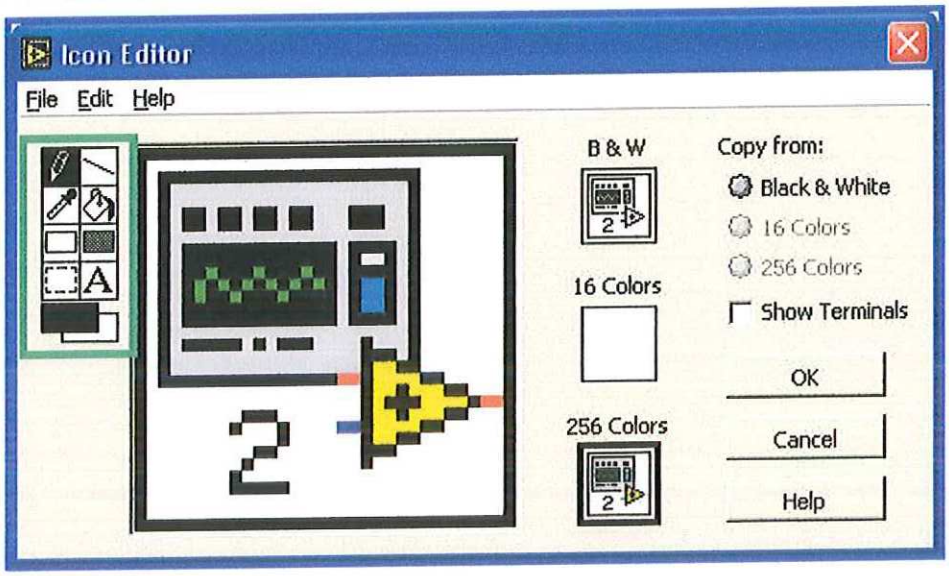
A cada programa creado en LabVIEW se le es asignado una gráfica o icono de identificación. Esta imagen se puede visualizar en la parte superior de la ventana Panel y Diagram.



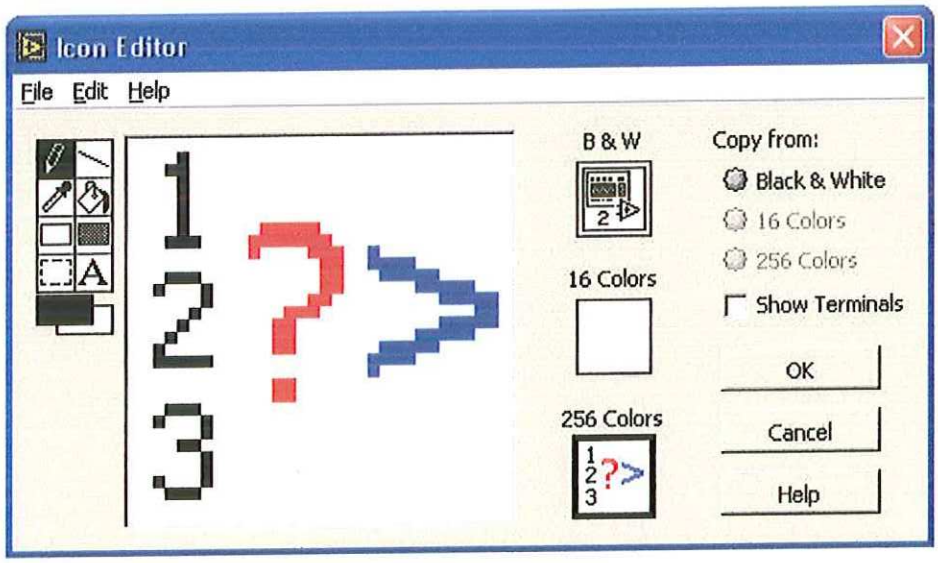
Se puede editar a gusto del usuario, para esto se da clic derecho sobre la imagen y se selecciona ***Edit Icon***.



Una nueva ventana aparece sobre el entorno de trabajo.

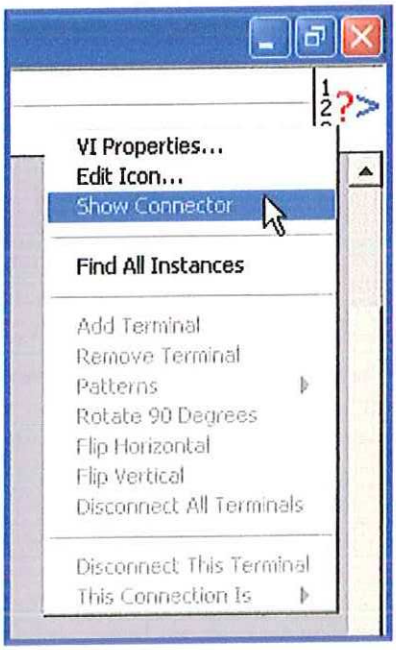


La imagen se amplifica y puede ser editada con las **herramientas disponibles**. El tratamiento de esta imagen es similar al de un mapa de bits trabajado desde Paint de Microsoft.

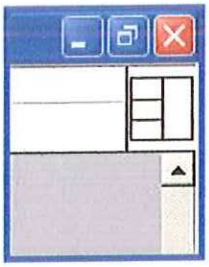


Hechos los cambios se da clic en ***OK***. La imagen sobre la ventana cambia.

Ahora desde Panel se da clic derecho sobre el icono y se selecciona ***Show Connector.***

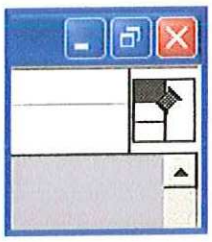


El icono cambia de figura. Y el mouse de forma, pasando al modo ***Connect Wire***.



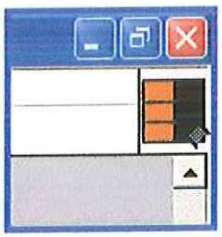
Con esta opción se enlazarán las salidas y las entradas del sistema con sus respectivos conectores. Las casillas ubicadas a la izquierda representan las entradas del sistema y las ubicadas a la derecha las salidas.

Ahora de clic sobre uno de los parámetros de entrada y posteriormente de clic sobre una de las casillas pequeñas.

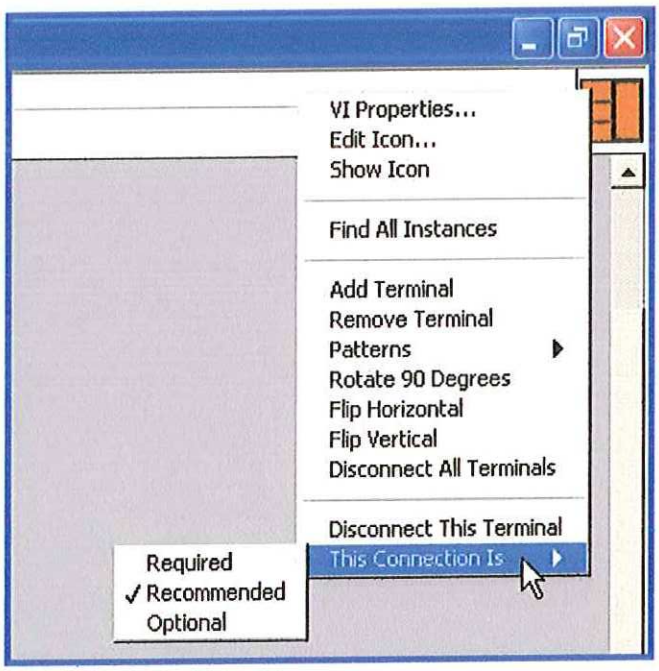


De esta forma la entrada Número 1 se conecta con la primera casilla.

Conecte las otras entradas y la salida en las casillas restantes.



Puede condicionar la conexión a una de las casillas dando clic derecho sobre ella y seleccionando, ***This Connection Is...***



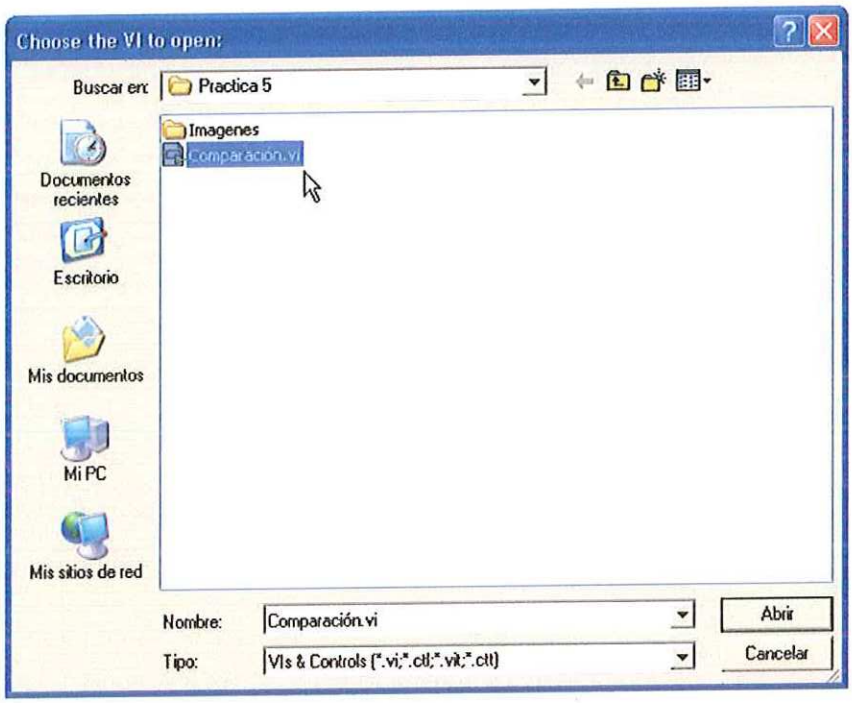
Establezca que todas las condiciones tengan seleccionado ***Recommended.***

En esta misma ventana desplegada existen mas opciones para la edición del Connector.

Para volver a ver el icono editado, de clic derecho sobre la imagen y seleccione ***Show Icon***.

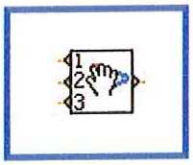
Guarde los cambios y cierre el programa. Después cree uno nuevo.

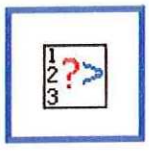
En este nuevo programa desde la ventana ***Diagram*** en el submenú ***Functions*** de clic sobre la casilla ***Select A VI***.



Una nueva ventana aparece, seleccione el programa de comparación que fue editado y de clic en ***Abrir***.

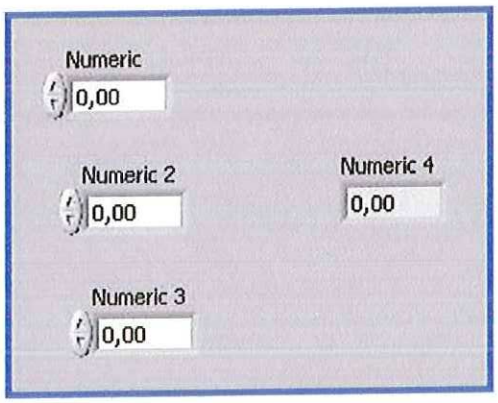
El puntero del Mouse toma la forma de la función, insértela sobre el espacio de trabajo.



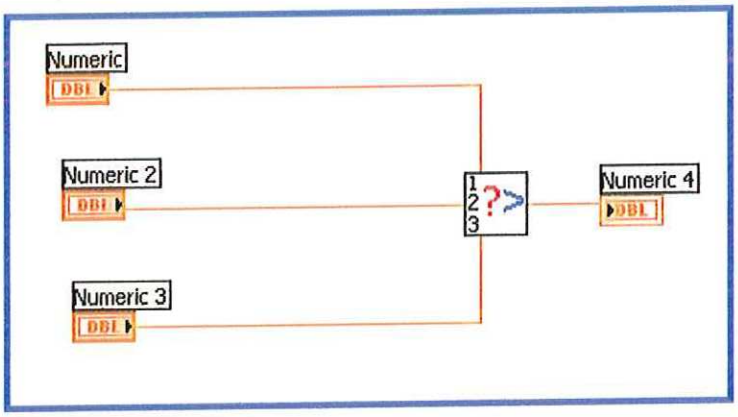


Al editar la función sobre el se pueden apreciar los conectores y las etiquetas con los nombres, de esta forma es muy fácil asociar los elementos requeridos.

Desde Panel Inserte las entradas y salidas necesarias para poder hacer trabajar la función.



Ahora conecte estos elementos sobre la función.



Pruebe los cambios y guarde el programa como ***Pruebadecomparacion***.vi.

5. ACTIVIDAD COMPLEMENTARIA

Nota: *todas las subVI aquí propuestas deben tener el formato de caja negra. (Como en el ejemplo del programa de Pruebadecomparación) En ningún momento debe requerir de funciones extras.*

- Del programa creado en la actividad complementaria pasada, el cual realizaba la suma de los componentes de una matriz. Transfórmelo en una SubVI y cree un nuevo programa en base a este donde se pueda sumar calcular la sumatoria a varias matrices.

- Diseñe una subVI para calcular el factorial y otra que calcule la sumatoria de un número. En un programa edite estas funciones para que realicen este trabajo sobre los componentes de una matriz. Procure tener cuidado con el formato de las variables a trabajar.

- Diseñe una SubVI que calcule el promedio de los valores contenidos dentro de un vector. Recuerde la expresión del promedio.

$$\Pr = \frac{\sum_{i=1}^{n} x_i}{n}$$

PRÁCTICA 17
GRÁFICAS

PRÁCTICA 17
GRÁFICAS

1. OBJETIVOS

- ✓ Generar gráficas a través de las herramientas que dispone LabVIEW para este propósito.
- ✓ Implementar las herramientas de graficación en proyectos .
- ✓ Implementar los conceptos aprendidos en la solución de problemas a través de LabVIEW.

2. CONCEPTOS FUNDAMENTALES

LabVIEW es un sistema SCADA, y como tal uno de sus propósitos es la de crear una interfaz con el usuario, la cual debe presentar información coherente y que sirva para tomar decisiones. Muchos de los datos provenientes del campo son generados por sensores, estos, son información que representa el estado de la temperatura, presión, caudal de un sistema, los cuales pueden presentar niveles críticos si se alcanza un valor específico.

Una herramienta de graficación permite al usuario obtener una información en tiempo real, cuyos datos se pueden comparar entre sí conociendo su cambio durante el tiempo, permitiendo así al operador desarrollar criterios de operación que le faciliten llegar a una conclusión. En casos menos críticos, una herramienta de graficación se puede usar para representar visualmente un gráfica de una ecuación o función matemática implementada en LabVIEW.

3. ACTIVIDAD PREVIA

Antes de proceder al desarrollo de esta práctica es necesario que elabore el siguiente cuestionario:

1) Enumere 3 ejemplos (nombrando la empresa donde proceden) de procesos donde se empleen sistemas SCADA para su control. Detalle que variables son analizadas y que tipo de sistema se usa para realizar el control.
2) ¿Qué elementos se necesitan para poder trazar una gráfica? ¿Qué softwares utiliza para este fin?

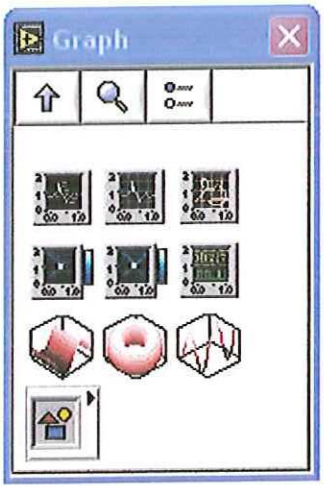
4. DESARROLLO METODOLÓGICO

Nota: *para el desarrollo de esta actividad es necesario haber elaborado la Práctica 0, 1 y 2.*

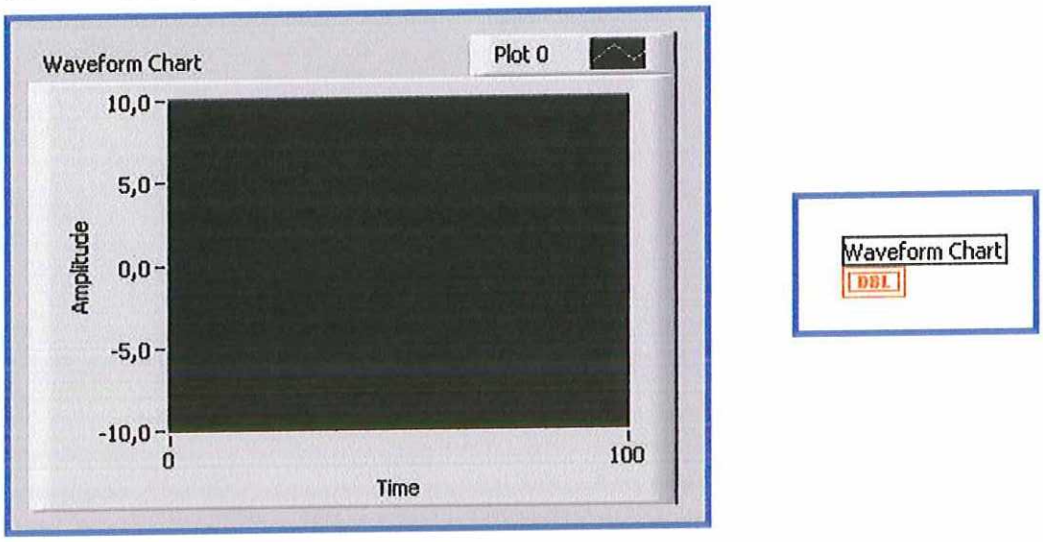
- **Creación de una gráfica simple.**

Se hace referencia a una gráfica simple cuando se desea indagar sobre el comportamiento de una sola variable en el tiempo.

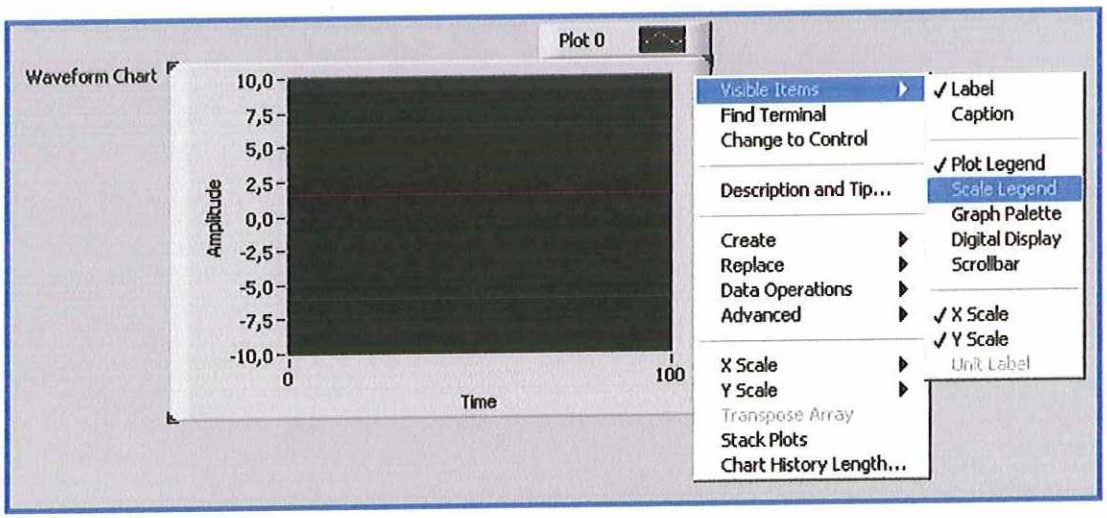
1) Para insertar una gráfica, en el submenú ***Controls*** seleccione la casilla ***Graph***. Graph despliega un menú con diversas opciones de graficado.



2) Seleccione ***Waveform Chart***, e insértela sobre el espacio de trabajo. Como su nombre lo indica *Waveform Chart* dibuja una forma de onda que varía en el tiempo, al dar clic derecho sobre la imagen se despliega un menú de edición.



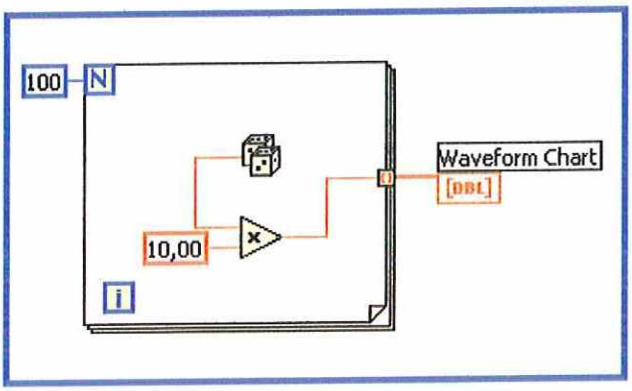
3) Seleccionando ***Visible Items***, puede agregar una serie de elementos que son de utilidad para corregir o mejorar la visualización de una gráfica. Observe los cambios que se dan cuando se selecciona una opción.

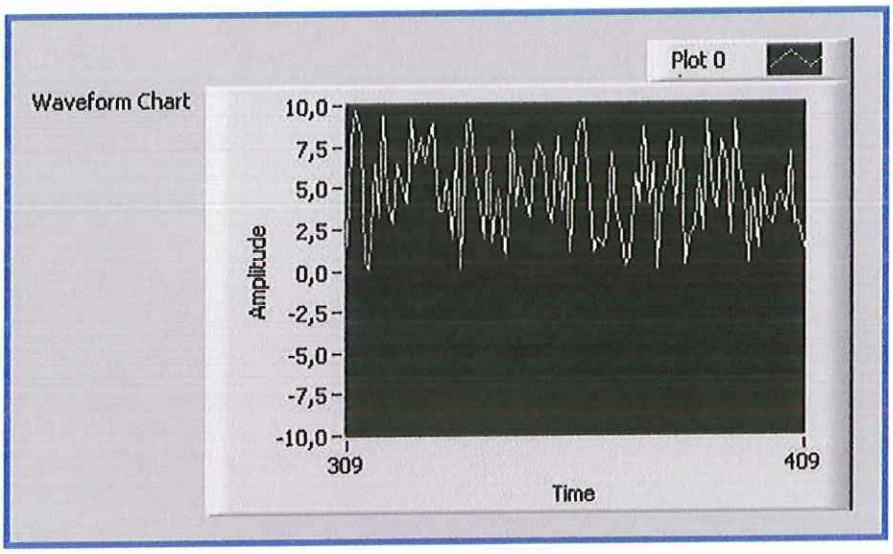


En esta misma ventana, puede configurar la escala de la gráfica. La opción ***AutoScale***, adapta la gráfica al tamaño más óptimo.

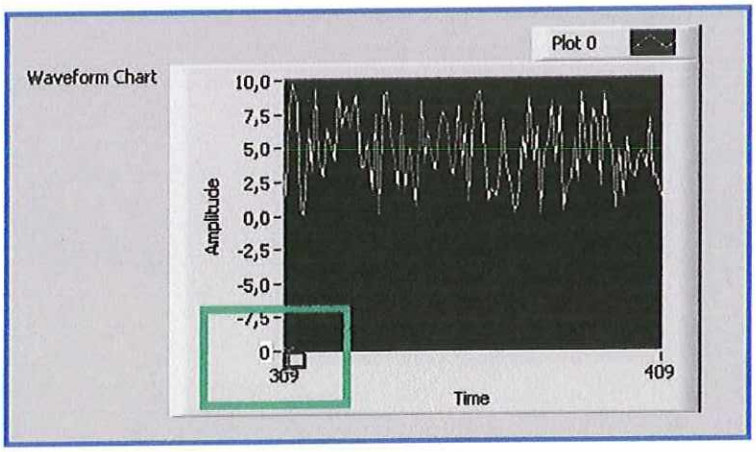
- **Utilización de Waveform Chart**

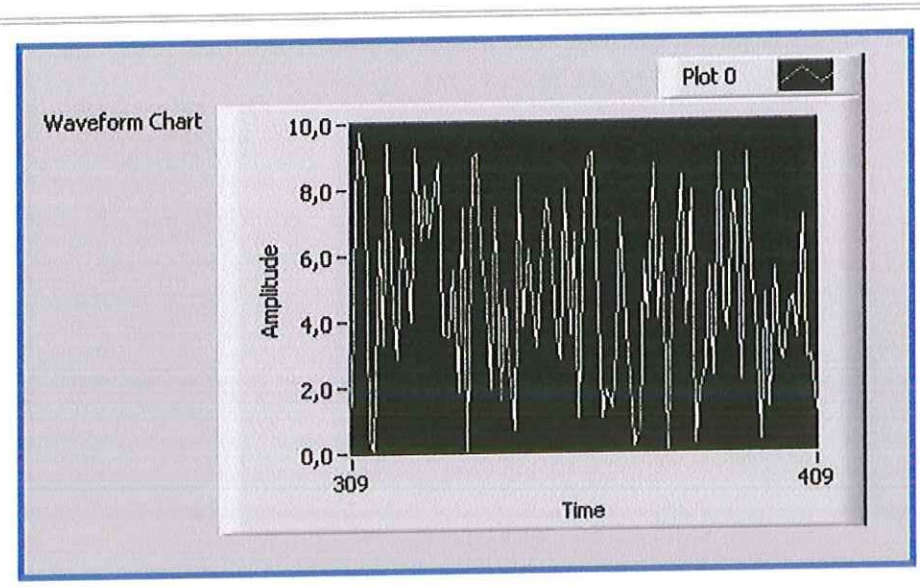
1) La siguiente gráfica describe una forma de utilización de ***Waveform Chart***.



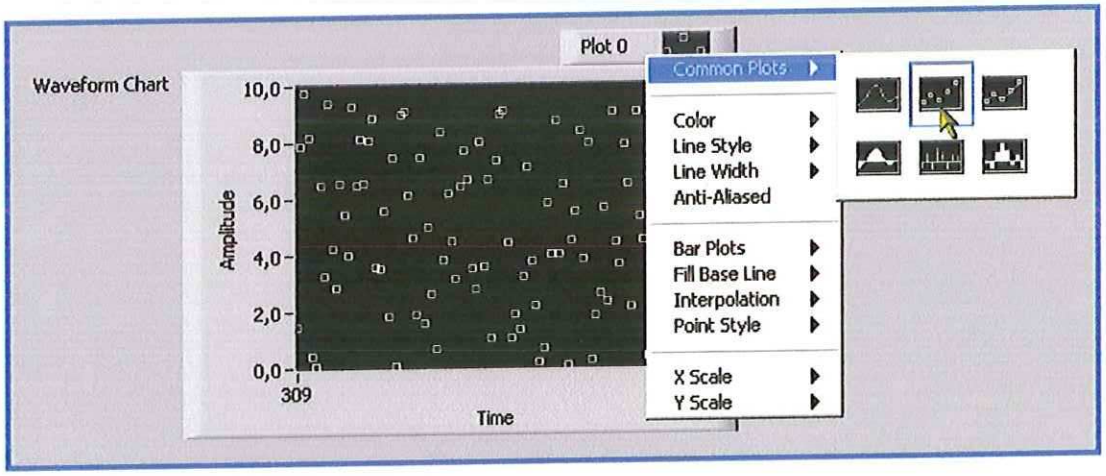


2) Con al herramienta de texto también se puede modificar la escala.

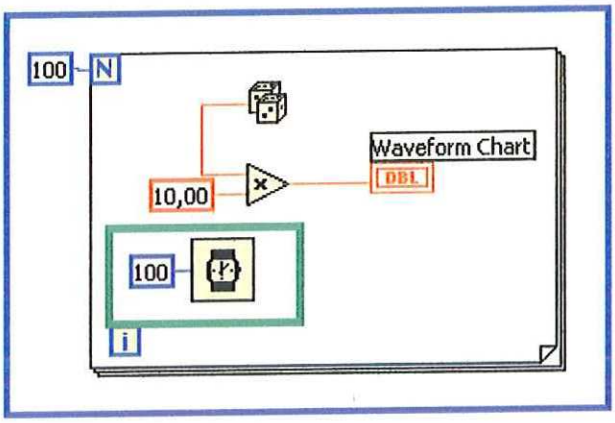




3) En la parte superior de la gráfica esta la opción ***Plot***, con ella, se puede configurar la gráfica para que genere diversas opciones de visualización, también se puede elegir el color o el estilo de la línea.

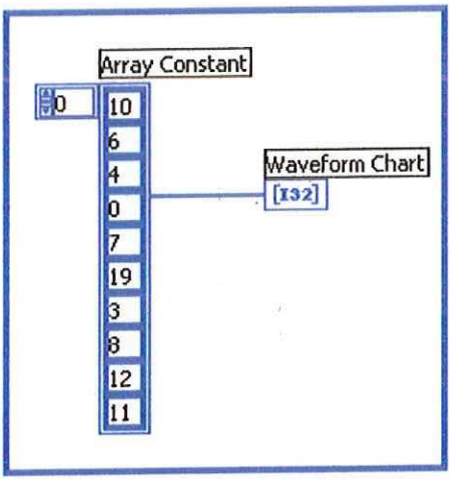


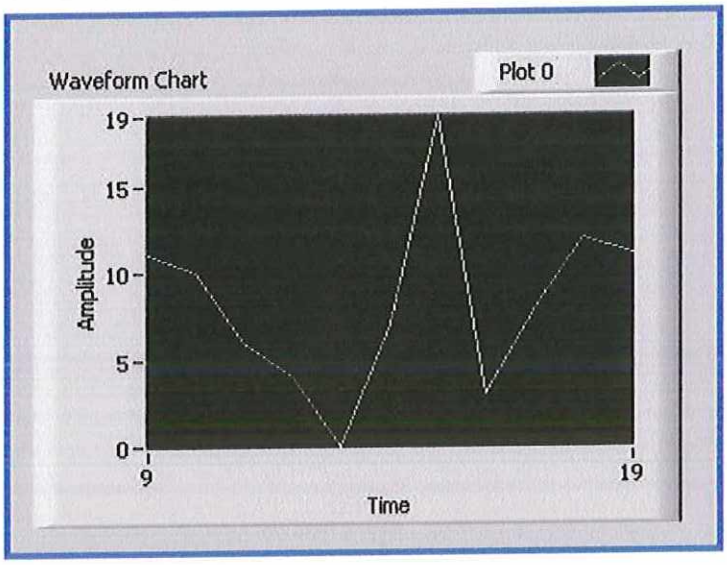
4) El siguiente diagrama muestra una forma diferente para la exhibición de las gráficas.



La nueva función utilizada se llama ***Whait*** (Espera), Whait genera un retardo entre ciclo y ciclo en un tiempo de x milisegundos, para el ejemplo 100 ms. Esta función de retardo se obtienen desde el submenú ***Functions***, la casilla ***Time & Dialog***. Realice el diagrama y observe el comportamiento.

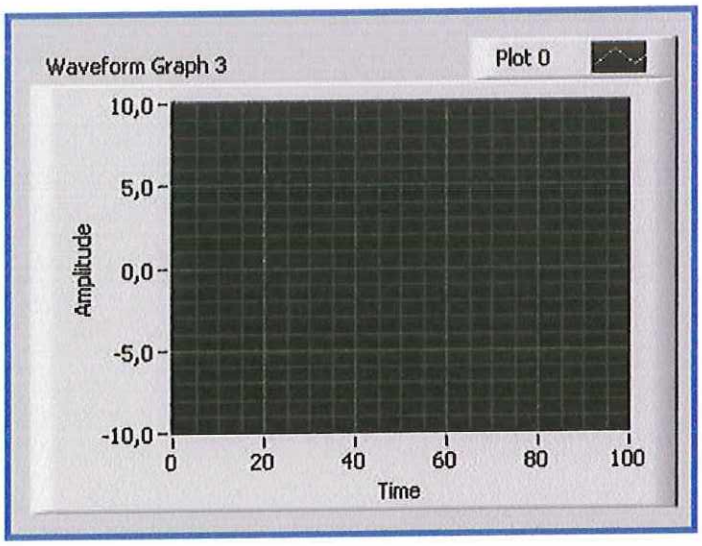
5) Waveform Chart también puede graficar datos contenidos en arrays. Un array se puede utilizar como un sistema de almacenamiento de datos, el resultado almacenado se puede después visualizar en una gráfica.

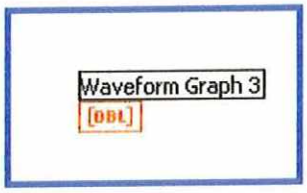




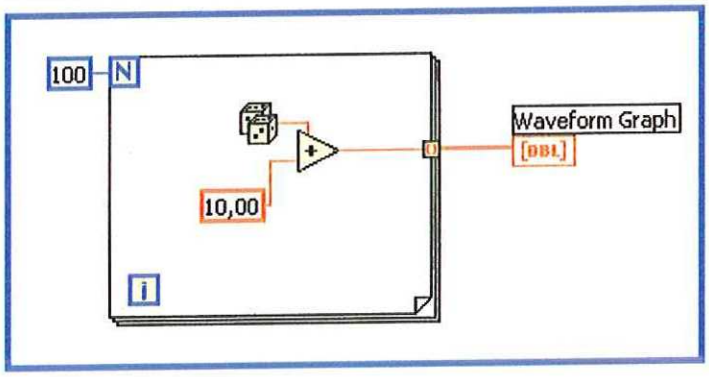
- **Waveform Graph**

Waveform Graph funciona igual que su contraparte *Waveform Chart*, tiene la ventaja de poder desplegar varias gráficas en un solo visor, además se ajusta automáticamente a los datos que estas despliegan. Sin embargo no tiene la capacidad de gráfica en tiempo real de *Waveform Chart*.

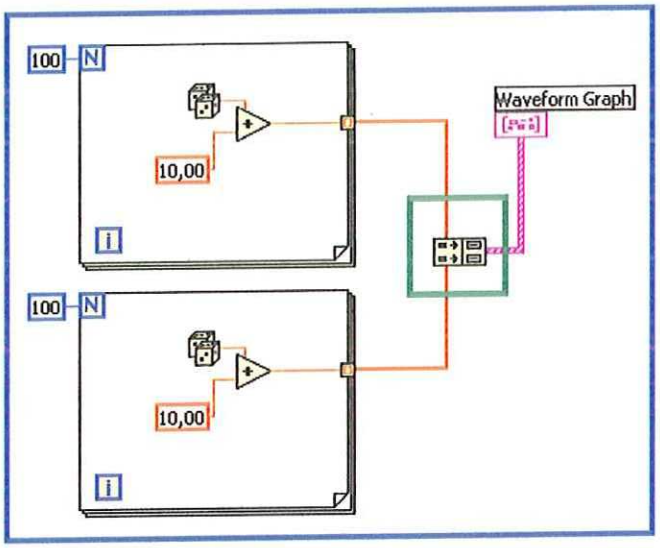


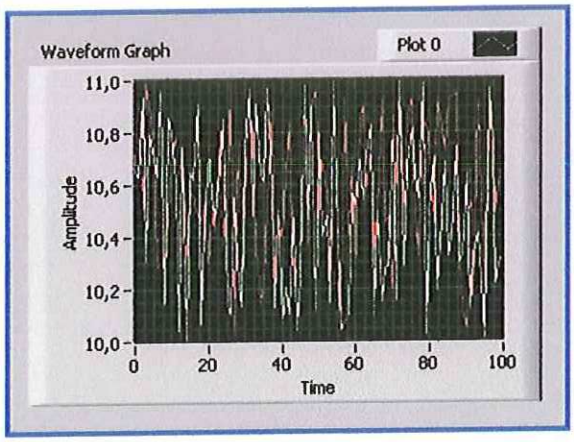


1) La siguiente gráfica describe una forma de utilización de *Waveform Graph.*



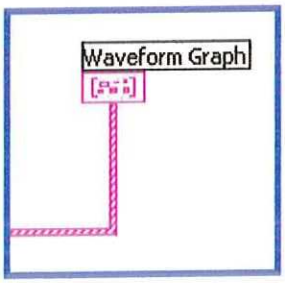
2) Cuando se quiere visualizar mas de una sola gráfica se utiliza un recurso llamado ***Cluster***. Un Cluster es un array que tiene la capacidad de almacenar datos de diferente tipo (en un solo cluster pueden guardarse valores booleanos y enteros), inclusive otros array.



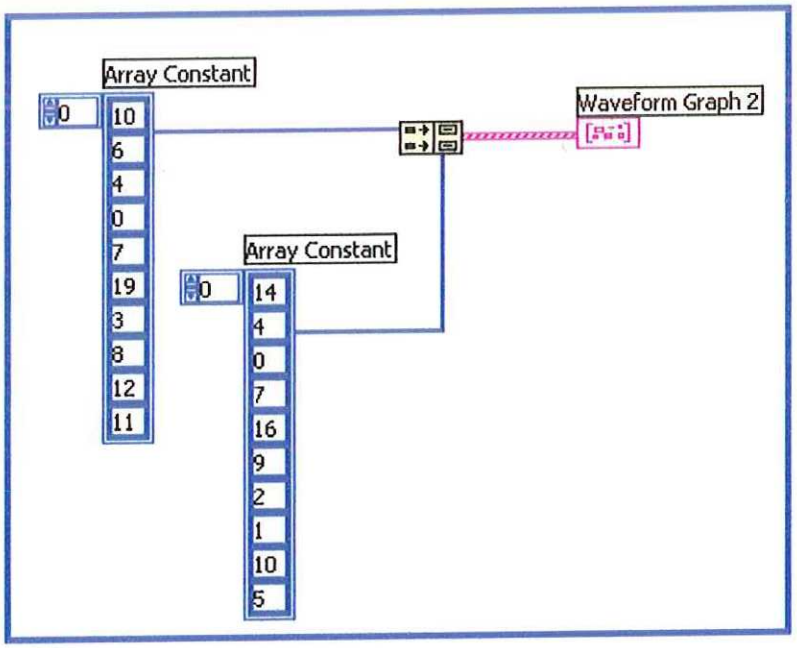


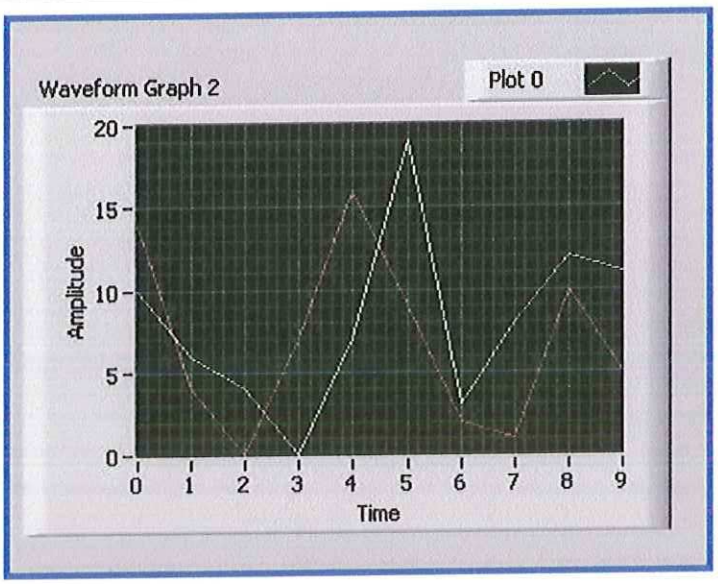
El objeto utilizado se denomina ***Build Cluster Array*** (Construir un Clúster de Arrays), el cual se obtiene del submenú ***Functions*** en la casilla ***Cluster***. Una vez insertado el objeto, modifique su tamaño con el puntero del Mouse para modificar su capacidad de almacenamiento.

Observe también la forma que tiene el conector, una línea gruesa la cual describe a las variables tipo Clúster.

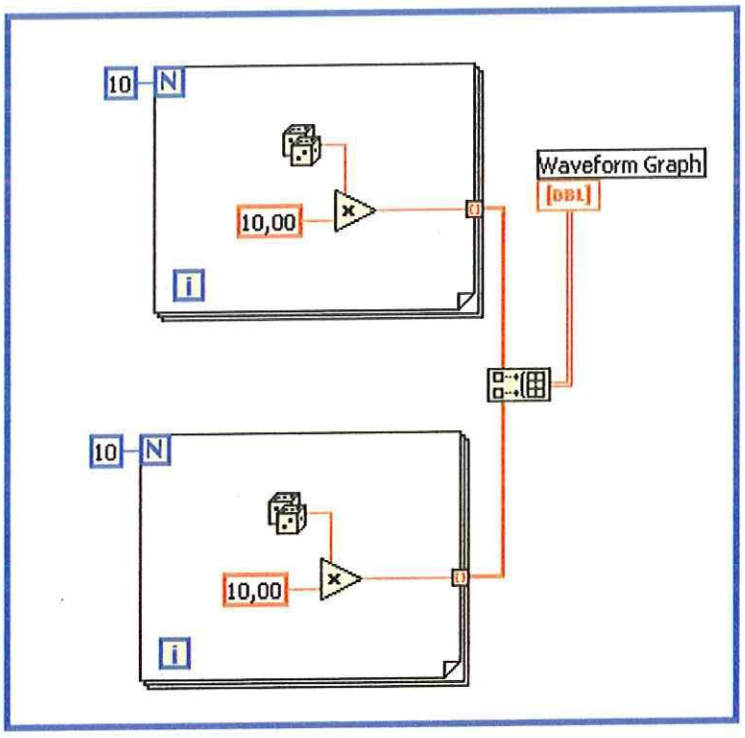


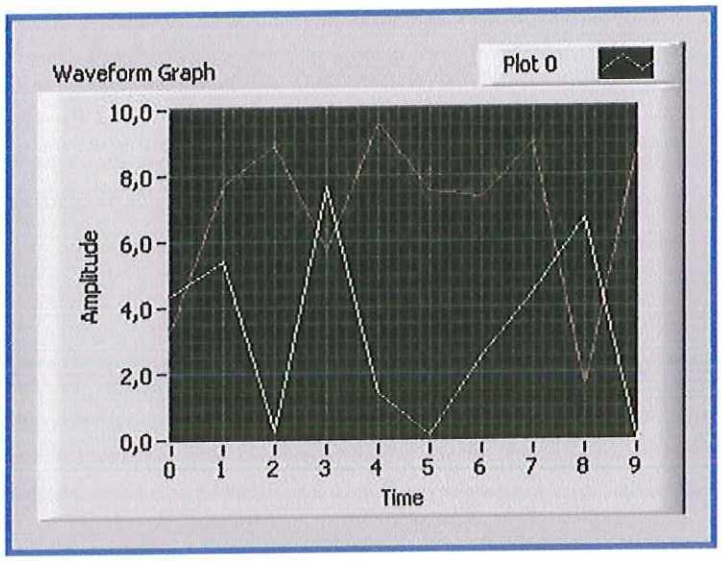
3) El siguiente diagrama grafica dos tablas independientes (vectores).





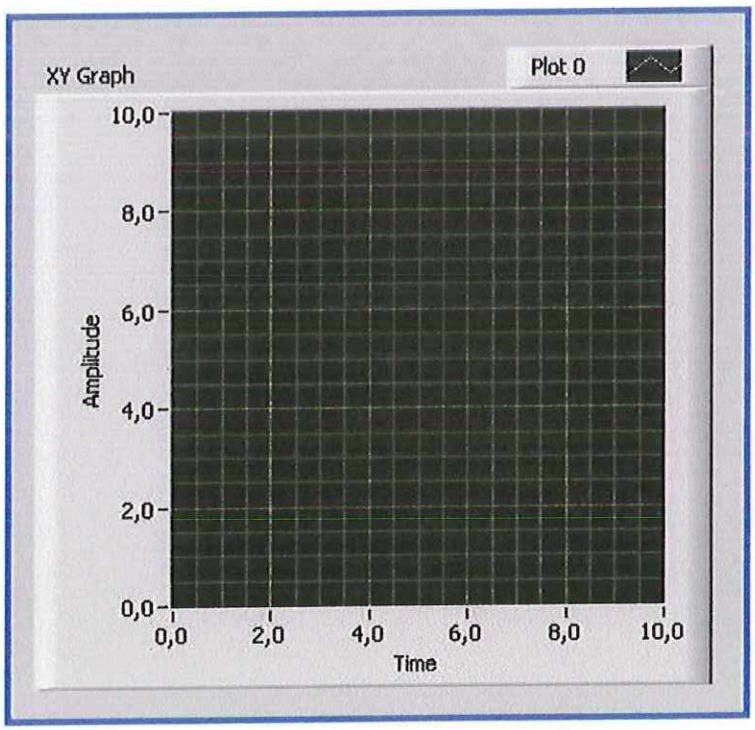
4) Otra forma de graficar varias imágenes es implementando la función ***Build Array*** (*Práctica 4*).



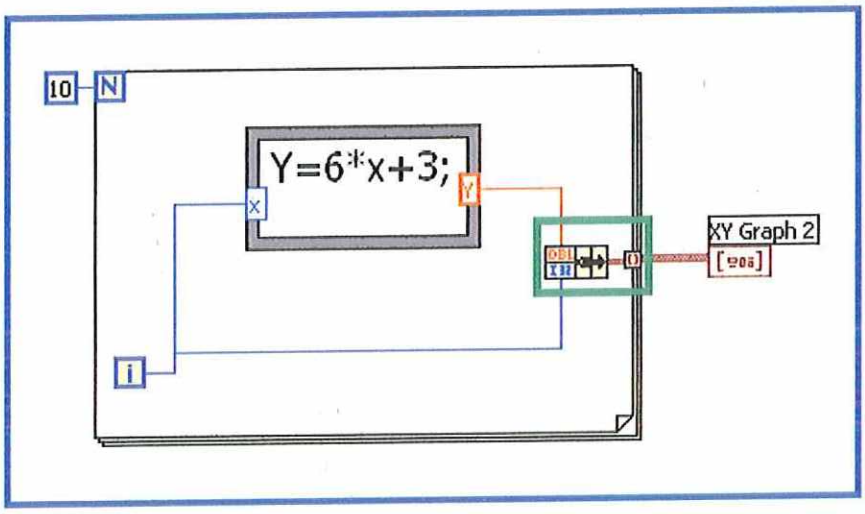


- **X Y Graph**

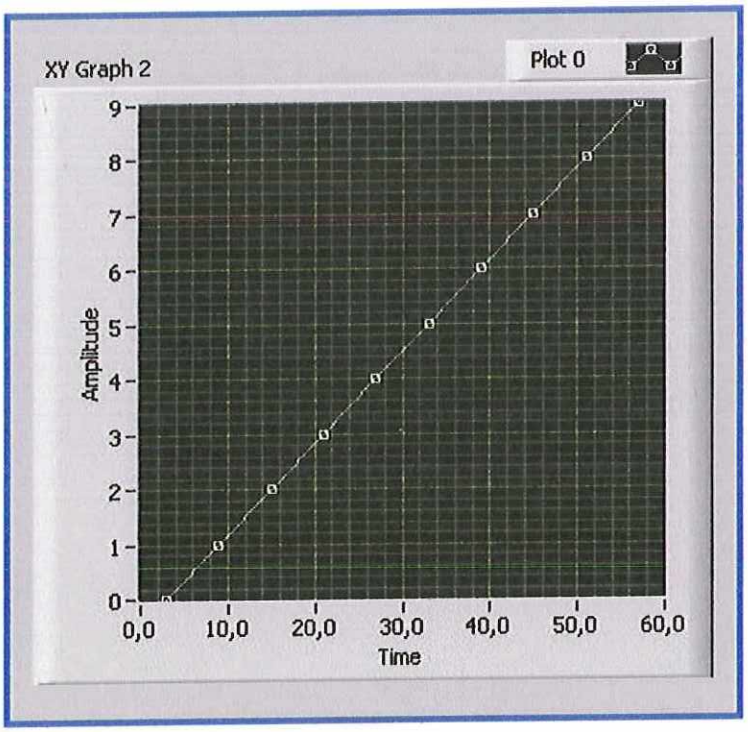
X Y Graph se utiliza para graficar una ecuación, donde existe una referencia entre dos variables.



El siguiente diagrama muestra un ejemplo de utilización de X Y Graph, implementado para este caso la estructura ***Formula Node*** y la función ***For***.



La función utilizada se denomina ***Bundle***, que fue extraída de la casilla ***Clusters***. *Bundle* almacena datos simples y los convierte en un array, luego transforma los array en variables tipo cluster, los cuales son los unicos que acepta X Y Graph.



5. ACTIVIDAD COMPLEMENTARIA

- Diseñe un programa que convierta una señal de grados centígrados a Fahrenheit y Kelvin. Los cambios en el flujo de temperatura se deben visualizar en una gráfica, una para cada escala.

- Dadas la ecuaciones:

$$y = 6 * sen(x)$$
$$z = 3 * \cos(x)$$
$$a = y + z$$

Diseñe un programa en LabVIEW para visualizar los resultados.

***Nota**: El rango de x va de 0 hasta 100 y su grado de precisión debe ser de 0.0, de esta forma la gráfica se puede visualizar correctamente.*

- Diseñe un programa en LabVIEW que dada una ecuación cuadrática:

$$y = ax^2 + bx + c$$

1) Pida los términos a, b y c y calcule sus raíces.

2) Grafique la ecuación.

***Nota**: El rango de x debe ser de –10 hasta 10. Con una precisión de 0.00. Implemente SubVI's en el desarrollo de este punto.*