

**UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO**

---

**FACULTAD DE ESTUDIOS SUPERIORES-CUAUTITLAN**

# **Fundamentos del PLC en la industria**

**TESIS**

Que para obtener el título de  
**Ingeniero Mecánico Electricista**

**P R E S E N T A**

Gustavo Rivero Hernández

**ASESOR DE TESIS**

M en I Ramon Osorio Galicia

**Cuautitlán Izcalli, Estado de México, 2018**



Universidad Nacional  
Autónoma de México



**UNAM – Dirección General de Bibliotecas**  
**Tesis Digitales**  
**Restricciones de uso**

**DERECHOS RESERVADOS ©**  
**PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL**

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.



UNIVERSIDAD NACIONAL  
AUTÓNOMA DE  
MÉXICO

**FACULTAD DE ESTUDIOS SUPERIORES CUAUTITLÁN  
UNIDAD DE ADMINISTRACIÓN ESCOLAR  
DEPARTAMENTO DE EXÁMENES PROFESIONALES**

U. N. A. M.  
FACULTAD DE ESTUDIOS  
SUPERIORES CUAUTITLÁN  
ASUNTO: VOTO APROBATORIO

**M. en C. JORGE ALFREDO CUÉLLAR ORDAZ  
DIRECTOR DE LA FES CUAUTITLÁN  
PRESENTE**

**ATN: LA. LAURA MARGARITA CORRALAR FIGUEROA**  
Jefa del Departamento de Exámenes  
Profesionales de la FES CUAUTITLÁN.

Con base en el Reglamento General de Exámenes, y la Dirección de la Facultad, nos permitimos comunicar a usted que revisamos **La Tesis:**

**"FUNDAMENTOS DEL PLC EN LA INDUSTRIA"**

Que presenta el pasante: **GUSTAVO RIVERO HERNÁNDEZ**  
Con número de cuenta: **40502356-1** para obtener el Título de: **Ingeniero Mecánico Electricista**

Considerando que dicho trabajo reúne los requisitos necesarios para ser discutido en el **EXAMEN PROFESIONAL** correspondiente, otorgamos nuestro **VOTO APROBATORIO**.

**ATENTAMENTE**  
**"POR MI RAZA HABLARA EL ESPÍRITU"**  
Cuautitlán Izcalli, Méx. a 01 de abril de 2018.

**PROFESORES QUE INTEGRAN EL JURADO**

	NOMBRE	FIRMA
<b>PRESIDENTE</b>	M. en I. Benjamín Contreras Santacruz	
<b>VOCAL</b>	M. en T.I. Jorge Buendía Gómez	
<b>SECRETARIO</b>	M. en I. Ramón Osorio Galicia	
<b>1er SUPLENTE</b>	M. en I. Ricardo Alberto Delgadillo Torres	
<b>2do SUPLENTE</b>	M. en C. Leopoldo Martín Del Campo Ramírez	

NOTA: Los sindicales suplentes están obligados a presentarse el día y hora del Examen Profesional (art. 127).  
En caso de que algún miembro del jurado no pueda asistir al examen profesional deberá dar aviso por anticipado al departamento.  
(Art 127 REP)

HHA/Vc

# **Agradecimientos**

**A mis padres María Julia Hernández Fragoso y Leobardo Rivero Martínez que sin escatimar esfuerzo alguno han dedicado gran parte de su vida para formarme, educarme y convertirme en persona de provecho. Por esto y más...Gracias.**

**A mis hermanos Raymundo Rivero Hernández, Janeth Rivero Hernández y Jesús Rivero Hernández por el apoyo incondicional recibido durante mis estudios.**

**A Tanya Gishel Coronel Zamora por la revisión de manera pedagógica que le realizo a mi trabajo de Tesis.**

**A todos los profesores y en especial a Leopoldo Martin del Campo Ramírez y Ramon Osorio Galicia que, con sus opiniones, aportaciones, juicios críticos, motivaciones y palabras de aliento contribuyeron para poder llevar a buen termino esta etapa de mi formación académica.**

# ÍNDICE

## AGRADECIMIENTO

## ÍNDICE

<b>MARCO TEORICO</b> .....	1
<b>INTRODUCCION</b> .....	2
<b>RESUMEN</b> .....	4
<b>OBJETIVO GENERAL</b> .....	5

## CAPÍTULO 1

### **Sistema de Control Programable**

1.1 Componentes del sistema de control programable.....	7
1.1.1 Sistema de programación.....	8
1.1.2 Red de comunicaciones.....	9
1.1.3 Controlador.....	10
1.1.4 Sistema de entradas y salidas.....	11
1.2 Sistema Logix 5000 y sus plataformas.....	12
1.2.1 ControlLogix.....	14
1.2.2 FlexLogix.....	16
1.2.3 CompactLogix.....	18
1.2.4 DriveLogix.....	19
1.2.5 SoftLogix.....	19

## CAPÍTULO 2

### **Nociones básicas de RSLogix 5000 y la lógica de escalera**

2.1 Elementos del software RSLogix5000.....	20
2.2 Que es un Proyecto en RSLogix5000.....	21
2.3 Definición de un TAG.....	22
2.4 Elementos de un proyecto.....	23
2.4.1 Definición de una tarea.....	23

2.4.2 Definición de un programa.....	25
2.4.3 Definición de una rutina.....	25
2.5 La lógica de escalera y sus elementos.....	26
2.5.1 Renglones.....	27
2.5.2 Instrucciones.....	27
2.5.3 Bifurcaciones paralela y anidada.....	29
2.5.4 Lógica de entrada.....	30
2.5.5 Lógica de salida.....	31
2.6 Temporizadores.....	32
2.7 Contadores.....	34
2.8 Documentación.....	35
2.9 Definición y creación de un comentario de escalón.....	36

## **CAPÍTULO 3**

### **Aplicación práctica utilizando un LOGO de SIEMENS**

3.1 Introducción a LOGO!8 y su conexión eléctrica.....	37
3.2 Desglosé del menú principal y los términos Básicos de LOGO!8.....	39
3.3 Funciones Generales de LOGO!8.....	42
3.3.1 AND y AND (Edge) con evaluación de flancos.....	42
3.3.2 NAND (AND negada) y AND (Edge) con evaluación de flancos.....	43
3.3.3 OR (O) y NOR (OR negada).....	44
3.3.4 XOR (OR exclusiva).....	45
3.3.5 NOT (inversor lógico).....	45
3.4 Funciones Especiales de LOGO!8.....	46
3.4.1 Temporizador ON-DELAY (retardo a la conexión).....	47
3.4.2 Relé Autoenclavador.....	49
3.4.3 Generador de impulsos asíncrono.....	50
3.5 LOGO!SoftComfort V8.0.....	51
3.5.1 Iniciar LOGO!SoftComfort V8.0.....	52

3.5.2 Programando mediante el software LOGO!SoftComfort V8.0.....	53
3.5.3 Simulación del circuito para el control de un portón industrial.....	59
3.5.4 Transferencia de un programa a LOGO!8.....	61
3.5.5 Practica 1 Semáforo Secuencial.....	63
3.5.5.1 Practica 2 Semáforo secuencial con retroalimentación.....	66
<b>CONCLUSIONES.....</b>	<b>69</b>
<b>BIBLIOGRAFIA.....</b>	<b>70</b>

## MARCO TEORICO

El ingeniero de hoy debe conocer bien las herramientas modernas que sean necesarias para lograr una alta producción con calidad. Una de las herramientas modernas de gran necesidad es sin duda el controlador lógico programable más conocido por sus siglas en inglés (**PLC, Programmable Logic Controller**) que en su estructura básica consta de:

- Módulos de entrada. Son tarjetas que obtienen las señales de campo mediante pulsadores, botones, sensores de presión, nivel, proximidad, temperatura, sensores inductivos, sensores capacitivos, entre otros.
- La unidad central del proceso. Contiene la memoria de solo lectura (**ROM, Read Only Memory**) y la memoria volátil (**RAM, Random Access Memory**).
- Tarjetas de salida. Permiten que el voltaje llegue a los dispositivos de salida como actuadores de mando, válvulas, motores etc.
- Fuente de alimentación. Dependiendo del tipo de PLC, se podrá conectar a una fuente de corriente alterna CA o a una fuente externa de corriente directa CD.
- El PLC maneja los siguientes lenguajes para su programación: lógica de escalera (**LD, Ladder Diagram**) basado en la representación lógica de contactos, Lista de instrucciones (**IL, Instruction List**) representado por expresiones nemotécnicas y lenguajes de bloque de funciones (**FBD, Function Block Diagram**) que se asemejan a las compuertas lógicas.

El PLC intercambia información mediante paquetes de software a través de una interfaz hombre-máquina (**HMI, Human Machine Interface**). Satisface las exigencias tanto de procesos continuos como discontinuos a través de entradas y salidas que pueden ser tanto analógicas como digitales. Regula presiones, temperaturas, niveles y caudales, así como todas las funciones asociadas de temporización, conteo y lógica. Sustituyen ampliamente a los relevadores eléctricos de contactos físicos, ya que tienen la desventaja de que sus contactos tienden a carbonizarse por el uso, lo que no ocurre con los contactos del PLC, un contacto es un elemento de entrada así lo lee el PLC. Las entradas se representan por medio de la letra **I**. Cuando un contacto se activa y éste se cierra (contacto normalmente abierto) este pasa de un estado lógico 0 a un estado lógico de 1. Cuando un contacto se activa y este se abre (contacto normalmente cerrado) este pasa de un estado lógico 1 a un estado lógico 0. Y la forma de programación está en función del software actual.



## INTRODUCCIÓN

La industria ha cambiado mucho con el paso del tiempo ya que las mismas necesidades del hombre así se lo han exigido y los grandes avances tecnológicos lo han provocado. Inicialmente el proceso de producción se llevaba manualmente casi en su totalidad a reserva de algunos instrumentos mecánicos, inicialmente había reguladores de vapor. Posteriormente con el surgimiento y desarrollo de la máquina de vapor a finales del siglo XIX, surgió la necesidad de desarrollar técnicas de medición, surgieron los primeros instrumentos de medición para indicar la presión de vapor en las calderas, con esto se pudo reducir la cantidad de accidentes que ocurrían debido a las frecuentes explosiones.

En un inicio la mayoría de las mediciones se realizaban por medio de los sentidos; si se requería de saber qué nivel tenía un tanque, era necesario trasladarse hasta éste y prácticamente asomarse a él para obtener el dato. Las regulaciones de las variables se realizaban manualmente, por ejemplo; la temperatura se regulaba por medio de válvulas manuales, las cuales eran controladas por los operadores que guiaban el proceso, quienes, por mera práctica, ya sabían cuántas vueltas había que girar el maneral de la válvula para elevar o disminuir a cierta temperatura la sustancia que se estaba calentando. Esto generaba muchos problemas como, por ejemplo: el producto terminado no siempre tenía las mismas características, ya que en cada cambio de operador cambiaban los criterios de control; existían muchos errores en las mediciones ya que los instrumentos y métodos de medición no eran lo suficientemente precisos. Para el año de 1938, el control automático era hecho básicamente con válvulas auto-accionadas. Con la evolución de los procesos industriales, fue necesario el control de fluidos donde las válvulas auto-accionadas no podrían ser aplicadas, y así empezaron los primeros instrumentos neumáticos de control de procesos que fueron cada vez más sofisticados, le permitían a un operador cuidar de varios procesos a la vez, disminuyendo el costo de la planta industrial.

Para el año de 1946 la universidad de Pensilvania presento el computador e integrador numérico electrónico (**ENIAC Electronic Numerical Integrator And Computer**), año en el que la humanidad se encontraba inmersa en uno de los conflictos más fatídicos de nuestra historia, la Segunda Guerra Mundial y como consecuencia de ello, la industria y otras áreas se favorecieron de los avances tecnológicos, por ejemplo el transistor desarrollado en 1947 que trajo consigo una nueva era electrónica, estos elementos de estado sólido son considerablemente pequeños, operan al instante que se les energiza y no requieren de precalentamiento, son mucho más económicos ya que la base de su fabricación es el silicio, y lo que es muy importante, consumen una mínima energía lo que los hace muy versátiles para usarse con baterías y darles aplicaciones portátiles, como por ejemplo los radios. Los instrumentos aplicados a la industria constaban básicamente de equipos neumáticos y electrónicos como transmisores, indicadores, registradores y tarjetas electrónicas que realizaban funciones específicas. Se desarrollaron procesos más complejos, donde se requería de decenas e incluso de cientos de mediciones y regulaciones de variables, control de motores, máquinas, etc.

Se vio la necesidad de centralizar todas las señales provenientes de las áreas de producción en una sala donde fueran monitoreadas por los operadores, a través de instrumentos montados en tableros. Así pues, las áreas en las que se montaban y se interconectaban las señales y alimentaciones de los instrumentos se dividieron en tres partes, al área de producción se le llamó **planta**, a la zona donde se interconectaban las señales y alimentaciones del área de producción se le llamó **cuarto de control** y al panel donde se montaban todos los instrumentos de medición **tablero**. Esto sólo fue logrado gracias a la etapa tecnológica que se vivía en aquel entonces.

A finales de 1958 se inventó el primer circuito integrado, con esto surgió la innovadora tecnología de los microchips o circuitos integrados que son pequeñas pastillas rectangulares de silicio de apenas algunos milímetros cuadrados de superficie, que en sus costados tienen varias terminales que están conectadas en el interior a miles de transistores. Algunos de estos circuitos trabajan con señales digitales aplicando el código binario que se basa en señales de dos estados, 1 y 0. Con este tipo de elementos era fácil, en un pequeño espacio, integrar una serie de circuitos electrónicos complejos en pequeñas tarjetas electrónicas.

En el año de 1960 la industria buscó una solución más eficiente para reemplazar los sistemas de control basados en circuitos eléctricos con relevadores, interruptores y otros componentes comúnmente utilizados para el control de los sistemas de lógica combinacional. En 1969 la empresa Bedford Associates propuso un sistema al que llamó controlador digital modular (MODICON, por sus siglas en inglés). Para 1971 los PLC's se extendían a otras industrias, por lo que nace el MODICON 084, primer PLC en ser producido comercialmente.

En 1980 se intentó estandarizar la comunicación entre PLC's con el protocolo de automatización de manufactura (**MAP**, **M**anufacturing **A**utomation **P**rotocol) de la empresa General Motors. En esos tiempos el tamaño del PLC se redujo, su programación se realizaba mediante computadoras personales en vez de terminales dedicadas sólo a ese propósito.

Y para el año de 1990 se introdujeron nuevos protocolos y se mejoraron algunos anteriores. El estándar IEC 1131-3 intentó combinar los lenguajes de programación de los PLC en un solo estándar internacional.

Actualmente se utilizan PLC's que contienen los circuitos necesarios para crear funciones lógicas. Su hardware interno se configura mediante puntos que se conectan y desconectan de manera electrónica en el circuito. Esta característica implica menos espacio para los tableros, menos consumo de energía, una mayor confiabilidad, menos inventario y una reducción en el costo total de fabricación. El PLC actual puede comunicarse con otros controladores y computadoras en redes de área local, y es una parte fundamental de los modernos sistemas de control distribuido. Por lo tanto, el técnico e ingeniero tendrán que familiarizarse con un controlador lógico programable aumentando de esta forma sus oportunidades de empleo con un nivel de paga satisfactorio.

## RESUMEN

Esta tesis presenta que es un PLC, explicando el sistema de control programable y el desglose de sus componentes, de forma teórica mediante el sistema Logix 5000 de Allen Bradley. Y de forma práctica mediante el mini PLC LOGO!8 de SIEMENS y su software de programación LOGO! SoftComfort V8.0.

En mi experiencia profesional, considero que, para tener un panorama actual del PLC, es necesario conocer las nociones básicas de Logix 5000 de Allen Bradley, así como sus plataformas (ControlLogix, FlexLogix, CompacLogix, DriveLogix y SofLogix). Mediante una explicación teórica de su hardware que engloba los componentes de cada plataforma, y de su software RSLogix5000 mediante los pasos a seguir para realizar un proyecto. Conocer el sistema Logix 5000 de Allen Bradley para el ingeniero debe de ser vista como una opción de estar actualizado e informado ya que este sistema es muy popular para la industria de la automatización en México.

Para fines prácticos a partir del capítulo 3 se utilizará el mini PLC LOGO!8 de la marca SIEMENS, también conocido como módulo lógico, se mostrará la forma de alambrado para este modelo y por ser una versión con display se visualizará el desglose del menú principal de LOGO!8, mediante la utilización de las teclas directamente desde el módulo lógico. Se retomarán las nociones básicas de lógica de escalera mencionadas en el capítulo 2, pero ahora serán vistas mediante bloques de funciones, como una manera de mostrar que lo único que cambia es el tipo de fabricante, pero la lógica o sus principios básicos son iguales o muy parecidos. Finalmente, para la programación se utilizará el software LOGO! SoftComfort V8.0 con el desarrollo de un programa de automatización para un portón industrial utilizando funciones básicas tales como OR, AND, NAND, NOT, XOR, y aplicando también funciones especiales como el generador de impulsos asíncrono, relé autoenclavador y temporizador retardo a la conexión. Al término de la programación se realizará una simulación y la transferencia del PC a LOGO!8. También se realizarán una práctica de un semáforo secuencial y otra más con realimentación que bien podría ser visto como el control secuencial de motores.

## **OBJETIVO GENERAL**

Entender que es un PLC, su relación con el sistema de control programable y adquirir las nociones básicas de programación en lenguaje gráfico: diagrama de escalera y diagrama de bloque de funciones.

## **OBJETIVOS**

- ✓ Saber que es un sistema de control programable y cuáles son sus componentes.
  
- ✓ Conocer el sistema Logix 5000 de Allen Bradley.
  
- ✓ Conocer y saber utilizar de forma básica el mini PLC LOGO!8 de SIEMENS y su software de programación LOGO! SoftComfort V8.0.

# CAPITULO 1 Sistema de Control programable

## Introducción

La comprensión de los componentes y las funciones comunes a todos los sistemas de control programables provee la información necesaria para la configuración de un sistema de control programable.

Un método de controlar máquinas o procesos en una planta consiste en utilizar relés electromecánicos. El relé o relevador funciona como un interruptor controlado por un circuito eléctrico en el que, por medio de un electroimán, se acciona un juego de uno o varios contactos que permiten abrir o cerrar otros circuitos eléctricos independientes. Cuando controlan grandes potencias se llaman contactores en lugar de relés.

Los sistemas de control programables se desarrollaron como alternativa de los sistemas de relés en ciertas situaciones. Estos sistemas se pueden programar para controlar automáticamente procesos, máquinas y dispositivos. Un sistema de control programable es básicamente un circuito electrónico que almacena un programa informático capaz de controlar el funcionamiento de algún dispositivo actuador, en función de la información recibida a través de los sensores que tenga conectados.

Los sistemas de control programables son flexibles y fáciles de mantener: Los cambios a los requisitos de producción solamente requieren modificaciones realizadas a un archivo de proyecto. Estos cambios de programación son introducidos por el personal de mantenimiento o por los ingenieros de la planta utilizando un lenguaje de programación estándar que es similar al de diagramas eléctricos.

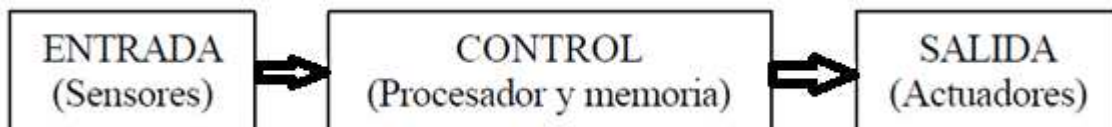


Figura 1 Diagrama del flujo de información en un sistema de control programable

1. El sistema monitorea la información de **entrada** proveniente de un proceso, una máquina o un dispositivo.
2. Un **controlador** evalúa la información de entrada en base a un conjunto dado de normas y luego genera la información de salida.
3. La información de **salida** se utiliza para controlar el proceso o la máquina.

## 1.1 Componentes del Sistema de Control Programable

Un sistema de control programable tiene cuatro componentes principales ver figura 1.1:

- Sistema de programación
- Red de comunicaciones
- Controlador
- Sistema de E/S (entrada/salida)

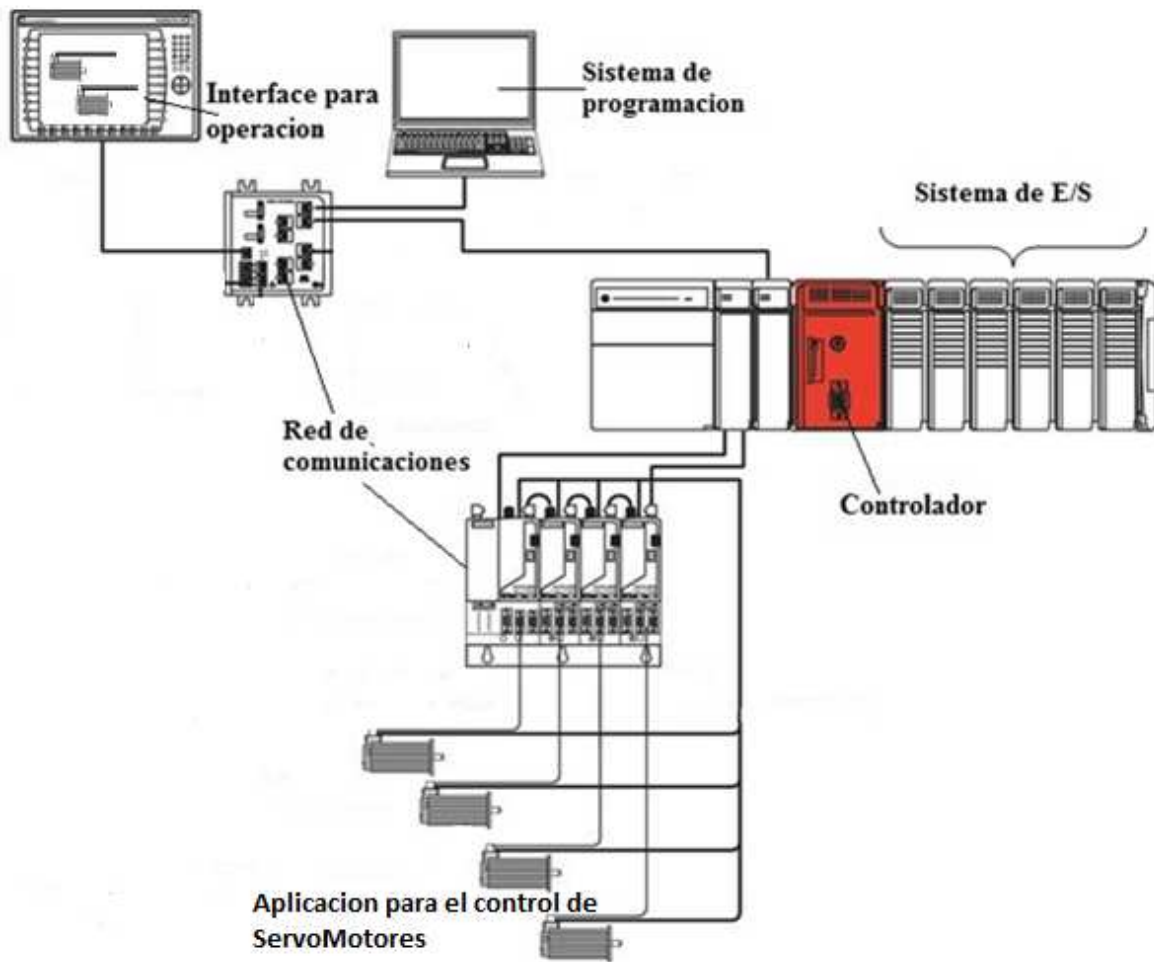


Figura 1.1 Componentes del sistema de control programable<sup>1</sup>

<sup>1</sup>Figura 1.1 obtenida de la URL ([http://sappi.ipn.mx/cgpi/archivos\\_anexo/20082329\\_5942.pdf](http://sappi.ipn.mx/cgpi/archivos_anexo/20082329_5942.pdf))

### 1.1.1 Sistema de Programación

Un sistema es un conjunto de componentes que interactúan entre sí para lograr un objetivo en común. Las personas se comunican con el lenguaje que es un sistema muy desarrollado formado por palabras y símbolos que tienen un significado. Lo mismo es para un controlador lógico programable utiliza un sistema de programación, compuesto por un software instalado en una computadora personal, computadora portátil, o estación de trabajo. Aunque no hay un lenguaje de programación común para todas las marcas de PLC's que existen en el mercado. La comisión electrotécnica internacional (IEC por sus siglas en inglés) desarrollo el estándar internacional IEC 61131 para Controladores Lógicos Programables (PLC), el IEC 61131-3 es la tercera parte (de 8) y fue publicada por primera vez en diciembre de 1993. La edición actual fue publicada en febrero del 2013. Esta parte trata los lenguajes de programación y define los estándares de dos lenguajes gráficos y dos lenguajes textuales para PLC.

Textuales:

**Lista de Instrucciones (IL, Instruction List);** Tiene su origen en Europa y utiliza un estilo muy similar al empleado por los lenguajes tipo ensamblador aplicados en la informática. Este tipo de lenguaje es una transcripción elemental e inmediata de las instrucciones del lenguaje máquina, las cuales están representadas por expresiones nemotécnicas.

**Texto estructurado(ST, Structured Text);**Facilita la programación de procesos que requieren instrucciones complejas y cálculos muy grandes por lo tanto es ideal para aplicaciones matemáticas (matriz compleja) o para programas convertidos desde otros proyectos creados en lenguajes de alto nivel como BASIC, PASCAL, FORTRAN, etc. De ellos, el lenguaje BASIC, convenientemente adaptado a las aplicaciones del autómatas, es el lenguaje de alto nivel más extendido debido a la facilidad de manejo. Este tipo de lenguaje permite resolver tareas de cálculo científico de alta resolución, clasificación de datos, estadísticas, etc., con mucha facilidad, y con acceso a módulos y subrutinas específicas ya escritas en estos lenguajes y de uso general en aplicaciones informáticas.

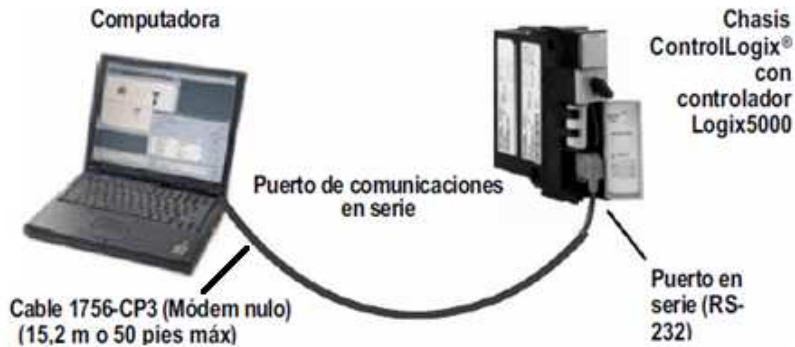
Grafico:

**Diagrama de escalera (LD, Ladder Diagram)** Tiene sus raíces en Estados Unidos. Está basado en la representación lógica de contactos. Fue el primero con el que se comenzó a programar, de ahí que presente grandes semejanzas con los diagramas eléctricos de escalera utilizados por los técnicos e ingenieros. Este lenguaje esta especialmente indicado para facilitar el cambio de un sistema de control realizado con relés por un autómatas programable.

**Diagrama de Bloques de Funciones (FBD, Function Block Diagram)** Es similar a diagramas de circuitos de electrónica. Su uso es común en algoritmos de control en procesos industriales y es ideal para aplicaciones de movimiento que siguen un proceso paso a paso.

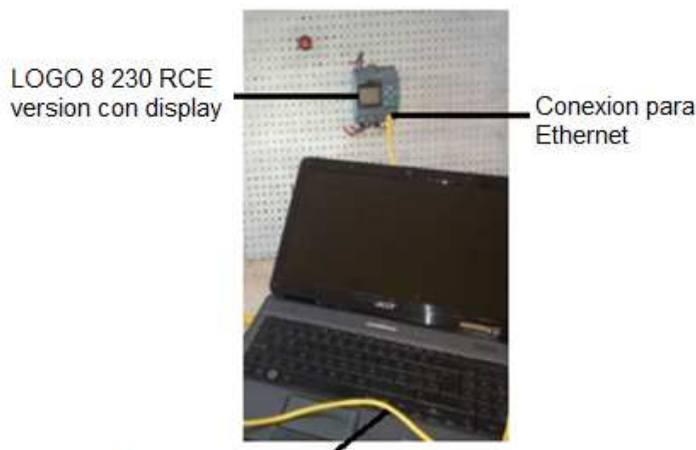
### 1.1.2 Red de comunicaciones

Una red de comunicaciones consiste en la conexión física entre una serie de componentes o dispositivos. Esta conexión se utiliza para transferir datos entre los componentes, como por ejemplo una computadora y un controlador utilizando un sistema de cables; conocida como conexión en serie ver figura 1.2:



El **módem nulo** es un método para conectar dos terminales usando un cable serie RS-232.

**Ilustración A**



Cable Ethernet es uno de los tipos de cable de red más comúnmente utilizados en redes cableadas. Los cables Ethernet conectan los dispositivos entre sí dentro de una red de área local

**Ilustración B**

Figura 1.2 Conexión en serie controlador Logix 5000 Ilustración A. Conexión en serie controlador LOGO!8 230 RCE (versión con display) de SIEMENS se aprecia en la ilustración B.



### 1.1.3 Controlador

Un controlador es el cerebro de un sistema de control programable. Consiste en un dispositivo en estado sólido, similar a una computadora, con memoria programable por el usuario y un procesador central (ver figura 1.3).



Ilustración A

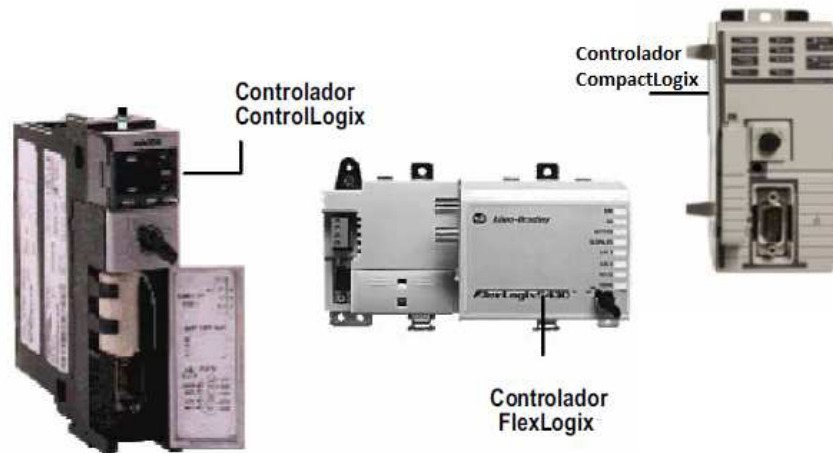


Ilustración B

Figura 1.3 Controladores Siemens ilustración A y controladores Allen Bradley ilustración B<sup>2</sup>

<sup>2</sup> Figura 1.3 obtenida de la URL (<https://www.siemens.com.mx/cms/mam/industry/Automatizacion/SIMATIC-sistemas-de-automatizacion-industrial/Pages/SIMATIC-sistemas-de-automatizacion-industrial.aspx>)

### 1.1.4 Sistema de entradas y salidas

Es la parte del controlador que interactúa con el medio externo. Lo hace a través de módulos de entradas y salidas (ver figura 1.4) cuyo objetivo básico es la transmisión de datos, mediante el envío de información de entrada y salida entre un controlador y un proceso/una máquina.

Un sistema de E/S consta de los siguientes componentes: módulos de E/S, que son parte del sistema de control programable, dispositivos de E/S, que son parte del proceso de la máquina. Un dispositivo de entrada es por ejemplo un sensor, que provee señales a un módulo de entrada. Un dispositivo de salida es el que se activa o energiza mediante un controlador por ejemplo un cilindro actuador.

Configurar módulos de E/S permite que el controlador envíe y reciba datos desde el proceso/máquina este tipo de módulos convierten señales analógicas a valores digitales para las entradas y convierten valores digitales a señales analógicas para las salidas. Sin módulos de E/S adecuadamente configurados, el controlador no se puede comunicar con el proceso/máquina.



Figura 1.4 Módulos de entradas y salidas<sup>3</sup>

---

<sup>3</sup>Figura 1.4 obtenida de la URL

([http://ab.rockwellautomation.com/resources/images/allenbradley/gl/medlrgprod/1746\\_SLCIOAnalogFamily\\_front1-large\\_312w255h.jpg](http://ab.rockwellautomation.com/resources/images/allenbradley/gl/medlrgprod/1746_SLCIOAnalogFamily_front1-large_312w255h.jpg))

## 1.2 Sistema Logix 5000y sus plataformas

Logix 5000 es un sistema operativo que utiliza el software de RSLogix 5000; este sistema tiene una gran variedad de aplicaciones, por ejemplo: control secuencial, control de movimiento, control de variadores, control de proceso o control de seguridad. La figura 1.5 muestra el desglose de la familia de PLC Allen Bradley.

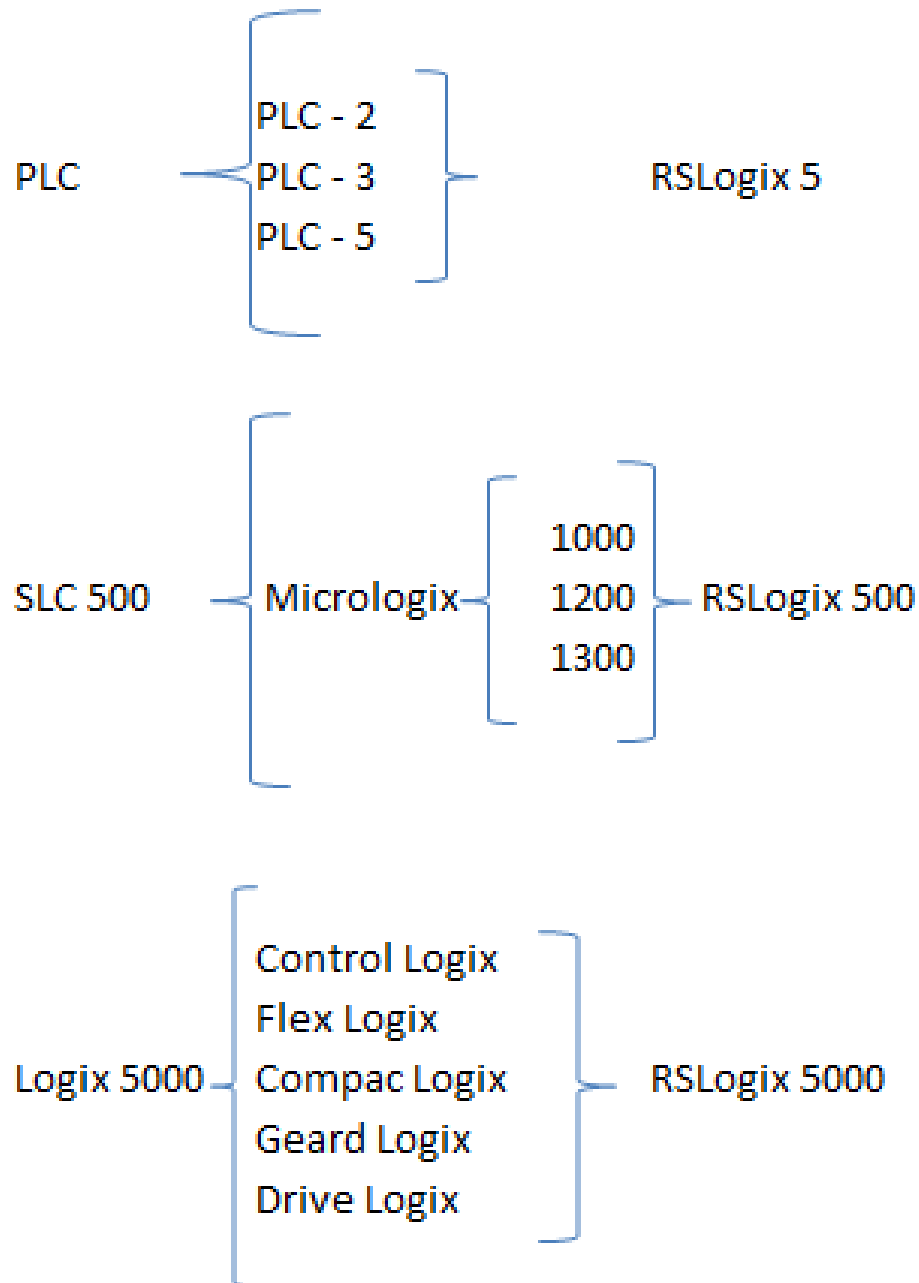


Figura 1.5 Cuadro sinóptico de la familia PLC Allen Bradley

La familia de Logix 5000 abarca lo que son los PLC; ControlLogix, CompactLogix; estos dispositivos son los más nuevos que ha sacado Allen Bradley para la industria de la automatización (ver figura 1.6).

**ControlLogix:** Sistema de controladores múltiples de alto rendimiento en un formato de chasis modular.

**CompactLogix:** Sistema modular pequeño para conectar sistemas de tamaños medianos.

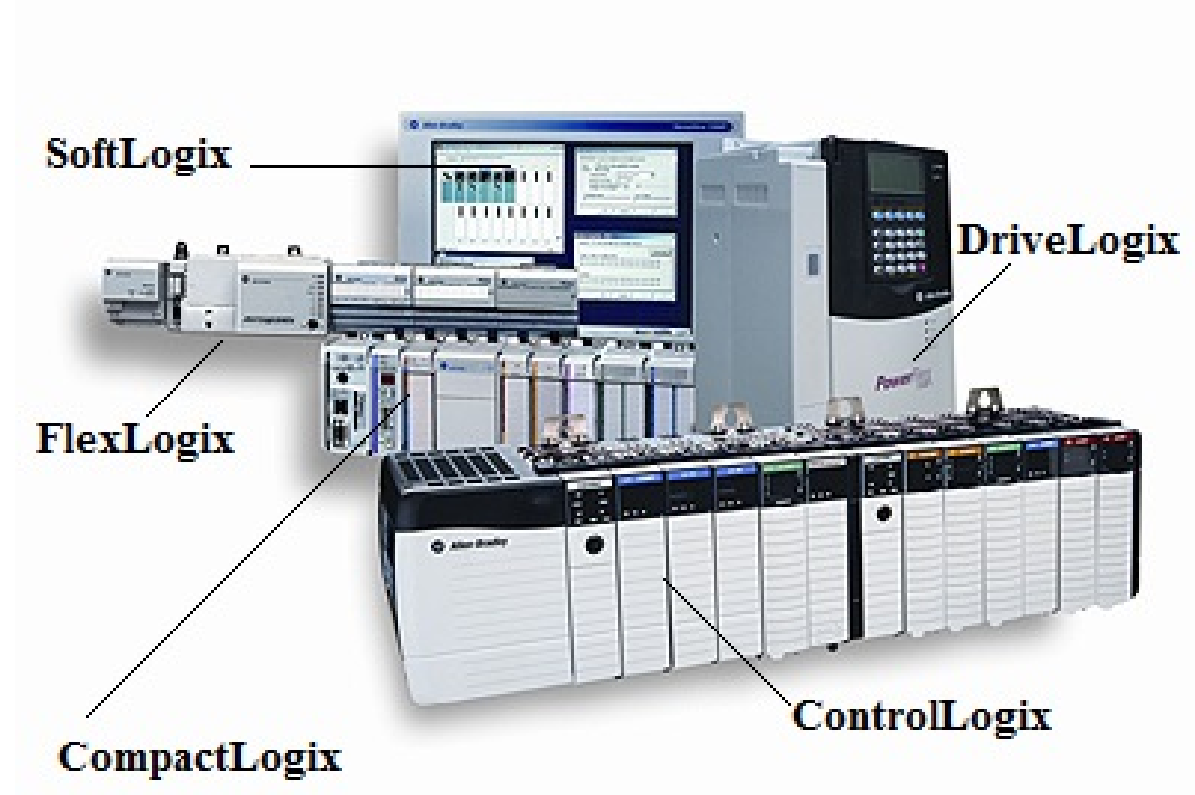


Figura 1.6 PLC's del sistema Logix 5000<sup>4</sup>

**DriveLogix:** Sistema para control de variadores distribuidos.

**FlexLogix:** Sistema para el control distribuido (es decir, la plataforma de control se encuentra ubicada en o cerca del proceso/la máquina).

**SoftLogix:** Sistema que combina control, información, y visualización en un sistema de control abierto (por ejemplo, el control de un motor se encuentra alojado en una computadora u otra terminal).

---

<sup>4</sup>Figura 1.6 obtenida de la URL ( <http://www.srcenergy.com.br/#!SRC-Energy-abre-inscrições-para-Treinamento-de-PLC-Rockwell/ch7/364F992E-20DE-44A7-A1B6-315BA68A4B50> )

## 1.2.1 ControlLogix

Es un PLC de última generación conformado por un sistema de controladores múltiples de alto rendimiento en un formato de chasis modular. El chasis es un ensamble de hardware que aloja dispositivos tales como controladores, módulos de E/S, módulos de interface de red y la fuente de alimentación (ver figura 1.7).

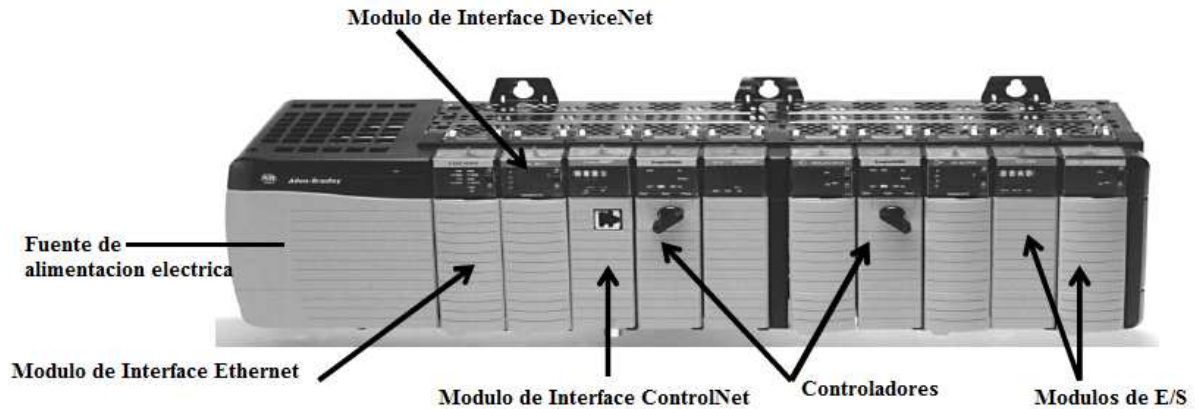


Figura 1.7 Aspecto físico del chasis modular controlLogix<sup>5</sup>

Uno de los aspectos más importantes de los PLC's de última generación es que están formados por ranuras o módulos individuales. Se encuentran disponibles en tamaños de 4, 7, 10, 13 y 17 ranuras. Las ranuras se numeran de izquierda a derecha comenzando con 0 (Ver figura 1.8).

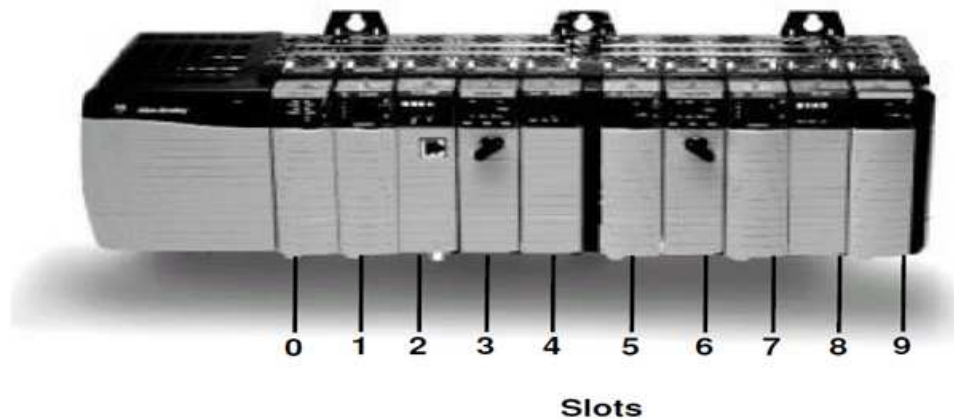


Figura 1.8 Numeración correcta de ranuras en el chasis de controlLogix<sup>6</sup>

<sup>5</sup>Figura 1.7 obtenida de la URL (<http://www.ab.com/en/epub/catalogs/12762/2181376/104830/2746667/print.html>)

<sup>6</sup>Figura 1.8 obtenida de la URL (<http://allenbradleyplctutorials.blogspot.mx/2013/06/controllogix-5000-system-components.html>)

ControlLogix cuenta con un **backplane** ControlBus; es una tarjeta de circuitos impresos en la parte posterior del chasis (ver figura 1.9), que provee una interconexión eléctrica entre módulos: el backplane permite multidifundir datos, es decir, un módulo de entrada envía datos una vez recibidos por los controladores múltiples simultáneamente.

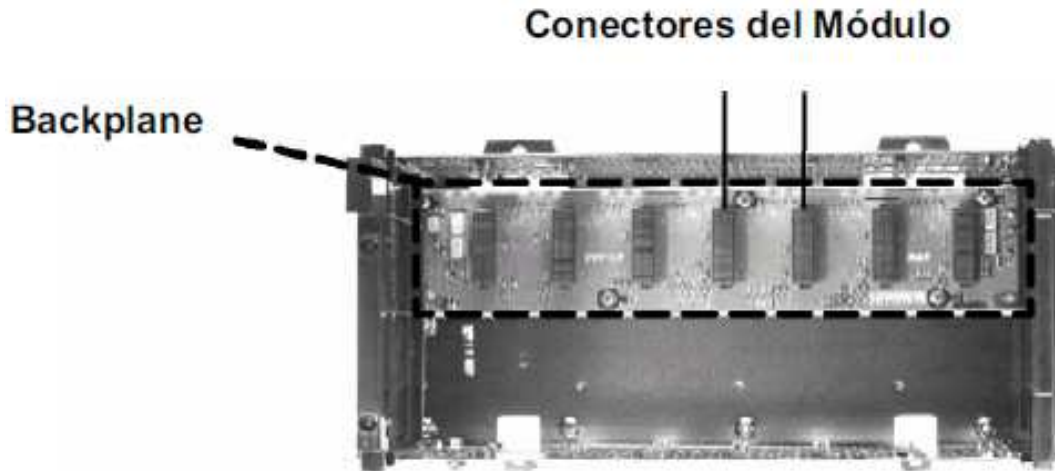


Figura 1.9 Backplane tarjeta de circuitos impresos parte posterior del chasis de ControlLogix<sup>7</sup>

ControlLogix utiliza el siguiente controlador (ver figura 1.10).

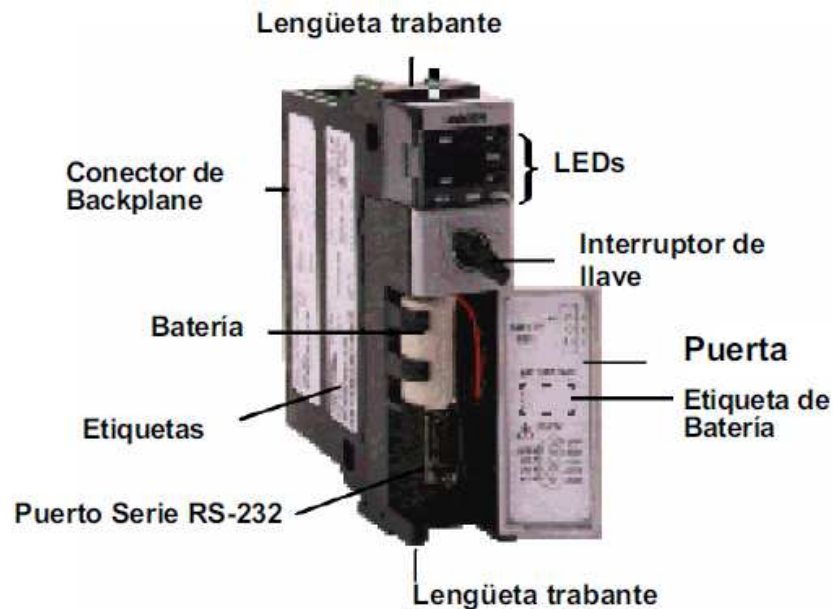


Figura 1.10 Partes del controlador ControlLogix<sup>8</sup>

<sup>7</sup>Figura 1.9 obtenida de la URL (<http://allenbradleyplctutorials.blogspot.mx/2013/06/controllogix-5000-system-components.html>)

<sup>8</sup>Figura 1.10 obtenida de la URL (<http://allenbradleyplctutorials.blogspot.mx/2013/06/controllogix-5000-system-components.html>)

## 1.2.2 FlexLogix

Es un sistema para el control distribuido; es decir, la plataforma de control se encuentra ubicada en o cerca del proceso de la máquina. Ello permite simplificar la aplicación dividiéndola.

Los ejemplos de aplicaciones de control distribuidas incluyen control de máquinas, control de supervisión y adquisición de datos (SCADA, por sus siglas en inglés), línea de ensambles, control de hornos, procesos pequeños y estaciones de relleno.

Este sistema posee las siguientes características:

- Es modular: los módulos de E/S FlexLogix soportan uno o dos bancos y hasta 8 módulos de E/S locales y 8 módulos de E/S locales extendidas.
- Puede montarse sobre un riel DIN (ver figura 1.11) que son dispositivos que proveen un montaje conveniente y simple de componentes para un fácil acceso.
- Puede montarse vertical u horizontalmente

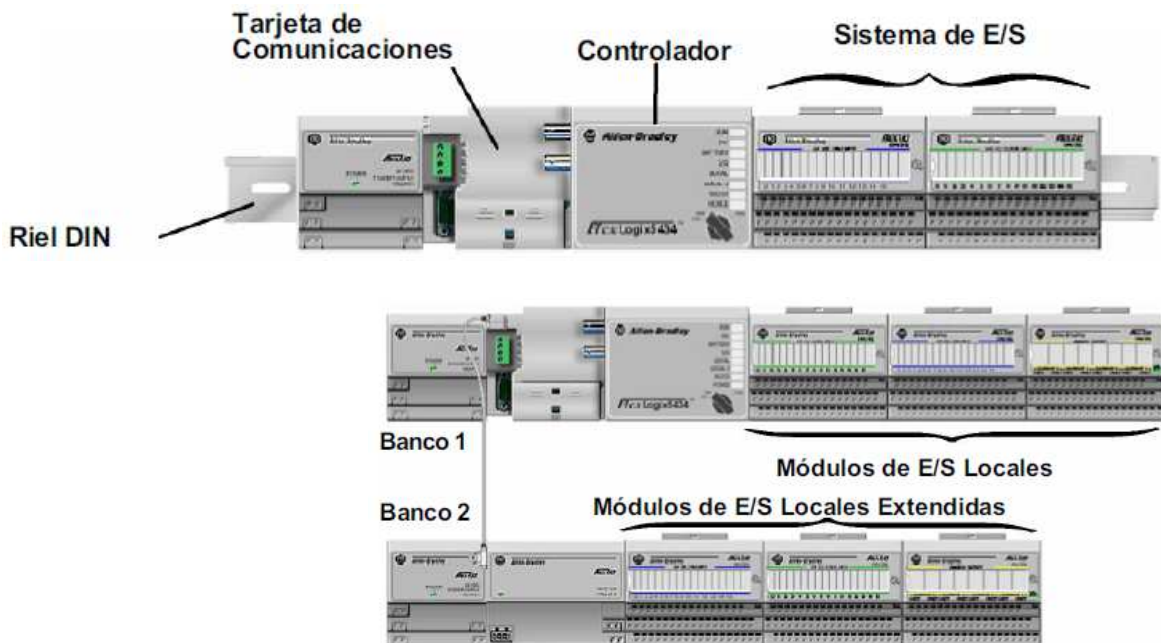


Figura 1.11 Plataforma de FlexLogix<sup>9</sup>

<sup>9</sup> Figura 1.11 obtenida de la URL (<http://allenbradleyplctutorials.blogspot.mx/2013/06/flexlogix-controller-system-components.html>)



El controlador FlexLogix (ver figura 1.12). Posee una memoria fija de 512 Kbytes, contiene una memoria no volátil para retener un proyecto sin batería y un soporte para redes EtherNet/IP, ControlNet y DeviceNet. Las tarjetas de comunicaciones FlexLogix se conectan directamente al controlador en las dos ranuras de tarjetas secundarias de comunicaciones.

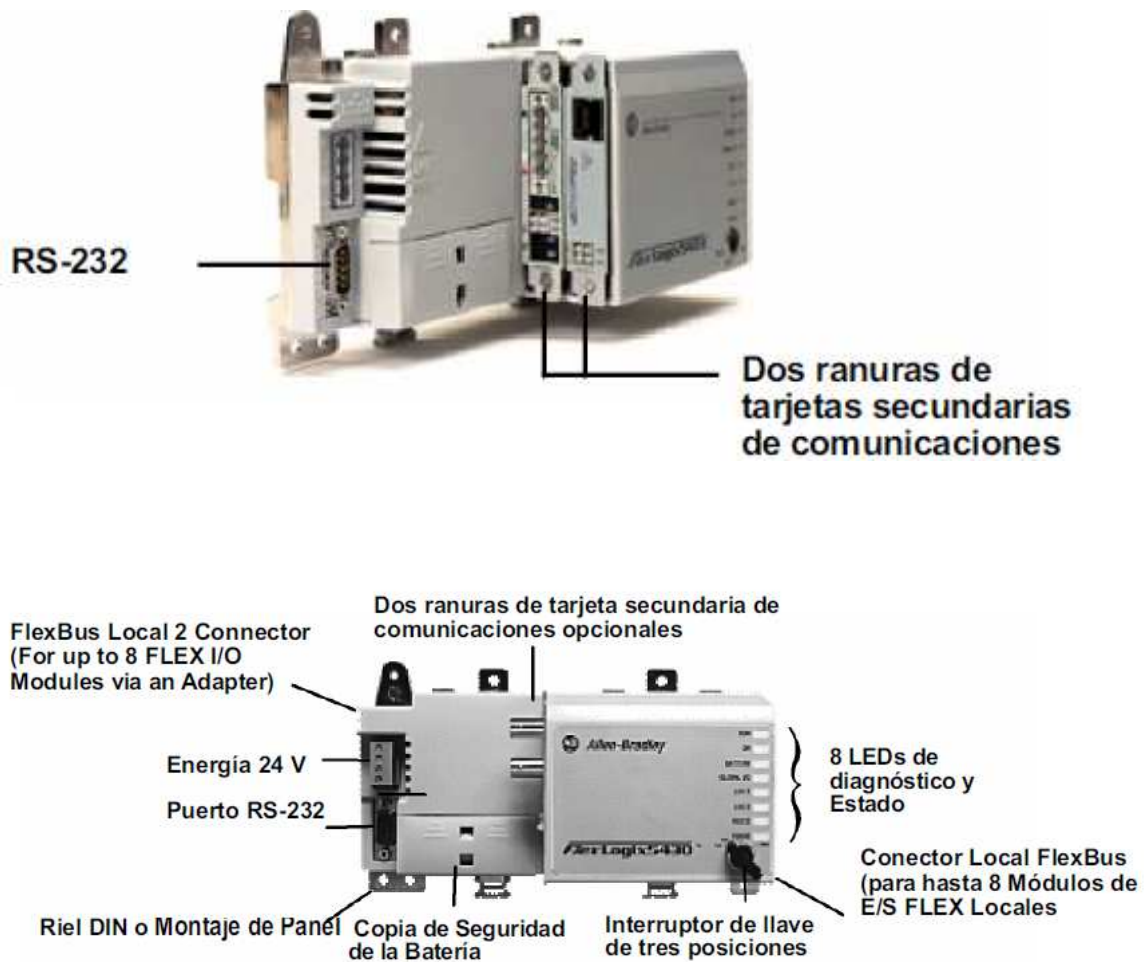


Figura 1.12 Componentes principales de un controlador FlexLogix<sup>10</sup>

<sup>10</sup>Figura 1.12 obtenida de la URL (<http://allenbradleyplctutorials.blogspot.mx/2013/06/flexlogix-controller-system-components.html>)



### 1.2.3 CompactLogix

La plataforma de CompactLogix consiste en un pequeño sistema modular. Se puede montar sobre rieles DIN o en paneles. El sistema de E/S está conformado por bloques terminales desmontables (ver figura 1.13). Soportan hasta 3 bancos de E/S locales y hasta 30 módulos de E/S CompactLogix.

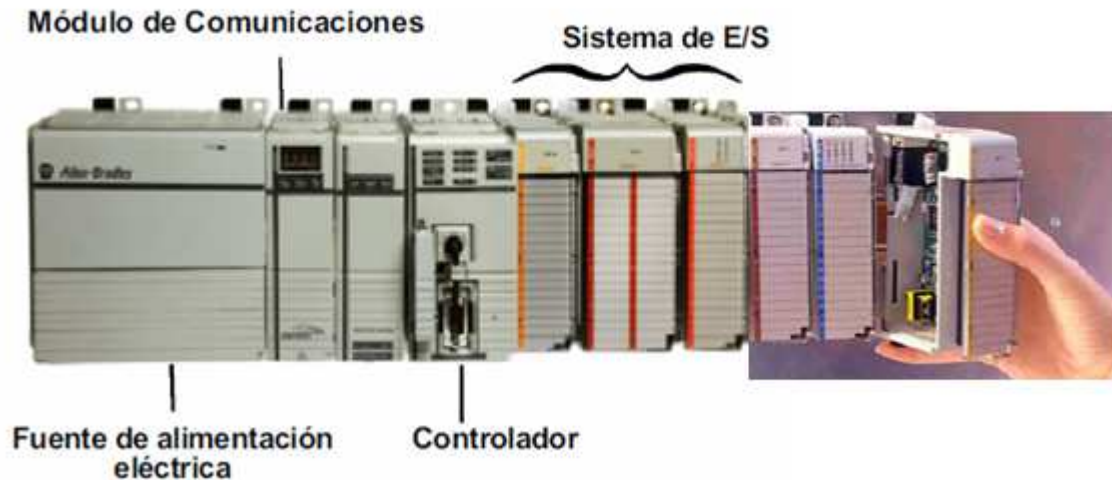


Figura 1.13 Plataforma CompactLogix<sup>11</sup>

Los controladores CompactLogix constan de leds de diagnóstico, interruptor de tres posiciones, puerto de comunicación RS-232(ver figura 1.14) y ofrecen los siguientes beneficios: poseen una capacidad de memoria de 512 Kbytes hasta 2 MBytes, soportan las redes EtherNet/IP, ControlNet y DeviceNet.

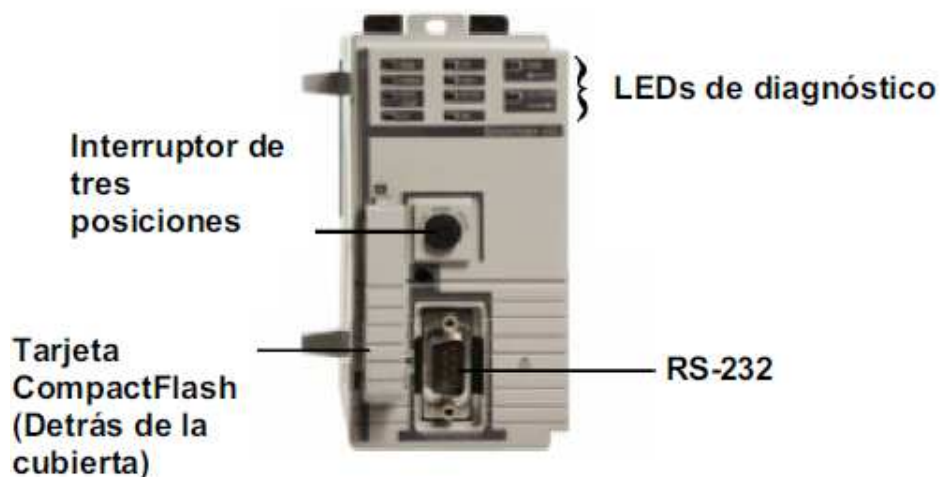


Figura 1.14 Controlador CompactLogix<sup>12</sup>

<sup>11</sup> Figura 1.13 obtenida de la URL (<http://allenbradleyplctutorials.blogspot.mx/2013/06/compact-logix-controller-system.html>)

<sup>12</sup> Figura 1.14 obtenida de la URL (<http://allenbradleyplctutorials.blogspot.mx/2013/06/compact-logix-controller-system.html>)

## 1.2.4 DriveLogix

La plataforma de DriveLogix es un sistema para el control de variadores distribuidos. Un ejemplo de utilización es con el variador de CA PowerFlex 700S (ver figura 1.15) que combina rendimiento y control flexible altamente funcional y rentable.



Figura 1.15 Un variador de Fase I de PowerFlex 700S con DriveLogix<sup>13</sup>

## 1.2.5 SoftLogix

La plataforma de SoftLogix combina control, información y visualización en un sistema de control abierto (Ver figura 1.16). Consta de un software en donde los módulos se pueden crear, configurar, y controlar a través de imágenes en un chasis virtual. La plataforma de SoftLogix es compatible con un conjunto de productos de Rockwell Automation y Microsoft, puede comunicarse con módulos de entradas, salidas ya existentes y soporta las redes NetLinx como son: DeviceNet, ControlNet, EtherNet/IP.



Figura 1.16 Plataforma SofLogix<sup>14</sup>

<sup>13</sup>Figura 1.15 obtenida de la URL <http://www.alltec-aps.com/rockwellone.html>

## Capítulo2 Nociones básicas de RSLogix 5000 y la lógica de escalera

### 2.1 Elementos del software RSLogix5000

El software RSLogix 5000 se utiliza para programar y configurar los controladores de la familia Logix5000. Sus elementos principales se muestran en la figura 2.1:

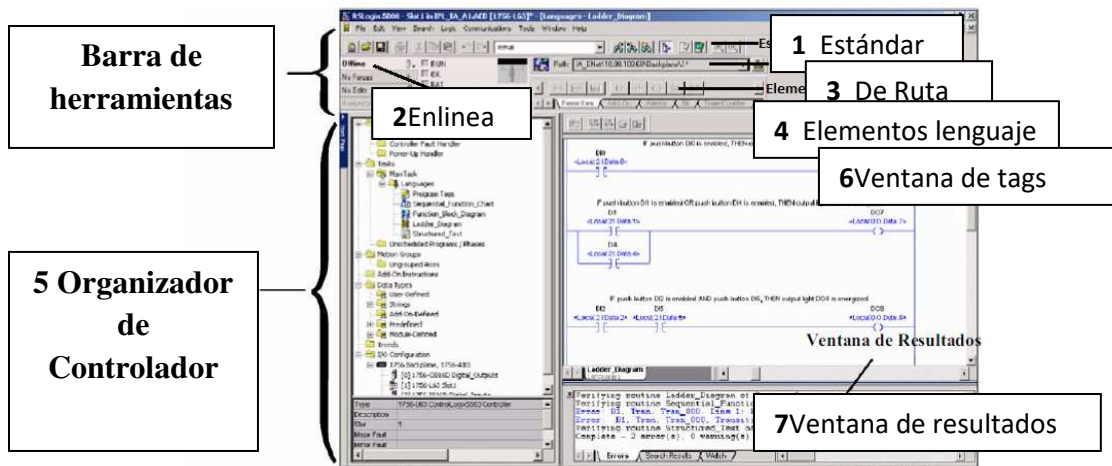


Figura 2.1 Elementos principales de la ventana del software RSLogix5000<sup>15</sup>

- 1. Barra de herramientas estándar:** Cuenta con alternativas estandarizadas de Microsoft por mencionar algunas la opción de nuevo, abrir, guardar, imprimir, cortar, copiar, pegar.
- 2. Barra de herramientas en línea:** Nos muestra el estado del controlador y se puede visualizar la información relacionada.
- 3. Barra de herramientas de ruta:** La función de esta barra es para comunicarse con el controlador y monitorear el estado de las comunicaciones.
- 4. Barra de herramientas de elementos del lenguaje:** Contiene elementos de programación, agrupados por lengüetas, para la introducción en una rutina activa. Cada lenguaje posee su propia barra de herramientas de elementos.
- 5. Organizador de Controlador:** Se muestra el desglose que se utiliza para organizar un proyecto completo. Muestra las áreas principales de un proyecto y la jerarquía de los componentes.
- 6. Ventana de tags:** Esta ventana muestra los nombres de los tag's que se almacenan en el controlador.
- 7. Ventana de resultados:** Aparecen los resultados tras la compilación de un programa.

<sup>14</sup>Figura 1.16 obtenida: <http://www.ab.com/en/epub/catalogs/12762/2181376/2416247/1239772/2185363/Introduction.html>

<sup>15</sup>Figura 2.1 obtenida de la URL [http://www.rocatek.com/forum\\_rslogix.php](http://www.rocatek.com/forum_rslogix.php)

## 2.2 Que es un Proyecto en RSLogix5000

Un proyecto es un archivo de software RSLogix 5000 almacena la totalidad de la información relacionada a la programación y configuración para un controlador. Las salidas se encuentran controladas en base a la organización y la ejecución de un proyecto. Los siguientes pasos permiten la creación de un proyecto nuevo.

1. Inicia el software RSLogix 5000 desde el menú de inicio de Windows o desde el acceso directo del escritorio.
2. En el menú archivo, selecciona nuevo. Aparecerá la ventana “new controller” que se muestra en la figura 2.2.
3. Dentro de esta ventana seleccionar el tipo de controlador
4. Escribe el nombre para el controlador
5. Escribe una breve descripción del nuevo proyecto.
6. Selecciona el tipo de chasis (número de ranuras) que contiene el controlador.
7. Selecciona el número de ranura donde está instalado el controlador.
8. El software automáticamente crea una rutina principal que utiliza el lenguaje de programación de diagrama de lógica de escalera.



Figura 2.2 Ventana para crear un proyecto en RSLogix 5000<sup>16</sup>

<sup>16</sup> Figura obtenida del software RSLogix5000 versión 19

### 2.3 Definición de un TAG

Un tag o etiqueta son un conjunto de palabras claves que se encuentran asociadas a una entrada. Una de las maneras de tener un programa bien organizado, claro y conciso; está en la declaración de variables (Tag's). Los controladores Logix5000 almacenan datos en tags, el tipo de datos para un tag se basa en la fuente de la información. Por ejemplo, un DINT (32 bits) es el tipo de datos principal utilizado en los sistemas Logix5000. Se trata del tipo de datos principal porque es la mínima asignación de memoria para cualquier tag. Sin embargo, se pueden utilizar otros tipos de datos más complejos que están compuestos por varios tipos de datos (ver la tabla 2.1). Por ejemplo, un tipo de datos de temporizador está compuesto por una combinación de DINT y BOOL.

Tipo de dato:	Seleccionar:
Dispositivo analógico en modo de coma flotante	REAL
Dispositivo analógico en modo entero	INT
Caracteres ASCII	STRING
Bit	BOOL
Contador	Counter
Señal de E/S digital	BOOL
Numero de coma flotante	REAL
Entero (número entero)	DINT
Secuenciador	CONTROL
Temporizador	TIMER

Tabla 2.1 Tipos de datos para un tag

Una forma de crear una nueva etiqueta es hacer clic derecho en los tags del controlador en el organizador del controlador y seleccione New Tag. Aún más rápido es la tecla de acceso rápido Ctrl + W (ver figura 2.3).

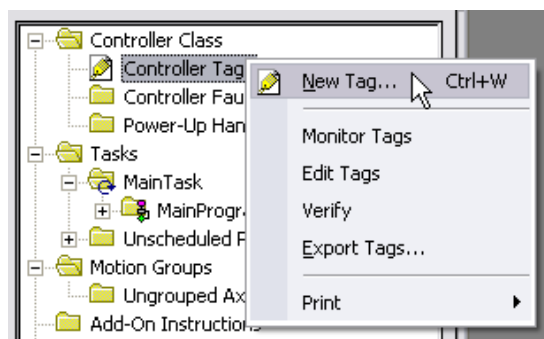


Figura 2.3 creación de un nuevo TAG

Cuando se obtiene acceso a los tags hay dos posibilidades. Se puede supervisar tags - esta opción permite ver los tags y cambiar sus valores, o editar tags - esta opción permite añadir o borrar tags, pero no cambiar valores.

## 2.4 Elementos de un proyecto

La estructura de un proyecto de aplicación puede llegar a ser muy compleja, para ello se han organizado los proyectos mediante tareas, programas, rutinas, siendo este el orden o la jerarquía a seguir en un proyecto de aplicación en RSLOGIX5000.

**Tarea:** Mecanismo de sincronización para ejecutar programas.

**Programa:** Conjunto de rutinas y tags relacionados.

**Rutina:** Secuencia de código de programación ejecutado en bloque. (Cada rutina en el proyecto utiliza un lenguaje de programación específico).

### 2.4.1 Definición de una tarea

Una tarea proporciona información de programación y prioridades para un conjunto de uno o más programas. Una vez que se activa, todos los programas asignados a la tarea se ejecutan en el orden como se muestran en el organizador del controlador. En un proyecto se pueden crear los siguientes tipos de tareas: continua, periódica y evento.

Tarea continua; se define como la tarea principal de manera predeterminada al crearse un proyecto. Se ejecuta todo el tiempo, pero puede ser interrumpida por tareas periódicas o de evento. Para cualquier controlador, sólo se puede configurar una sola tarea como continua, por ejemplo, la tarea continua POS (ver figura 2.4) contiene los siguientes programas:

PLC: Intercambio de datos entre PLC

SCADA: Datos de hacia el sistema supervisor

ALARMING: Rutinas de programa que monitorean los límites permitidos y las retroalimentaciones para manejo de alarmas antes de ser mostradas en el HMI.

SIMULATION: Este programa ejecuta la simulación de los motores, las válvulas, los interruptores y los valores de proceso.

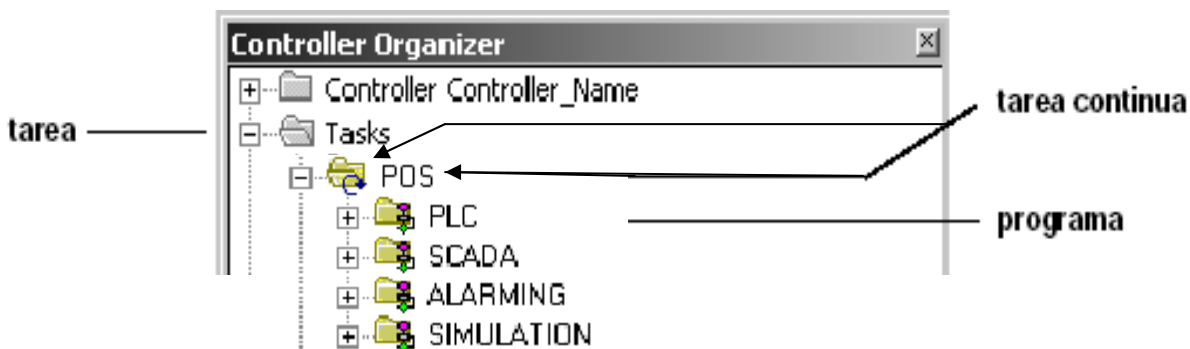


Figura 2.4 Icono que identifica una tarea continua y como ejemplo la tarea POS<sup>17</sup>

<sup>17</sup>Figura obtenida de [http://www.rocatek.com/forum\\_rslogix.php](http://www.rocatek.com/forum_rslogix.php)

Tarea Periódica; se ejecuta de modo regular a una velocidad definida por el usuario. Cuando se le llama, interrumpirá cualquier tarea de prioridad inferior. Una tarea periódica realiza una función según un régimen específico. Cada vez que caduca el tiempo de la tarea periódica esta interrumpe a la tarea continua, se ejecuta una vez y luego devuelve el control donde se interrumpió la tarea continua. La opción predeterminada para el periodo de tiempo es de 10 ms.

Como ejemplo la tarea periódica BASIC\_CONTROL de 200ms (ver figura 2.5) contiene las rutinas de los lazos, las interfaces de entrada/salidas análogas con su escalamiento necesario y los módulos de control de los motores y válvulas. Las interfaces de entrada y salida hacen los enlaces entre los módulos de entrada/salida y los elementos de las estructuras (retroalimentaciones, relevadores, etc.). Los programas para motor y para válvula realizan el control básico de estos elementos.

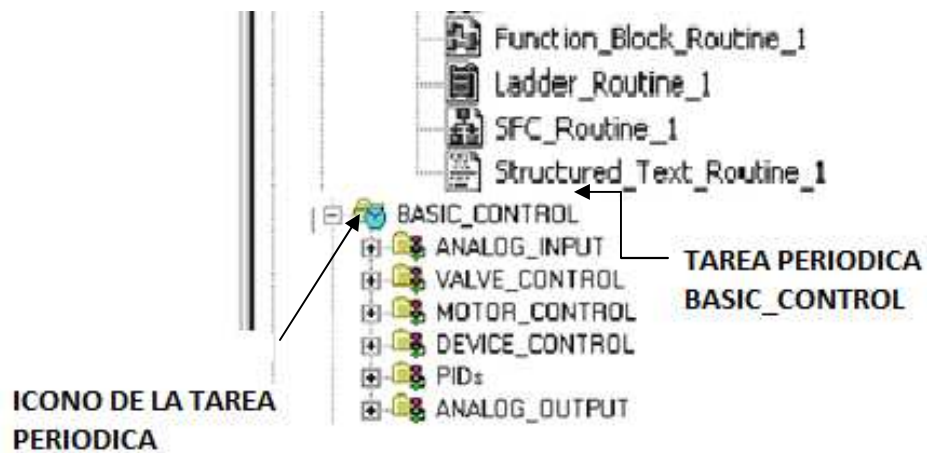


Figura 2.5 Icono que identifica una tarea periódica y como ejemplo la tarea periódica BASIC\_CONTROL<sup>18</sup>

La tarea PHASES\_FUNCTIONS es una tarea periódica de 250ms. contiene todas las rutinas para las fases del equipo. Una rutina de salidas de comando es utilizada para unir todos los comandos de las diferentes fases y funciones a los dispositivos.

La tarea SEQUENCE es una tarea periódica de 500ms y contiene todas las secuencias necesarias para controlar los procedimientos y operaciones. Procedimiento de producción; controlan el modo de producción de las diferentes unidades, controlando la secuencia de las diferentes fases en la receta.

Evento: tarea que se activa sólo cuando ocurre un evento específico. También se pueden usar eventos para aplicaciones de conteo de alta velocidad.

<sup>18</sup> Figura 2.5 obtenida de [http://www.rocatek.com/forum\\_rslogix.php](http://www.rocatek.com/forum_rslogix.php)



## 2.4.2 Definición de un programa

Un programa es una subdivisión de una tarea. Cada programa contiene tags, donde se guardan los datos provenientes de dispositivos. Por lo tanto, los programadores deben definir la memoria del controlador mediante la creación de tags, de una rutina principal, de otras rutinas y una rutina de fallo opcional.

Un programa constituye el segundo nivel de sincronización dentro de un proyecto, cada tarea ControlLogix o SoftLogix puede contener (sincronizar) hasta 100 programas. Cada tarea CompactLogix o FlexLogix puede contener (sincronizar) hasta 32 programas.

Los programas dentro de un proyecto que no se encuentran sincronizados por ninguna tarea, no se ejecutan. Los programas pueden quedar sin sincronización hasta que se los necesite (para añadir funcionalidad futura o para la resolución de problemas).

## 2.4.3 Definición de una rutina

Una rutina proporciona el código ejecutable, o las instrucciones de toma de decisiones, para un proyecto en un controlador. Cada rutina contiene un conjunto de elementos para un lenguaje de programación específico. Para seleccionar el lenguaje de programación se debe crear una rutina como se muestra en la figura 2.6.

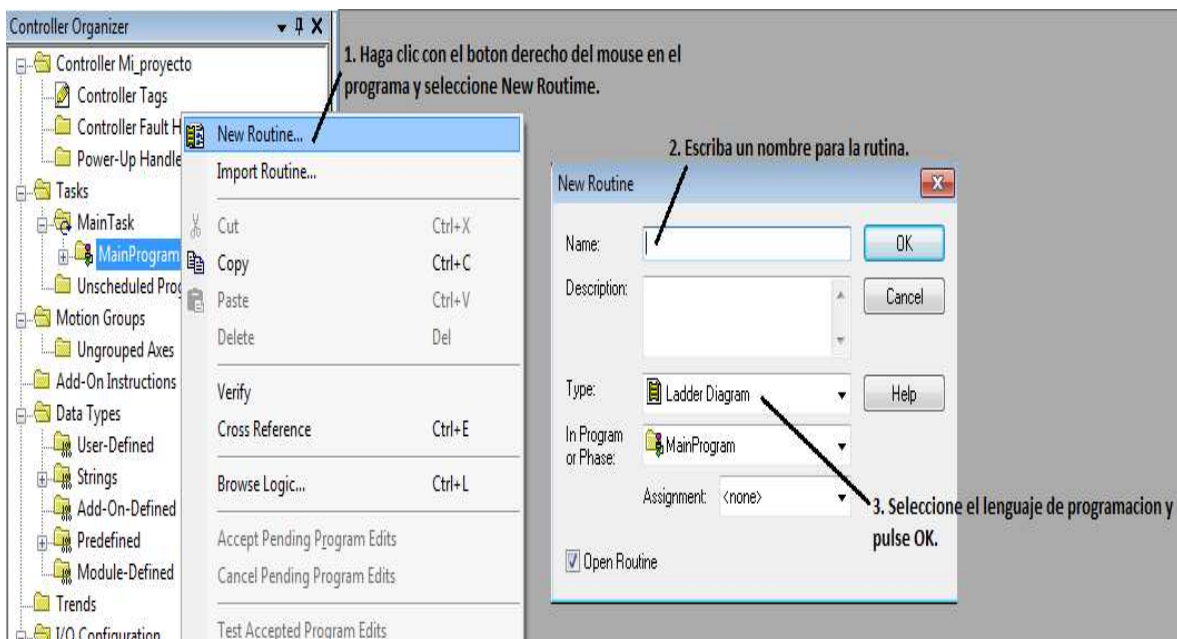


Figura 2.6 Creación de una rutina<sup>19</sup>

Una vez seleccionado el lenguaje de programación estará creada la rutina y será posible introducir la lógica.

<sup>19</sup> Figura 2.6 obtenida de: Software RSLogix5000 versión 19



## 2.5 La lógica de escalera y sus elementos

Un diagrama de escalera es un esquema eléctrico estandarizado que emplea símbolos para describir la lógica de un circuito eléctrico de control. Es importante hacer notar que un diagrama de escalera no indica la localización física de los componentes. Es llamado diagrama de escalera, debido a que varios de los dispositivos del circuito están conectados en paralelo a través de una línea de CD o CA, lo cual, todo en conjunto se asemeja a una escalera, en donde cada conexión en paralelo es un escalón de la escalera.

Las principales características son: las instrucciones de entrada tales como botones pulsadores, interruptores de límite y cualquier otro elemento de mando, deben localizarse a la izquierda estas son las condiciones que tiene el circuito para dejar o no dejar pasar la corriente de una línea a la otra. Estas condiciones se manejan comúnmente con contactos normalmente abiertos o normalmente cerrados los cuales interpretan las señales de alto y bajo de sensores o interruptores. Si las condiciones son verdaderas la corriente llega a las instrucciones de salida las cuales generan acciones como energizar la bobina de un motor o energizar una lámpara, las instrucciones de salida se situarán en el lado derecho. Los carriles de alimentación son las líneas de suministro de energía L1 y L2 para los circuitos de corriente alterna y 24 V y tierra para los circuitos de CD. El controlador explora peldaños de la escalera de arriba a abajo y de izquierda a derecha utilizando RSLogix5000 instalado en el sistema operativo Windows, los elementos de lógica de escalera se pueden arrastrar desde la barra de herramientas elemento de lenguaje hasta una ubicación válida (ver figura 2.7). Este software soporta otras funciones muy utilizadas con base en Windows entre las cuales se incluyen: cortar, copiar, pegar, eliminar, deshacer y rehacer.

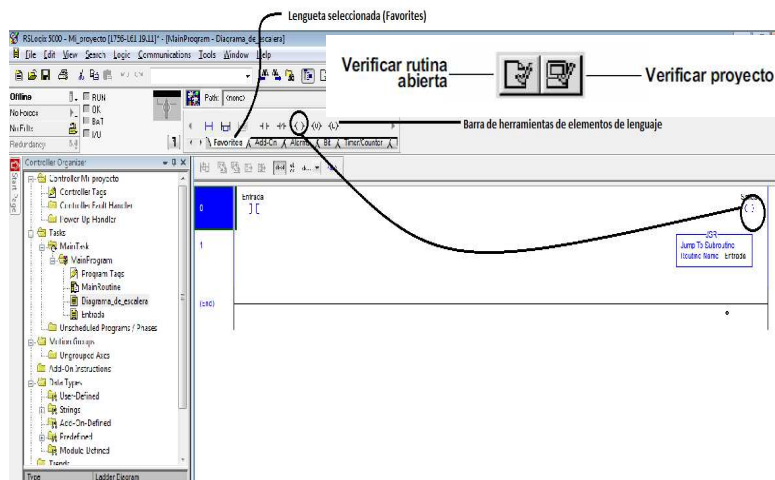


Figura 2.7 Arrastre de elementos de lenguaje en RSLogix 5000 e iconos de rutina abierta y proyecto.

Antes de ejecutar un proyecto en software RSLogix5000, debe verificarse la totalidad de la lógica de escalera para evitar tener errores como: ubicación inadecuada de la instrucción, tags inadecuados o incompletos, tags con tipos de datos que no son válidos para la instrucción, renglones vacíos. La verificación puede realizarse utilizando el botón de verificación de rutina abierta o el de verificar proyecto iconos que se aprecian en la figura 2.7. La lógica de escalera utiliza los siguientes elementos básicos: renglones, instrucciones y bifurcaciones.

## 2.5.1 Renglones

La programación de lógica de escalera se organiza en renglones que ordenan los pasos en el proceso de lectura de una instrucción. Los renglones se enumeran desde cero hasta el número más elevado (de arriba hacia abajo). Cada renglón se lee de izquierda a derecha. Los renglones no pueden estar vacíos y el último renglón es la instrucción fin. Cada renglón de un programa de lógica de escalera debe contener por lo menos una instrucción de control (salida) y generalmente, contiene una o más instrucciones condicionales (entradas).

## 2.5.2 Instrucciones

Las instrucciones son comandos que definen operaciones a ser realizadas por un controlador, a cada instrucción se le asigna un tag, cada instrucción de entrada mira al valor de su tag correspondiente a fin de determinar si la instrucción de entrada es verdadera o falsa, en el software, las instrucciones aparecen resaltadas cuando son verdaderas o son habilitadas. Existen dos tipos de instrucciones de entrada y de salida (ver figura 2.8), son instrucciones de bit condicional que cambian su estado verdadero/falso a fin de reflejar el valor del bit al que corresponde.

**Instrucción de entrada:** Instrucción que evalúa los datos en un controlador.

La instrucción de entrada **XIC** examina si el bit de datos está cerrado, es decir, que tenga un valor de 1 lógico (no confundir esta instrucción con un contacto normalmente abierto).

La instrucción de entrada **XIO** examina si el bit de datos está abierto, es decir, que tenga un valor de 0 lógico (no confundir esta instrucción con un contacto normalmente cerrado).

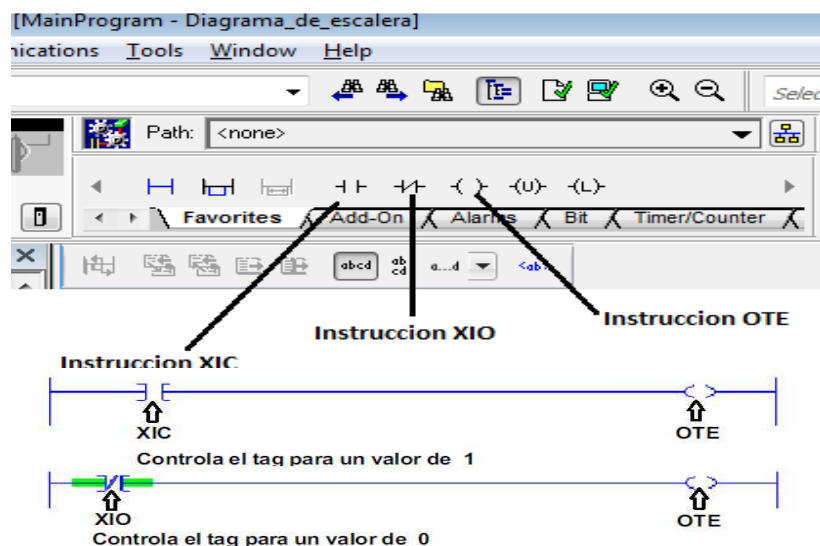


Figura 2.8 Instrucciones condicionales de entrada XIC, XIO y de salida OTE.

**Instrucción de salida:** Instrucción que establece datos desde un controlador.

La instrucción de salida no-retentiva restablece automáticamente sus datos cuando el controlador pasa a modo marcha. Un ejemplo de instrucción de salida no-retentiva es la instrucción **OTE** de salida que establece el bit que opera en 1 cuando la instrucción es verdadera y restablece el bit que opera en 0 cuando la instrucción es falsa o luego de una desconexión y reconexión de la alimentación eléctrica.

Las instrucciones de salida de bit retentivas son conocidas de esta forma porque mantienen el estado de la salida luego de que se vuelve verdadera, aún si las condiciones cambian a falso un ejemplo son las instrucciones **OTL** (enclavamiento de salida) y **OTU** (desenclavamiento de salida) que por lo común se utilizan en pares. Una vez habilitada, la instrucción OTL establece el bit de datos y permanece establecido aun si se desconecta de la red eléctrica y solo hasta que se restablece, típicamente por una instrucción OTU. Una vez inhabilitada, la instrucción OTL no cambia el estado del bit de datos (ver figura 2.9).

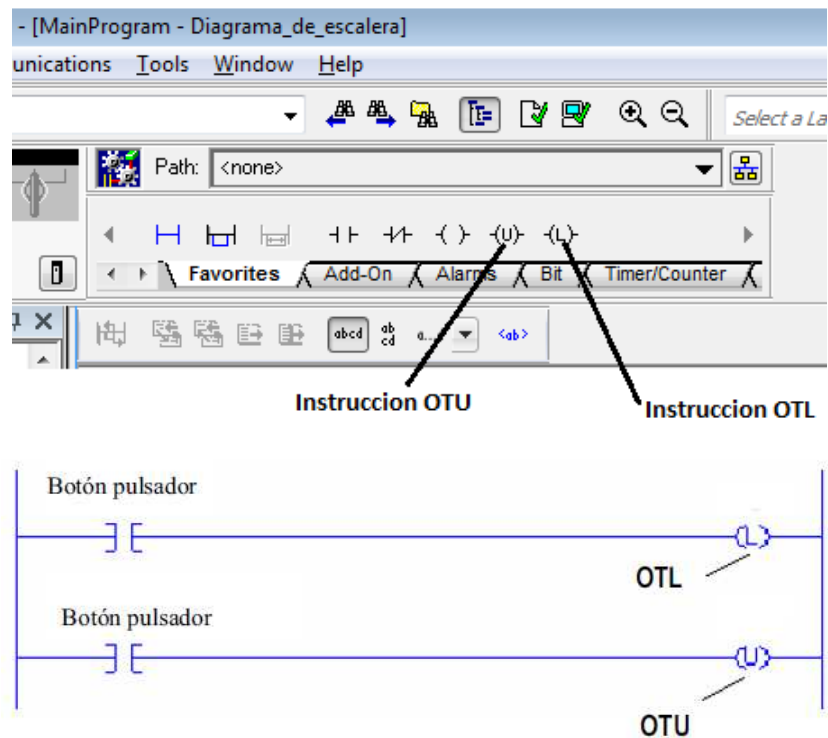


Figura 2.9 Instrucciones de salida OTL y OTU

Un buen programador utiliza salidas no-retentivas, en lugar de salidas retentivas, siempre que sea posible debido a que las salidas no-retentivas tienen las siguientes ventajas; la lógica es más fácil de leer, todas las condiciones para la salida están en una línea, esta permanece activa aún si las condiciones se hacen falsas. La lógica de auto retención es una técnica común en programación que ayuda a utilizar salidas no retentivas en una diversa cantidad de situaciones. Existen varias reglas para colocar instrucciones en renglones, un renglón no necesita contener ninguna instrucción de entrada, pero debe contener al menos una instrucción de salida, las instrucciones de entrada y de salida pueden estar mezcladas en un renglón, y la última instrucción en un renglón debe ser una instrucción de salida.

Se debe tener en cuenta que la condición de entrada (condición de entrada\_1) de toda instrucción que se encuentre al comienzo de un renglón siempre es verdadera, pero eso no significa que la condición de salida (condición de salida\_1) vaya a serlo, ya que dependerá de la instrucción. La selección de instrucciones de lógica de escalera correctas constituye una habilidad importante para la creación del código que evaluará las entradas y controlará las salidas en un sistema de control.

### 2.5.3 Bifurcación paralela y anidada

Una bifurcación es la división en dos partes o ramales y permite ejecutar una de entre varias acciones en valor de una expresión lógica.

Las bifurcaciones se utilizan para crear una ruta alternativa para leer entradas y salidas y pueden tener más de un nivel. Se leen de izquierda a derecha, de arriba hacia abajo. Se tratan de estructuras importantes ya que son las encargadas de controlar el flujo de ejecución de un programa.

Una bifurcación paralela tiene una rama de entrada y varias de salida (ver figura 2.10), es un lugar del flujo de proceso en el que pueden realizarse actividades simultáneamente, en vez de secuencialmente, se evalúan más rápido que las bifurcaciones anidadas.

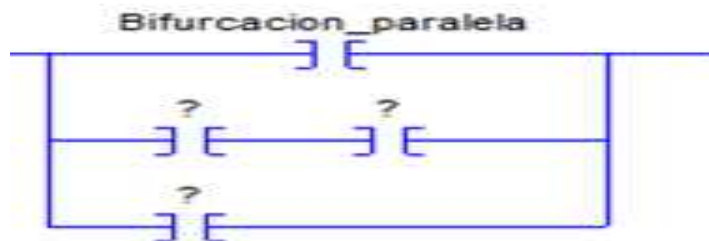


Figura 2.10 Representación de bifurcación paralela

Una bifurcación anidada contiene dos o más bifurcaciones (ver figura 2.11), se basa en la evaluación de una condición (compuesta por una o varias expresiones lógicas) y en función de esta condición el ordenador ejecuta unas u otras instrucciones.

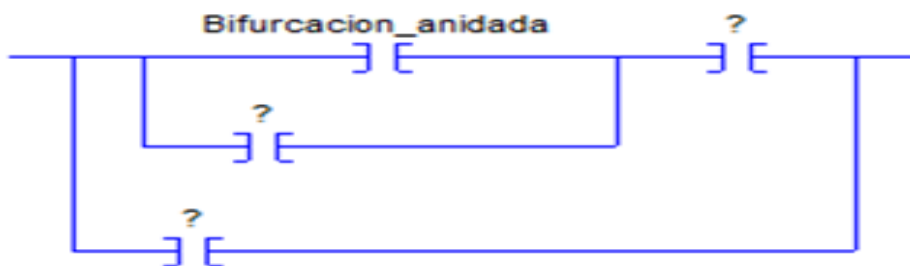


Figura 2.11 Representación de bifurcación anidada

## 2.5.4 Lógica de entrada

El PLC e incluso cualquier sistema digital, se basan en lógica Booleana. La lógica Booleana está basada en la interpretación de señales binarias conjuntadas en ecuaciones, las cuales determinan las condiciones que anteceden a una acción. Existen tres combinaciones de entrada posibles utilizadas para determinar la continuidad lógica.

**La lógica AND** se selecciona cuando todas las condiciones deben ser verdaderas. Ejemplo: en la figura 2.12, las instrucciones de entrada (Uno y Dos) deben ser verdaderas en el mismo renglón del nivel para que la salida X sea verdadera.

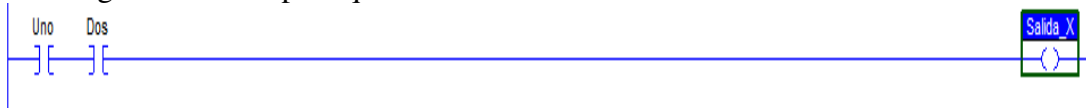


Figura 2.12 Lógica AND

**La Lógica OR** se selecciona cuando cualquiera de ambas condiciones es suficiente para que una salida sea verdadera. Ejemplo: en la figura 2.13, las instrucciones de entrada (Uno o Dos) pueden ser verdaderas es decir con que solo una de ellas sea verdadera, la salida X será verdadera.

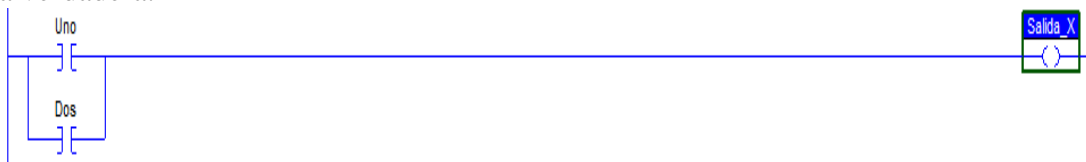


Figura 2.13 Lógica OR

**La lógica AND con OR:** una combinación de la lógica AND y OR se selecciona para evaluaciones más complejas. Ejemplo: en la figura 2.14 el renglón usa tanto AND como OR lógico, si las instrucciones (Uno y Dos y Tres) son verdaderas, entonces la instrucción de salida X es verdadera. “O” Si las instrucciones (Cuatro y Tres) son verdaderas, entonces la instrucción de salida X es verdadera.

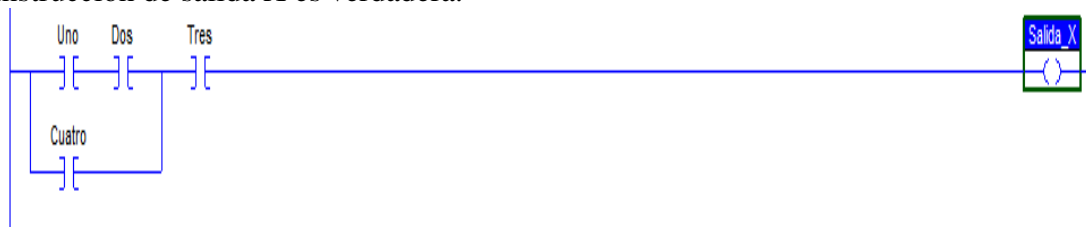


Figura 2.14 Lógica AND con OR

Los controladores Logix5000 soportan entradas y salidas entrelazadas. Para esta lógica las instrucciones de entrada a la izquierda de una salida deben ser verdaderas para que la salida sea verdadera. Si la instrucción (Uno) es verdadera entonces la **salida\_X** es verdadera. Si la instrucción (Uno) es verdadera y la instrucción (Dos) es verdadera entonces la **salida\_Y** también es verdadera (ver figura 2.15).



Figura 2.15 Entradas y salidas entrelazadas

## 2.5.5 Lógica de salida

**Salida no condicionada;** este tipo de salida no requiere ninguna instrucción de entrada. Ejemplo: en la figura 2.16 no existen condiciones, por lo que la salida A es siempre verdadera.



Figura 2.16 Salida No condicionada

**Salidas múltiples;** las bifurcaciones paralelas se pueden utilizar para programar salidas múltiples (ver figura 2.17). También, se pueden utilizar salidas en serie. Si la instrucción de entrada (Uno) es verdadera, tanto la salida A como la B son verdaderas:

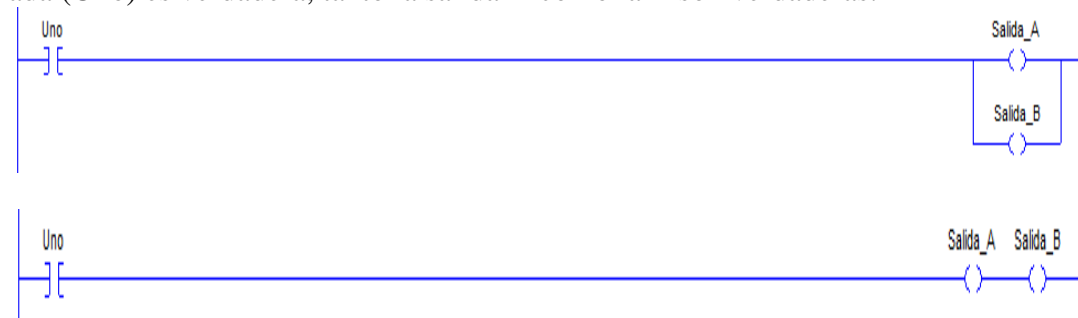


Figura 2.17 Salidas múltiples en paralelo y en serie

**Salidas con entradas diferentes;** Si las salidas comparten condiciones comunes, introducir las condiciones comunes una vez. Y solo utilizar una bifurcación para colocar cualquier condición adicional. Ejemplo: en la figura 2.18, ambas salidas requieren que las instrucciones (Uno y Dos) sean verdaderas; sin embargo, la ruta hacia la salida B también requiere que la instrucción (Tres) sea verdadera.

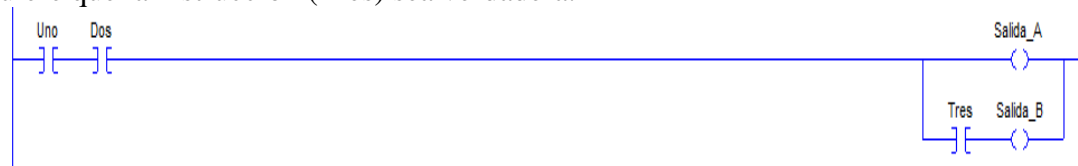


Figura 2.18 Salidas que requieren entradas diferentes

**Análisis de un estado de salida;** El estado de una salida puede ser examinado como condición del renglón utilizando una instrucción condicional con la dirección de la salida. Ejemplo: en la figura 2.19, cuando (Uno y Dos) son verdaderos, el renglón es verdadero. Una vez que el renglón es verdadero, seguirá siendo verdadero hasta que la condición (Dos) se convierta en falsa y rompa el sello. Este tipo de “lógica de enclavamiento” se utiliza a menudo en programación. Por ejemplo, si se utiliza un botón pulsador momentáneo para encender un motor, el motor permanecerá encendido aun cuando el operador libere el botón pulsador.

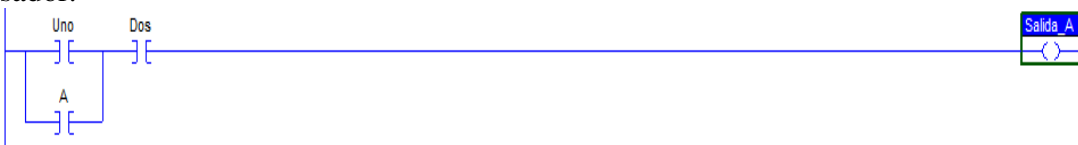


Figura 2.19 Análisis de un estado de salida

## 2.6 Temporizadores

El temporizador es una herramienta que puede ser utilizada para activar y desactivar una bobina o memoria dentro del programa de acuerdo con un tiempo especificado. Así es posible programar una salida, para que en un determinado tiempo encienda o apague un dispositivo externo, por medio de un temporizador que solo existe a nivel lógico, es decir que esta internamente en el PLC y no como un dispositivo externo, cuando se crea un temporizador se crea la siguiente estructura de datos.

Preset (**.PRE**); tiempo final o meta al cual el temporizador quiere llegar y puede estar dentro del rango de 0 a 2,147,483,647 milisegundos.

Acumulado (**.ACC**); es el total del tiempo que ha transcurrido desde que el renglón se hizo verdadero dado en milisegundos.

Enable (**.EN**); especifica si el temporizador está o no habilitado, funciona como bit de control.

. EN = 1 habilitado.

. EN = 0 deshabilitado.

Timer Timing (**.TT**); especifica si el temporizador está o no contando.

.TT = 1 habilitado.

.TT = 0 deshabilitado.

Done (**.DN**); especifica cuando el preset es igual al acumulado.

Existen varios tipos de temporizadores los más comunes son:

### On Delay Timer (TON)

La instrucción TON o temporizador de retardo de conexión se utiliza para un intervalo de tiempo específico (ver figura 2.20), este tipo de temporizador retarda la conexión de la bobina, el tiempo que uno determina es el que nosotros deseamos que se retrase el encendido, cuando el temporizador es habilitado empieza a contar hasta un valor preseleccionado cuando el renglón es verdadero, y reinicia el acumulador cuando el renglón es falso.

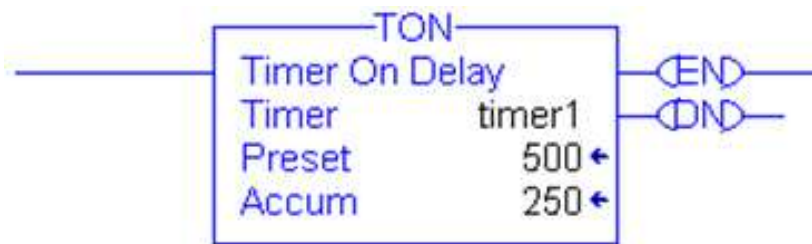


Figura 2.20 Símbolo de la instrucción TON

### Off Delay Timer (TOF)

La instrucción TOF o temporizador de retardo de desconexión (ver figura 2.21) retarda por un intervalo de tiempo específico el apagado de la bobina, después de este el temporizador es deshabilitado. Empieza a contar cuando el renglón es falso, y reinicia el acumulador cuando el renglón es verdadero.

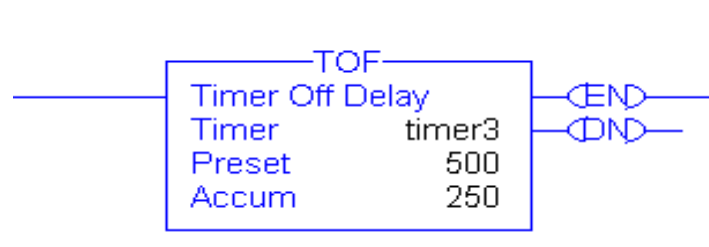


Figura 2.21 Símbolo de la instrucción TOF

### Retentive timer on delay RTO

La instrucción RTO (ver figura 2.22), empieza a contar cuando el renglón es verdadero, y mantiene el acumulador cuando el renglón es falso.

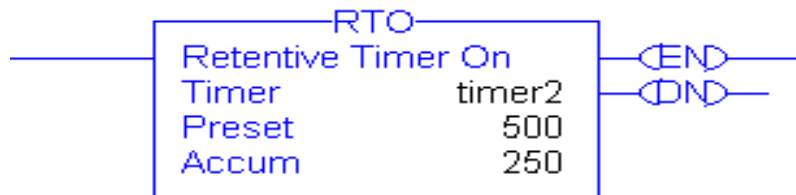


Figura 2.22 Símbolo de la instrucción RTO

### Reset (RES)

La instrucción Reset (ver figura 2.23), reinicia el acumulador del temporizador a cero, es una instrucción de salida utilizada para restablecer temporizadores y contadores.



Figura 2.23 Símbolo de la instrucción RES

Nunca se debe utilizar una instrucción RES para restablecer una instrucción TOF porque el RES borra los bits de estado, así como el valor acumulado.



## 2.7 Contadores

El contador va almacenando o contando eventos, puede contar de forma descendente o ascendente. Su estructura de datos es la siguiente.

Preset (.PRE) Valor que va a ser contado puede estar en un rango de -2,147,483,647 a 2,147,483,646.

Acumulado (.ACC) Numero de eventos que ha computado.

Count Down (.CD) Especifica si el contador esta designado en su conteo descendente.

Count (.UP) Especifica si el contador esta designado en su conteo ascendente.

### Count Up (CTU)

Es usado para contar en forma ascendente cuando el renglón se hace verdadero (ver figura 2.24).

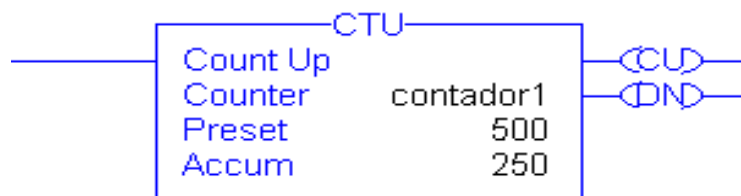


Figura 2.24 Símbolo de la instrucción CTU

### Count Down (CTD)

Es usado para contar en forma descendente cuando el renglón se hace verdadero (ver figura 2.25).

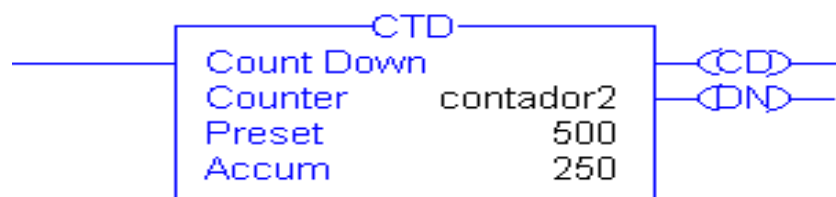


Figura 2.25 Símbolo de la instrucción CTD

Se pueden combinar ambos contadores para que una sola estructura de datos cuente en forma ascendente o descendente. Y al igual que en el temporizador también se utiliza la instrucción reset (RES) para iniciar el acumulador a cero.

## 2.8 Documentación

Una de las cosas más importantes que un programador debe hacer cuando programa lógica de escalera es incluir documentación en el proyecto. Esto es crítico porque generalmente las personas responsables de brindar soluciones a los problemas de un proyecto son distintas a las personas que lo desarrollan.

La documentación reside en un documento en la PC, es decir no se descarga al controlador. Por lo que se tiene que importar o exportar el documento ver figura 2.26.

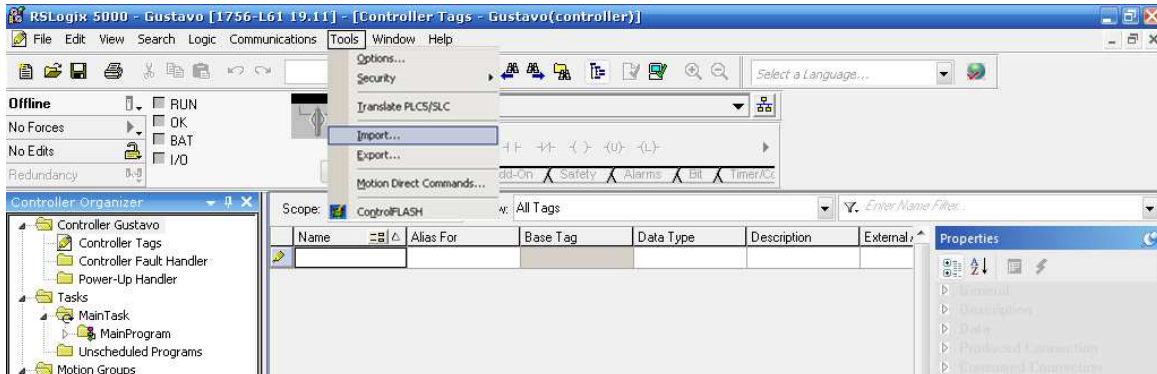


Figura 2.26 Importar un documento

Una función de la documentación es proporcionar descripciones de la lógica de escalera para que programadores y encargados de mantenimiento puedan interpretar rápidamente el programa. Permite una rápida búsqueda en la lógica de escalera. En la tabla 2 se muestran las opciones de búsqueda que están disponibles en RSLogix 5000. Inclusive para el propio programador, es una herramienta indispensable, tanto como para el momento de estar programando y para futuras intervenciones.

Menú de búsqueda	Función
GoTo	Ir a un componente específico, como un tag, documentación, etc.
Find	Encontrar y desplegar hechos de un componente especificado.
Replaced	Encontrar y cambiar un componente de proyecto especificado.
Cross Reference	Crea una tabla de referencia cruzada completa de cada tag específico
BrowseLogic	Crea una descripción general de los componentes del proyecto.

Tabla 2.2 Opciones en el menú de búsqueda para RSLogix 5000

## 2.9 Definición y creación de un comentario de escalón

Un comentario de escalón se encarga de describir la función que realiza el renglón de la lógica de escalera (ver figura 2.28). Para poder crear dentro del programa un comentario de escalón es necesario desplegar el menú contextual dentro del renglón y seleccionar Edit Rung Comment o mediante el teclado pulsando (Ctrl + D) como se muestra en la figura 2.27.

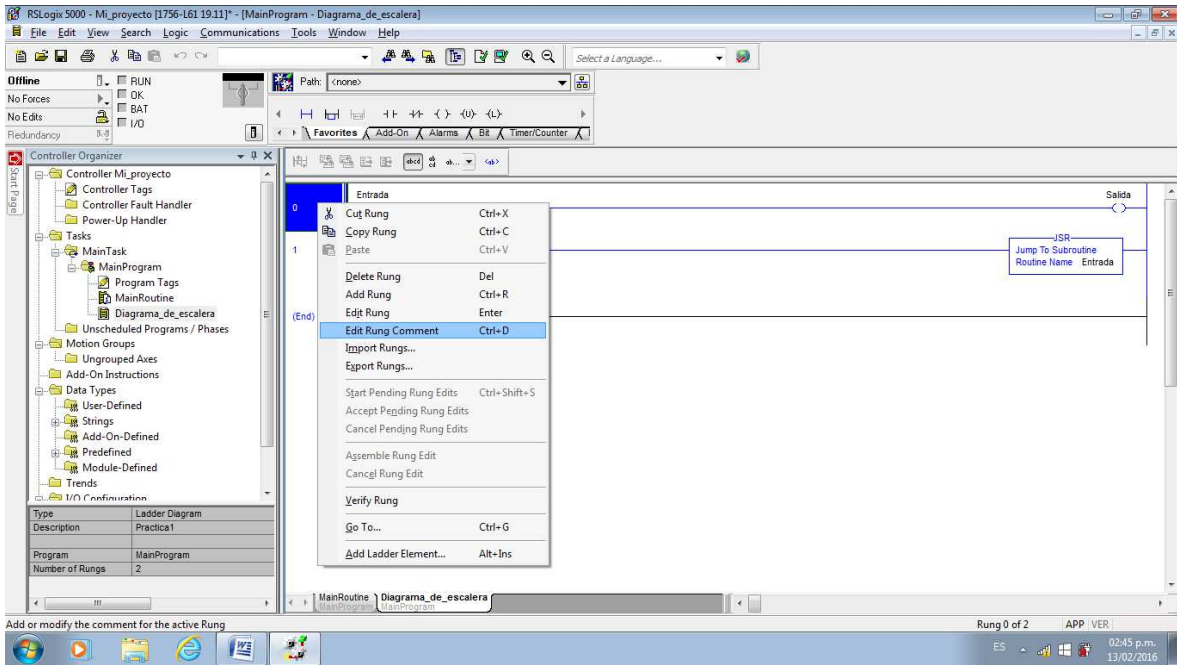


Figura 2.27 Desplegando menú contextual para editar un comentario de escalón.

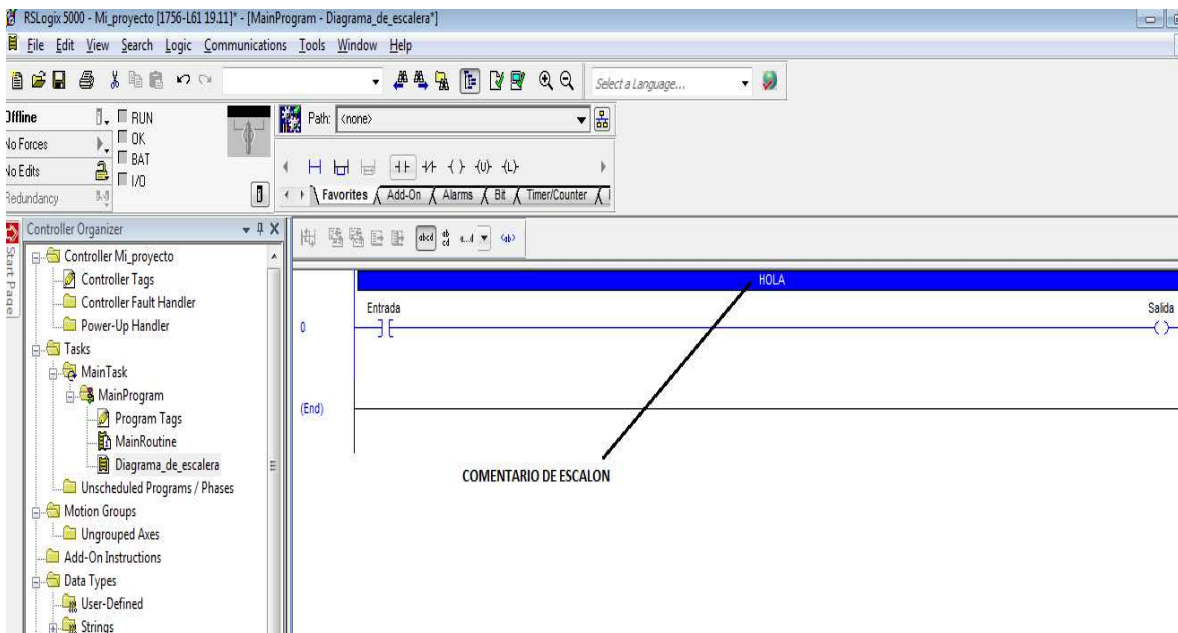


Figura 2.28 Vista de un comentario de escalón dentro del programa, como ejemplo un “HOLA”.

## CAPITULO 3 Aplicación práctica utilizando un LOGO de SIEMENS

### 3.1 Introducción a LOGO!8 y su conexión eléctrica

Logo es el módulo lógico universal de Siemens, ideal para tareas sencillas de automatización industrial y de edificios. Este módulo lógico inteligente destaca por su extraordinaria facilidad de manejo y lo tiene todo en cuanto a funcionalidades.

Utilizaremos un LOGO!8 (230 RCE versión con display): 8 entradas digitales, 4 salidas de tipo relevador a 10A, es de clase 2 debido a su voltaje de alimentación 115a 240 VCA.

Las entradas digitales se identifican mediante una “I” seguido de un numero para cada borne de entrada. Las salidas se designan con la letra “Q” seguido de un número para cada borne de salida (ver figura 3).

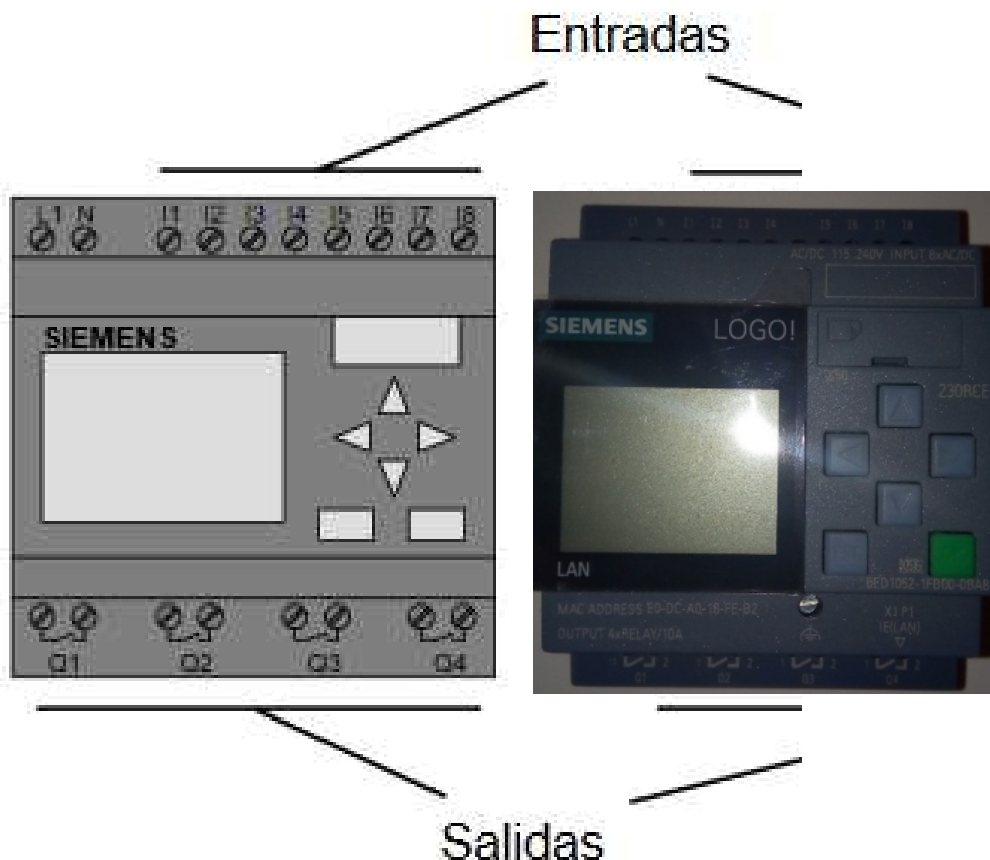


Figura 3 LOGO!8 Basic 230 RCE (versión con display), 8 entradas y 4 salidas.

LOGO!8 cuenta con una pantalla y un teclado para poder programarlo directamente sin necesidad de utilizar una PC, el modo de programación es a través del lenguaje de programación FBD (diagrama de bloques de funciones), que utiliza los símbolos gráficos del álgebra booleana, para representar la lógica. También es posible representar en conexión directa con los cuadros lógicos funciones complejas, por ejemplo, funciones matemáticas, aunque también cuenta con esta opción, mediante el software de programación **LOGO!SoftComfort** que permite crear, comprobar, modificar, guardar e imprimir programas rápida y fácilmente en un PC.

La conexión de LOGO!8, se realizó como se muestra en la figura 3.1, con un voltaje de alimentación de 127 V. Se colocaron 4 focos en las salidas para visualizar el funcionamiento.

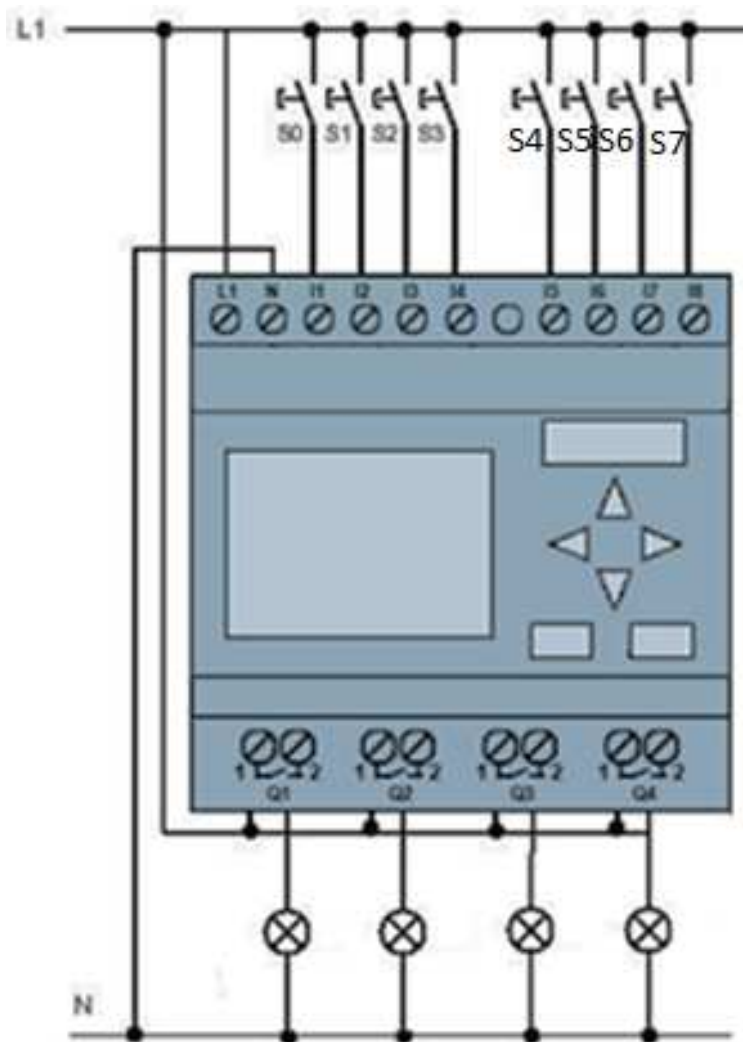


Figura 3.1 Alambrado de un LOGO!8 (230 RCE versión con display)

### 3.2 Desglose del menú principal y los términos básicos de LOGO!8

Al energizar el PLC LOGO!8 se puede observar la pantalla de inicio también conocida como pantalla de menú principal (ver figura 3.2).



Figura 3.2 Menú principal de LOGO!8

Al posicionar el cursor en “Program” y presionar “OK” (botón verde). Aparecerán las opciones del menú de programación que se muestran en la Figura 3.3.

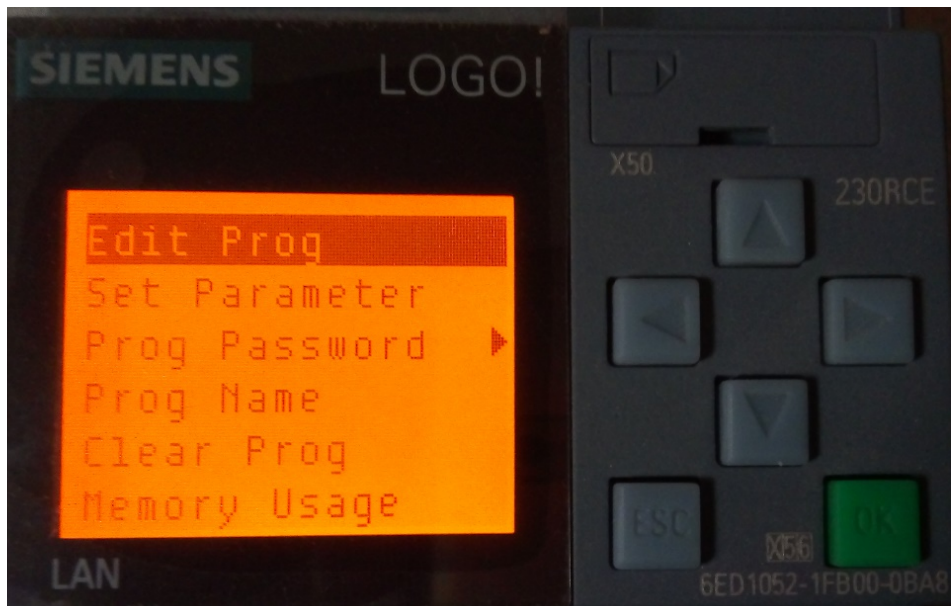


Figura 3.3 Opciones del Menú de programación de LOGO!8



Al posicionar el cursor en cualquier opción del menú principal y posteriormente presionando “OK” se puede acceder a las opciones de los demás menús y al presionar la tecla “ESC” se regresa de nuevo al menú principal, el desglose completo se muestra en la figura 3.4.

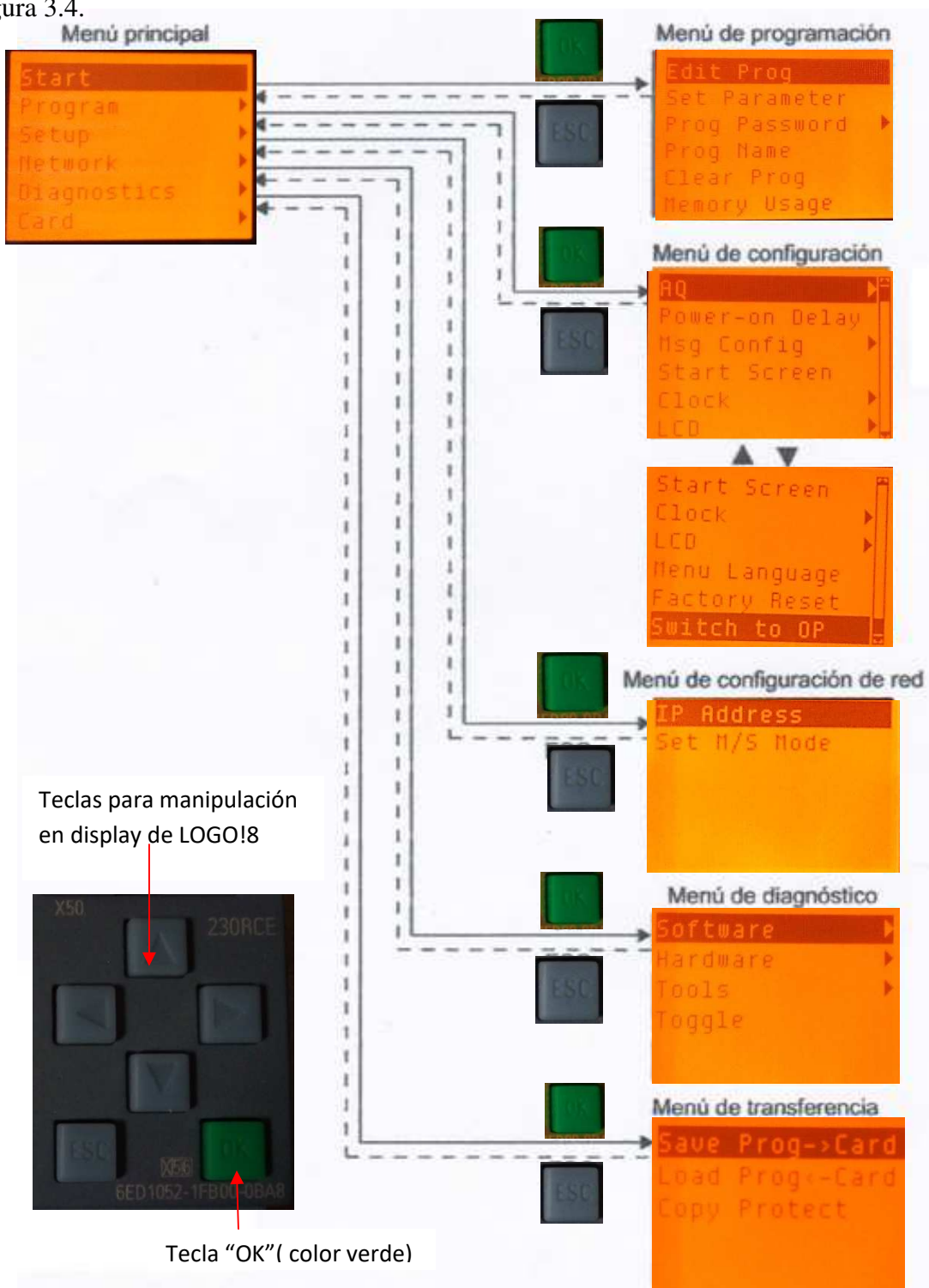


Figura 3.4 Vista general de los menús de LOGO!8RCE

Para utilizar y crear programas en LOGO!8 es necesario saber qué significan los dos términos básicos conector y bloque:

1.- El término “conector” designa todas las conexiones y estados de LOGO. Las E/S digitales pueden tener el estado lógico '0' o '1'. El estado '0' significa que la entrada no tiene aplicado un voltaje específico. El estado '1' significa que la entrada tiene aplicado un voltaje específico. Los conectores 'hi' y 'lo' facilitan la creación de programas. 'hi' (high) tiene asignado el estado '1' y 'lo' (low) el estado '0'. No es necesario utilizar todos los conectores de un bloque. Para conexiones no utilizadas, el programa adopta automáticamente el estado que garantiza el funcionamiento del bloque en cuestión.

2.- El término “bloque” es una función que sirve para convertir información de entrada en información de salida. Antes era necesario cablear los distintos elementos en un armario eléctrico o una caja de bornes. Pero utilizando LOGO!8 a través de su número de bloque, es posible añadir casi cualquier bloque a una entrada del bloque actual. Cada vez que se inserta un bloque en un programa, LOGO!8 adjudica un número a ese bloque (ver figura 3.5). A través del número de bloque, muestra la relación existente entre los bloques.

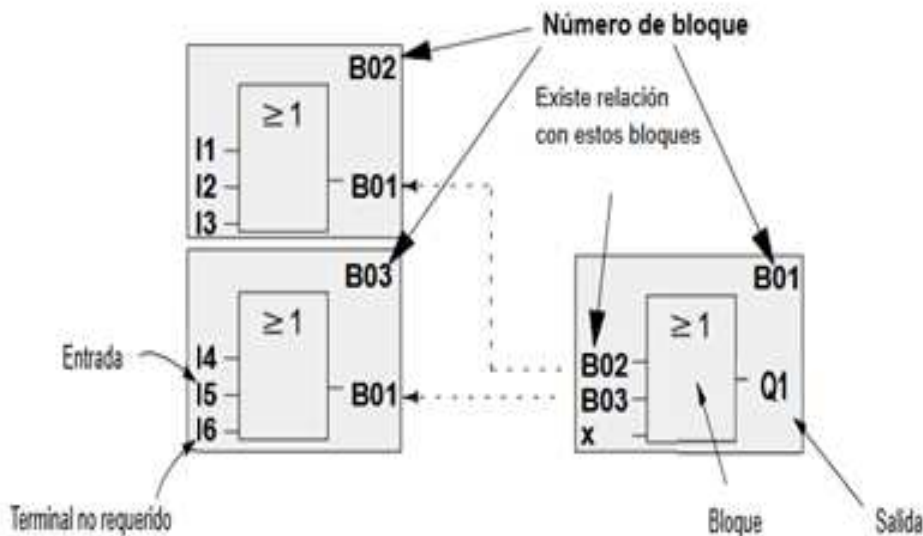


Figura 3.5 Representación de bloque y como LOGO!8 relaciona los números de bloque entre ellos<sup>20</sup>.

Los bloques más elementales son las operaciones lógicas, LOGO!8 cuenta con una programación basada en el uso de compuertas lógicas y bloques de funciones, que permiten la elaboración de algoritmos de control simplificados y eficaces. Al unir varios bloques de funciones, de forma específica, se pueden implementar programas de control complejos.

<sup>20</sup> Figura obtenida

de [http://biblioteca.upnfm.edu.hn/images/directorios%20tematicos/xxtindustrial/libros%20de%20electricidad/Controles%20Electromecanicos/LOGO\\_PROGRAMA%20DE%20INSTALACIONES%20ELECTRICAS.PDF.pdf](http://biblioteca.upnfm.edu.hn/images/directorios%20tematicos/xxtindustrial/libros%20de%20electricidad/Controles%20Electromecanicos/LOGO_PROGRAMA%20DE%20INSTALACIONES%20ELECTRICAS.PDF.pdf)



### 3.3 Funciones generales de LOGO!8

Las General Función (GF, por sus siglas en inglés) o funciones generales en el LOGO!8, están basadas en el Álgebra de Boole, la cual está definida por operaciones lógicas como Y, O ó NO (And, Or, Not). La electrónica digital emplea este sistema en conjunto con los números binarios, donde, un nivel bajo de señal significa “0” que a su vez significa “Falso” y un nivel alto de señal significa “1” o “verdadero”. Con estos valores se realizan operaciones booleanas a través de dispositivos físicos llamados compuertas lógicas que se encuentran como funciones en el relevador, de las cuales, las principales son:

#### 3.3.1 Función AND y AND (Edge) con evaluación de flancos

Esta función también conocida como producto lógico, se puede entender como una conexión en serie con varios contactos normalmente abiertos. La salida de la función AND solo es 1 si todas las entradas son 1, es decir, si están cerrados todos los contactos.

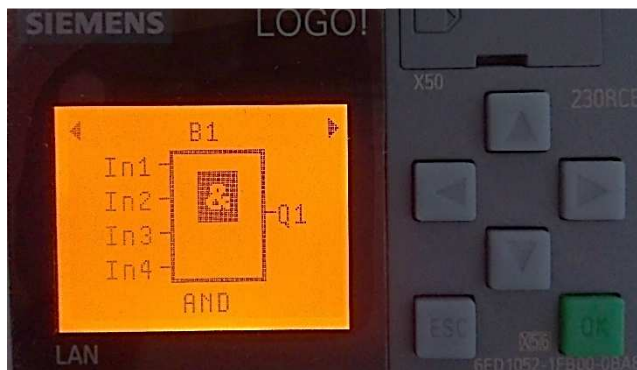


Figura 3.6 La función AND su simbología ANSI e IEC y como se visualiza en LOGO!8

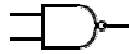
La salida de la función AND activada por flancos solo es 1 si todas las entradas son 1 y por lo menos una de ellas tenía el estado "0" en el ciclo anterior.



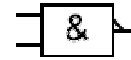
Figura 3.7 Visualización en LOGO!8 de la función AND (Edge)

### 3.3.2 NAND (AND negada) y NAND (Edge) con evaluación de flancos

Se puede entender como una conexión en paralelo con varios contactos normalmente cerrados. La salida de la función NAND solo es 0 si todas las entradas tienen el estado 1, es decir, si los contactos están cerrados.



NAND Símbolo sistema ANSI



NAND Símbolo sistema IEC

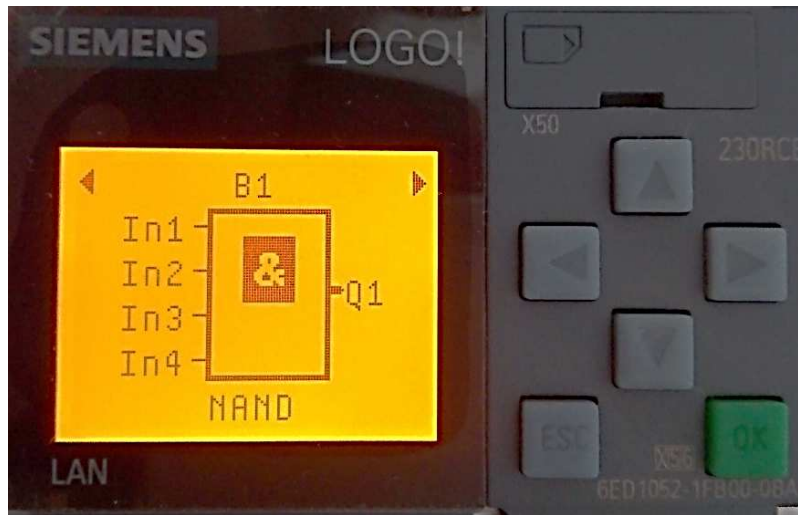


Figura 3.8 La función NAND su simbología ANSI e IEC y como se visualiza en LOGO!8

La salida de la función NAND con evaluación de flancos solo adopta el estado 1 si por lo menos una entrada tiene el estado 0 y en el ciclo anterior todas las entradas tenían el estado 1.



Figura 3.9 Visualización en LOGO!8 de la función NAND (Edge)

### 3.3.3 OR (O) y NOR (OR negada)

Un OR se puede entender como un circuito en paralelo con varios contactos normalmente abiertos, de tal forma que la salida de la función OR solo adopta el estado 1 si por lo menos una entrada tiene el estado 1, es decir, si por lo menos uno de los contactos está cerrado.

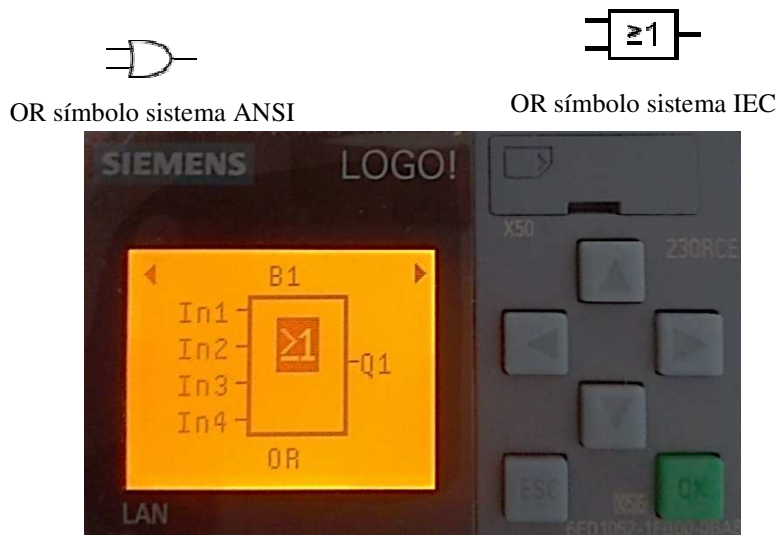
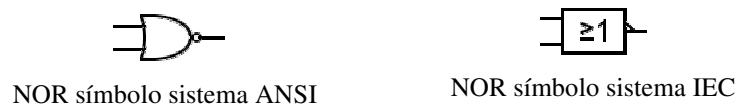


Figura 3.10 La función OR su simbología ANSI e IEC y como se visualiza en LOGO!8

Un NOR se puede entender como un circuito en serie con varios contactos normalmente cerrados. La salida de la función NOR solo adopta el estado 1 si todas las entradas tienen el estado 0, es decir, si están desactivadas. La salida de NOR se pone a 0 tan pronto como se activa una de las entradas (estado lógico 1).



NOR símbolo sistema ANSI

NOR símbolo sistema IEC

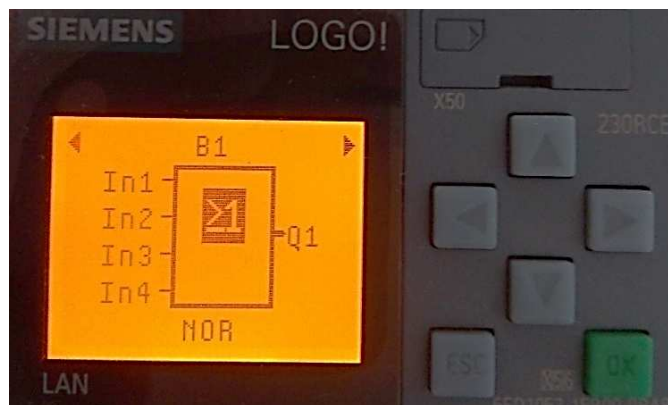


Figura 3.11 La función NOR su simbología ANSI e IEC y como se visualiza en LOGO!8

### 3.3.4 XOR (OR exclusiva)

Un XOR se puede entender como un circuito en serie con dos contactos inversores, de tal forma que la salida de la función XOR adopta el estado 1 si las entradas tienen diferentes estados.



Figura 3.12 La función XOR su simbología ANSI e IEC y como se visualiza en LOGO!8

### 3.3.5 NOT (inversor lógico)

Se puede entender como un contacto normalmente cerrado. La salida adopta el estado 1 si la entrada es 0. El bloque NOT invierte el estado de la entrada. Una ventaja del bloque NOT, por ejemplo, es que no es necesario utilizar contactos normalmente cerrados. Solo tiene que utilizar un contacto normalmente abierto (NA) y mediante el bloque NOT, convertirlo en un contacto NC.

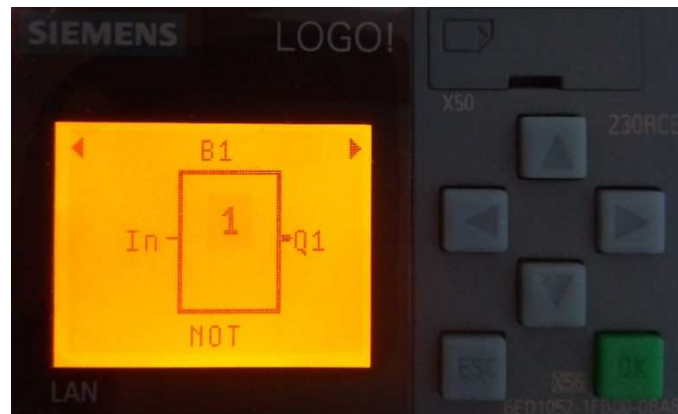


Figura 3.13 La función NOT su simbología ANSI e IEC y como se visualiza en LOGO!8

### 3.4 Funciones especiales de LOGO!8

LOGO!8 no solamente cuenta con funciones generales como las mencionadas anteriormente, este dispositivo también cuenta con temporizadores, contadores, generadores de pulsos y memorias de estados, que conforman las Special Functions (SF, por sus siglas en inglés) o funciones especiales, se distinguen a primera vista de las funciones básicas en la denominación diferente de sus entradas. La entrada "x" de conector para cualquier SF es "low". La entrada x es considerada una entrada no utilizada de funciones especiales y por lo tanto su valor está predeterminado en un "0" lógico.

Los conectores siguientes son entradas lógicas y permiten crear un enlace lógico con otros bloques o con las entradas de la unidad LOGO!8:

- S (Set): Una señal en la entrada S pone la salida a un "1" lógico.
- R (Reset): La entrada de reset R tiene prioridad sobre todas las demás entradas y desactiva las salidas.
- Trg (Trigger): Esta entrada dispara el inicio de una función.
- Cnt (Count): Esta entrada cuenta impulsos.
- Fre (Frequency): LOGO aplica a esta entrada señales de frecuencia que deben evaluarse.
- Dir (Direction): Esta entrada determina el sentido (+ o -).
- En (Enable): Esta entrada habilita la función de un bloque. Si la entrada es "0", el bloque ignora todas las demás señales.
- Inv (Invert): Una señal aplicada en esta entrada invierte la señal de salida del bloque.
- Ral (Resetall): Una señal aplicada en esta entrada da reset a todos los valores internos.
- Lap (para la función de cronómetro) Una señal en esta entrada detiene el cronómetro.

En algunas entradas no se aplica ninguna señal, sino que se configuran los valores relevantes del bloque, por ejemplo:

- Par (Parameter): El parámetro Par no debe conectarse. En su lugar se ajustan los parámetros relevantes del bloque (tiempos, umbrales de conexión/desconexión, entre otros).

Las funciones especiales permiten al usuario realizar algoritmos de control más avanzados y complejos, por lo tanto, solo serán vistas las que se ocuparan para el ejemplo práctico de un sistema que realizara el control para el acceso a través de un portón y son las siguientes:

### 3.4.1 Temporizador ON-DELAY (retardo a la conexión)

En ocasiones es necesario retardar en algún punto el paso de una señal a través de un circuito. Esta labor la realizan los temporizadores y son parte fundamental de la automatización porque nos pueden indicar el momento justo para que inicie o termine un proceso.

Los temporizadores son ampliamente utilizados en aplicaciones industriales para controlar el tiempo de arranque de los motores más grandes, de modo que no todos tratan de iniciar al mismo tiempo. Cuando se inicia un motor, consumirá una corriente más grande que cuando está funcionando a plena velocidad. Esta corriente de arranque se llama corriente de rotor bloqueado y es causada por la armadura cuando no gira y se le aplica una corriente al motor. A medida que el eje del motor empieza a girar, la corriente disminuye a niveles normales y así hasta que el motor se estabiliza a velocidad plena. Para una máquina que tiene varios motores grandes para bombas hidráulicas, y todos ellos trataran de comenzar en el mismo momento en que se enciende la máquina, se debe aplicar el ciclo de arranque escalonado mediante el uso de temporizadores de retardo a la conexión de manera que se reduzcan al mínimo los efectos de la corriente de entrada. De cuando se aplica energía a los motores, se permite que el primer motor se inicie normalmente, pero la alimentación para el segundo arrancador del motor se controla mediante el temporizador de retardo a la conexión.

El temporizador TON (Retardo a la conexión) ver figura 3.14, se utiliza para un intervalo de tiempo  $T$  específico por ejemplo si el temporizador se establece en 10 segundos, los contactos del temporizador se cerrarán después de un retraso de 10 segundos, el tiempo que uno determina es el que nosotros deseamos que se retrase el encendido en el cual la salida  $Q$  es activada siempre y cuando la entrada  $Trg$  siga activada. Una señal en la entrada  $Trg$  (Trigger) dispara el temporizador de retardo a la conexión. Sus parámetros son:  $T$  representa el tiempo tras el que se activa la salida (transición de 0 a 1 de la señal de salida) y remanencia activada el estado se guarda de forma permanente.

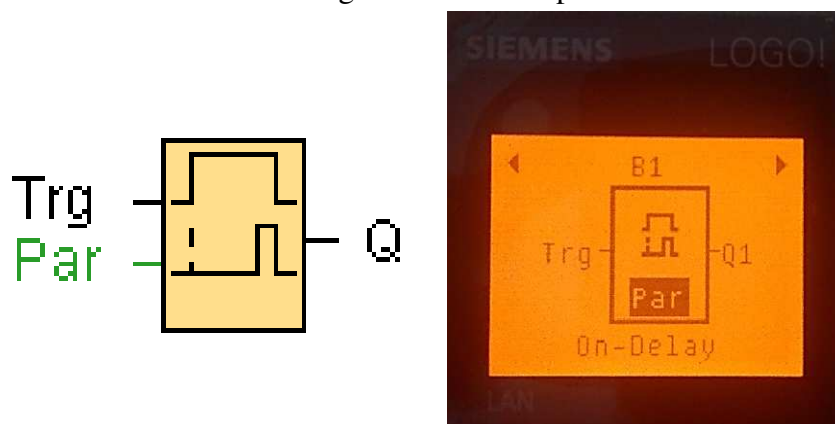


Figura 3.14 Símbolo de retardo a la conexión y como se visualiza en LOGO!8.

La respuesta de tiempo será mediante el **Parámetro T** en donde los valores de entrada se rigen por la base de tiempo ajustada, base de tiempo s (segundos), m (minutos) y h (horas), ver tabla 3.3.

Base de tiempo	Valor máx.	Resolución mín.	Precisión
s (segundos)	99:99	10 ms	+ 10 ms
m (minutos)	99:59	1s	+ 1 s
h (horas)	99:59	1 min	+ 1 min

Tabla 3.3 Rangos válidos para la base de tiempo

La descripción de la función retardo a la conexión se explica con el cronograma de la figura 3.15. Mediante una transición de 0 a 1 se dispara el tiempo **Ta** (Ta es el tiempo actual en LOGO!8). Si la entrada Trg tiene el estado 1 por lo menos durante el tiempo **T** configurado, pondrá la salida a 1 (la salida se activa con retardo respecto a la entrada) tal y como se muestra en el cronograma de la figura 3.15.

LOGO!8 reinicia el temporizador es decir pone la salida nuevamente a 0 si la señal en la entrada Trg es 0. Si el bloque es remanente LOGO!8 pone la salida Q y el tiempo transcurrido a los valores que tenían antes de un corte de alimentación; si el bloque no es remanente, tras un corte de alimentación LOGO!8 pone la salida Q y el tiempo transcurrido a los valores predeterminados.

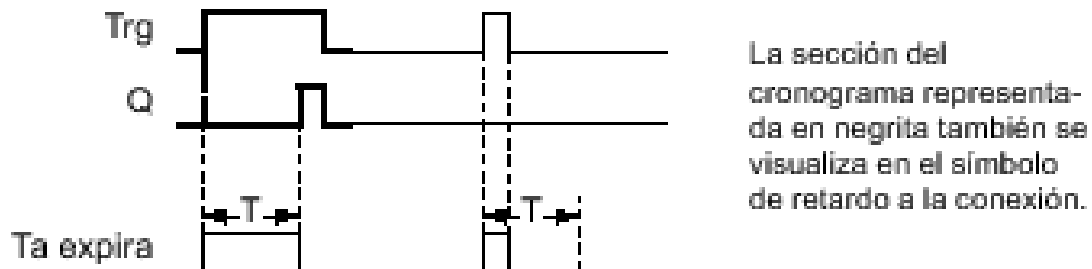


Figura 3.15 Cronograma de un retardo a la conexión<sup>21</sup>

Un diagrama de tiempos o cronograma (ver figura 3.15). Es una gráfica de formas de onda digitales que muestra la relación temporal entre varias señales, y cómo varía cada señal en relación a las demás. Se emplea para describir en una forma concreta el funcionamiento del circuito de control, se pueden apreciar con claridad, qué condiciones se deben cumplir para hacer que un elemento de salida sea energizado o no, lográndose apreciar también, la relación que existe entre los elementos de entrada y salida en un tiempo determinado.

<sup>21</sup>Imagen obtenida de <https://support.industry.siemens.com/cs/mdm/100782807?c=87556823179&dl=es&lc=de-DE>

### 3.4.2 Relé Autoenclavador

Una señal en la entrada S (Set) activa la salida Q y permanece activada hasta que es desactivada con una señal en la entrada R (reset) desactiva la salida Q.

La salida Q se desactiva si están activadas tanto S como R (la desactivación tiene prioridad sobre la activación).

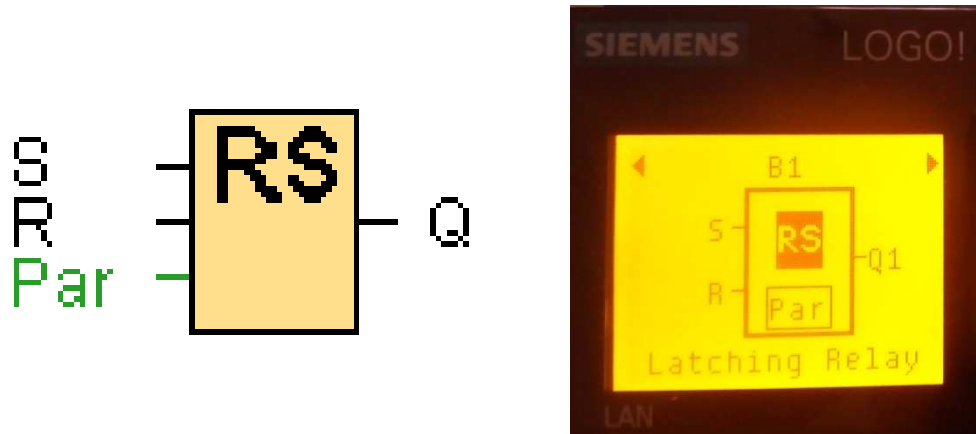


Figura 3.16 Símbolo del relé de enclavamiento y como se visualiza en LOGO!8.

Un relé autoenclavador o también conocido como relé de enclavamiento es un elemento de memoria binario simple. El valor de la salida depende del estado de las entradas y del estado anterior de la salida.

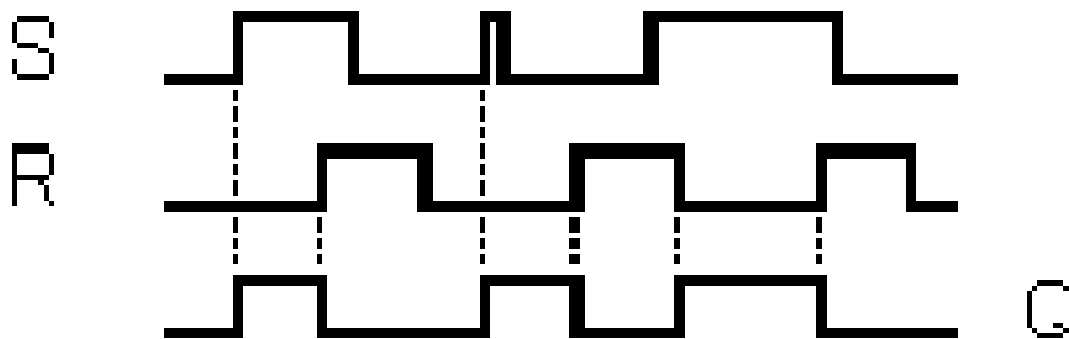


Figura 3.17 Cronograma de relé autoenclavador.<sup>22</sup>

<sup>22</sup> Imagen obtenida de <https://support.industry.siemens.com/cs/mdm/100782807?c=87556823179&dl=es&lc=de-DE>



### 3.4.3 Generador de impulsos asíncrono

El generador de impulsos asíncrono ver figura 3.18, consta de una entrada **En** que habilita y deshabilita esta función especial. La entrada **Inv** permite invertir la señal de salida del generador de impulsos asíncrono activo, solo si el bloque se ha activado por medio de una señal en la entrada EN.

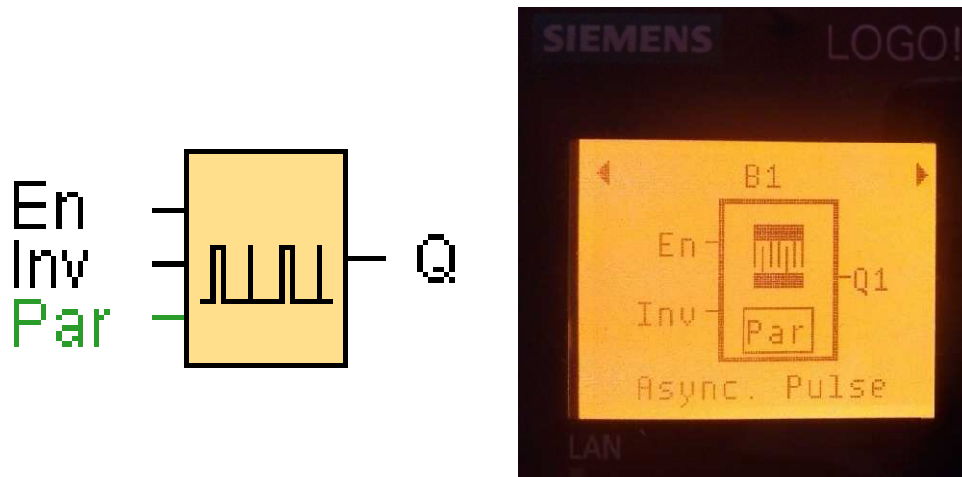


Figura 3.18 Símbolo del generador de impulsos asíncrono y como se visualiza en LOGO!8.

Y la salida **Q** se activa y desactiva cíclicamente con los tiempos impulso/pausa  $T_H$  y  $T_L$ . La forma del impulso de salida puede modificarse mediante una relación impulso/pausa configurable. La relación impulso/pausa se puede configurar en los parámetros TH (Time High) y TL (Time Low), ver figura 3.19. El ancho de impulsos  $T_H$  y la duración de pausa entre impulsos  $T_L$  también pueden ser el valor real de otra función preprogramada.

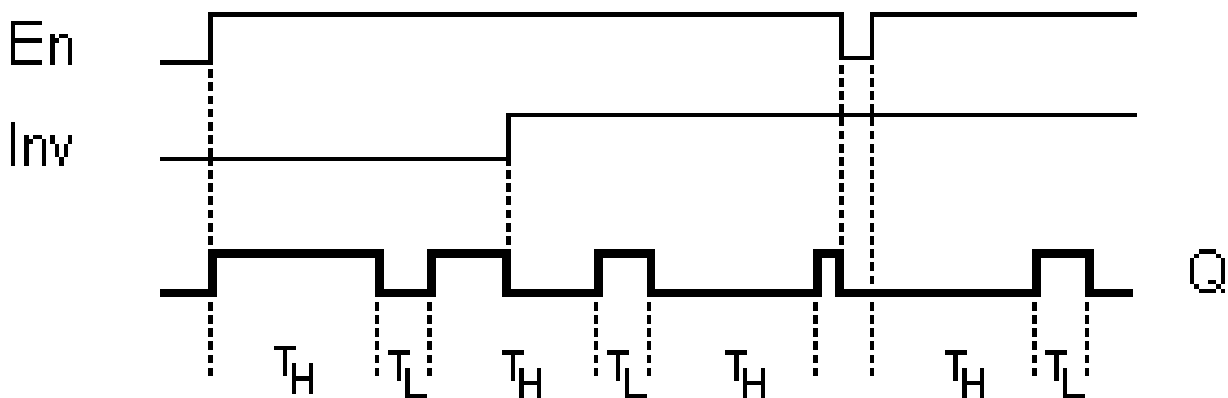


Figura 3.19 Cronograma del generador de impulsos asíncrono.<sup>23</sup>

<sup>23</sup> Imagen obtenida de <https://support.industry.siemens.com/cs/mdm/100782807?c=87556823179&dl=es&lc=de-DE>

### 3.5 LOGO!SoftComfort V8.0

Este software constituye una interfaz de usuario que permite una visualización de los menús de aplicaciones. El modo de programación en LOGO!SoftComfort V8.0 se inicia con un diagrama vacío.

La mayor parte de la pantalla la ocupa entonces el área dedicada a la creación de esquemas de conexiones. Esta área se denomina interfaz de programación (ver la figura 3.20).

En esta interfaz de programación se disponen los símbolos y enlaces del programa. Para no perder la vista de conjunto, especialmente en el caso de programas grandes, en los extremos inferior y derecho de la interfaz de programación se dispone de barras de desplazamiento que permiten mover el programa en sentido horizontal y vertical.

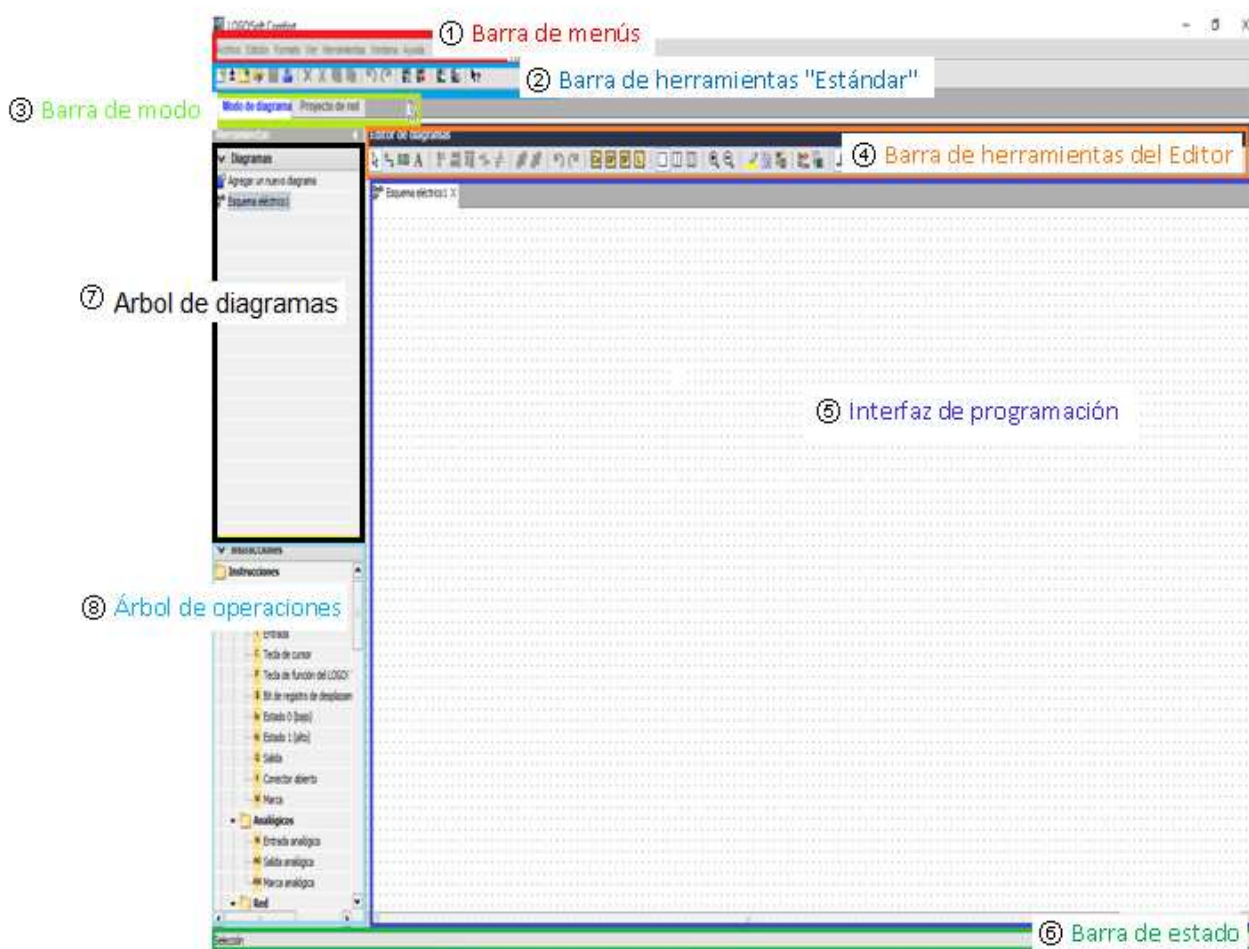


Figura 3.20 Interfaz de usuario de LOGO!SoftComfort V8.0

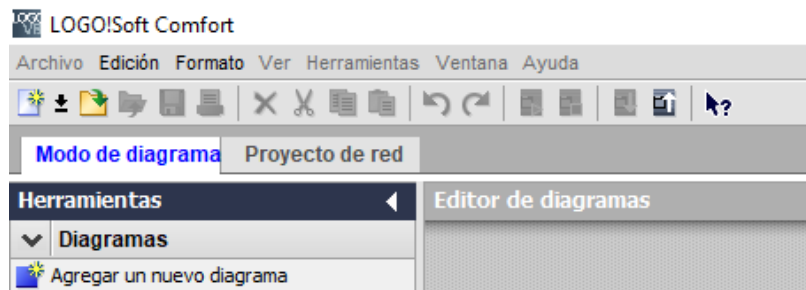
### 3.5.1 Iniciar LOGO!SoftComfort V8.0

Iniciar el software LOGO!SoftComfort V8.0 desde el acceso directo del escritorio de Windows (ver figura 3.21).

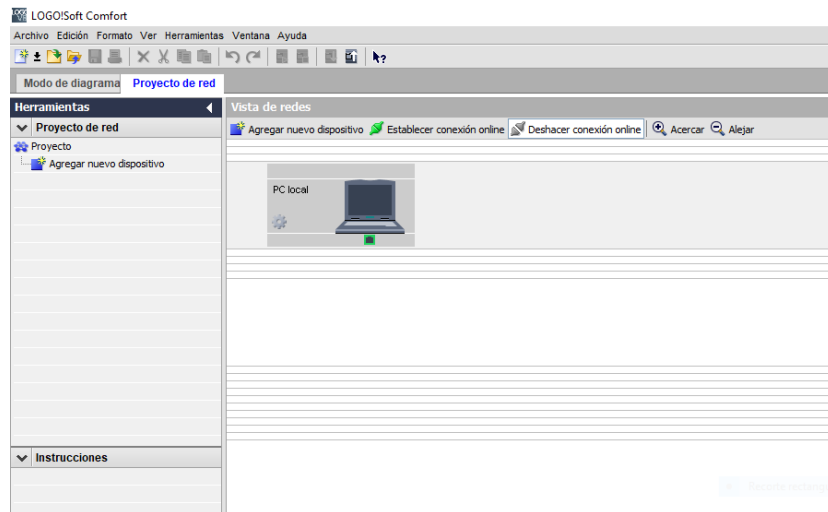


Figura 3.21 Acceso directo en escritorio de Windows para software LOGO!SoftComfort V8.0

Se abrirá en el modo de diagrama como se muestra en la ilustración 1 de la figura 3.22. Haga clic en la ficha proyecto de red y podrá visualizar el modo proyecto de red como se muestra en la ilustración 2 de la figura 3.22.



1



2

Figura 3.22 Modo de diagrama ilustración 1 y el modo proyecto de red ilustración 2

En el modo proyecto de red, haga doble clic en agregar nuevo dispositivo, aparecerá el cuadro de dialogo de la selección de dispositivos, escoja LOGO!OBA8 porque será el dispositivo que se utilizará en esta tesis. En la parte de configuración introduzca los ajustes de red, y confirme su selección con aceptar. Estos datos suelen aparecer ya predefinidos por el software tal y como se aprecia en la figura 3.23.

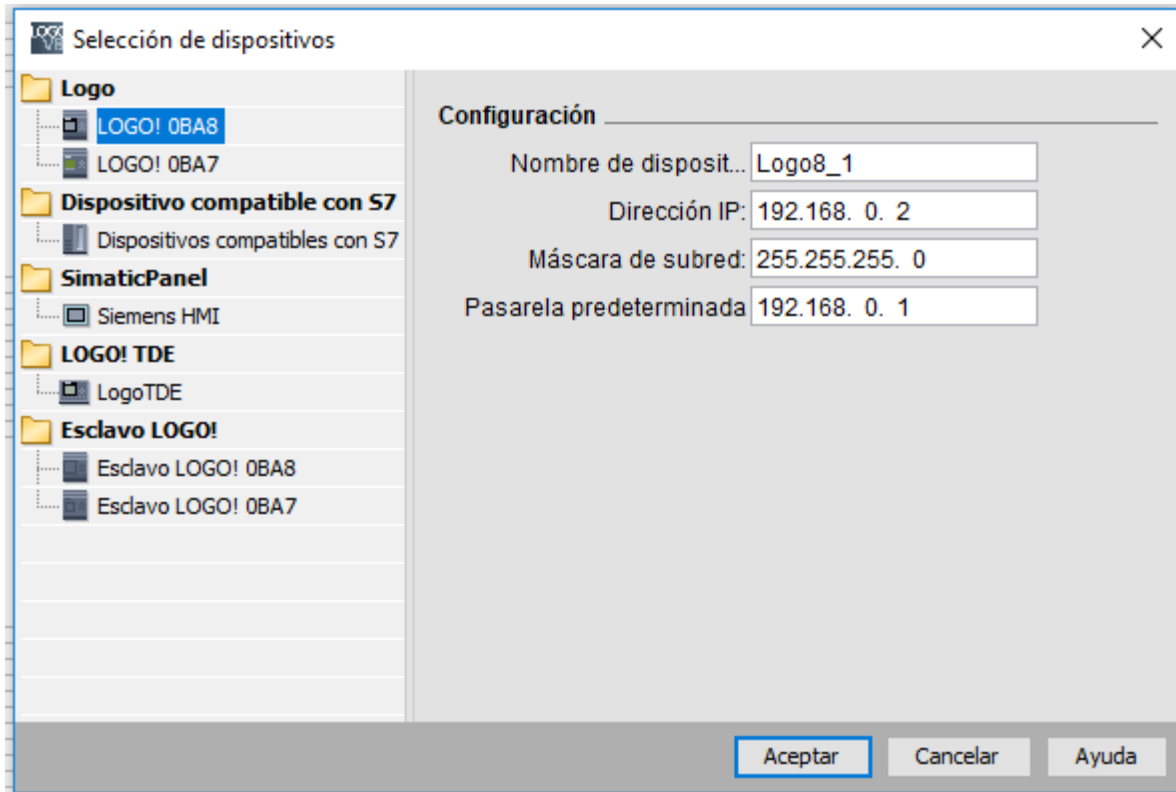


Figura 3.23 Proyecto de red en el software LOGO!SoftComfort V8.0

### 3.5.2 Programando mediante el software LOGO!SoftComfort V8.0

El sistema que se realizara debe controlar el acceso a través de un portón, garantizando la posibilidad de abrir y cerrar mediante un interruptor.

Para la automatización se utilizará un LOGO!OBA8. El portón se abrirá y cerrará por completo mediante un interruptor. Una luz intermitente se enciende 5 segundos antes de iniciar el movimiento y permanece encendida mientras se desplaza el portón. Por medio de una barra de seguridad se garantiza que, al cerrar el portón, no se lesionen personas ni se aprisionen objetos.

Estando en el modo proyecto de red en la sección árbol de diagramas de la interfaz de programación seleccionamos Logo8\_1diagrama y mediante el despliegue del menú contextual dando clic con el botón derecho del mouse seleccionamos cambio de nombre asignándole “Portón industrial”.

En esta misma sección árbol de diagramas de la interfaz de programación de este software se encuentra la opción ajustes haciendo doble clic accedemos al cuadro de dialogo configuración de LOGO!8, aquí puede editar todos los ajustes offline/online de LOGO OBA8,ver figura 3.24.

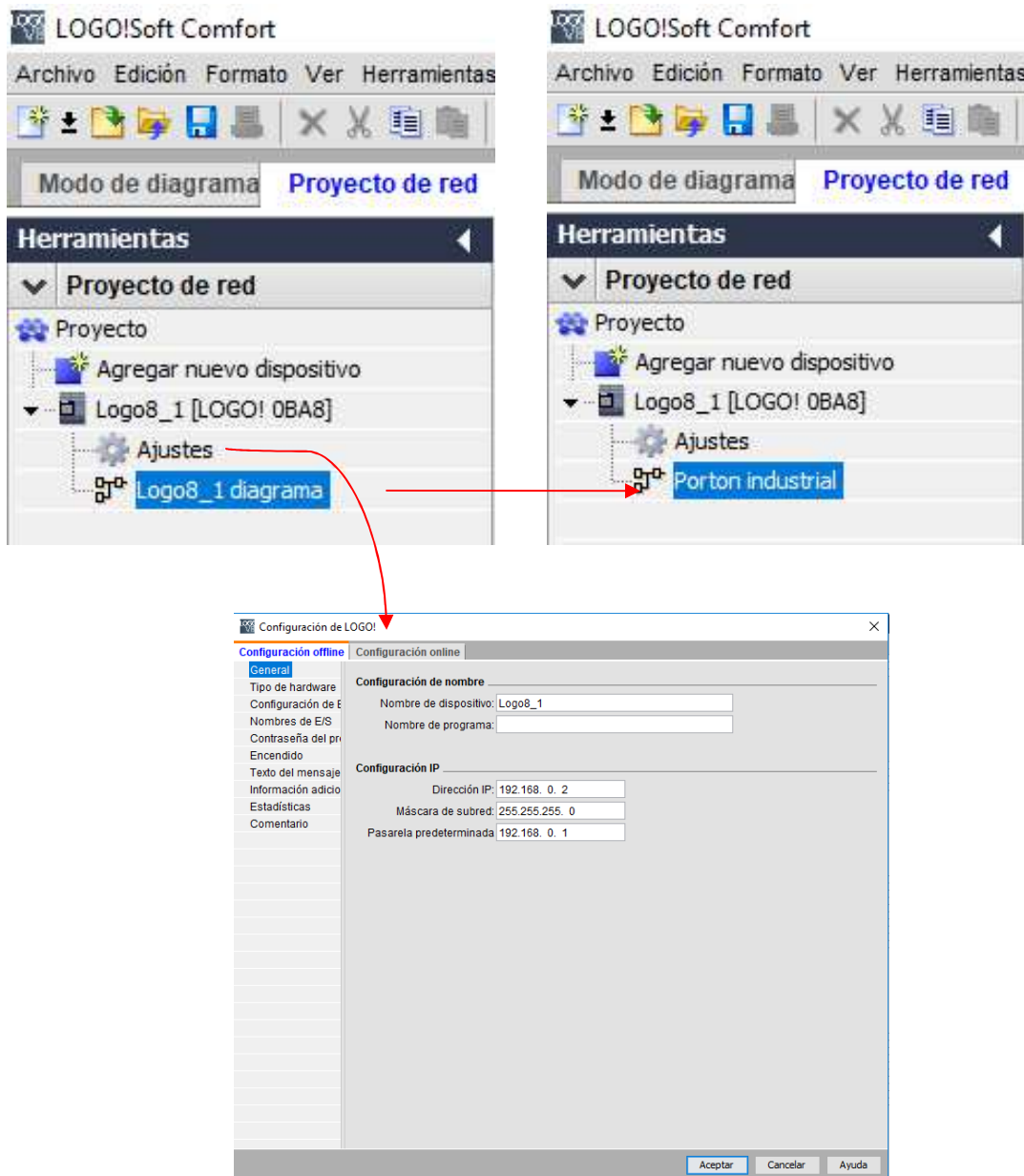


Figura 3.24 Asignación del nombre “Portón industrial” y acceso a Configuración de LOGO!8

Enseguida determinar el número de entradas digitales para la programación del portón automático quedando como se aprecian en la tabla 3.1.

ENTRADA DIGITAL	INTERRUPTOR	FUNCION	CONTACTOS NC/NO
I1	S0	Interruptor ABRIR PORTÓN	NO
I2	S1	Interruptor CERRAR PORTÓN	NO
I3	S2	Pulsador APERTURA MANUAL PORTÓN	NO
I4	S3	Pulsador CIERRE MANUAL PORTÓN	NO
I5	S4	Interruptor de posición PORTÓN ABIERTO	NC
I6	S5	Interruptor de posición PORTÓN CERRADO	NC
I7	S6	Barra de seguridad	NC

Tabla 3.1 Definición de las entradas digitales

De acuerdo con la tabla 3.1, dentro del modo de diagrama seleccionamos en la sección del árbol de operaciones en la carpeta digital la **I de entrada** la arrastramos hasta el editor de diagramas se mostrara como **I1** (ver figura 3.25) de esta misma forma con las demás entradas digitales colocándolas en el siguiente orden **I1, I3, I5, I2, I4, I6** e **I7**. Pulsando la tecla Ctrl+clic seleccionamos cada una de las entradas insertadas enseguida las alineamos con el botón “alinear verticalmente” que se encuentra en la barra de herramientas del editor de diagramas.

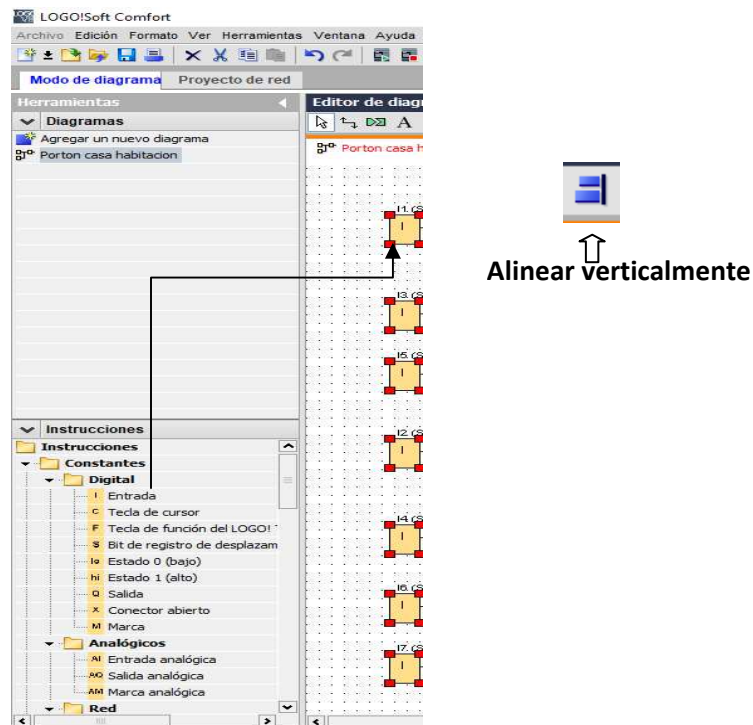


Figura 3.25 Colocación de entradas digitales I1, I3, I5, I2, I4, I6 e I7 y su alineación vertical.

Al dar un clic “distribuir espacio verticalmente” aparece el cuadro de dialogo separación en donde podemos introducir una distancia de separación de 10 como se aprecia en la figura 3.26, damos clic en aceptar.

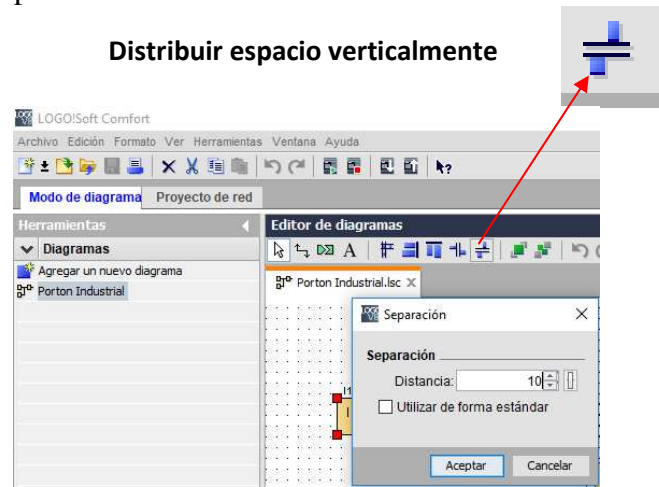


Figura 3.26 Opción de distribuir espacio verticalmente mediante el cuadro de dialogo separación.

Posteriormente determinar las salidas digitales quedando como se aprecian en la tabla 3.2.

SALIDA DIGITAL		FUNCION
Q1	K1	Abrir contactor principal
Q2	K2	Cerrar contactor principal
Q3	H1	Lampara de señalización

Tabla 3.2 Definición de las salidas digitales

De acuerdo con la tabla 3.2 elegimos de la sección del árbol de operaciones la Q de salida digital arrastramos **Q1, Q2 y Q3** con el mouse al editor de diagramas y las colocamos como se muestra en la figura 3.27. Seleccionamos las salidas insertadas mediante **Ctrl+clik** enseguida las alineamos con el botón **alinear verticalmente** y mediante **distribuir espacio verticalmente** introducimos para la distancia de separación un valor de 180 y damos clic en aceptar como se aprecia en la figura 3.27.

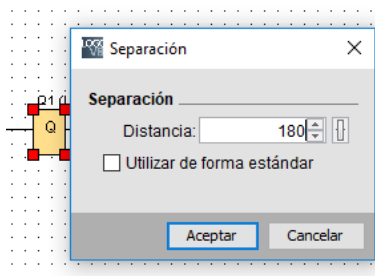


Figura 3.27 Salidas Q1, Q2 y Q3, alineación y distribución de espacio verticalmente.

Seleccione y haga doble clic en **B001** (retardo a la conexión) para el ajuste de tiempo a **5** segundos. (Ver figura 3.28)

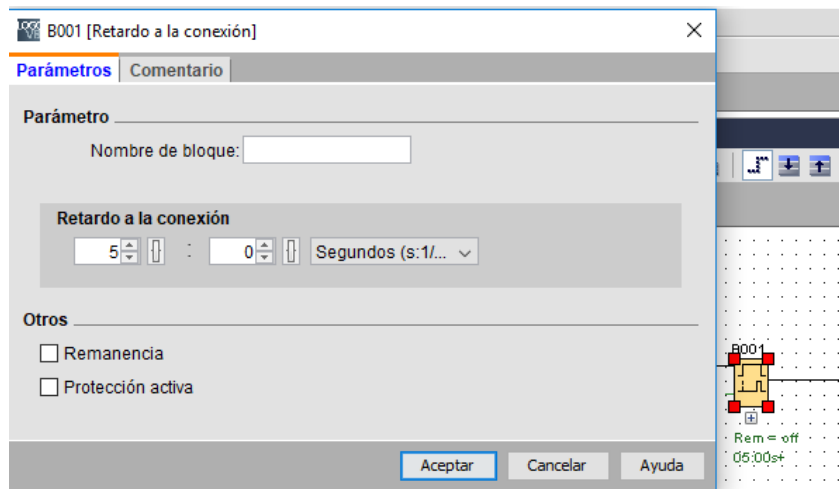


Figura 3.28 Temporizador retardo a la conexión y su cuadro de diálogo para su ajuste de tiempo.

Para introducir los nombres de las conexiones se puede realizar a través del menú edición y dando clic en nombre de conexiones. Introduzca apoyándose de los nombres definidos en las tablas 3.1 y 3.2, al término de la identificación de los bornes de entradas y bornes de salidas, cierre la ventana de configuración de LOGO! (ver figura 3.29) con aceptar.

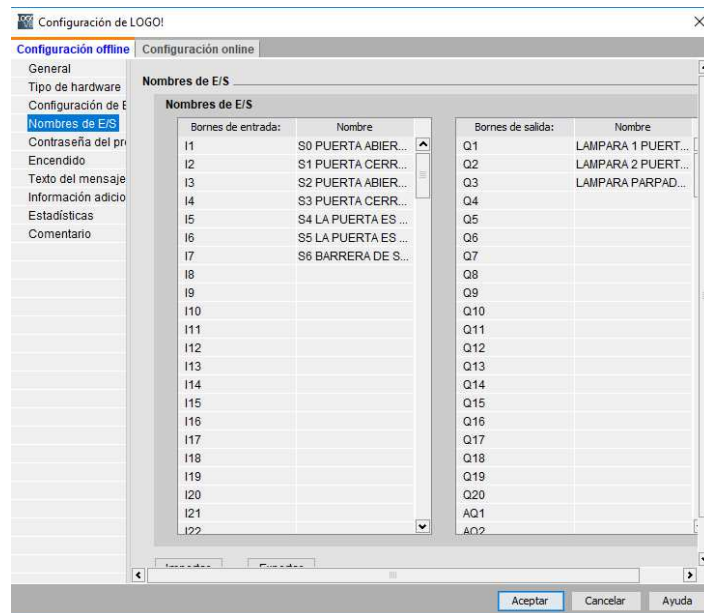



Figura 3.29 Asignando los nombres de las conexiones de acuerdo con las tablas 3.1 y 3.2.



Al dar clic en aceptar las entradas y salidas ahora se muestran con el nombre de borne asignado tal y como se aprecia en la figura 3.30. El seguimiento ahora consiste en colocar las siguientes funciones básicas; tres compuertas OR identificadas en la figura 3.30, como B002, B010 y B015, colocar 5 compuertas AND identificadas como B004, B006, B008, B011, B013, una compuerta NAND identificada como B014 una compuerta NOT identificada como B005, una compuerta XOR identificada como B009 y acomodarlas lo más parecido a la figura 3.30.

Enseguida colocar las siguientes funciones especiales; un generador de impulsos asíncrono identificado como B016, dos relés autoenclavador identificados como B003, B012, otro retardo a la conexión aparte de B001 que sería el B007 con un ajuste de tiempo de 5 segundos. Estas funciones se encuentran marcadas con cuatro cuadros rojos en la figura 3.30.

Para terminar el esquema de conexiones deben conectarse los distintos bloques entre si tal y como se muestra en la figura 3.30. Por lo tanto, en la barra de herramientas del editor de diagramas haga clic en el icono conectar  para ir conectando los bloques.

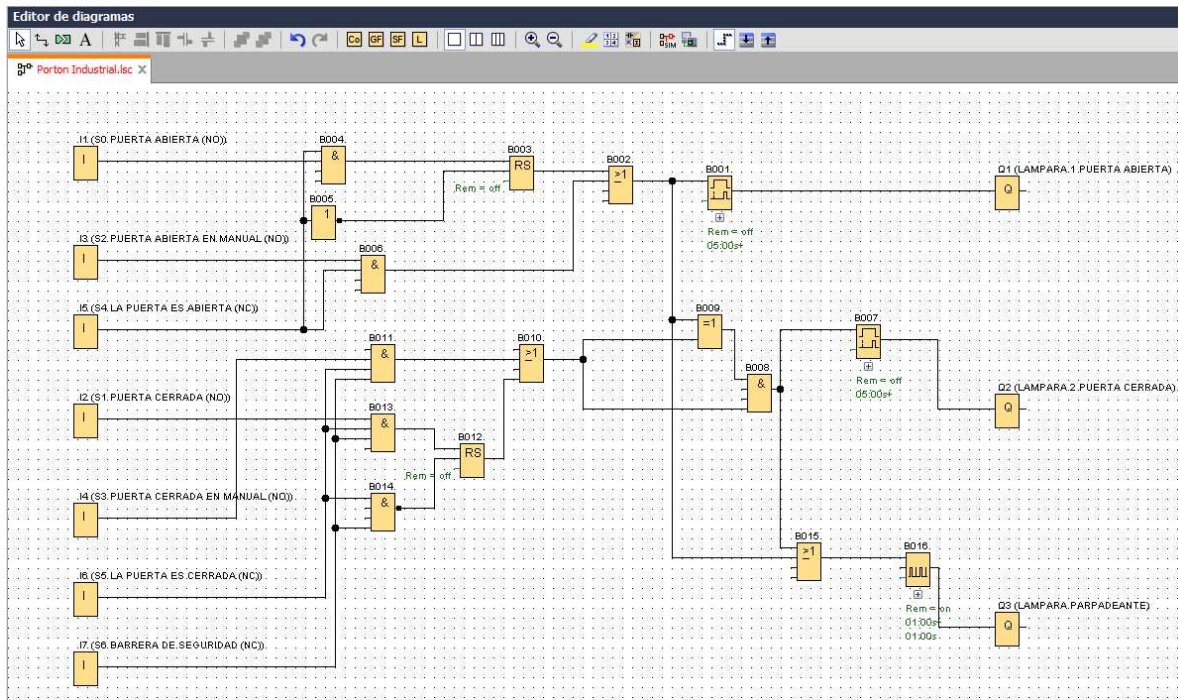



Figura 3.30 Inserción de funciones básicas y funciones especiales.

Al terminar el esquema de conexiones como se muestra en la figura 3.30 hacer clic en el icono del disquete , o mediante guardar como colocar el nombre del archivo portón industrial y dar clic en guardar ver figura 3.31.

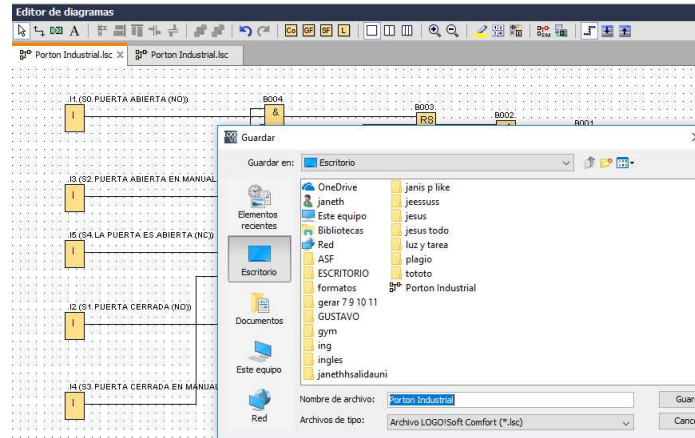


Figura 3.31 forma de guardar el esquema de conexiones terminado.

### 3.5.3 Simulación del circuito para el control de un portón industrial.

La simulación del programa nos permitirá comprobarlo y así poder garantizar que se transfiera al dispositivo LOGO!8 un programa con un funcionamiento óptimo. Para la simulación las señales de entrada deben preajustarse, mediante un doble clic sobre la entrada I1 se accede a su ficha y en la pestaña de simulación elija “pulsador (contacto normalmente abierto)” este mismo ajuste para las entradas I2, I3 e I4 en pulsador (NA).

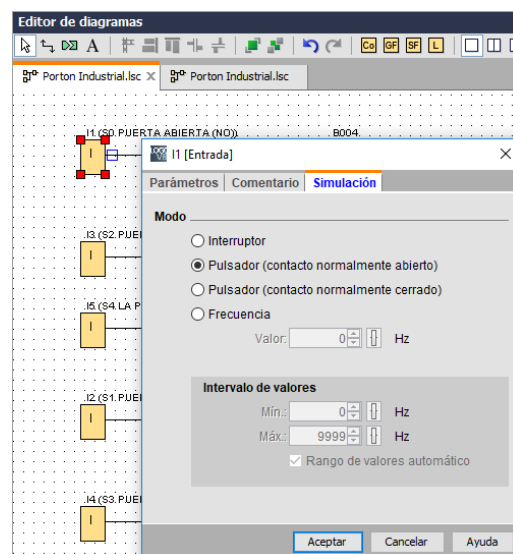


Figura 3.32 Ajustes para modo pulsador (contacto normalmente abierto)

Mediante un doble clic sobre la entrada I5 se accede a su ficha y en la pestaña de simulación elija “pulsador (contacto normalmente cerrado)” este mismo ajuste para las entradas I6 e I7 en pulsador (NC), al finalizar los ajustes guarde el esquema de conexiones.

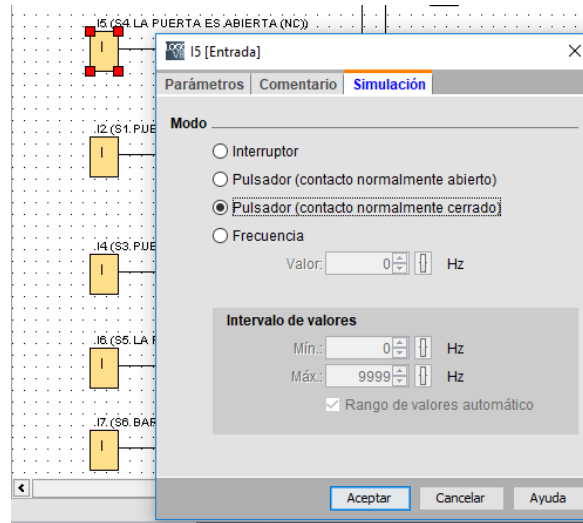



Figura 3.33 Ajustes para modo pulsador (contacto normalmente cerrado)

Para comenzar la simulación, haga clic en el icono simulación  de la barra de herramientas del editor del diagrama, esto nos dará el acceso al modo simulación.

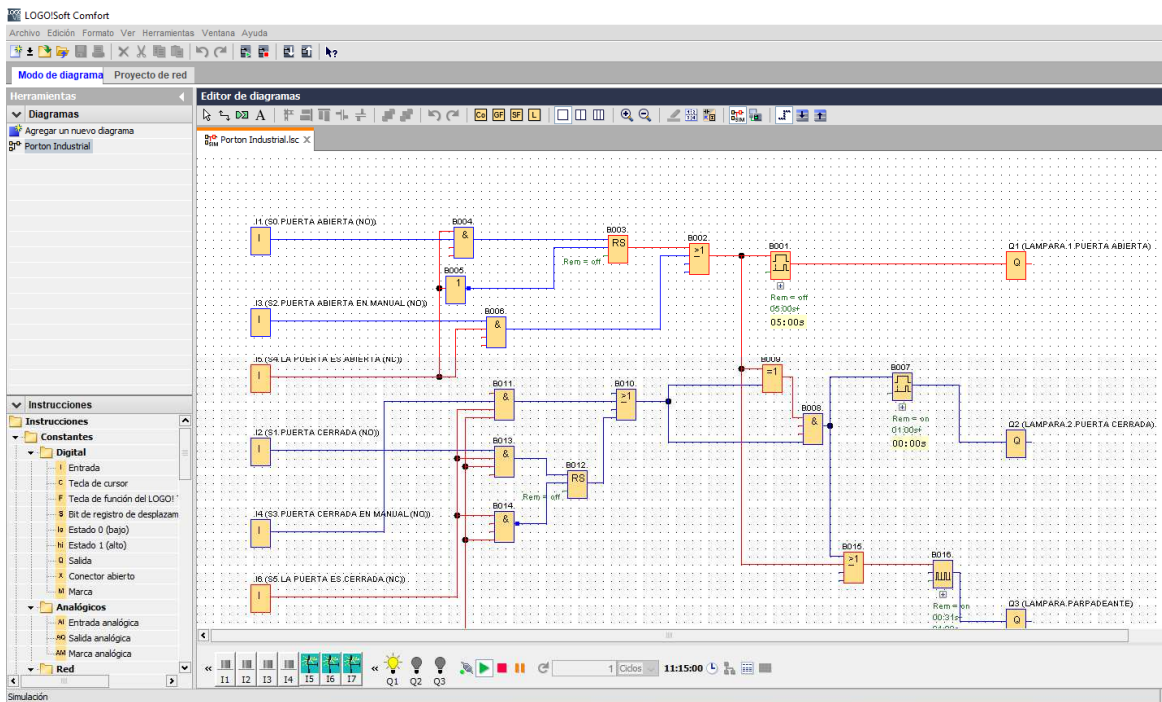


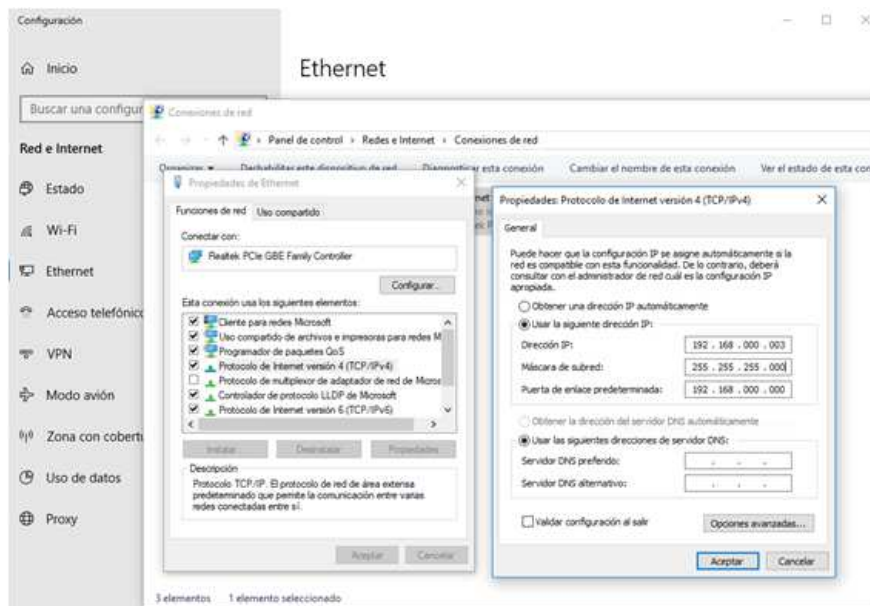
Figura 3.34 Modo simulación proyecto porton industrial

### 3.5.4 Transferencia de un programa a LOGO!8

Después de haber probado el programa en la simulación, podemos transferirlo del PC al PLC, por lo tanto primero energizamos el PLC LOGO!8 se puede observar su pantalla de menú principal, seleccionamos **Network**, enseguida nos posicionamos en **IP Address** y presionando “OK”(botón verde) aparecerán las direcciones que trae de fabricación el PLC LOGO!8 como se muestra en la ilustración 1 de la figura 3.35. Estas direcciones serán utilizadas para configurar la red Ethernet de la PC donde se escribirán de acuerdo a los espacios predefinidos como se muestra en ilustración 2 de la figura 3.35.




Ilustración 1



En la computadora escogemos la red identificada por la PC que es Ethernet, ingresamos a propiedades de Ethernet y nos dirigimos al protocolo de internet versión 4 (TCP/IPv4) y en esa ventana seleccionamos **usar la siguiente dirección IP**

Ilustración 2

Figura 3.35 Ilustración 1 acceso a Network en el PLC LOGO!8 e ilustración 2 configuración de red de área local en la PC

Para continuar desde la PC iniciamos el software LOGO!SoftComfort V8.0 y para comenzar la transferencia del programa desde el menú herramientas, dando clic en transferir y enseguida en la opción **PC a LOGO!** o mediante el teclado pulsando **Ctrl + D**, se mostrara la ventana de interfaz y mediante un clic en el **botón Actualizar**  para ver los dispositivos LOGO!8 accesibles siempre y cuando la configuración de Ethernet se haya realizado adecuadamente. Seleccionamos la dirección IP damos clic en probar obteniendo la palomita de transferencia adecuada tal y como se aprecia en la figura 3.36

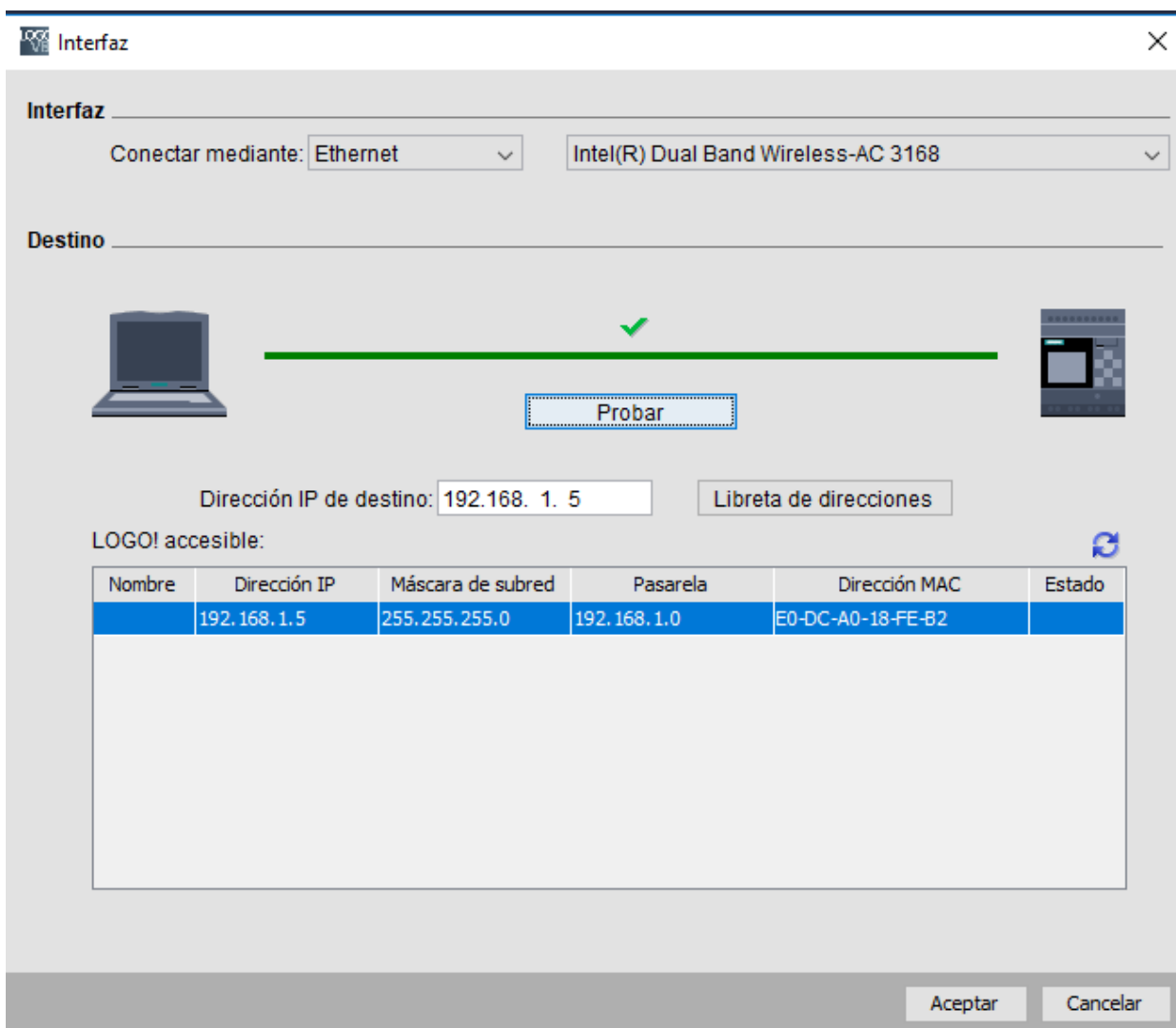


Figura 3.36 Ventana de Interfaz para transferencia del PC a LOGO!8

### 3.5.5 Practica 1 Semáforo secuencial

Primero seleccionamos una entrada I1 la configuramos dando doble clic y en la pestaña de simulación seleccionamos pulsador (contacto normalmente abierto) tal y como se muestra en la figura 3.37.

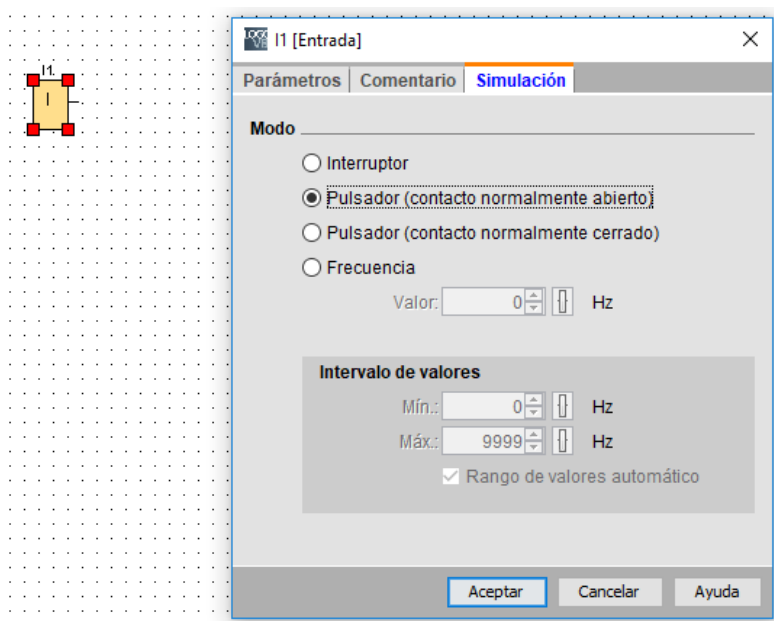


Figura 3.37 Configuración de la entrada I1 como contacto normalmente abierto

En la pestaña comentario anotamos el nombre de iniciar tal y como se aprecia en la figura 3.38 y damos clic en aceptar.

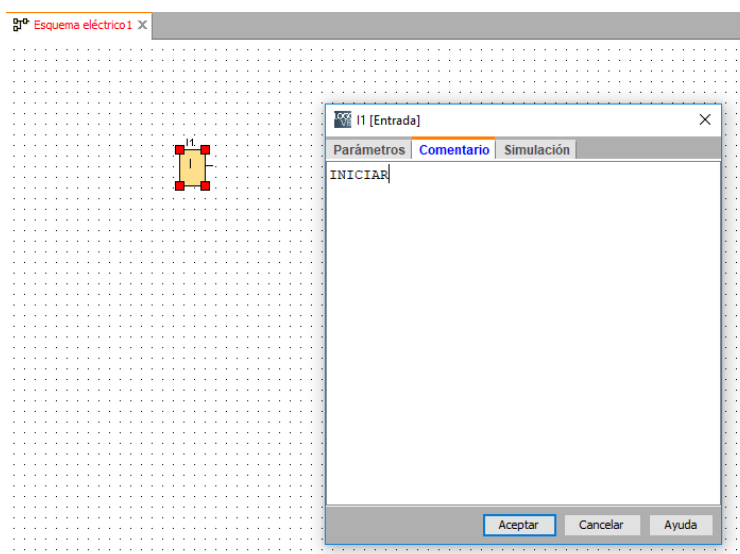


Figura 3.38 Asignación de nombre a la entrada I1

En segundo paso agregaremos un temporizador a la desconexión (B001) y conectamos la entrada I1 a la que nombramos INICIAR. Tal y como se muestra en la figura 3.39.

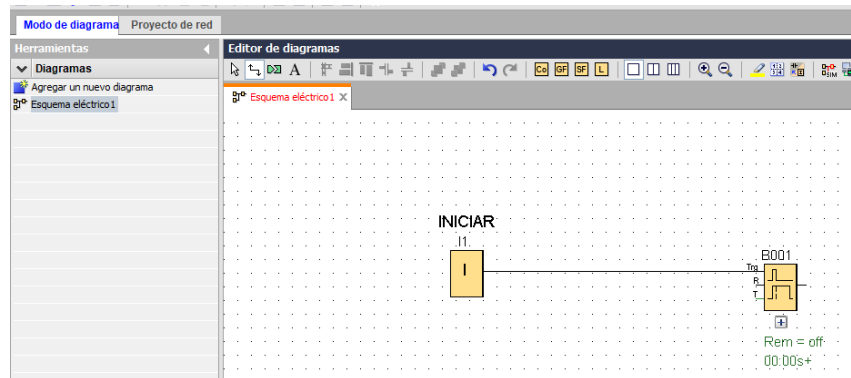


Figura 3.39 Conexion de entrada I1 al temporizador a la desconexion B001

Agregamos una salida Q1 y la salida del temporizador a la desconexión lo conectamos a esta salida. Tal y como se muestra en la figura 3.40.

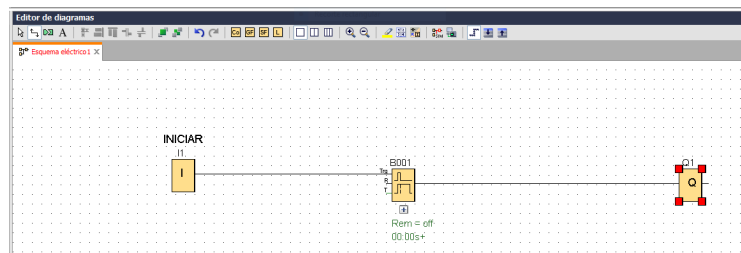


Figura 3.40 Conexion del temporizador a la desconexion B001 la salida Q1

Ahora agregamos un temporizador retardo a la conexión (B002) y un temporizador retardo a la desconexión (B003), y vamos a conectar la salida del temporizador B002 a la entrada del temporizador B003 y a conectar la salida del temporizador B001 a la entrada del B002 tal y como se muestra en la figura 3.41.

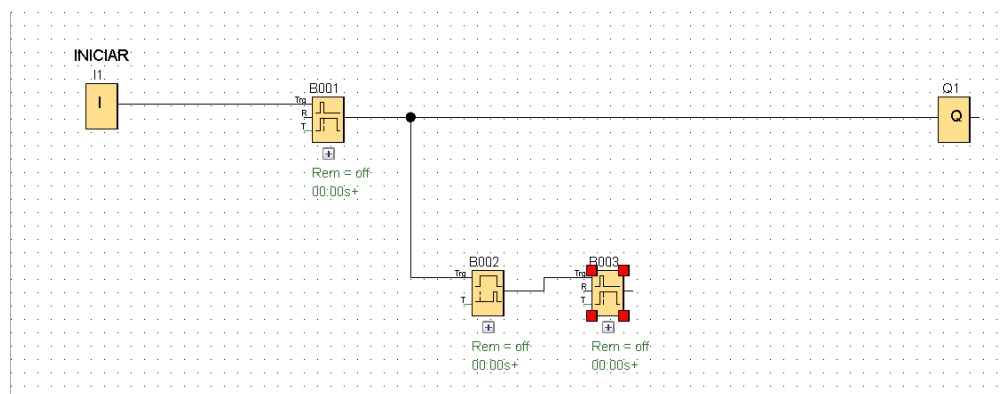


Figura 3.41 Insercion y conexion de las funciones B002 y B003

Enseguida vamos a insertar una salida Q2 y la salida del temporizador B003 la vamos a conectar a la salida Q2, y volvemos a insertar dos temporizadores uno con retardo a la conexión (B004) y otro con retardo a la desconexión (B005) y una salida Q3 y los conectamos tal y como se muestra en la imagen.

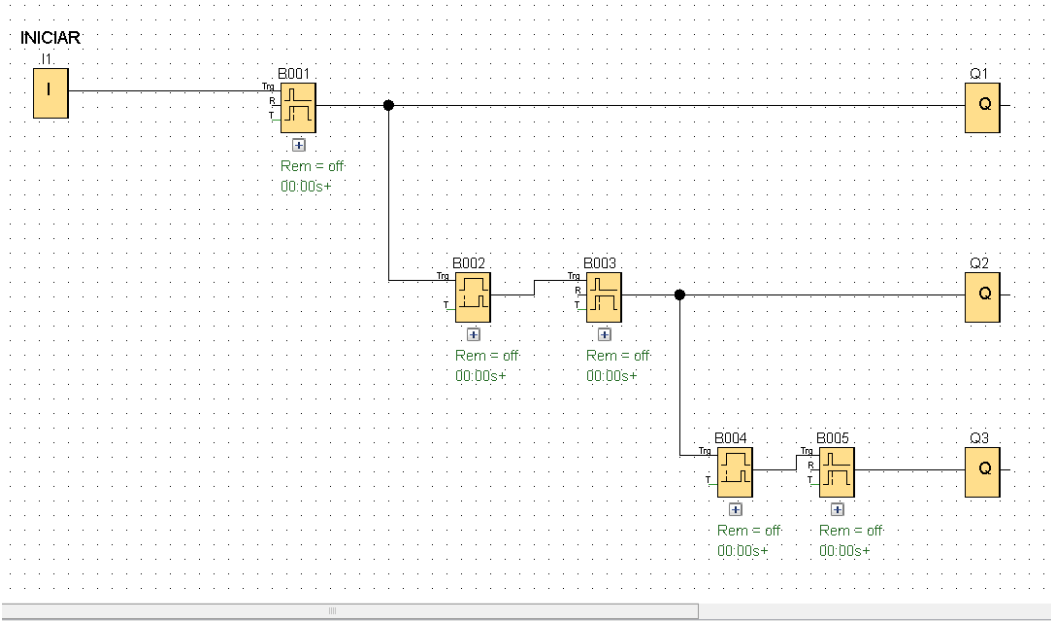


Figura 3.42 Insercion de Q2, Q3, B004 y B005

Ahora configuramos a 10 segundos el tiempo de los temporizadores, quedando de la siguiente manera tal y como se aprecian en la imagen.

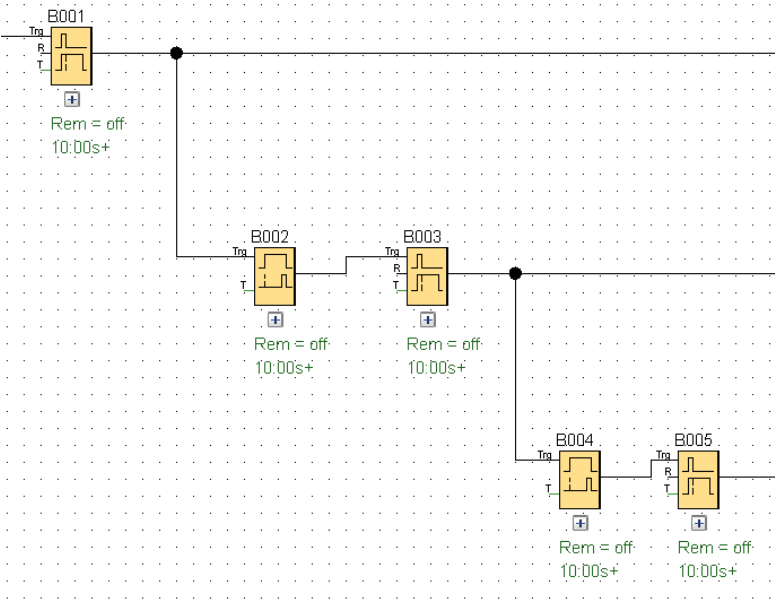


Figura 3.43 Insercion de Q2, Q3, B004 y B005



Podemos iniciar la simulación y lo que obtendremos será que una vez pulsado iniciar la salida Q1 se mantendrá encendida 10 segundos, posteriormente se apagará y se encenderá la salida Q2 de igual forma se mantendrá encendida 10s hasta apagarse y dar inicio al encendido de la salida Q3.

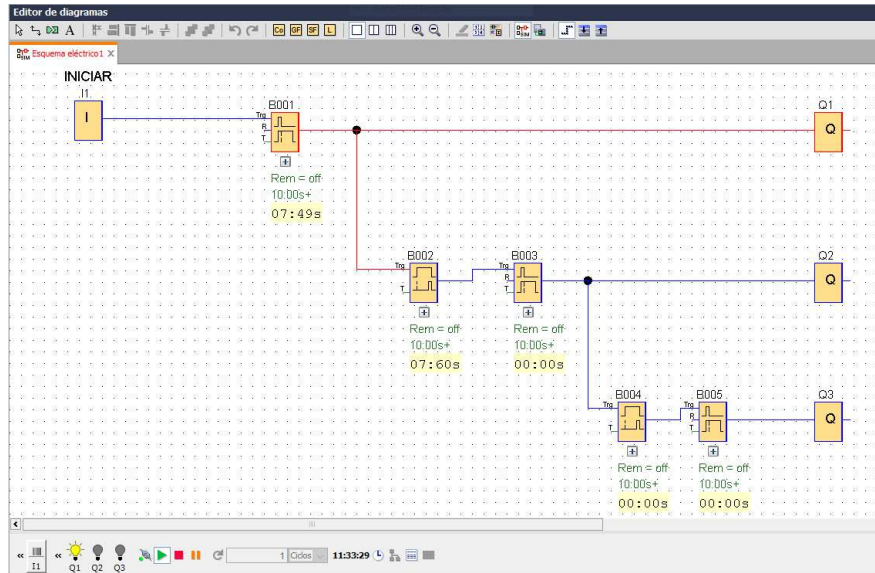


Figura 3.44 Simulación de programa semaforo secuencial

## Practica 2 semáforo secuencial con retroalimentación

Vamos a utilizar el diagrama de la practica 1 de semáforo secuencial pero ahora vamos a hacer algunas modificaciones. Primeramente, vamos a desconectar la salida Q2 e insertar una compuerta OR (B006) así como un temporizador a la conexión B007 y un temporizador a la desconexión B008. Tal y como se muestra en la figura 3.45.

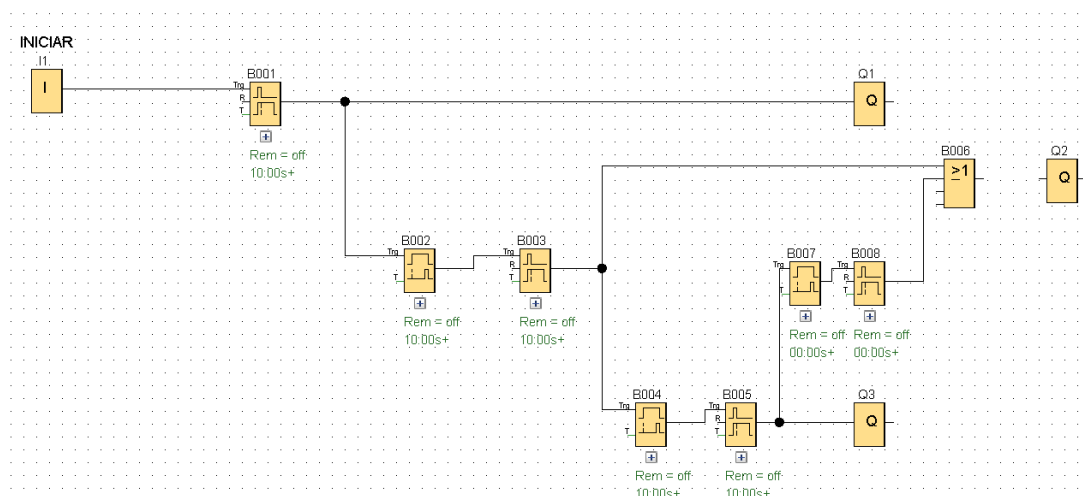


Figura 3.45 Desconexión de Q2, e inserción de la compuerta OR(B006)

Enseguida vamos a insertar un temporizador a la conexión (B009) y procedemos a desconectar el botón de iniciar del temporizador B001 y vamos a insertar una función OR(B010). Enseguida insertamos una marca (M1) y conectamos la salida del temporizador B009 a la marca M1 y la salida de M1 a la entrada de la compuerta OR B010. Conectamos el temporizador B006 a la salida Q2 tal y como se muestra en la figura 3.46.

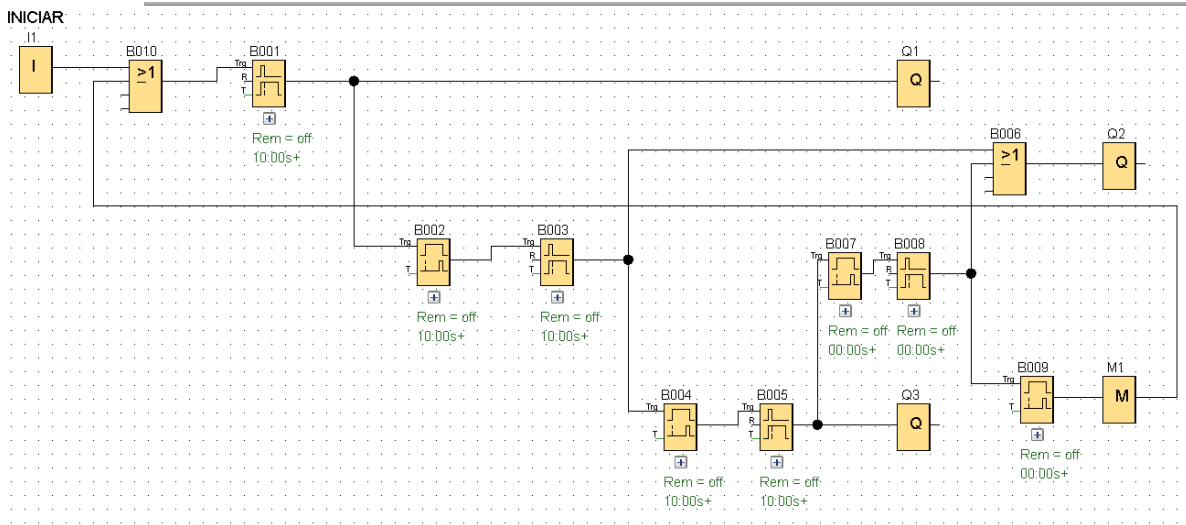


Figura 3.46 Insercion de B009, B010 y una marca M1

Ahora solo realizaremos un ajuste para la presentación final de este diagrama, utilizando la herramienta deshacer y unir conexión de la marca M1 a la compuerta OR B010 quedando tal y como se muestra en la figura 3.47.

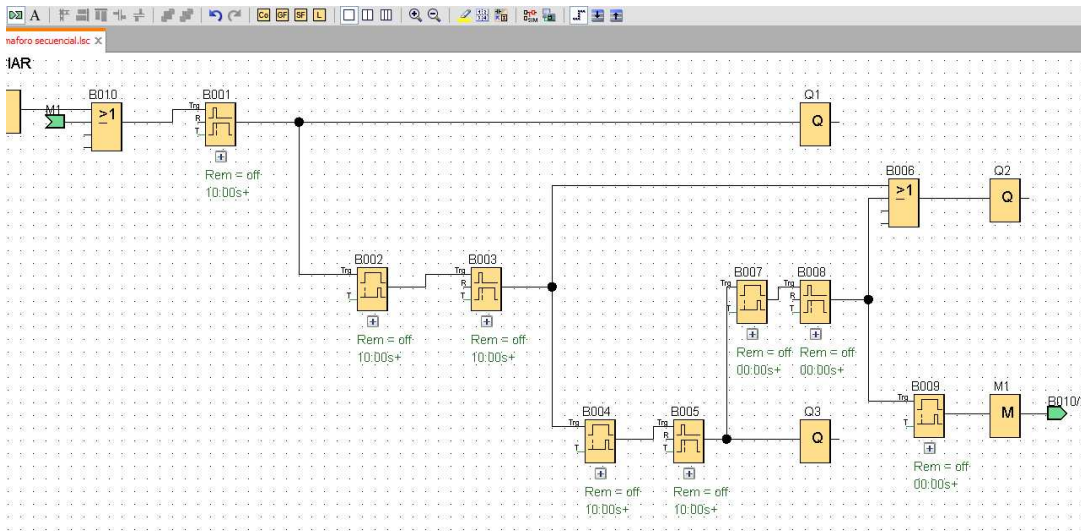


Figura 3.47 Conexion de la marca M1 a la compuerta OR B010

Ahora configuramos el tiempo de los temporizadores B007, B008 y B009 a 10 segundos y después de esto podemos mandar a simular el programa tal y como se muestra en la siguiente figura.

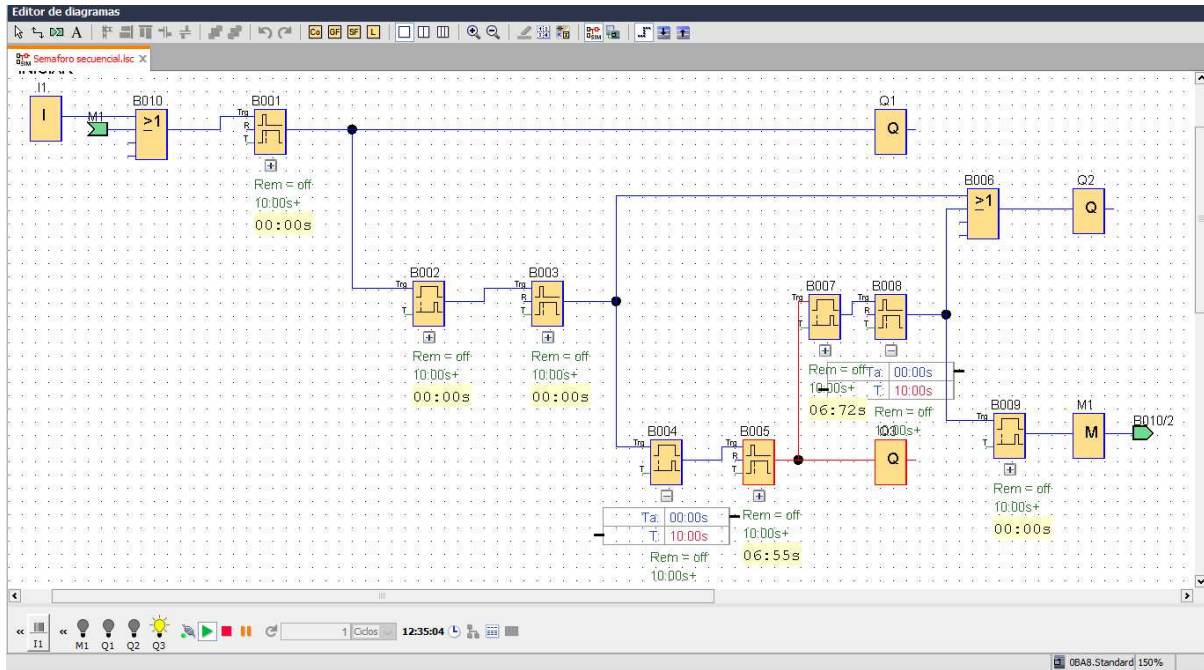


Figura 3.48 Simulación del programa semaforo secuencial

## **CONCLUSIONES**

Puedo concluir que al estudiar esta tesis se adquieren las nociones básicas de lo que es un PLC, su relación con el sistema de control programable y la programación en lenguaje gráfico: diagrama de escalera y diagrama de bloque de funciones.

De igual manera se adquirió un panorama actual de la familia Logix 5000 de Allen Bradley, así como de su software RSLogix5000 y con las practicas del mini PLC LOGO!8 de la marca SIEMENS, también conocido como modulo lógico. Se reafirman las nociones básicas para lógica de escalera mencionadas en el capítulo 2, pero aplicadas mediante bloques de funciones, demostrando que lo único que cambia es el tipo de fabricante. Cumpliendo así con los objetivos particulares de esta tesis.

## **BIBLIOGRAFIA**

Manual del estudiante RSLOGIX 5000 PROGRAMACION BASICA Rockwell  
Automatización

Manual de baja tensión control instalación y automatización. Catalogo Siemens

<http://allenbradleyplctutorials.blogspot.mx/2013/06/compact-logix-controller-system.html>

[http://www.rocatek.com/forum\\_rslogix.php](http://www.rocatek.com/forum_rslogix.php)

<https://www.siemens.com.mx/cms/mam/industry/Automatizacion/SIMATIC-sistemas-de-automatizacion-industrial/Pages/SIMATIC-sistemas-de-automatizacion-industrial.aspx>

[http://sappi.ipn.mx/cgpi/archivos\\_anexo/20082329\\_5942.pdf](http://sappi.ipn.mx/cgpi/archivos_anexo/20082329_5942.pdf)

<http://www.simbologia-electronica.com/simbolos-electricos-electronicos/simbolos-electronica-digital.htm>

[http://www.infoplcn.net/files/descargas/siemens/infoPLC\\_net\\_Datos\\_adjuntos\\_sin\\_t%C3%ADtulo\\_00048.pdf](http://www.infoplcn.net/files/descargas/siemens/infoPLC_net_Datos_adjuntos_sin_t%C3%ADtulo_00048.pdf)

[http://biblioteca.upnfm.edu.hn/images/directorios%20tematicos/xxtindustrial/libros%20de%20electricidad/Controles%20Electromecanicos/LOGO\\_PROGRAMA%20DE%20INSTALACIONES%20ELECTRICAS.PDF.pdf](http://biblioteca.upnfm.edu.hn/images/directorios%20tematicos/xxtindustrial/libros%20de%20electricidad/Controles%20Electromecanicos/LOGO_PROGRAMA%20DE%20INSTALACIONES%20ELECTRICAS.PDF.pdf)

<https://support.industry.siemens.com/cs/mdm/100782807?c=87556823179&dl=es&lc=de-DE>

<https://www.youtube.com/watch?v=TwU1qTg94vo>