UNIVERSIDAD DEL AZUAY

FACULTAD DE CIENCIA Y TECNOLOGÍA

ESCUELA DE INGENIERÍA ELECTRÓNICA

GUÍA DE PRÁCTICAS PARA AUTÓMATA PROGRAMABLE BASADO EN EL S7-200 Y EL EM-235

TRABAJO DE GRADUACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA

Autor

Francisco Eduardo Alvarado Correa

Director

Ing. Francisco Eugenio Vázquez Calero

TOMO 1

Cuenca - Ecuador

2010

DEDICATORIA

Dedico con mucho cariño éste trabajo a mis Padres.

A mis hermanos que de una u otra manera me han apoyado, especialmente a mi hermana que me supo dar la ayuda en esos momentos imponderables.

A mis hijos, para que lo tomen como un ejemplo a seguir y así puedan alcanzar sus más caros proyectos de vida.

AGRADECIMIENTO

En primera instancia deseo expresar mis agradecimientos a Dios por permitirme haber llegado a ésta meta propuesta.

A mis Padres, por su apoyo en todo lo que me he propuesto.

A todos los Profesores de la Facultad de Ciencia y Tecnología de la Universidad del Azuay, que supieron inculcar los principios fundamentales de la Ciencia y Tecnología, mezclados con valores de ética y responsabilidad.

Al Cuerpo de Profesores de la Universidad de Buenos Aires, Facultad de Ingeniería, FIUBA, que nos proporcionaron un complemento importantísimo en nuestra formación académica.

Al Ing. Francisco Vázquez Calero (*Director*), al Ing. Cristian Beltrán (*Técnico Siemens*), e Ing. Eduardo Maldonado (*Profesional externo*), por la ayuda brindada.

A Paul Peralta y Pedro Coronel por permitirme usar la información de su Trabajo de Graduación, para desarrollar una práctica de ésta Guía.

RESUMEN

En éste trabajo se plantea una Guía de Prácticas para el autómata programable S7-200 con sus módulos de ampliación, analógico EM235 y de comunicaciones CP243-1, que servirá de apoyo a los estudiantes de Ingeniería Electrónica dentro del Área de Automatización.

Además, mediante la interface gráfica HMI WinCC [SCADA] se visualiza la planta y se podría tomar alguna decisión sobre ella. La programación se efectúa mediante el MicroWin, pero en forma modular, apoyándose en el lenguaje de programación Grafcet. Lográndose totalmente la practicidad de lo planteado.

ABSTRACT

This work is a Guide of Practices for the Programmable Automaton S7-200 with its modules of extension, analogical EM235 and of communications CP243-1, which will use as support for the students of Electronic Engineering on the Automation Area.

In addition, by means of the graphical interface HMI WinCC [SCADA] the plant is visualized and it might take some decision on it. The programming is effected by means of the MicroWin, but in form modular , resting on the lenguage of programming grafcet. The practicality of the raised being achieved totally.

RESPONSABILIDAD

Las ideas y opiniones vertidas, sin perder la óptica y finalidad de los fabricantes [*propietarios de las marcas comerciales Siemens y LabView, y de los equipos aquí usados, respectivamente*], en éste trabajo de graduación son de exclusiva responsabilidad del Autor. Además, cada práctica se debe tomar como una posibilidad; y no como una solución definitiva.

Esto porque cada tema puede resolverse de otra manera, que optimice aun más el automatismo.

Francisco Eduardo Alvarado Correa

edalva01@hotmail.com

2010

INDICE DE CONTENIDO

TOMO 1

INDICE DE FIGURAS

CAPÍTULO 1: SESIONES DE FUNDAMENTACIÓN TEÓRICA

Sesión 1

Sesión 2

Sesión 3

Sesión 4

Sesión 5

Sesión 6

CAPÍTULO 4: PROYECTOS

CAPÍTULO 5: PRÁCTICAS DE DESAFÍO

INDICE DE TABLAS

Alvarado Correa Francisco Eduardo

Trabajo de Graduación

Vázquez Calero, Francisco, Ing.

Junio / 2010

GUÍA DE PRÁCTICAS PARA AUTÓMATA PROGRAMABLE BASADO EN EL S7-200 Y EL EM-235

INTRODUCCION

El avance de la tecnología ha hecho posible que cada vez se pueda manejar procesos más complejos, con las variables que se encuentran muy a menudo en la naturaleza. Lo que ha permitido establecer controles mucho más precisos sobre la gran mayoría de procesos que se desarrollan en la industria en general. Es por esto que la capacitación del personal que va a manejar dicha equipos debe estar a la altura de los mismos, para poder tomar decisiones acorde a la complejidad de la tecnología que se tiene.

Uno de los equipos que más auge ha tenido son los PLC o también llamados Autómatas Programables, que son empleados en controles simples como en los complejos. Siendo una de los tantos, que existen el mercado mundial, los de la Siemens, de la familia de los S7-200. Que presentan gran versatilidad de uso, ya que se puede ampliar colocando módulos I/O analógicos o digitales, de posicionamiento, de comunicaciones, de control para stepper motors, así como también pantallas táctiles, que conjuntamente con el software HMI WinCC permiten obtener detalles y datos que posibilitan la toma de decisiones importantes de la empresa.

Planteado ésto, la presente Guía de Prácticas pretende cubrir esa necesidad de información que todo Técnico requiere, sin pretender reemplazar a los manuales del fabricante. Comprende dos tomos:

- ✓ El primer Tomo es toda la información teórica necesaria, así como también los enunciados de las prácticas.
- ✓ El segundo Tomo son las respuestas a las prácticas plantadas.

Además, en el CD-1 se cuenta con información extra que le ayudará a entender varios temas. Y softwares que le servirán de apoyo para el estudio.

Capítulo 1: Sesiones de Fundamentación Teórica.

Iniciar al estudiante en los fundamentos teóricos necesarios e indispensables, para poder desarrollar una aplicación, con sesiones de trabajo interactivas. Basándose en lenguajes de programación, el conocimiento de los entornos de programación de cada software involucrado. Y por último, la implementación de una práctica demostrativa; con la finalidad de afianzar dicha teoría.

El tiempo estimado (*en minutos*) de cada sesión es:

Sesión 1:	El Autómata Programable o PLC	50
Sesión 2:	Entorno de Programación MicroWin	60
Sesión 3:	Lenguaje de Programación Grafcet	60
Sesión 4:	Entorno de Programación WinCC	80
Sesión 5:	Ejemplo de aplicación	160
Sesión 6:	Redes de Trabajo	60

Entonces, al finalizar éstas sesiones de trabajo, el Profesor elaborará un banco de preguntas para determinar si el estudiante ha asimilado los nuevos conceptos. Si es el caso, se pasará al siguiente capítulo [*que ya es eminentemente práctico*], de lo

contrario se tendrá que efectuar un *feedback*. Para lo cual, y salvando el mejor criterio del Profesor, se propone:

- Antes de empezar cada sesión de trabajo:
  - Realizar una serie dirigida de preguntas relativas al tema anterior y/o a las experiencias adquiridas de cada estudiante.
  - Determinar las incógnitas que tengan los estudiantes, y tratar de aclararlas, y si es posible concatenarlas con la nueva información.

- Antes de finalizar cada sesión de trabajo:
  - Efectuar preguntas dirigidas a investigar el grado de conocimientos adquiridos del estudiante.
  - Enviar trabajos de investigación, para complementar lo estudiado.

Capítulo 2: *Prácticas de Secuencias Básicas*.

Aquí se expondrá varios automatismos básicos generales, en los cuales el estudiante tendrá que implementarlos acorde a las exigencias de cada uno. Además, se plantean una serie de pasos que le servirán para obtener una solides de conocimientos cognitivos, aptitudinales y de procedimientos.

Capítulo 3: *Aplicaciones de Control*.

Siendo el punto de unión de los dos anteriores capítulos, vienen aquellos automatismos que tienden a encontrarse en la cotidianidad del control industrial. Estos deberán ser resueltos considerando la viabilidad y versatilidad de los equipos disponibles.

Capítulo 4: *Proyectos Varios*.

Este conjunto de prácticas le van a servir al Técnico de Automatización para poder diseñar un plan de trabajo para la resolución de cada una de ellas.

Despertando su iniciativa y sentido común. Dado que podrá verificar rutinas que ya hizo, y chequear si con un mínimo de esfuerzo pueden ser nuevamente implementadas en el nuevo control.

Capítulo 5: Prácticas de desafío [Proyectos finales]

Esta parte es fundamental, porque se pondrá a prueba la visión y proactividad del estudiante, puesto que se tendrá que realizar una investigación antes de desarrollar la práctica. Lo que permitirá unir los conocimientos adquiridos anteriores con los nuevos, generándole una matríz que le permitirá estar en capacidad de diseñar cualquier sistema de automatización con éste equipo.

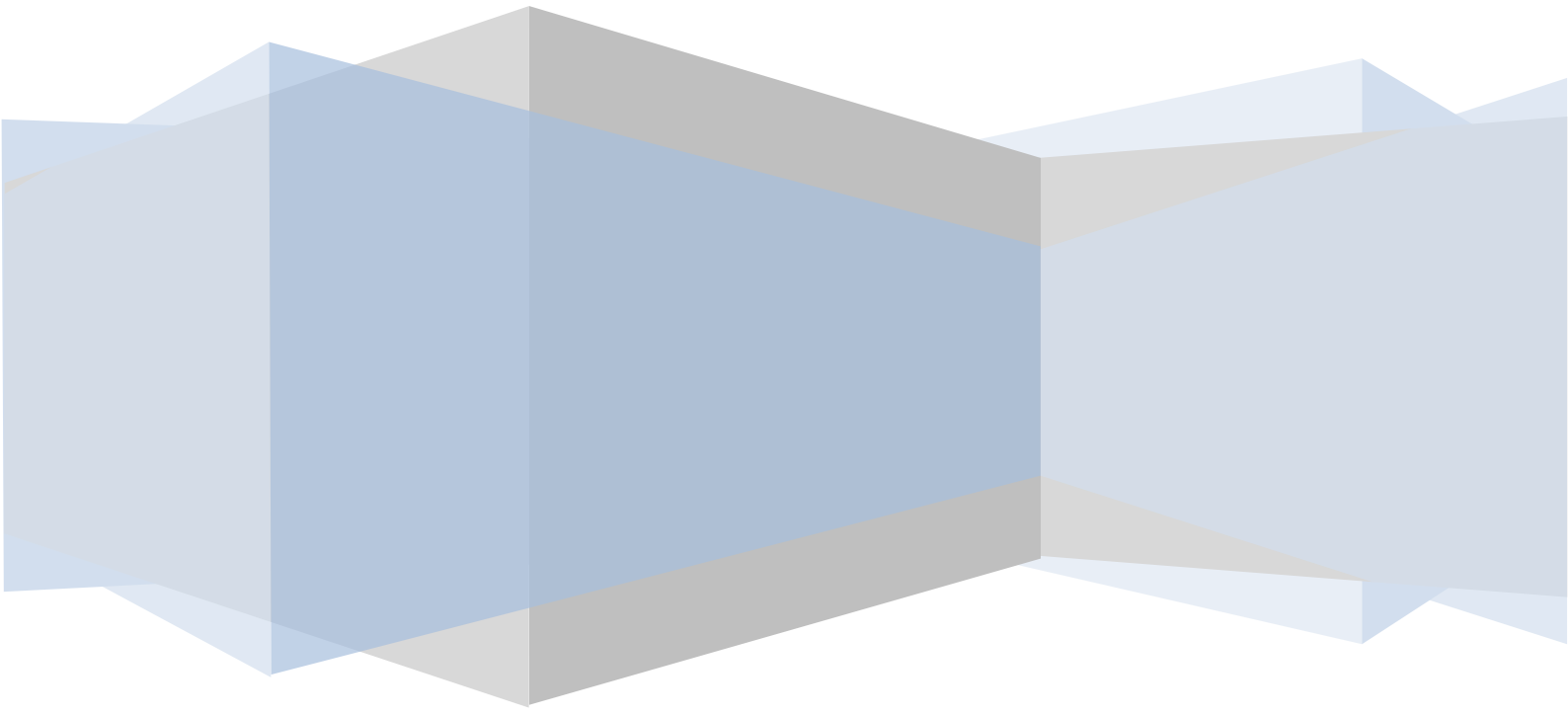
Los desafíos expuestos son iniciales, pudiendo adjuntarse otros conforme lo estime el Profesor, y estarán establecidos por el nivel de exigencia que el Catedrático creyera conveniente, además, de la capacidad de los estudiantes.

Ante todo lo expuesto, esta Guía facilitará la comprensión de la implementación de automatismos con el S7-200 y sus módulos de ampliación, con la interactividad de su SCADA. Logrando que el estudiante esté capacitado, y pueda migrar fácilmente a otro equipo.

"Todo buen trabajo se daña al no trabajar un poco más"

CAPITULO 1

Sesiones de Fundamentación Teórica



ESCUELA DE INGENIERIA ELECTRONICA
LABORATORIO DE AUTOMATAS PROGRAMABLES

GUIA DE PRACTICAS PARA AUTOMATA PROGRAMABLE

Sesión **1**

Tema **El Autómata Programable o PLC (Programmer Logic Controller)**

1.1 Introducción

El Autómata Programable, PA por su denominación en inglés, es un equipo capaz de ejecutar programas constituídos por una serie de instrucciones; ésto para cumplir tareas específicas. Poseen gran capacidad para manejar dispositivos de ingreso de señales, y controlar en sus salidas a otros dispositivos relacionados a las señales de ingreso.

Según la norma IEC 61131, se define a un PLC o PA así:

"*Un autómata programable es una máquina electrónica programable diseñada para ser utilizada en un entorno industrial (hostil), que utiliza una memoria programable para el almacenamiento interno de instrucciones orientadas al usuario, para implantar soluciones específicas. Tales como: funciones lógicas, secuencias, temporizaciones, recuentos y funciones aritméticas; con el fin de controlar, mediante entradas y salidas (digitales y análogas), diversos tipos de maquinas o procesos. (Figura 1.1)*"

Algunas cualidades dadas por su:

- Sistema embebido

  En la automatización permite reducir el espacio de elementos de control, minimizando el cableado en relación a su contraparte electromecánica.

Facilita grandemente el control, ya que por medio de su software de programación se obtiene una gran versatilidad de operaciones.

- Por su mantenimiento

  De igual manera maximiza el tiempo empleado en el mantenimiento preventivo o mantenimiento correctivo. El personal que maneja éstos equipos debe ser medianamente preparado. La interface hombre-máquina HMI es muy potente, facilitando la interacción.

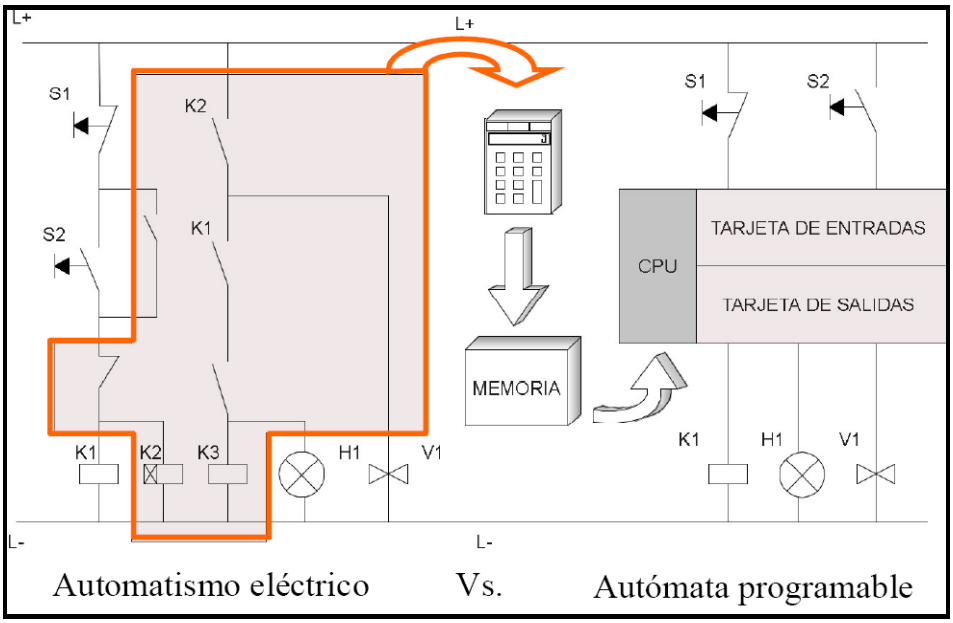


Figura 1.1: Definición gráfica de un PLC.

Fuente: González Víctor, Autómatas Programables (Visión General), Universidad de Oviedo, Programa de Estudios 2009 – 2010, publicación pdf, España.

1.2 Arquitectura de un PLC

Todo PLC puede ser subdivido acorde al hardware y software que maneja, dado por las características que le da su fabricante. A pesar de esto, se mantiene una composición genérica válida para todos éstos equipos.

Siendo:

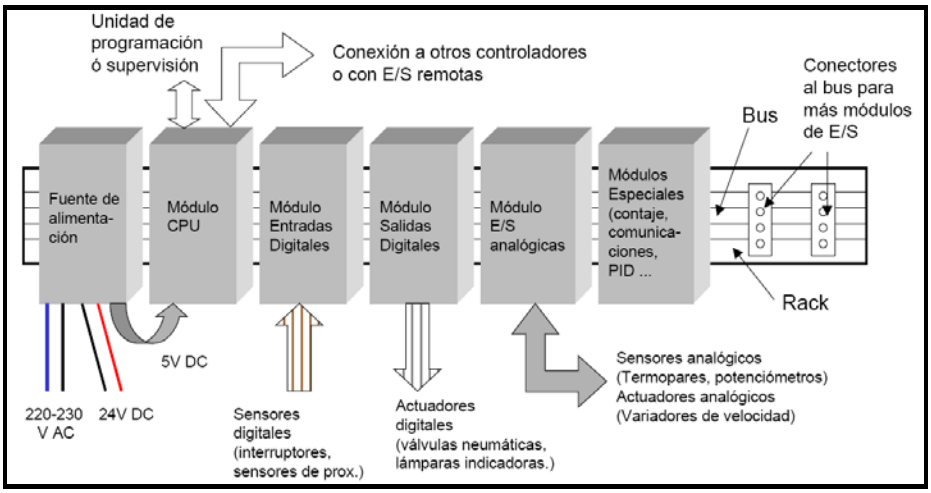


Figura 1. 2: Composición genérica de un PLC.

Fuente: González Víctor, Autómatas Programables (Visión General), Universidad de Oviedo, Programa de Estudios 2009 – 2010, publicación pdf, España.

- *La fuente de alimentación*, suple de energía eléctrica a todos los módulos para su funcionamiento. Puede ser conectada directamente a la red de VAC, en cuyo caso alimenta con VCD a los módulos; o ser directamente de VCD, para lo cual se requiere de un bloque transformador y acondicionador de voltaje AC/DC. Es importante dimensionar correctamente.

- *Bloque de la CPU*, es la que divide, procesa y decide. Esto en función de una programación lógica almacenada en su memoria. Es de una arquitectura similar a un ordenador, consta de un microprocesador, una unidad de memoria, una unidad aritmético lógica (ALU), y demás circuitos de apoyo para la interconexión con los módulos periféricos.

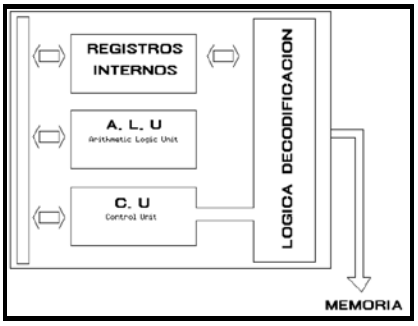


Figura 1. 3: Desglose de una CPU.

Fuente: Modesti, Mario Ing, Controladores de Lógica Programable, Universidad Tecnológica Nacional, publicación pdf, Córdova – Argentina, 2008.

- *Los Módulos de entrada/salida* son interfaces de acople entre los dispositivos de entrada/salida. Pudiendo ser digitales o analógicos.

- *Módulos especiales,* son los necesarios para tareas específicas especiales como: contaje de eventos de elevada frecuencia, PID, comunicaciones en varios estándares, posicionar ejes de motores, etc.

Considerando la forma constructiva, sus clases son:

Compacto: Tienen todos los módulos bajo un misma carcaza.

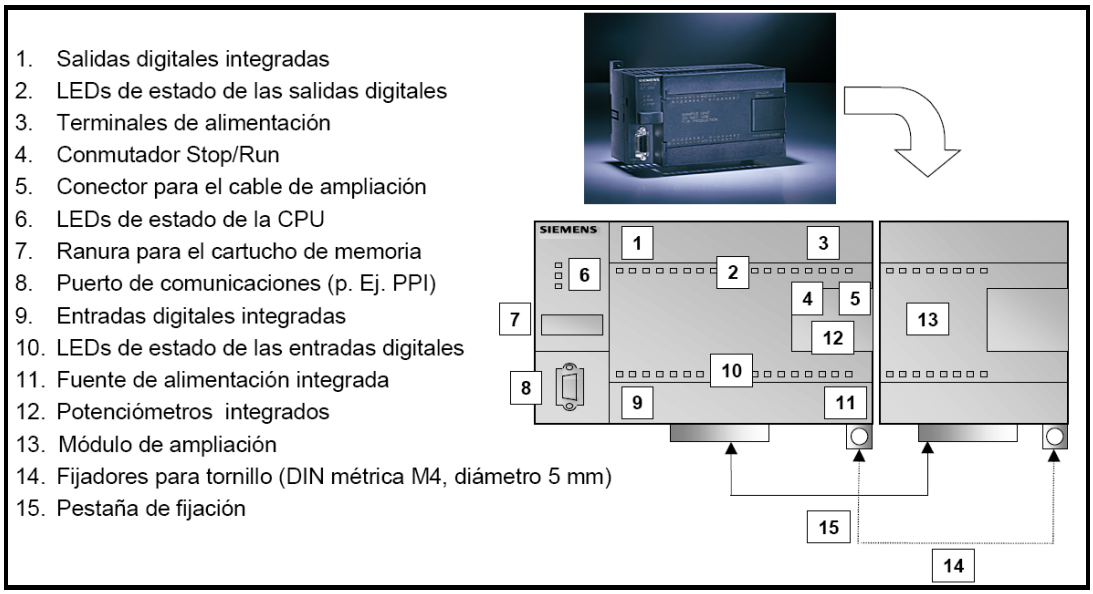


Figura 1.4: PLC´s comerciales compactos.

Fuente: González Víctor, Autómatas Programables (Visión General), Universidad de Oviedo, Programa de Estudios 2009 – 2010, publicación pdf, España.

Modulares: Módulos por separado, considerar compatibilidad al armar un PLC.

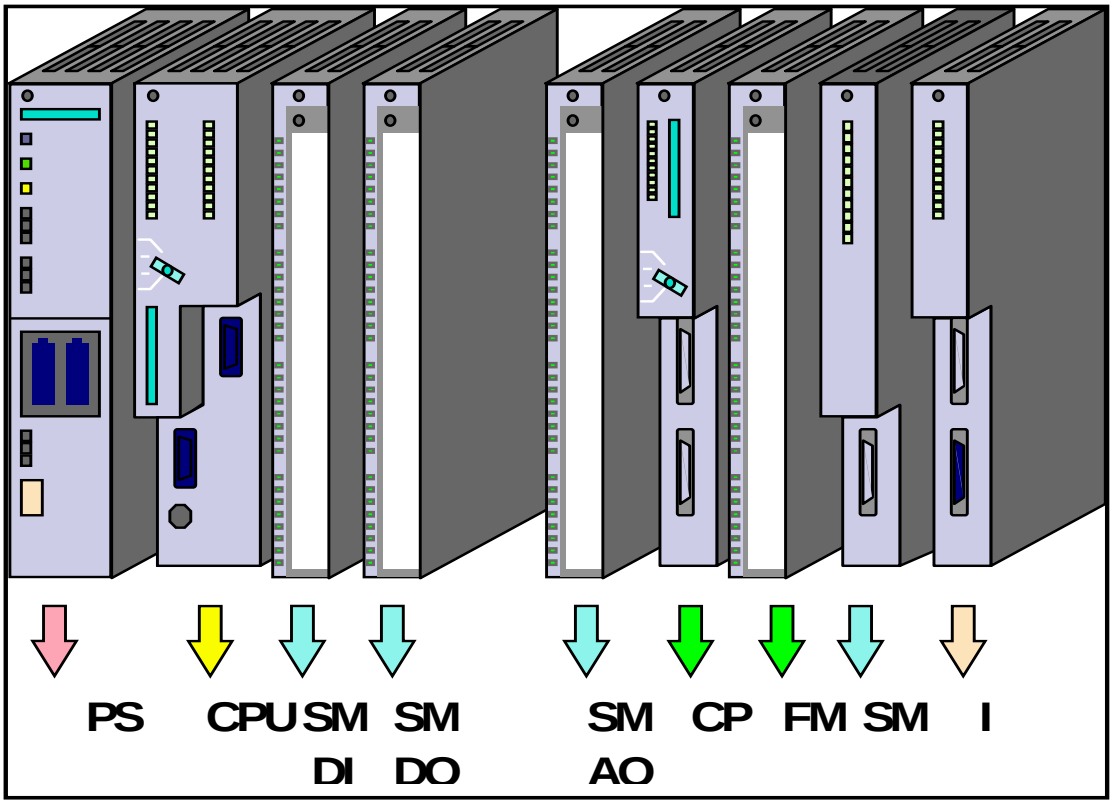


Figura 1.5: PLC´s comerciales modulares.

Fuente: http://www.isaatc.ull.es/asignaturas/Curso2003-2004/aindustrial/descargas/Clase1.ppt

1.3 Principio de operación

Considerando que el PLC está energizado y en modo RUN, se conoce como "*ciclo de scan*" a la ejecución cíclica del programa guardado en la memoria.

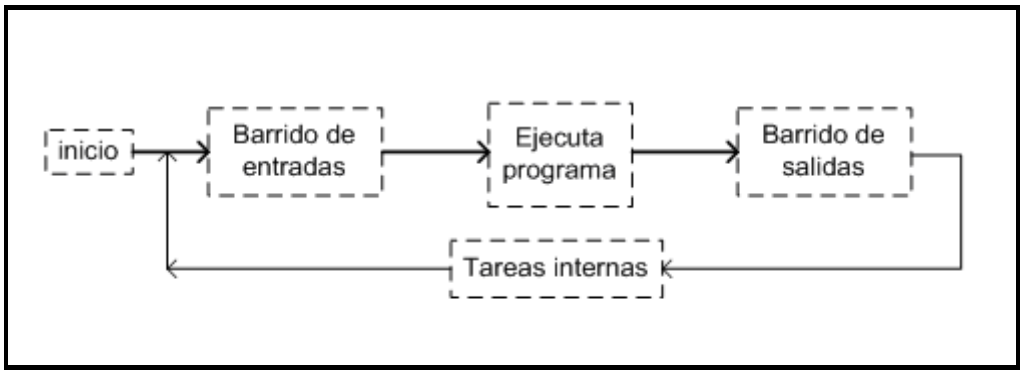


Figura 1.6: Ciclo de scan del PLC →barrido de entradas, actualizar salidas.

Fuente: Autor.

Al iniciar su funcionamiento, lee las entradas para luego ejecutar el programa; si detecta que hubo un cambio responde de acuerdo a su programación lógica. Luego actualiza las salidas con los cambios en memoria realizados, y efectúa sus propias tareas.

El tiempo de barrido es el *total empleado para ejecutar todas las operaciones internas del microprocesador, el watchdog timer que vigila que todo vaya correctamente; que se efectúen correctamente las actualizaciones de las entradas y de las salidas con sus imágenes respectivas.* Es un ciclo cerrado indefinido.

1.4 Autómata S7-200

La CPU S7-200 incorpora en una carcasa compacta un microprocesador, una fuente de alimentación integrada, así como circuitos interfaces de entrada y de salida que conforman un Micro-PLC. Luego de haber cargado el programa en el S7-

200, éste contendrá la lógica necesaria para observar y controlar los dispositivos de entrada y salida de la aplicación.

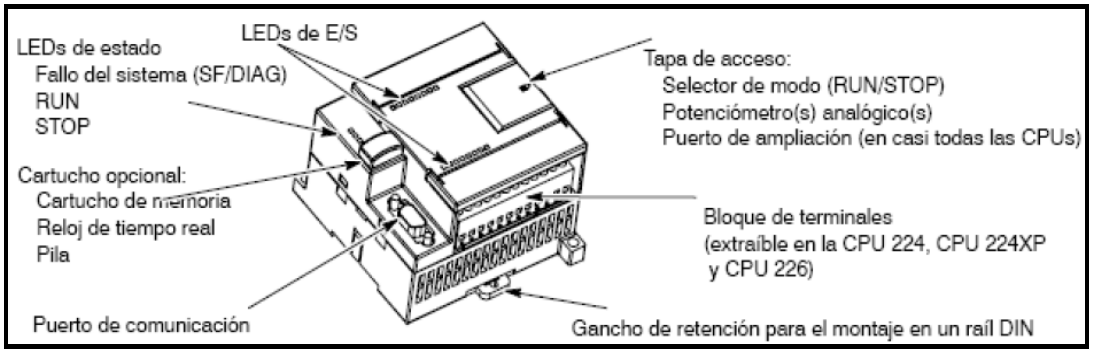


Figura 1.7: Constitución de un S7-200.

Fuente: Siemens, Manual del sistema de automatización S7-200, Germany – 08/2005.

El funcionamiento básico del S7-200 es muy sencillo:

- El S7-200 lee el estado de las entradas.
- El programa almacenado en el S7-200 utiliza las entradas para evaluar la lógica.
- Durante la ejecución del programa, el S7-200 actualiza los datos.
- El S7-200 escribe los datos en las salidas.

El resultado obtenido establece entonces el estado de la salida que corresponde a un actuador, el cual activa o desactiva la variable que está controlando. Al realizar la lectura de las entradas, procede:

Entradas digitales: Al principio de cada ciclo se leen los valores actuales de las entradas digitales y se escriben luego en la imagen del proceso de las entradas.

Entradas analógicas: El S7-200 no actualiza las entradas analógicas de los módulos de ampliación como parte del ciclo normal, a menos que se haya habilitado la filtración de las mismas. Existe un filtro analógico que permite disponer de una señal más estable. Este filtro se puede habilitar para cada una de las entradas analógicas.

Si se habilita la filtración de una entrada analógica, el S7-200 actualizará esa entrada una vez por ciclo, efectuará la filtración y almacenará internamente el valor filtrado. El valor filtrado se suministrará cada vez que el programa accede a la entrada analógica.

Si no se habilita la filtración, el S7-200 leerá de los módulos de ampliación el valor de la entrada analógica cada vez que el programa de usuario acceda a esa entrada.

Las entradas analógicas AIW0 y AIW2 incorporadas en la CPU 224XP se actualizan en cada ciclo con el resultado más reciente del convertidor analógico/digital. Este convertidor es de tipo promedio (sigmadelta) y, por lo general, no es necesario filtrar las entradas en el software.

Función	CPU 221	CPU 222	CPU 224	CPU 224XP	CPU 226
Dimensiones físicas (mm)	90 x 80 x 62	90 x 80 x 62	120,5 x 80 x 62	140 x 80 x 62	190 x 80 x 62
Memoria del programa con edición en runtime sin edición en runtime	 4096 bytes 4096 bytes	 4096 bytes 4096 bytes	 8192 bytes 12288 bytes	 12288 bytes 16384 bytes	 16384 bytes 24576 bytes
Memoria de datos	2048 bytes	2048 bytes	8192 bytes	10240 bytes	10240 bytes
Memoria de backup	50 horas (típ.)	50 horas (típ.)	100 horas (típ.)	100 horas (típ.)	100 horas (típ.)
E/S integradas Digitales Analógicas	 6 E/4 S -	 8 E/6 S -	 14 E/10 S -	 14 E/10 S 2 E/1 S	 24 E/16 S -
Módulos de ampliación	0 módulos	2 módulos[1]	7 módulos[1]	7 módulos[1]	7 módulos[1]
Contadores rápidos Fase simple Dos fases	 4 a 30 kHz 2 a 20 kHz	 4 a 30 kHz 2 a 20 kHz	 6 a 30 kHz 4 a 20 kHz	 4 a 30 kHz 2 a 200 kHz 3 a 20 kHz 1 a 100 kHz	 6 a 30 kHz 4 a 20 kHz
Salidas de impulsos (c.c.)	2 a 20 kHz	2 a 20 kHz	2 a 20 kHz	2 a 100 kHz	2 a 20 kHz
Potenciómetros analógicos	1	1	2	2	2
Reloj de tiempo real	Cartucho	Cartucho	Incorporado	Incorporado	Incorporado
Puertos de comunicación	1 RS-485	1 RS-485	1 RS-485	2 RS-485	2 RS-485
Aritmética en coma flotante	Sí				
Tamaño de la imagen de E/S digitales	256 (128 E / 128 S)				
Velocidad de ejecución booleana	0,22 microsegundos/operación				

Tabla 1.1: Características de las CPU´s S7-200.

Fuente: Siemens, Manual del sistema de automatización S7-200, Germany – 08/2005.

Las salidas que tienen pueden ser a relés o a semiconductor. Las I/O se pueden ampliar por medio de módulos:

Módulos de ampliación	Tipos		
Módulos digitales			
Entradas	8 entradas c.c..	8 entradas a.c.	16 entradas c.c.
Salidas	4 entradas c.c. 8 salidas c.c.	4 salidas de relé 8 salidas a.c.	8 salidas de relé
Combinación	4 entradas c.c. / 4 salidas c.c.	8 entradas c.c. / 8 salidas c.c.	16 entradas c.c. / 16 salidas c.c.
	4 entradas c.c. / 4 salidas de relé	8 entradas c.c. / 8 salidas de relé	16 entradas c.c. / 16 salidas de relé
Módulos analógicos			
Entradas	4 entradas	4 entradas termopar	2 entradas RTD
Salidas	2 salidas		
Combinación	4 entradas / 1 salida		
Módulos inteligentes	Posicionamiento Ethernet	Módem Internet	PROFIBUS-DP
Otros módulos	AS-Interface		

Tabla 1.2: Tipos de módulos de ampliación acoplables a la serie S7-2XX.

Fuente: Siemens, Manual del sistema de automatización S7-200, Germany – 08/2005.

Los almacenamientos de información se hacen en diferentes áreas de la memoria que tienen direcciones unívocas. Al explicitar la dirección de acceso el programa lo hace directamente a la información. (Tabla 1.3)

Representación	Byte (B)	Palabra (W)	Palabra doble (D)
Entero sin signo	0 a 255 0 a FF	0 a 65.535 0 a FFFF	0 a 4.294.967.295 0 a FFFF FFFF
Entero con signo	-128 a +127 80 a 7F	-32.768 a +32.767 8000 a 7FFF	-2.147.483.648 a +2.147.483.647 8000 0000 a 7FFF FFFF
Real IEEE de 32 bits en coma flotante	*No aplicable*	*No aplicable*	+1,175495E-38 a +3,402823E+38 (positivo) -1,175495E-38 a -3,402823E+38 (negativo)

Tabla1.3: Rangos en tamaño que tienen los datos.

Fuente: Siemens, Manual del sistema de automatización S7-200, Germany – 08/2005.

Para acceder a un bit (Fig. 1.8) en un área de memoria, es preciso indicar la dirección de éste compuesta por un identificador de área, la dirección del byte y el número del bit.

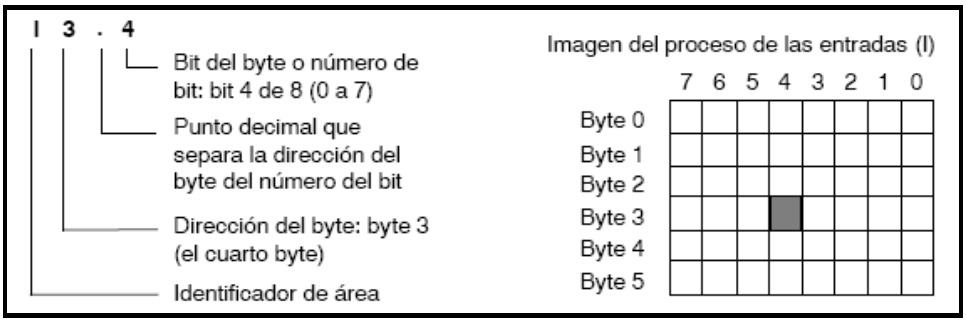


Figura 1.8: Formato de direccionamiento.- Llamado también "byte.bit".

Fuente: Siemens, Manual del sistema de automatización S7-200, Germany – 08/2005.

Utilizando el formato de dirección de byte se puede acceder a los datos de la mayoría de las áreas de memoria (V, I, Q, M, S y SM) en formato de bytes, palabras o palabras dobles. Estas se indicas de forma similar a la dirección de un bit. Esta última está compuesta por un identificador de área, el tamaño de los datos y la dirección inicial del valor del byte, de la palabra o de la palabra doble.

Para acceder a los datos comprendidos en otras áreas de la memoria, por ejemplo, T, C, HC y ACCs, es preciso utilizar una dirección compuesta por un identificador de área y un número de elemento. (Fig.1.9)

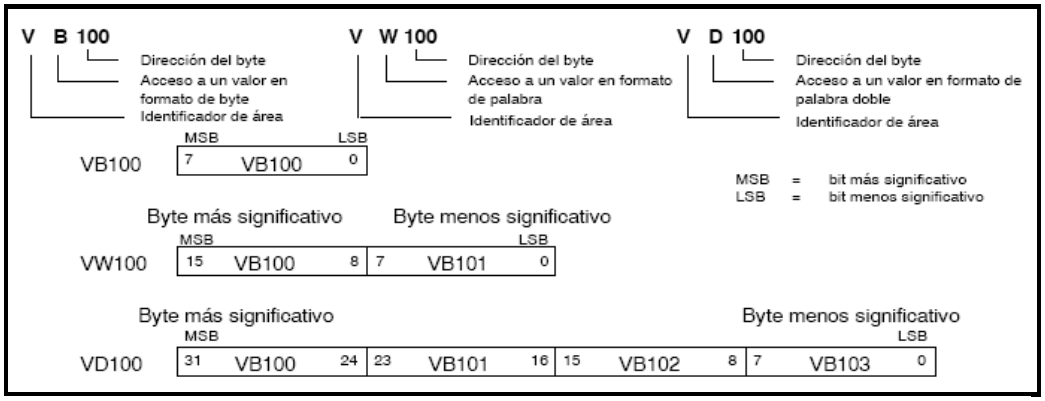


Figura 1.9: Formato de direccionamiento.- En byte, Word o Word Double.

Fuente: Siemens, Manual del sistema de automatización S7-200, Germany – 08/2005.

1.5 Módulo de expansión analógica EM-235

El modulo EM-235 es una interface de 4 entradas y 1 salida. Que permite conectar señales analógicas al autómata, y controlar una señal aplicada a un equipo a controlar.

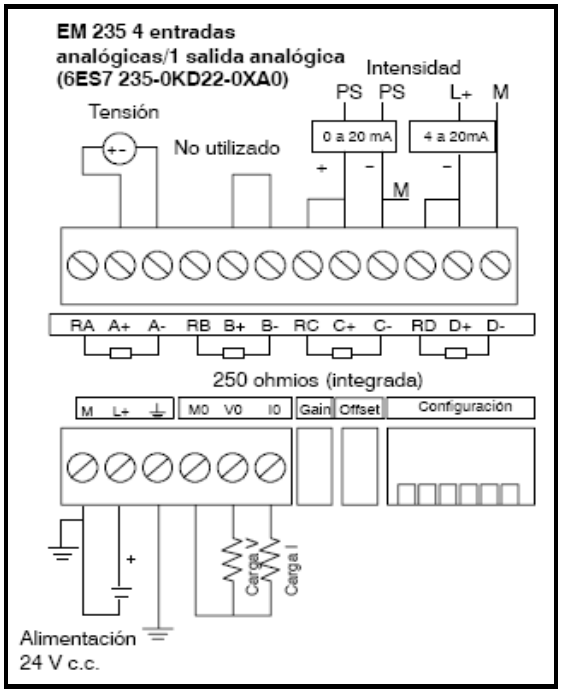


Figura 1.10: Conexiones del módulo analógico EM-235.

Fuente: Siemens, Manual del sistema de automatización S7-200, Germany – 08/2005.

Datos generales	6ES7 231-0HC22-0XA0	6ES7 235-0KD22-0XA0
Formato palabra de datos	(v. fig. A-14)	(v. fig. A-14)
Bipolar, rango máx.	-32000 a +32000	-32000 a +32000
Unipolar, rango máx.	0 a 32000	0 a 32000
Impedancia de entrada DC	≥10 MΩ entrada de tensión, 250 Ω entrada de intensidad	≥ 10 MΩ entrada de tensión, 250 Ω entrada de intensidad
Atenuación del filtro de entrada	-3 db a 3,1 kHz	-3 db a 3,1 kHz
Tensión de entrada máxima	30 V c.c.	30 V c.c.
Intensidad de entrada máx.	32 mA	32 mA
Resolución		
Bipolar	11 bits más 1 bit de signo	
Unipolar	12 bits	
Aislamiento (campo a circuito lógico)	Ninguno	Ninguno
Tipo de entrada	Diferencial	Diferencial
Rangos de entradas		
Tensión	Seleccionable (rangos disponibles, v. tabla A-20)	Seleccionable (rangos disponibles, v. tabla A-21)
Intensidad	0 a 20 mA	0 a 20 mA
Resolución de las entradas	V. tabla A-20	V. tabla A-21
Tiempo de conversión analógica/digital	< 250 µs	< 250 µs
Respuesta de salto de la entrada analógica	1,5 ms a 95%	1,5 ms a 95%
Rechazo en modo común	40 dB, c.c. a 60 Hz	40 dB, c.c. a 60 Hz
Tensión en modo común	Tensión de señal más tensión en modo común (debe ser ≤ ±12 V)	Tensión de señal más tensión en modo común (debe ser ≤ ±12 V)
Rango de tensión de alimentación 24 V c.c.	20,4 a 28,8 V c.c. (clase 2, potencia limitada o alimentación de sensores de la CPU)	

Tabla 1.4: Datos de las entradas de los módulos analógicos.

Fuente: Siemens, Manual del sistema de automatización S7-200, Germany – 08/2005.

Datos generales	6ES7 232-0HB22-0XA0	6ES7 235-0KD22-0XA0
Aislamiento (campo a circuito lógico)	Ninguno	Ninguno
Rango de señales		
Salida de tensión	± 10 V	± 10 V
Salida de intensidad	0 a 20 mA	0 a 20 mA
Resolución, rango máx.		
Tensión	12 bits más bit de signo	11 bits más bit de signo
Intensidad	11 bits	11 bits
Formato palabra de datos		
Tensión	-32000 a +32000	-32000 a +32000
Intensidad	0 a +32000	0 a +32000
Precisión		
Caso más desfavorable, 0° a 55° C		
Salida de tensión	± 2% de rango máx.	± 2% de rango máx.
Salida de intensidad	± 2% de rango máx.	± 2% de rango máx.
Típico, 25° C		
Salida de tensión	± 0,5% de rango máx.	± 0,5% de rango máx.
Salida de intensidad	± 0,5% de rango máx.	± 0,5% de rango máx.
Tiempo de ajuste		
Salida de tensión	100 µS	100 µS
Salida de intensidad	2 mS	2 mS
Accionamiento máx.		
Salida de tensión	Mín. 5000 Ω	Mín. 5000 Ω
Salida de intensidad	Máx. 500 Ω	Máx. 500 Ω
Rango de tensión de alimentación 24 V c.c.	20,4 a 28,8 V c.c. (clase 2, potencia limitada o alimentación de sensores de la CPU)	

Tabla 1.5: Datos de las salidas de los módulos analógicos.

Fuente: Siemens, Manual del sistema de automatización S7-200, Germany – 08/2005.

1.5.1 Calibración de las entradas

Los ajustes de calibración afectan a la fase de amplificación de la instrumentación que sigue al multiplexor analógico. Por consiguiente, el calibrado afecta a todos los canales de entrada del usuario, por lo que cualquier variación de los valores de los circuitos de entrada que preceden al multiplexor analógico provocará diferencias mínimas entre los valores de los distintos canales que estén conectados a la misma señal, incluso después de la calibración.

Con objeto de cumplir las especificaciones es preciso utilizar filtros de entrada para todas las entradas analógicas del módulo. Elija 64 o más muestreos para calcular el valor promedio. Para calibrar una entrada, proceda de la manera siguiente:

1. Desconecte la alimentación del módulo. Seleccione el rango de entrada deseado.
2. Conecte la alimentación de la CPU y del módulo. Espere unos 15 minutos para que el módulo pueda estabilizarse.

3. Mediante una fuente de tensión o de intensidad, aplique a una de las entradas una señal de valor cero.
4. Lea el valor que la CPU ha recibido del correspondiente canal de entrada.
5. Con el potenciómetro OFFSET, seleccione el valor cero u otro valor digital.
6. Aplique una señal de rango máximo a una entrada. Lea el valor que ha recibido la CPU.
7. Con el potenciómetro GAIN, seleccione el valor 32000 u otro valor digital.
8. En caso necesario, vuelva a calibrar el desplazamiento (OFFSET) y la ganancia (GAIN).

1.5.2 Calibración y configuración del módulo EM 235

La Figura 1.11 muestra el potenciómetro de calibración y los interruptores DIP de configuración ubicados en el lado derecho del bloque de terminales inferior del módulo.

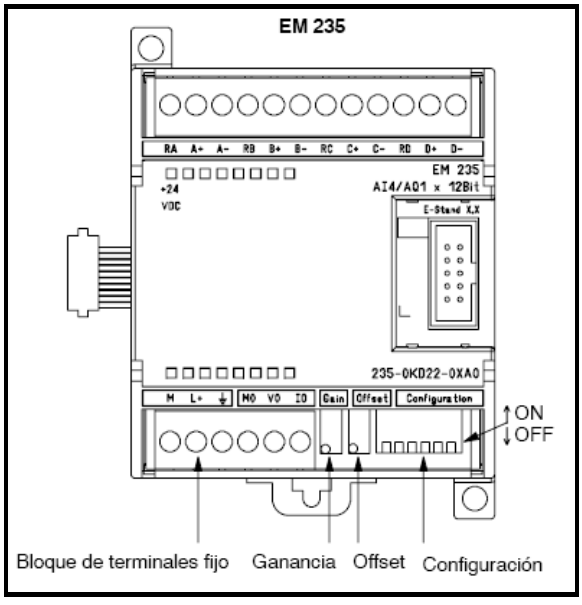


Figura 1.11: Terminales de configuración modular.

Fuente: Siemens, Manual del sistema de automatización S7-200, Germany – 08/2005.

Unipolar						Rango máx.	Resolución
Int. 1	Int. 2	Int. 3	Int. 4	Int. 5	Int. 6		
ON	OFF	OFF	ON	OFF	ON	0 a 50 mV	12,5 µV
OFF	ON	OFF	ON	OFF	ON	0 a 100 mV	25 µV
ON	OFF	OFF	OFF	ON	ON	0 a 500 mV	125 µV
OFF	ON	OFF	OFF	ON	ON	0 a 1 V	250 µV
ON	OFF	OFF	OFF	OFF	ON	0 a 5 V	1,25 mV
ON	OFF	OFF	OFF	OFF	ON	0 a 20 mA	5 µA
OFF	ON	OFF	OFF	OFF	ON	0 a 10 V	2,5 mV
Bipolar						Rango máx.	Resolución
Int. 1	Int. 2	Int. 3	Int. 4	Int. 5	Int. 6		
ON	OFF	OFF	ON	OFF	OFF	±25 mV	12,5 µV
OFF	ON	OFF	ON	OFF	OFF	±50 mV	25 µV
OFF	OFF	ON	ON	OFF	OFF	±100 mV	50 µV
ON	OFF	OFF	OFF	ON	OFF	±250 mV	125 µV
OFF	ON	OFF	OFF	ON	OFF	±500 mV	250 µV
OFF	OFF	ON	OFF	ON	OFF	±1 V	500 µV
ON	OFF	OFF	OFF	OFF	OFF	±2,5 V	1,25 mV
OFF	ON	OFF	OFF	OFF	OFF	±5 V	2,5 mV
OFF	OFF	ON	OFF	OFF	OFF	±10 V	5 mV

Tabla 1.6: Valores de configuración lograda con la combinación de los dipswithes.

Fuente: Siemens, Manual del sistema de automatización S7-200, Germany – 08/2005.

1.5.3 Formato de la palabra de datos de entrada y de salida de los módulos de ampliación EM 235

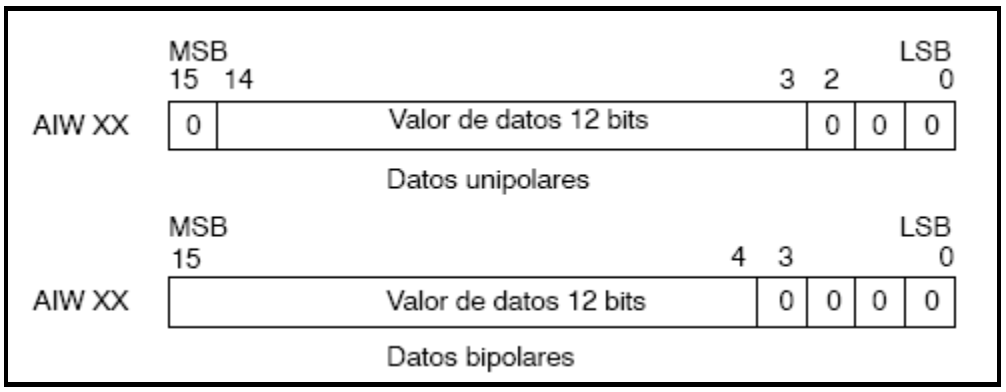


Figura 1.12: Formato de la palabra de datos de ingreso al módulo.

Fuente: Siemens, Manual del sistema de automatización S7-200, Germany – 08/2005.

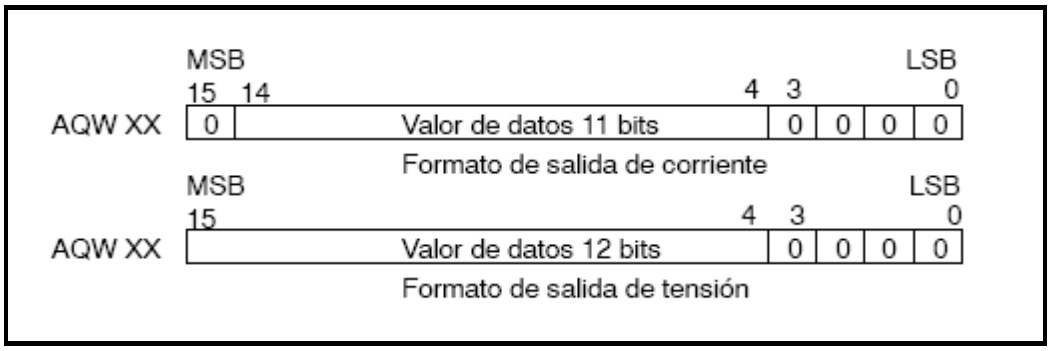


Figura 1.13: Formato de la palabra de datos de salida del módulo.

Fuente: Siemens, Manual del sistema de automatización S7-200, Germany – 08/2005.

1.5.4 Reglas de instalación

Tenga en cuenta las siguientes reglas para asegurar la precisión y la repetibilidad:

- Asegúrese de que la alimentación de sensores 24 V CD. sea estable y esté exenta de interferencias.
- Utilice cables lo más cortos posible para la alimentación de sensores.
- Utilice cables dobles trenzados apantallados para el cableado de la alimentación de sensores.
- Conecte el apantallado sólo del lado de los sensores.
- Desvíe las entradas de los canales no utilizados.
- Evite doblar excesivamente los cables.
- Conduzca los cables a través de canales.
- Evite colocar los cables de señales en paralelo con cables de alta tensión. Si los cables se deben cruzar, hágalo en ángulo recto.
- Verifique que las señales de entrada se encuentren dentro de los límites de tensión en modo común, aislando dichas señales o referenciándolas al hilo común externo de 24V del módulo analógico.

ESCUELA DE INGENIERIA ELECTRONICA
LABORATORIO DE AUTOMATAS PROGRAMABLES

GUIA DE PRACTICAS PARA AUTOMATA PROGRAMABLE

Sesión **2**

Tema **Entorno de Programación MicroWin.**

2.1 Software de Programación STEP7 / MicroWin V4.05.08

Para elaborar el *conjunto de instrucciones* se emplea el **Step7 Micro/Win 32 V4.0.5.08** de Siemens, el cual permite crear secuencias de control, para luego bajarlos a la memoria del PLC. El entorno de programación presenta la siguiente interfaz gráfica:

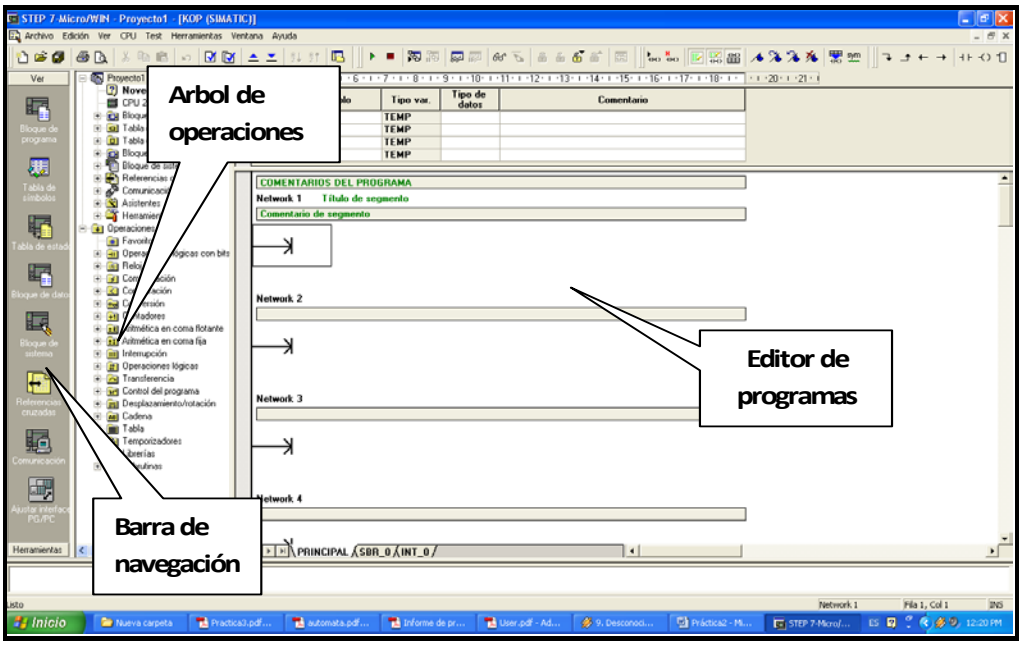


Figura 2.1: Pantalla principal del MicroWin.

Fuente: Autor.- Obtenido del MicroWin V4.05.08.

Barra de Navegación, están los iconos que permiten ingresar a las varias funciones del programa.

Editor de programas, es en donde se efectúa la programación. Además, se tiene una tabla de variables locales temporales; como también, las rutinas e interrupciones que se ven como tags (parte inferior izquierda).

Árbol de operaciones, contiene los objetos del proyecto y las operaciones de control para la programación.

Se tiene tres maneras de programar, o podría decirse tres ambientes de programación:

2.1.1 Funciones del editor AWL

El editor AWL visualiza el programa textualmente. Permite crear programas de control introduciendo la nemotécnica de las operaciones. El editor AWL sirve para crear ciertos programas que, de otra forma, no se podrían programar con los editores KOP ni FUP. Ello se debe a que AWL es el lenguaje nativo del S7-200, a diferencia de los editores gráficos, sujetos a ciertas restricciones para poder dibujar los diagramas correctamente.

Como se muestra en las instrucciones precedentes, esta forma textual es muy similar a la programación en lenguaje ensamblador.

```
LD I0.0      //Leer una entrada
A I0.1       //AND con otra entrada
= Q0.0       //Escribir en el valor en
               //la salida 1
```

El S7-200 ejecuta cada operación en el orden determinado por el programa, de arriba a abajo, reiniciando después arriba. AWL utiliza una pila lógica para resolver la lógica de control. El usuario inserta las operaciones AWL para procesar las operaciones de pila.

Considere los siguientes aspectos importantes cuando desee utilizar el editor AWL:

- El lenguaje AWL es más apropiado para los programadores expertos.
- En algunos casos, AWL permite solucionar problemas que no se podrían resolver fácilmente con los editores KOP o FUP.
- El editor AWL soporta sólo el juego de operaciones SIMATIC.
- En tanto que el editor AWL se puede utilizar siempre para ver o editar programas creados con los editores KOP o FUP, lo contrario no es posible en todos los casos.

Los editores KOP o FUP no siempre se pueden utilizar para visualizar un programa que se haya creado en AWL.

2.1.2 Funciones del editor KOP

El editor KOP visualiza el programa gráficamente, de forma similar a un esquema de circuitos. Los programas KOP hacen que el programa emule la circulación de corriente eléctrica desde una fuente de alimentación, a través de una serie de condiciones lógicas de entrada que, a su vez, habilitan condiciones lógicas de salida.

Los programas KOP incluyen una barra de alimentación izquierda que está energizada. Los contactos cerrados permiten que la corriente circule por ellos hasta el siguiente elemento, en tanto que los contactos abiertos bloquean el flujo de energía.

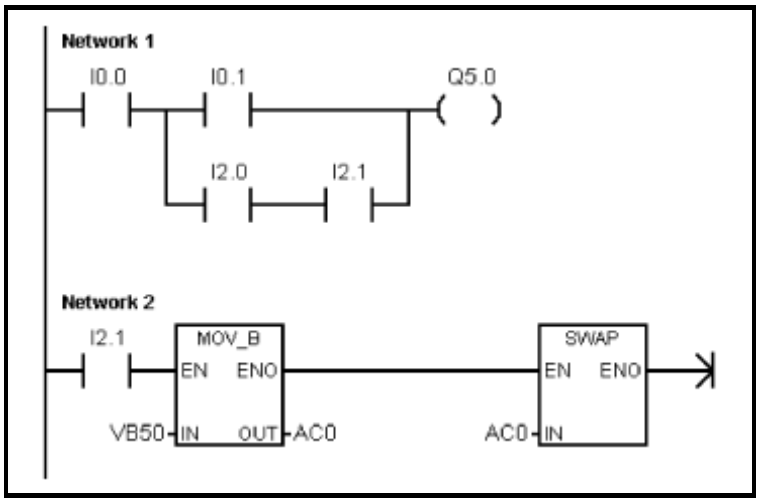


Figura 2.2: Programación en bloques de contactos.

Fuente: Siemens, Manual del sistema de automatización S7-200, Germany – 08/2005.

La lógica se divide en segmentos (”networks”). El programa se ejecuta un segmento tras otro, de izquierda a derecha y luego de arriba a abajo. La figura 6 muestra un ejemplo de un programa KOP. Las operaciones se representan mediante símbolos gráficos que incluyen tres formas básicas. Los contactos representan condiciones lógicas de entrada, tales como interruptores, botones o condiciones internas.

Las bobinas representan condiciones lógicas de salida, tales como lámparas, arrancadores de motor, relés interpuestos o condiciones internas de salida. Los cuadros representan operaciones adicionales, tales como temporizadores, contadores u operaciones aritméticas.

Considere los siguientes aspectos importantes cuando desee utilizar el editor KOP:

- El lenguaje KOP les facilita el trabajo a los programadores principiantes.
- La representación gráfica es fácil de comprender, siendo popular en el mundo entero.
- El editor KOP se puede utilizar con los juegos de operaciones SIMATIC e IEC 1131-3.

• El editor AWL se puede utilizar siempre para visualizar un programa creado en KOP SIMATIC.

2.1.3 Funciones del editor FUP

El editor FUP visualiza el programa gráficamente, de forma similar a los circuitos de puertas lógicas. En FUP no existen contactos ni bobinas como en el editor KOP, pero sí hay operaciones equivalentes que se representan en forma de cuadros.

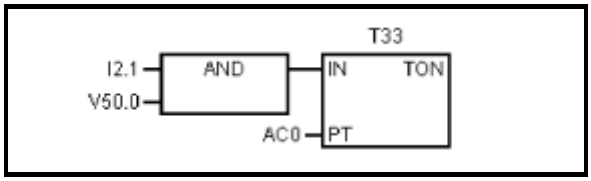


Figura 2.3: Programación por funciones.

Fuente: Siemens, Manual del sistema de automatización S7-200, Germany – 08/2005.

El lenguaje de programación FUP no utiliza las barras de alimentación izquierda ni derecha. Sin embargo, el término "circulación de corriente" se utiliza para expresar el concepto análogo del flujo de señales por los bloques lógicos FUP; el recorrido "1" lógico por los elementos FUP se denomina circulación de corriente.

El orígen de una entrada de circulación de corriente y el destino de una salida de circulación de corriente se pueden asignar directamente a un operando. La lógica del programa se deriva de las conexiones entre las operaciones de cuadro, lo cual significa que la salida de una operación (por ejemplo, un cuadro AND) se puede utilizar para habilitar otra operación (por ejemplo, un temporizador), con objeto de crear la lógica de control necesaria. Estas conexiones permiten solucionar numerosos problemas lógicos.

Considere los siguientes aspectos importantes cuando desee utilizar el editor FUP:

• El estilo de representación en forma de puertas gráficas se adecúa especialmente para observar el flujo del programa.

• El editor FUP soporta los juegos de operaciones SIMATIC e IEC 1131-3.

• El editor AWL se puede utilizar siempre para visualizar un programa creado en SIMATIC FUP.

2.2 Estructuración del software

Las programaciones efectuadas para la CPU S7-200 comprenden tres partes fundamentales: el programa principal, las subrutinas (opcional) y las rutinas de interrupción (opcional).

§ *Programación principal:* En esta parte en donde se disponen las operaciones que controlan la aplicación. Se ejecutan de forma secuencial en cada ciclo de la CPU. Para terminar el programa principal, utilice en KOP una bobina absoluta Finalizar programa principal, o en AWL una operación Finalizar programa principal (MEND).

§ *Subrutinas:* Son elementos opcionales del programa que se ejecutan sólo cuando se llaman desde el programa principal. Se deben añadir siempre al final del programa principal (detrás de la bobina absoluta Finalizar programa principal en KOP o detrás de la operación MEND en AWL). Utilice siempre una operación Retorno absoluto

(RET) para terminar cada subrutina.

§ *Rutinas de interrupción:* Son elementos opcionales del programa que se ejecutan cada vez que se presente el correspondiente evento de interrupción. Se deben añadir siempre al final del programa principal (detrás de la bobina absoluta Finalizar programa principal en KOP o detrás de la operación MEND en AWL). Utilice siempre una operación Retorno absoluto desde rutina de interrupción (RETI) para terminar cada rutina de interrupción.

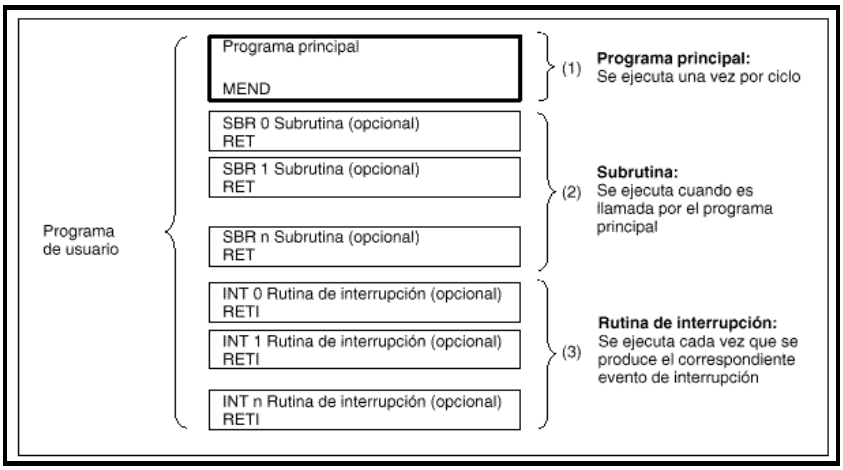


Figura 2.4: Recomendaciones de programación.

Fuente: Siemens, Manual del sistema de automatización S7-200, Germany - 1998.

Subrutinas y las rutinas de interrupción se deben añadir detrás de la bobina absoluta Finalizar programa principal en KOP o detrás de la operación MEND en AWL. No hay reglas adicionales en lo relativo a su disposición en el programa de usuario. Las subrutinas y las rutinas de interrupción se pueden mezclar a voluntad después del programa principal.

No obstante, para que la estructura del programa sea fácil de leer y comprender, es recomendable agrupar al final del programa principal primero todas las subrutinas y, después, todas las rutinas de interrupción.

2.3 Creación de un programa

Para los cual se recomienda crear una carpeta con un nombre, y guardar el área de trabajo en esa carpeta. Para crear un nuevo proyecto en KOP, se tiene que selectar en menú **Archivo** el submenú **Nuevo**.

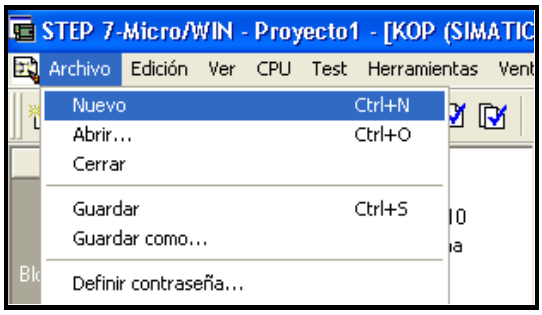


Figura 2.5: Abrir un nuevo documento.

Abriéndose un nuevo documento de trabajo.

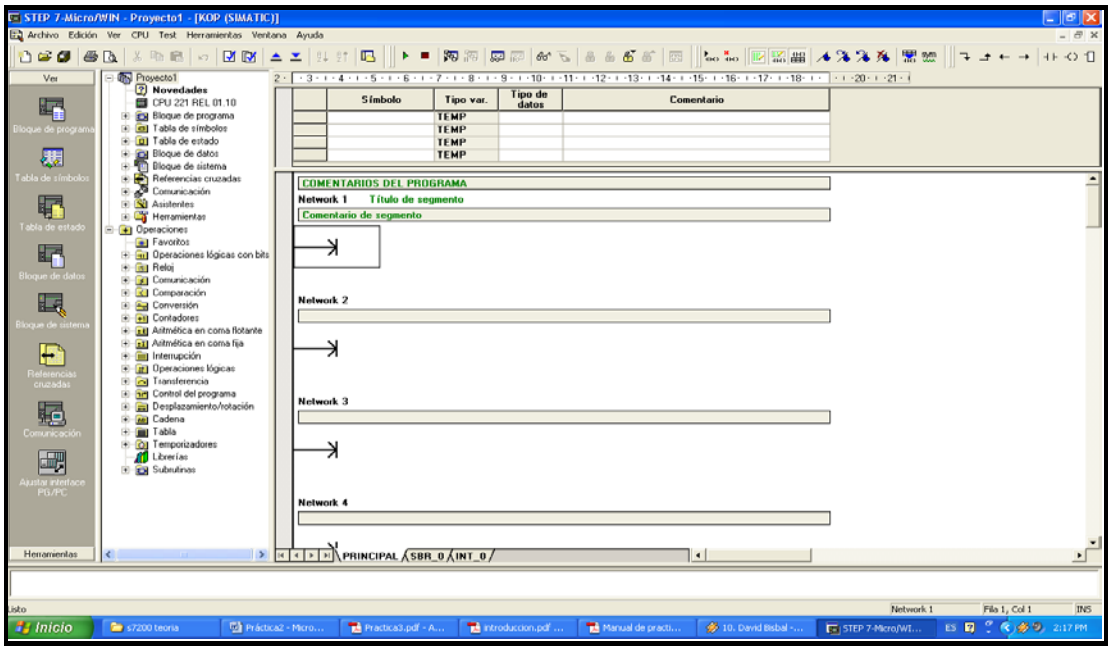


Figura 2.6: Nueva área de trabajo para usar.

Luego escoger el tipo de CPU, haciendo clic en **CPU** >> **Tipo**.

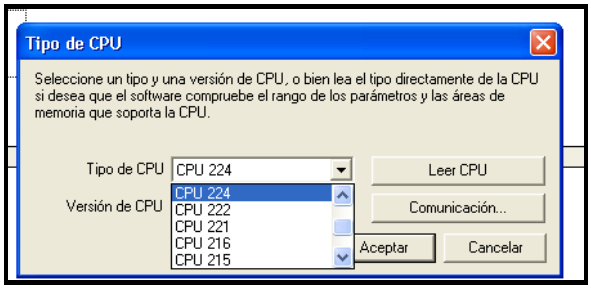


Figura 2.7: Ventana para escoger el tipo de CPU.

Si se tiene conectado el PLC a la PC, se tiene que ajustar el tipo de interface.

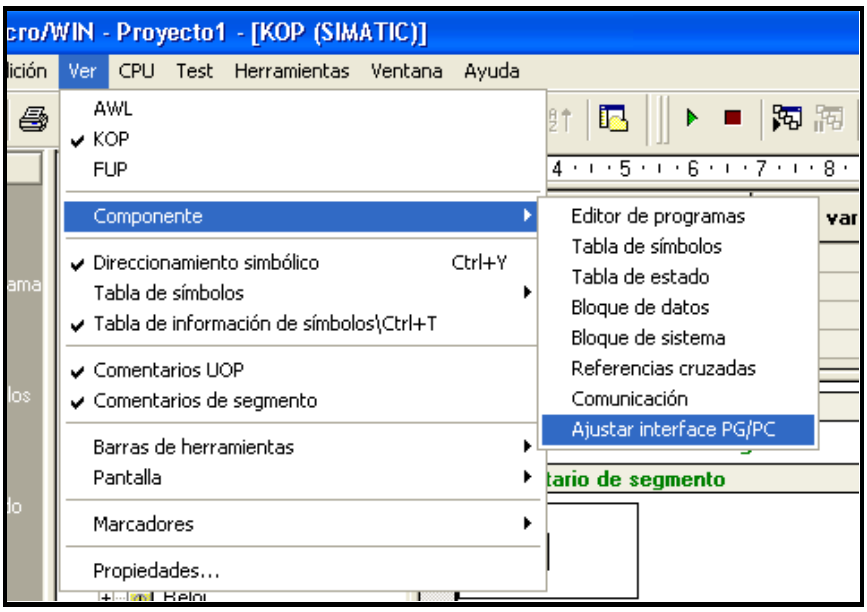


Figura2.8: Ajuste de interface PC →PLC.

Asegurándose que se encuentre activo la opción de cable PPI.

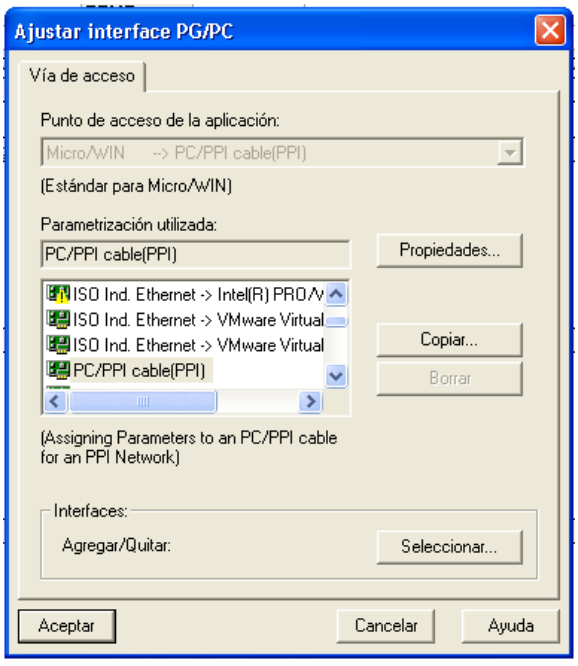


Figura 2.9: Ajuste de cable de interface.

Al comenzar a escribir un programa se debe tener en cuenta éstos aspectos importantes:

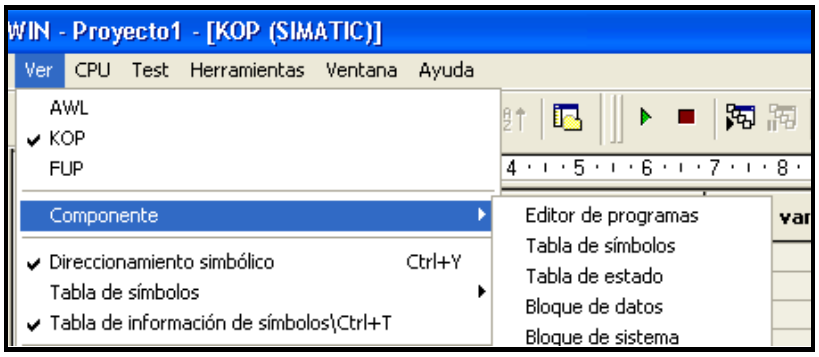


Figura 2.10: Partes importantes al realizar un programa.

Editor de programa: El editor de programa incluye el código ejecutable y los comentarios. El código se compila y se carga en la CPU, más no los comentarios del programa.

Bloque de datos: El bloque de datos comprende datos (valores iniciales de memoria, valores de constantes) y comentarios. Los datos se compilan y se cargan en la CPU, más no los comentarios.

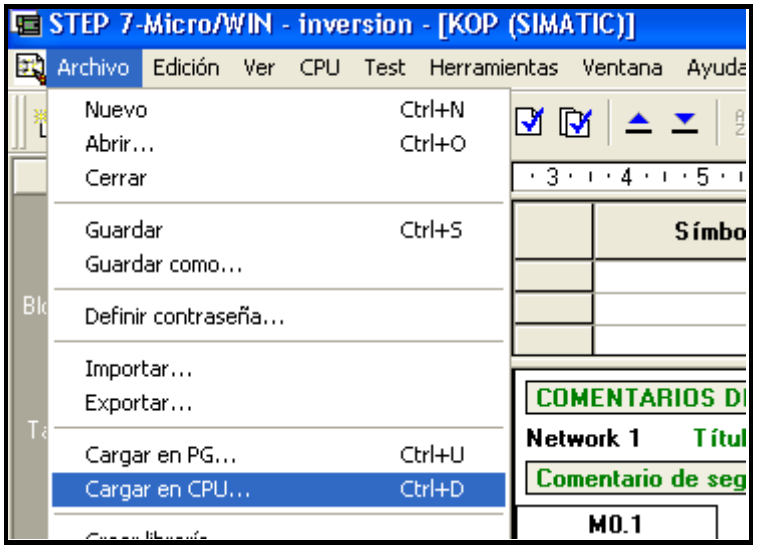
Bloque de sistema: El bloque de sistema comprende los datos de configuración, tales como los parámetros de comunicación, las áreas remanentes, los filtros de las entradas analógicas y digitales, los valores de las salidas en caso de un cambio a STOP y las informaciones sobre la protección con contraseña. Las informaciones contenidas en el bloque de sistema se cargan en la CPU.

Tabla de símbolos: La tabla de símbolos permite utilizar el direccionamiento simbólico para la programación. En algunos casos es más conveniente programar con símbolos, puesto que facilitan el entendimiento del programa. El programa compilado que se carga en la CPU convierte todos los símbolos a direcciones absolutas. Las informaciones contenidas en la tabla de símbolos no se cargan en la CPU.

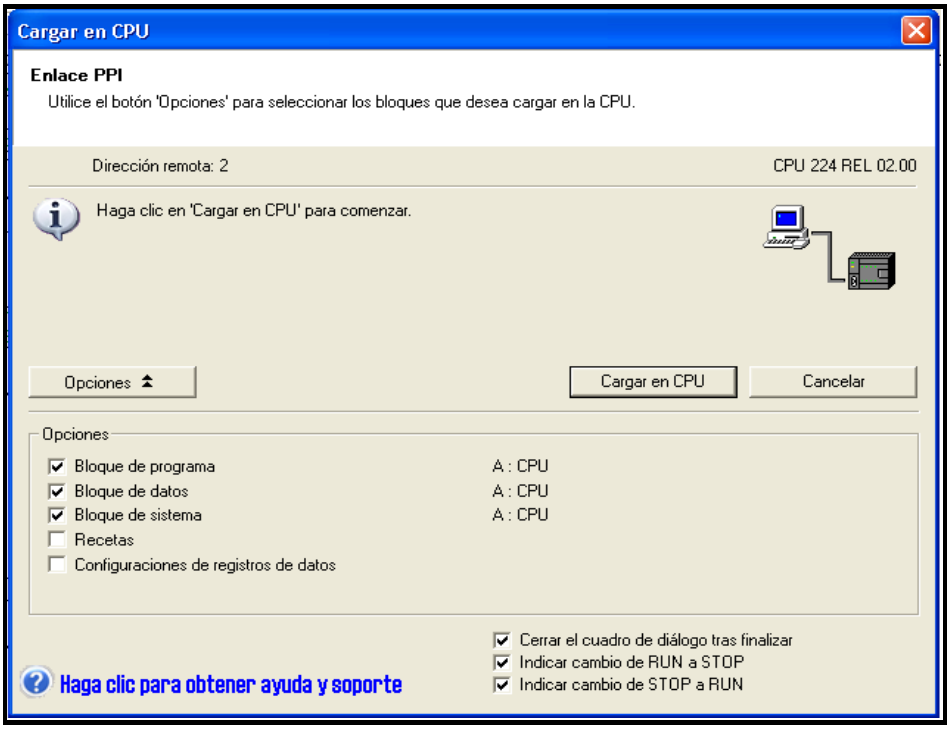
Tabla de estado: Las informaciones contenidas en la tabla de estado no se cargan en la CPU.

2.4 Pasos para correr un programa en una CPU

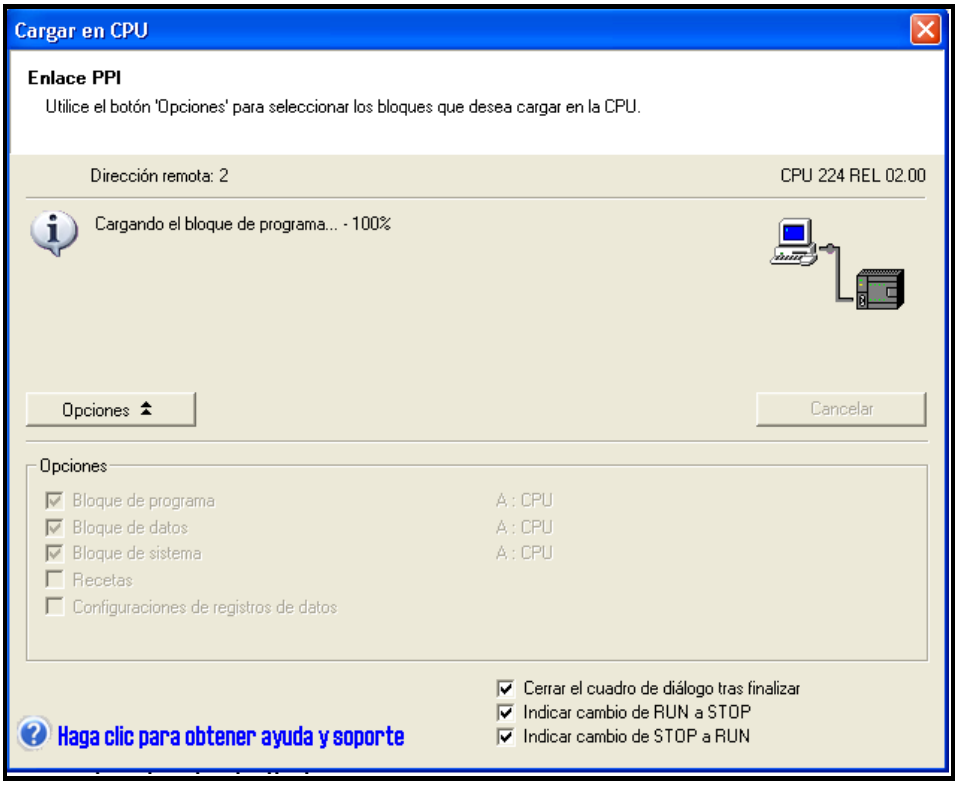
1) Seleccionar cargar.



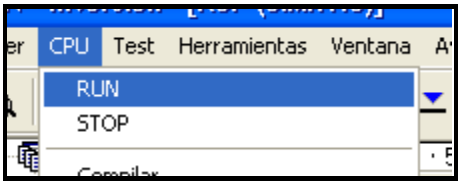
2) Aceptar la carga.



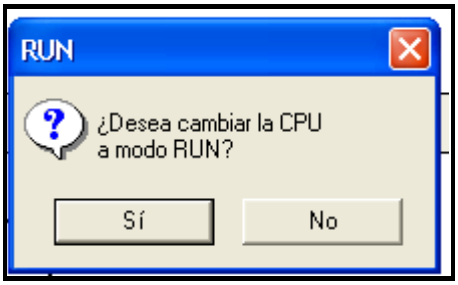
Apareciendo:



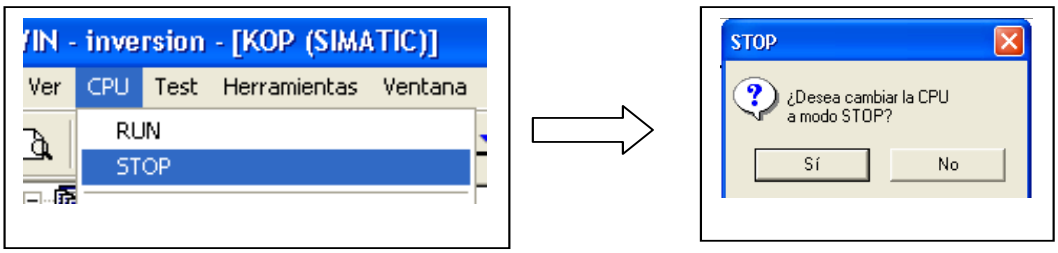
3) Escoger RUN.



4) Presionar Si.



Una vez que se ha probado el programa, o se desea corregir errores, se tiene que detener:



Sesión 3

Tema Lenguaje de programación gráfica "GRAFCET".

3.1 Introducción

Constituye un método gráfico de simple sintaxis que tiene por finalidad describir procesos secuenciales o en línea, de una manera más simple y eficaz y, que sean independientes de la tecnología a usar. Además, que fuera fácilmente entendible por los técnicos de otras ramas. Así se origino el grafcet (**GRA**fico **F**uncional de **C**ontrol de **E**tapas y **T**ransiciones), que está basado en el Standard IEC848.

En conclusión, "ésta metodología permite describir los comportamientos del automatismo en relación a las informaciones que recibe, imponiendo un funcionamiento riguroso, evitando de esta forma incoherencias, bloqueos o conflictos en el funcionamiento. En cada nivel de descripción, este diagrama puede ser modificado o corregido, sin necesidad de volver a partes ya estudiadas[1]."

3.2 Tipos de Grafcet

3.2.1 *Grafcet de Nivel 1*: Opera con las normativas funcionales del automatismo. Por lo tanto describe las acciones que realizaran los elementos involucrados en el automatismo, sin especificar la tecnología que hará práctico.

3.2.2 *Grafcet de Nivel 2*: Constituye ya la descripción tecnológica de los elementos del automatismo. Dando las especificaciones de cada uno.

---

1 ET_AL, *Resumen de Grafcet, Universidad de Oviedo, Programa de Estudios 2009 – 2010, publicación pdf, España. (http://isa.uniovi.es/~vsuarez/ii/index.htm)*

3.2.3 *Grafcet de Nivel 3*: Es la descripción operativa de los elementos, con sus nombres propios de entrada o salida dentro del programa. Como también las marcas que serán usadas.

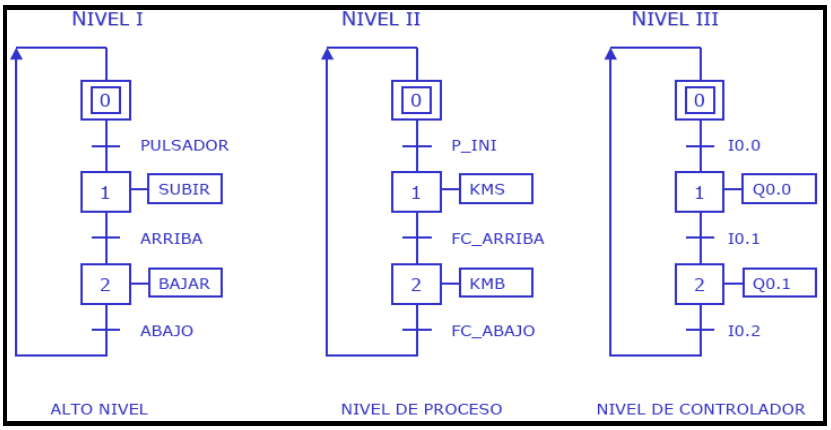


Figura 3.1: Niveles del Grafcet.

Fuente: ET_AL, Grafcet, Universidad de Oviedo, Programa de Estudios 2009 – 2010, publicación pdf, España. (http://isa.uniovi.es/~vsuarez/ii/index.htm)

3.3 Elementos básicos

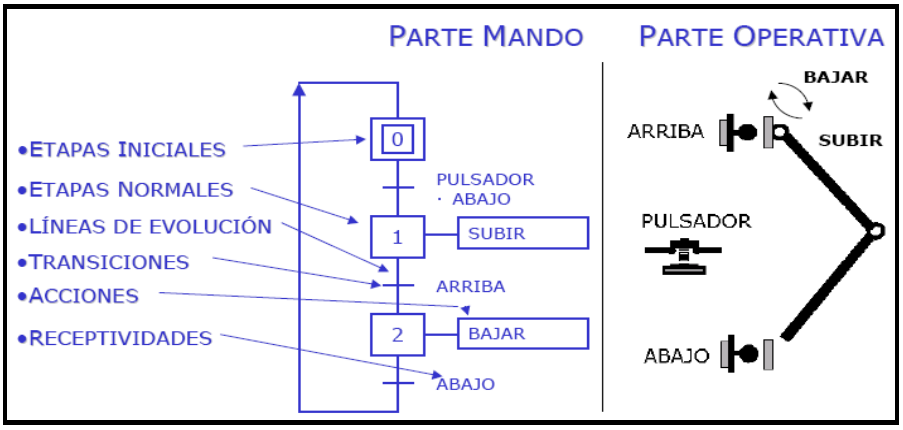


Figura 3.2: Elementos de base para el grafcet.

Fuente: ET_AL, Grafcet, Universidad de Oviedo, Programa de Estudios 2009 – 2010, publicación pdf, España. (http://isa.uniovi.es/~vsuarez/ii/index.htm)

3.3.1 *Etapas*: comprenden los estados estables en una parte o en la totalidad de la lógica del programa. Pueden estar activas o inactivas, dependiendo de las condicionantes de programación. Y se le puede asociar o no acciones a desarrollar.

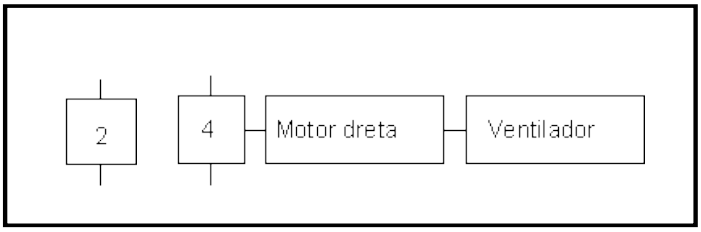


Figura 3.3: Acciones asociadas a las etapas.

Fuente: Fabrizzio, Rubén Ing., Grafcet y Gemma, FIUBA – Curso de PLCs, Buenos Aires – 2008.

3.3.2 *Transiciones*: Representan las condiciones que el sistema debe superar para poder pasar de una etapa a la siguiente. Validar la transición implica que las etapas asociadas a esa transición deben estar activas.

La norma IEC-848 propone las representaciones siguientes para las acciones asociadas condicionadas

C	**Acción condicionada**
D	**Acción retardada**
L	**Acción limitada en el tiempo**
P	**Impulso**
S	**Acción memorizada**

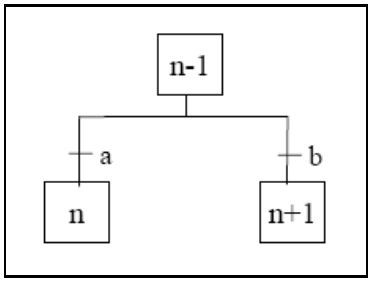
La condición o condiciones que se deben superar para poder pasar una transición, reciben el nombre de receptividades. En una transición podemos tener:

- Una condición simple,
- Una función booleanas,
- La señal de un temporizador o contador, (en este caso, es habitual que el temporizador se haya activado en la acción asociada de la etapa de entrada.),
- La activación de otra etapa del grafcet.

3.3.3 *Líneas de evolución*: son líneas verticales u horizontales, que unen con una dirección significativa (a no ser que se indique lo contrario de arriba a abajo), las distintas etapas con las transiciones, y las transiciones con las etapas.

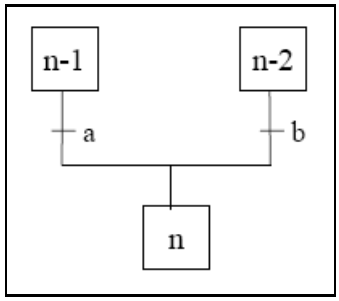
3.4 Transiciones condicionales[2]

Divergencia en OR: Estando activa la etapa *n-1* se pasa a la etapa *n* o a la *n+1* según este activa *a* o *b*.



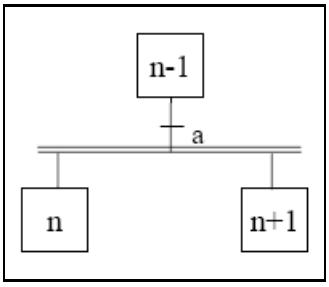
Convergencia en OR:

Para pasar a la etapa *n* debe estar activa la etapa *n-1* y cumplirse la receptividad *a* o estar activa la etapa *n-2* y cumplirse la receptividad *b*.



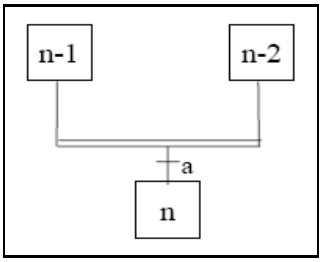
Divergencia en AND:

Estando activa la etapa *n-1* al verificarse la receptividad *a* se pasa simultáneamente a las etapa *n* y *n+1*.



Convergencia en AND:

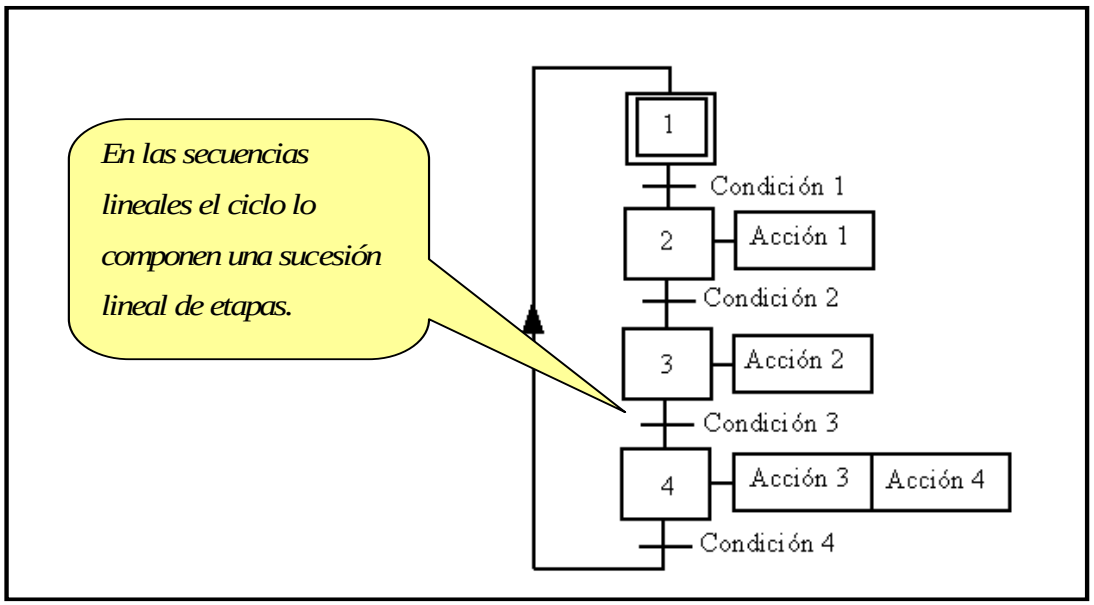
Si las etapas *n-1* y *n-2* están activas simultáneamente y se cumple la condición *a* se pasa a la etapa *n*.



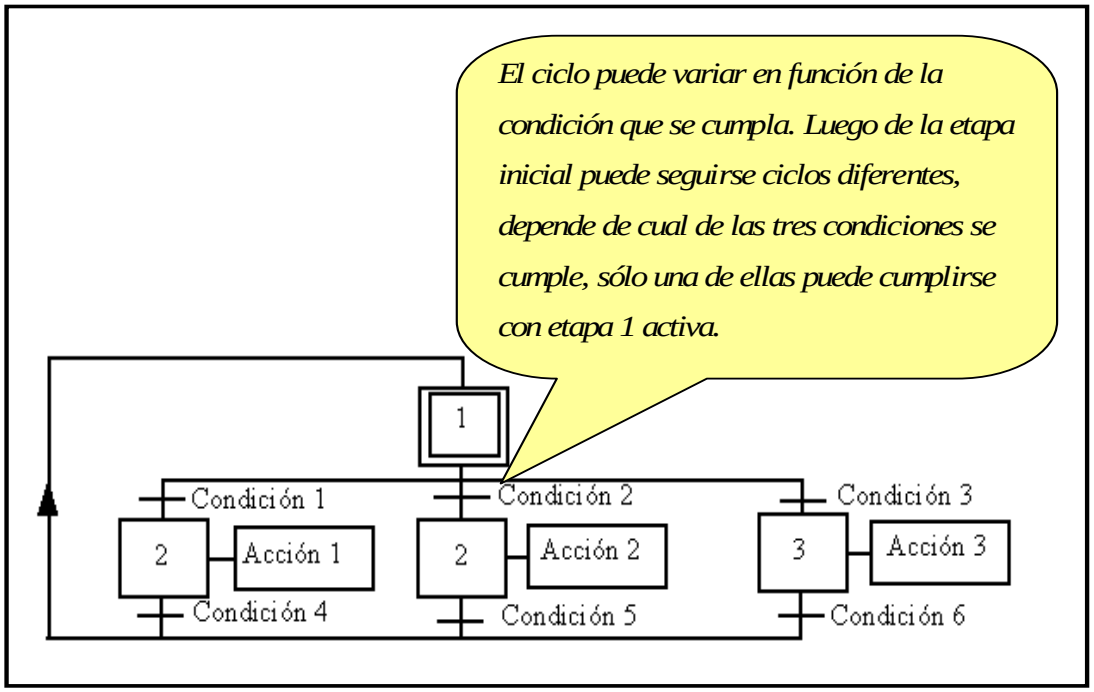
2 ET_AL, *Resumen de Grafcet, Universidad de Oviedo, Programa de Estudios 2009 – 2010, publicación pdf, España. (http://isa.uniovi.es/~vsuarez/ii/index.htm)*

3.5 Estructuras de programación

3.5.1 Lineales[3]

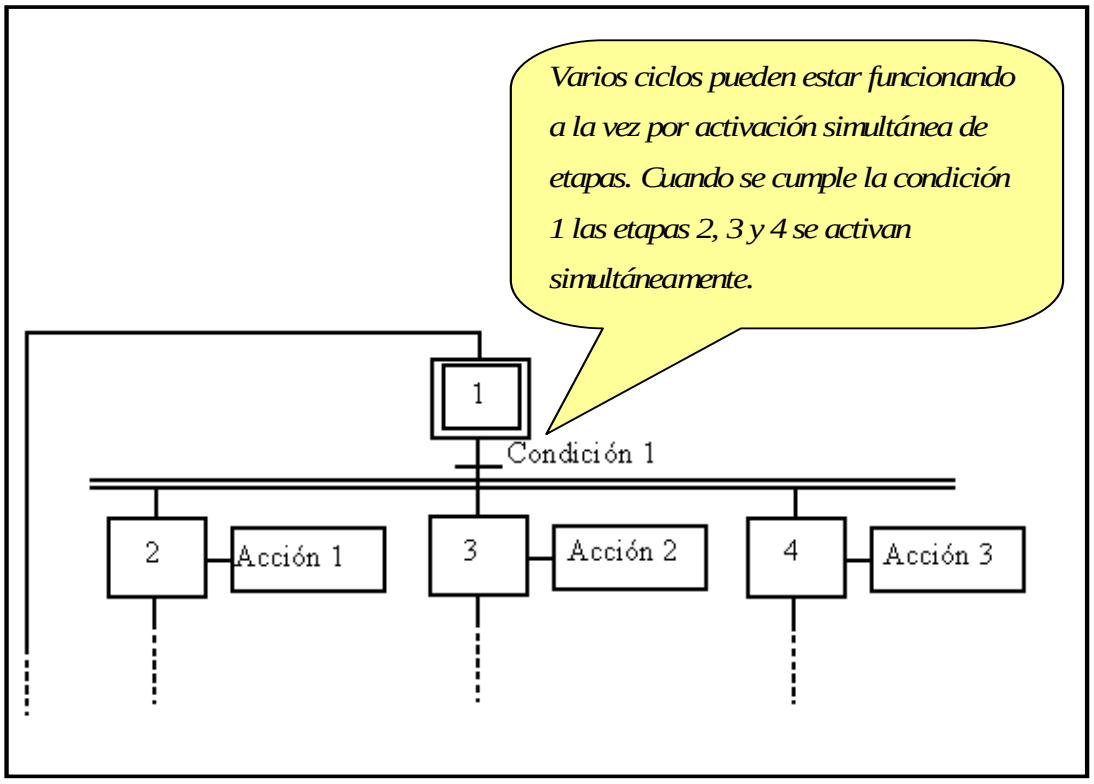


3.5.2 Con direccionamiento[4]

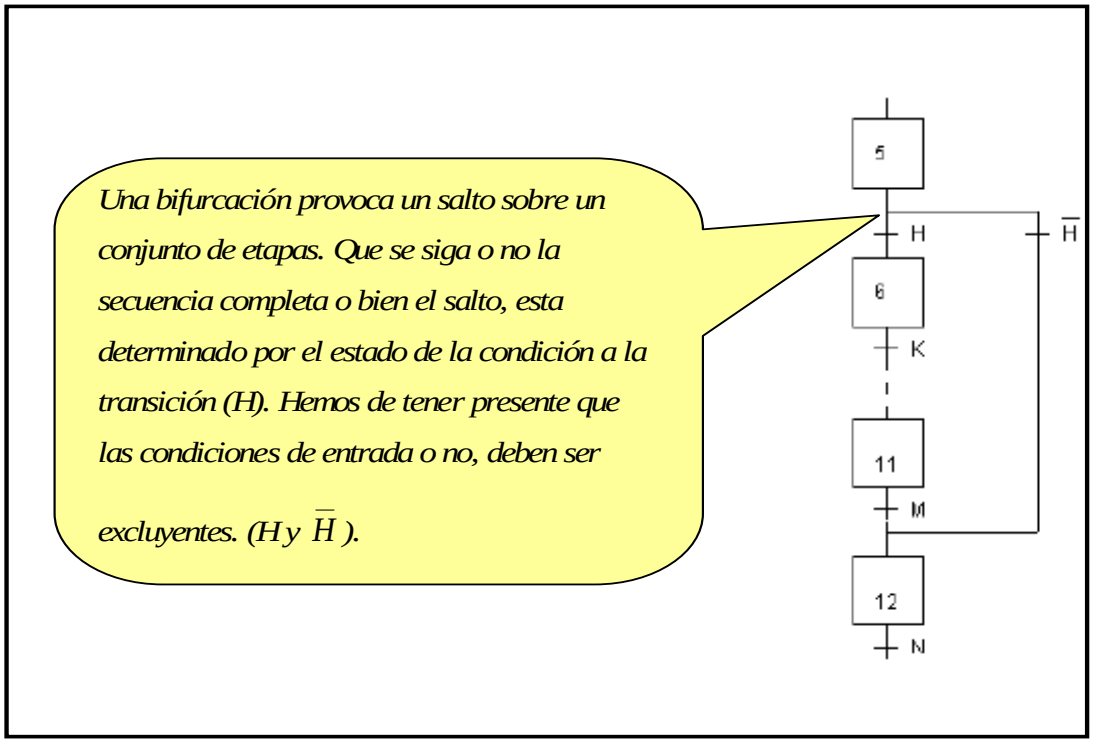


3,4 www.wikipedia.com/Grafcet/Diagrama_de_etapa-transicion.htm#Clasificaci.C3.B3n_de_las_secuencias

3.5.3 Simultáneas[5]



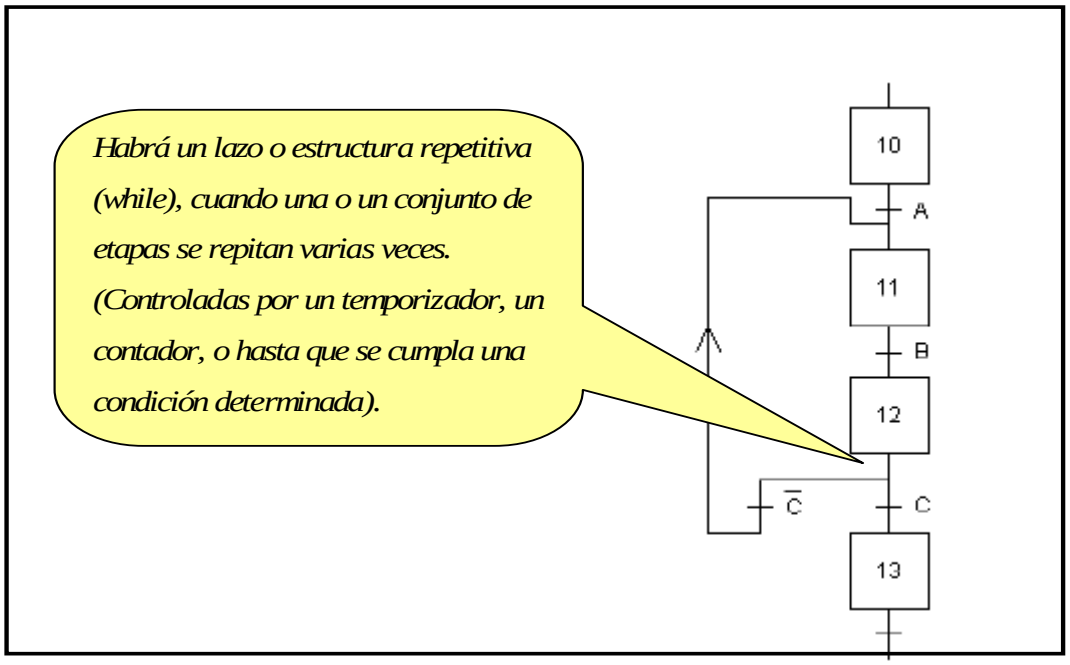
3.5.4 Saltos de etapas[6]



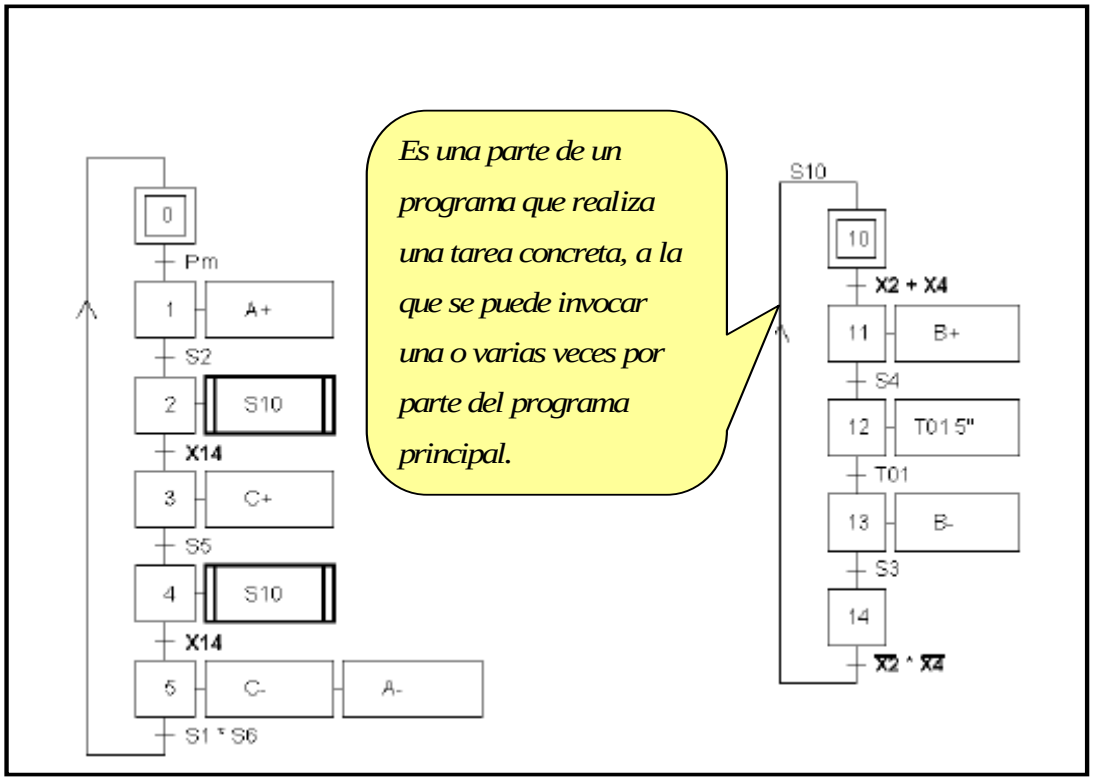
5 www.wikipedia.com/Grafcet/Diagrama_de_etapa-transicion.htm#Clasificacio.C3.B3n_de_las_secuencias

6 www.cursos.femz.es/Automatas/tema05/02tema5.htm#salts

3.5.5 Lazos repetitivos[7]



3.5.6 Subrutinas[8]



7,8 www.cursos.femz.es/Automatas/tema05/02tema5.htm#salts

3.6 Consideraciones para la programación

Para empezar a realizar programaciones correspondiente a un ciclo de trabajo en lenguaje GRAFCET, se deberán tener en cuenta lo siguiente:

- Se descompone el proceso en etapas que serán activadas una tras otra.
- A cada etapa se le asocia una o varias acciones que sólo serán efectivas cuando la etapa esté activa.
- Una etapa se activa cuando se cumple la condición de transición.
- El cumplimiento de una condición de transición implica la activación de la etapa siguiente y la desactivación de la etapa precedente.

De igual manera, considerar las Reglas de evolución del Grafcet:

- La etapa inicial de un Grafcet se activan de forma incondicional. Esta situación inicial se corresponde en general con una situación de reposo.
- Una transición está en disposición de ser validada cuando todas las etapas inmediatamente precedentes, unidas a dicha transición, están activadas. La activación de una transición se produce cuando está validada y la condición de transición o receptividad es verdadera. Se podría definir una etapa como activable cuando la transición precedente esta validada.
- Franquear una transición implica la activación de todas las etapas siguientes inmediatas, y la desactivación de las inmediatas precedentes.
- Transiciones conectadas en paralelo, se activan de forma simultánea si se cumplen las condiciones para ello.
- Una o varias acciones se asocian a cada etapa. Estas acciones sólo están activas cuando la etapa esta activa.

ESCUELA DE INGENIERIA ELECTRONICA
LABORATORIO DE AUTOMATAS PROGRAMABLES

GUIA DE PRACTICAS PARA AUTOMATA PROGRAMABLE

Sesión 4

Tema Entorno de Programación WinCC.

4.1 Generalidades

El WinCC (*Windows Control Center*) es una aplicación HMI (*Human Machine Interface*), que integra al software de control del proceso de la planta y los varios elementos que intervienen. Además, combina las arquitecturas de las aplicaciones de Windows con un entorno de programación grafica. Lo que permite establecer niveles de supervisión y control en los procesos.

Entre las características más importantes se tiene:

- Posee un esquema abierto de programación en C.
- Puede soportar tecnologías ActiveX.
- Comunicación OPC con otras aplicaciones.
- Fácil comunicación por medio de drivers.
- Se puede programar on-line.

El entorno de ingeniería de proyectos de WinCC engloba:

 Dibujos.- Para diseñar representaciones de planta.

 Estructura de archivos.- Para guardar datos/eventos marcados con fecha y hora en una base de datos SQL.

 Generador de informes.- Para generar informes sobre los datos solicitados.

□ **Administración de datos.-** Para definir y recopilar datos de toda la planta.

□ **RunTime.-** Tiempo de ejecución de WinCC: "Permite a los operarios interactuar con la aplicación directamente en la máquina o desde un centro de control". (Pérez Fede.)

4.2 Empezar a usarlo

Considerar que para comenzar a trabajar con este software, se requiere la licencia original que viene con el paquete de instalación. La que se puede instalar en el momento de instalación del paquete de WinCC o luego de ésta. Si no se dispone de la licencia, funcionará solo en intervalos de 10 minutos; teniendo que lanzar el programa constantemente en modo programación. Solo el explorador sigue operando continuamente.

Una vez instalado, se puede ingresar ya sea por el ícono que se coloca en el escritorio, o por medio del botón de inicio.

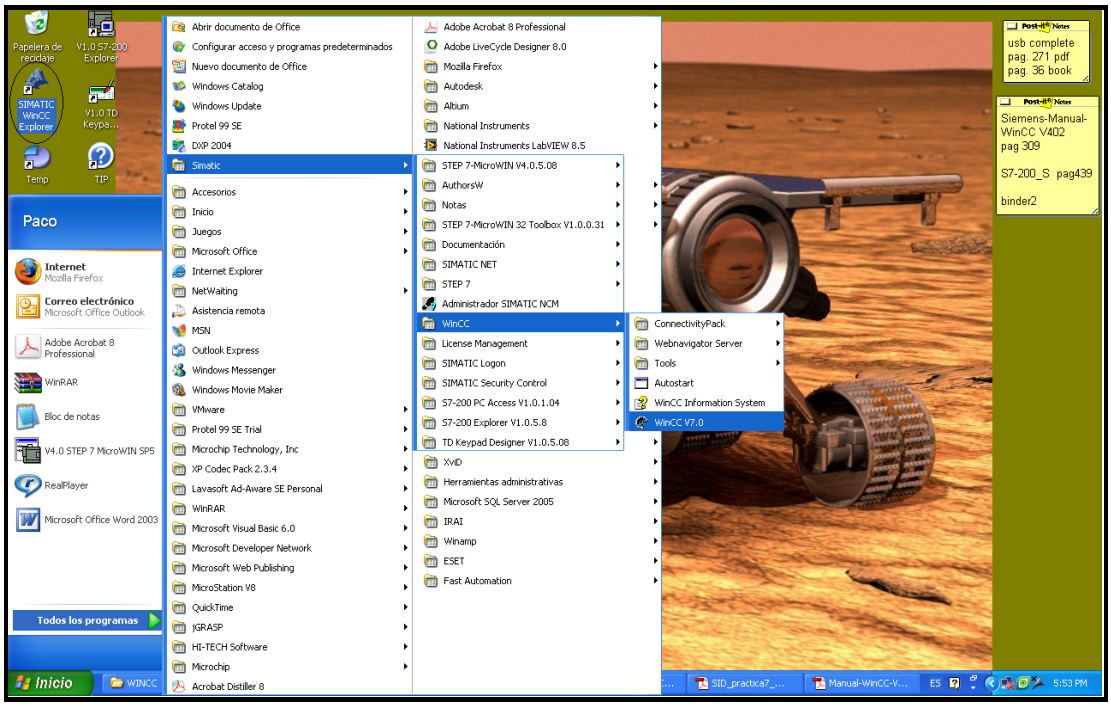


Figura 4.1: Entrar al WinCC por inicio del desk, o por el ícono.

Ya lanzado presenta la ventana del explorador que muestra el último proyecto activo:

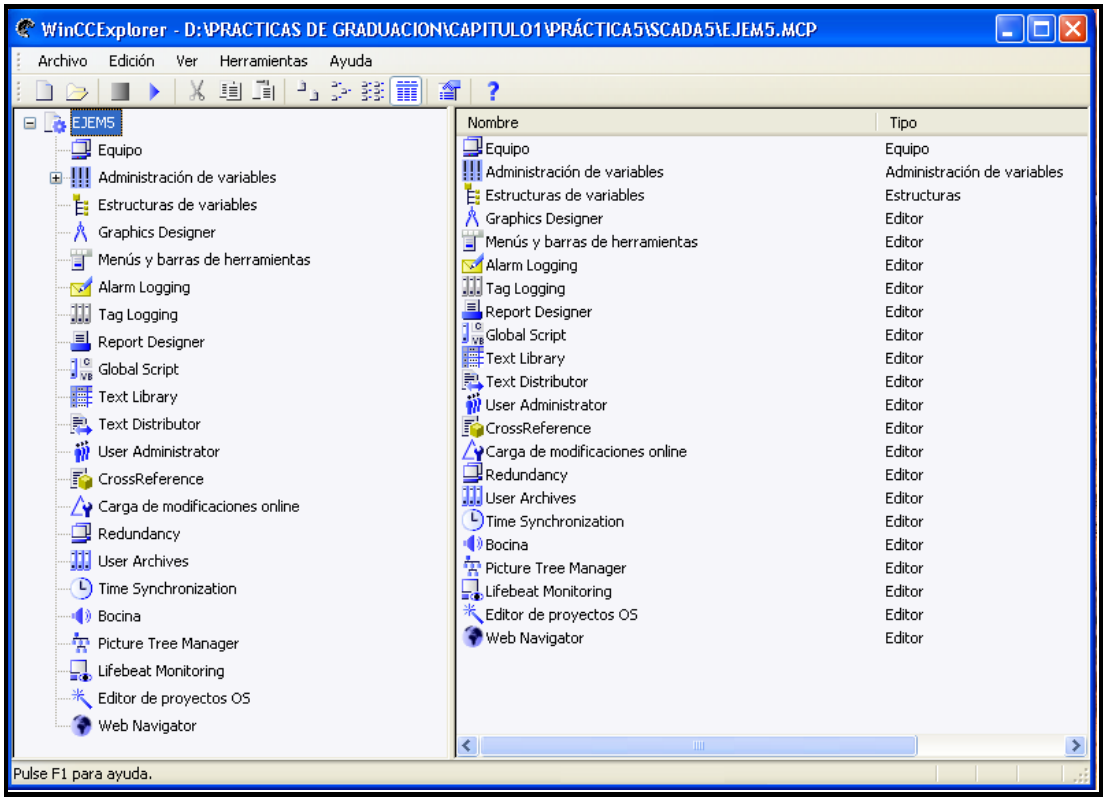


Figura 4.2: Ventana del Explorer del WinCC.

Caso contrario, aparece un box en el cual se escoge el tipo de proyecto.

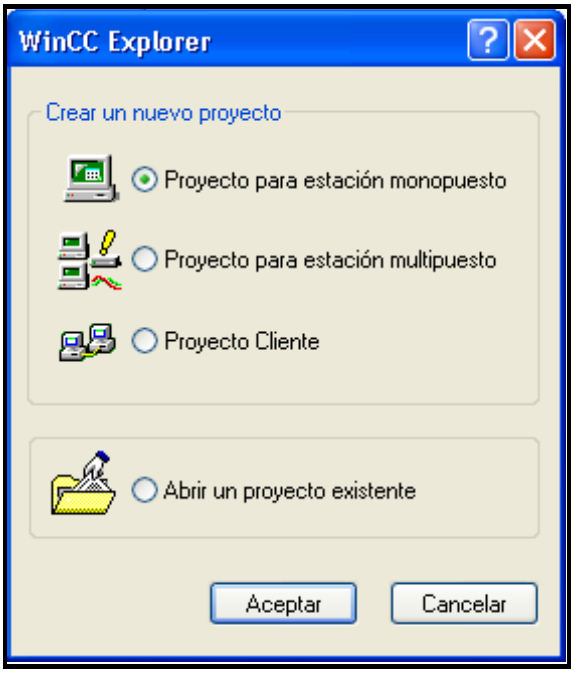


Figura 4.3: Box de inicio de proyecto.

Proyecto para estación monopuesto: es para un único ordenador.

Proyecto para estación multipuesto: un servidor y varios clientes.

Proyecto cliente: varios servidores y un cliente.

Abrir un proyecto existente: abre uno que ya existe con anterioridad.

Para luego crear el proyecto, el cual se almacena en el path que se tipee. Es recomendable colocar nombres cortos, menos de 32 caracteres, que permitan una rápida identificación del proyecto. Y un lugar en el cual no peligre por acción de los virus sobre el sistema operativo.

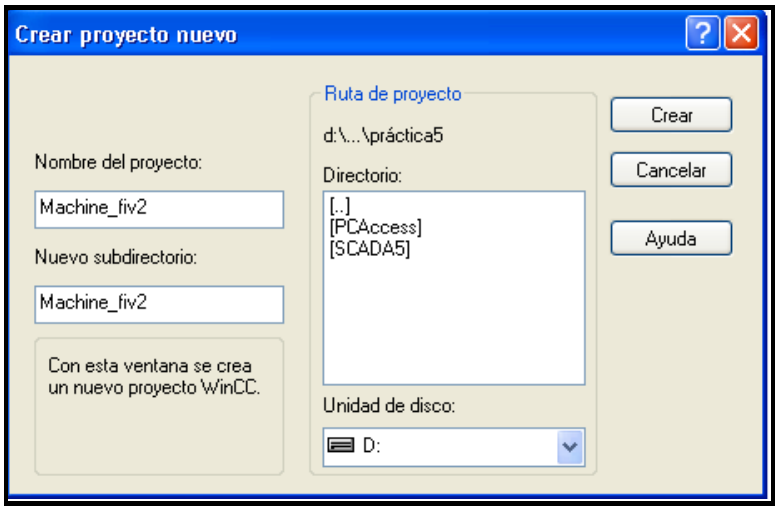


Figura 4.4: Box de creación del proyecto.

Ahora se escoge el **tipo de drive** que se usará para establecer la comunicación con el equipo exterior.

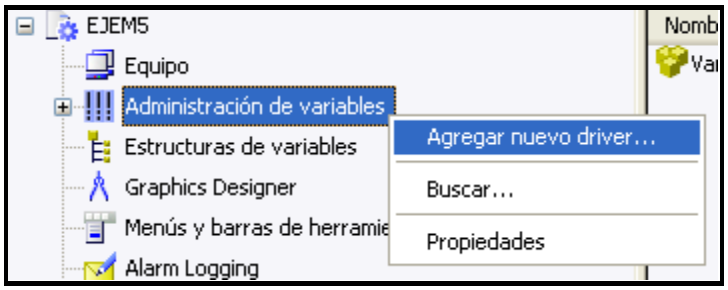


Figura 4.5: Forma de agregar un drive al proyecto.

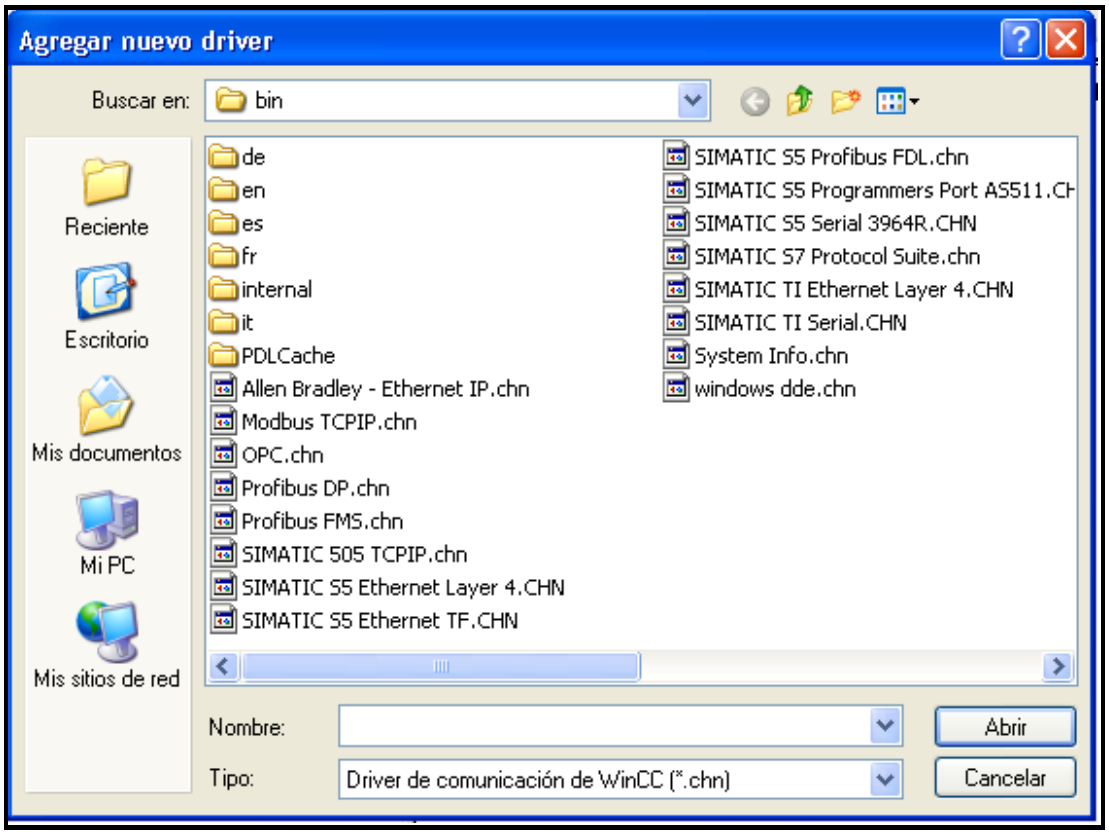


Figura 4.6: Varios drives soportados por el WinCC.

Escogido el driver (*canal de comunicaciones*), se tiene que agregar las variables que se usaran.

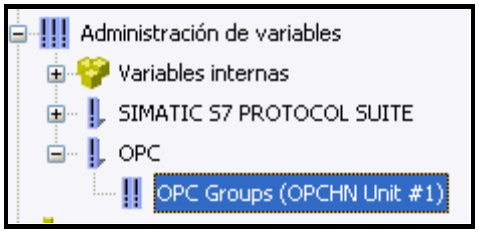


Figura 4.7: Drives escogidos para comunicarse.

Se pueden crear dos **tipos de variables**:

- *Internas*: el valor de almacenamiento no depende del exterior. Aunque pueden almacenar valores a partir de procesos externos. Son ilimitadas.
- *De Proceso*: almacenan valores provenientes de procesos externos al ordenador. Para operar se requiere que se coloque primero el canal de comunicaciones. Sirven para dimensionar el WinCC.

Ambas pueden colocarse en **grupos** que faciliten su localización y aplicación, esta estructura es conveniente cuando se tiene que controlar varios subprocesos de un proceso general. A pesar que pueden trabajar individualmente.

Para colocar una variable interna, se procede así (es similar para las de proceso):

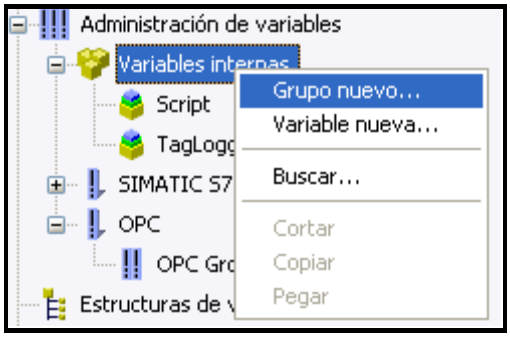


Figura 4.8: Creación de un Grupo.

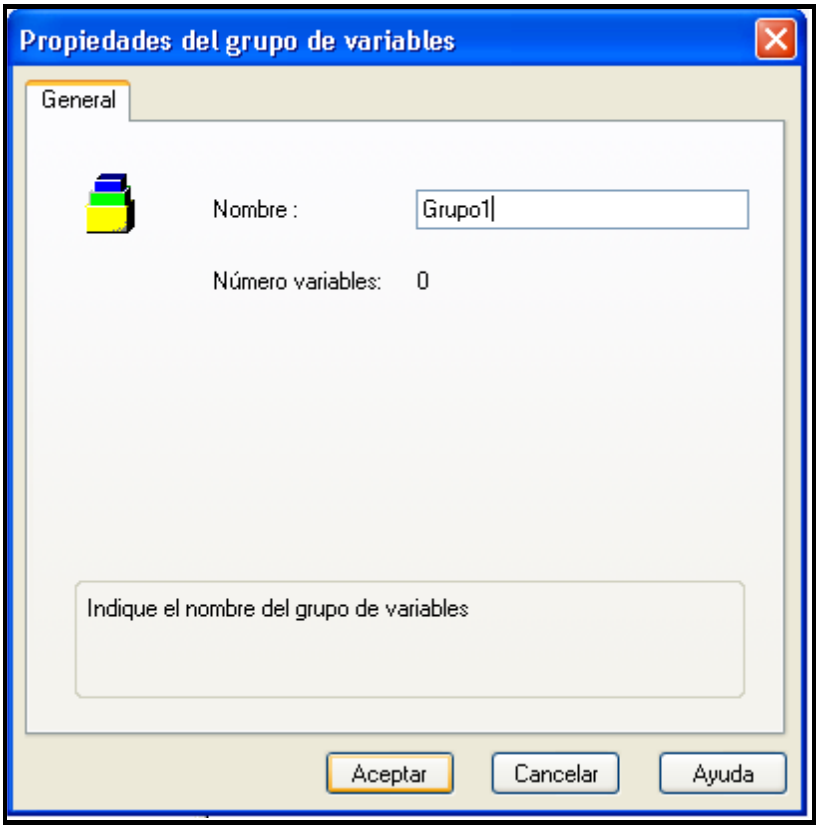


Figura 4.9: Propiedades del grupo: se coloca el nombre.

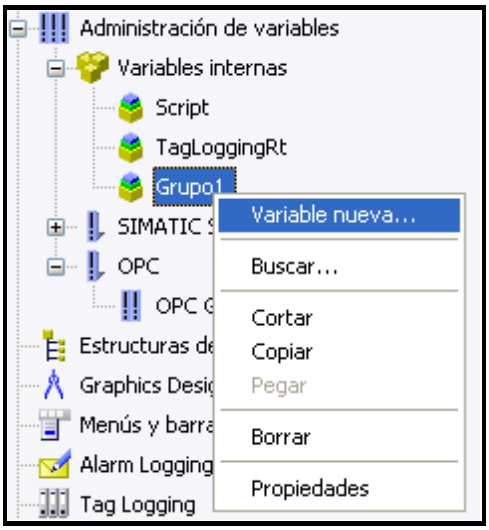


Figura 4.10: Crear una Variable interna.

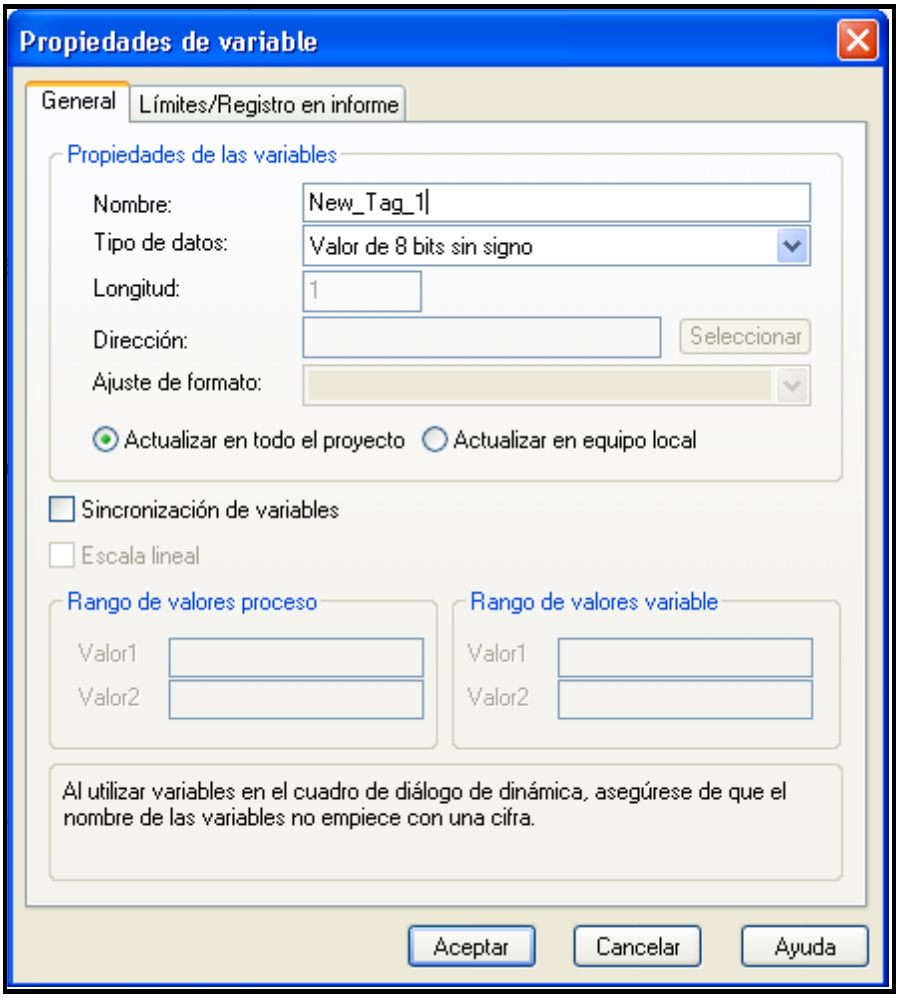


Figura 4.11: Presetear los valores de operación.

En **data type (** *tipo de datos* **)** podemos seleccionar el tamaño de la variable que vamos a utilizar:

· *Binary tag*: Un bit

· *Signed 8 bit value*: Un byte con signo (-128 a 127)

· *Unsigned 8 bit value*: un byte sin signo (0 a 255).

· *Signed 16 bit value*: Una palabra con signo (-32768 a 32767)

· *Unsigned 16 bit value*: Una palabra sin signo (0 a 65535)

· *Signed 32 bit value*: Una doble palabra con signo (-2147483647 a 2147483647)

· *Unsigned 32 bit value*: Una doble palabra sin signo (0 a 4294967295)

· *Floating Point 32 bits*: Una doble palabra en coma flotante de 32 bits de resolución

· *Floating Point 64 bits*: Una doble palabra en coma flotante de 64 bits de resolución

· *Text tag 8 bit Character Set*: Una cadena de texto de la longitud que deseemos de caracteres de 8 bits (caracteres ASCII).

· *Text tag 16 bit Character Set*: Una cadena de texto de la longitud que deseemos de caracteres de 16 bits (caracteres Unicode).

· *Raw data type*: Una telegrama de datos que no es tratado por el procesador del PLC utilizando el protocolo propio de comunicaciones.

· *Text Reference*: Un puntero a una cadena de texto que se encuentra en el Text Library. Asociándole a la variable el numero identificador del Text library, soporta el valor de la cadena de texto que definamos allí.

· *Structure Types*: Una estructura es un conjunto de variables de igual o diferentes tamaño agrupadas debido a una determinada propiedad que las relaciona. Para poder seleccionar una propiedad en esta pestaña es necesario anteriormente haber generado la estructura en Data Types.

En el caso de manejar variables por medio del OPCServer, se debe seguir:

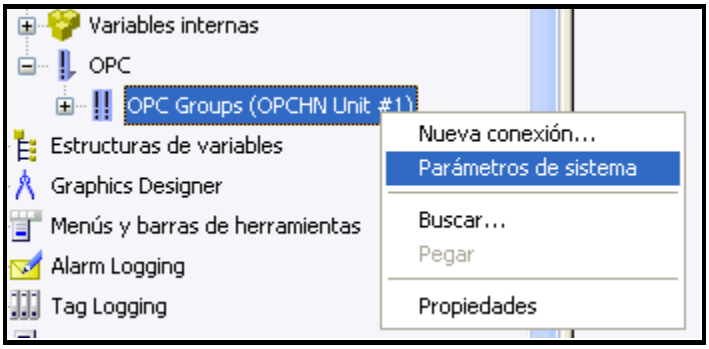


Figura 4.12: Buscar el OPCServer.

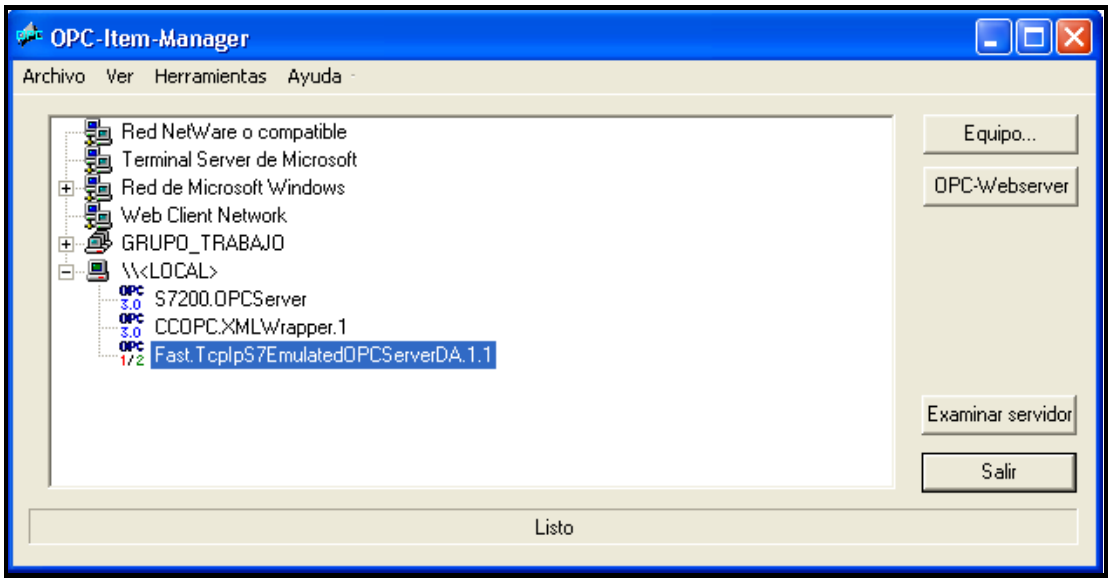


Figura 4.13: Se señala el OPCServer y luego se pulsa "Examinar servidor".

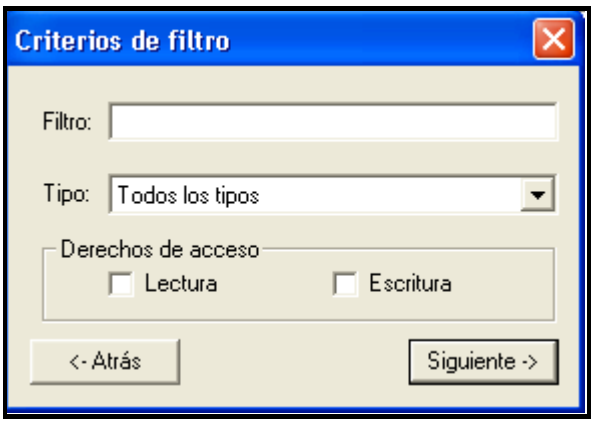


Figura 4. 14: Establece criterios de búsqueda, pulsa "Siguiente".

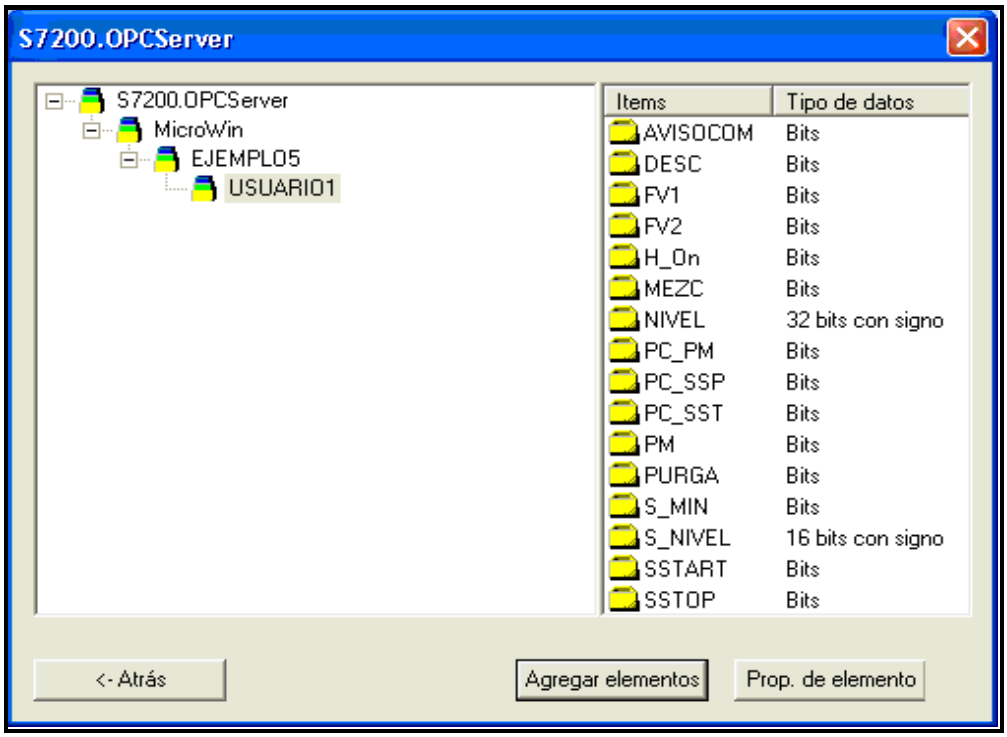


Figura 4.15: Desplegando "+" se accede al "USUARIO1" conectado, y a sus variables.

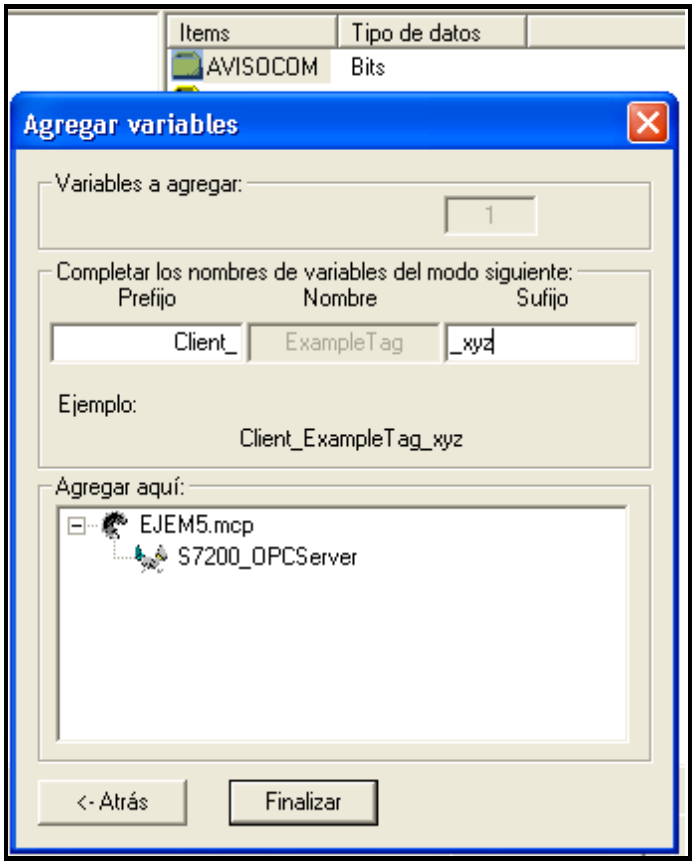


Figura 4.16: Señalando la variable y pulsando "Agregar elementos" aparece este box, escribir el Prefijo y el Sufijo y luego "Finalizar".

Una vez hecha la creación de la variable interna o externa (*de proceso*), está lista para ser usada en la programación. Para comenzar a dibujar la planta se selecta el **Graphic Designer** y se agrega una nueva imagen (*Picture*), así:

Figura 4.17: Agregar una nueva picture para dibujo.

Es mejor cambiar el nombre a la imagen para poder trabajar.

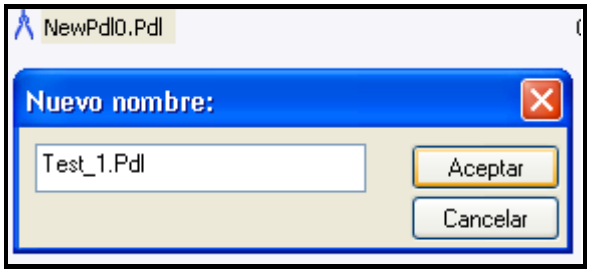


Figura 4.18: Box de cambio de nombre de imagen.

Haciendo doble clic en el nombre de la imagen se accede a la **hoja de graficación**, y se procede a armar la planta con los objetos que dispone la librería del WinCC. A pesar que se puede crear los íconos necesarios para un proyecto, se dispone de una gama razonable de íconos que se puede usar.

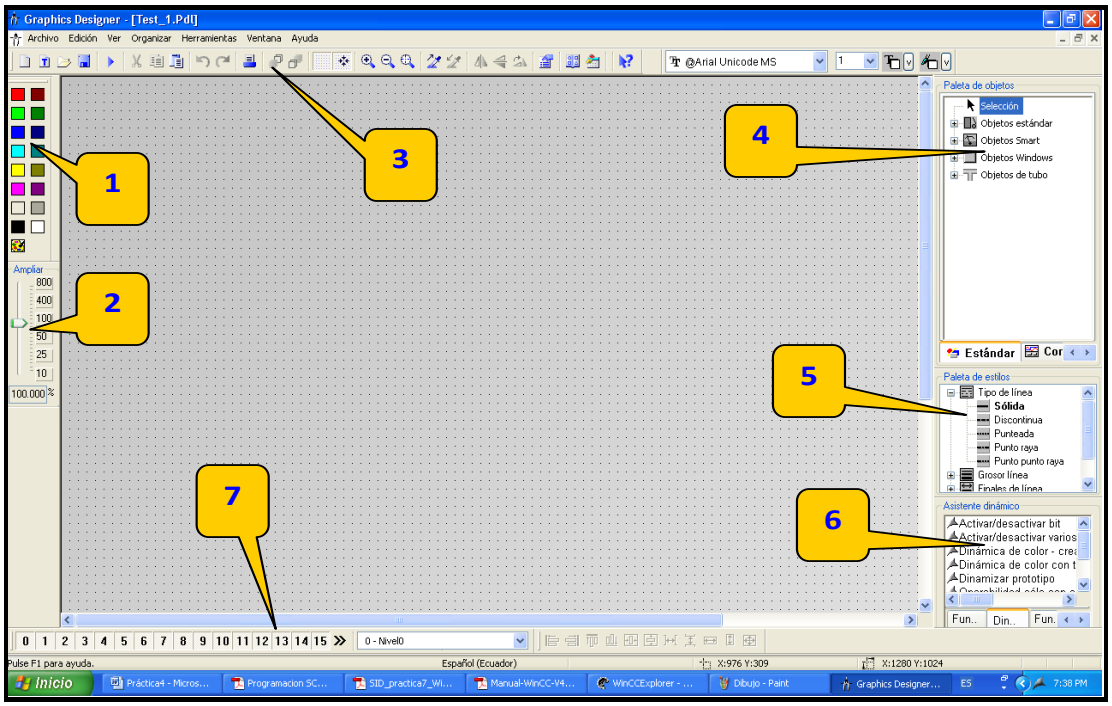


Figura 4.19: Entorno de dibujo.

1: Paleta de colores.

2: Slider del Zoom.

3: Barra de menúes y botones de control.

4: Paleta de objetos: dispone de Objetos estándar, Objetos Smart, Objetos Windows, Objetos de tubo.

5: Paleta de estilos: se escoge el tipo de líneas y de rellenos.

6: Asistente dinámico: constituye el Wizard para dinamizar los objetos.

7: Indicador de capa activa, y botones de manejo de objetos.

Para manejar los varios objetos se tiene los asistentes de las propiedades. En estos se setea los atributos en su propiedad, y permite la dinámica por eventos de la imagen.

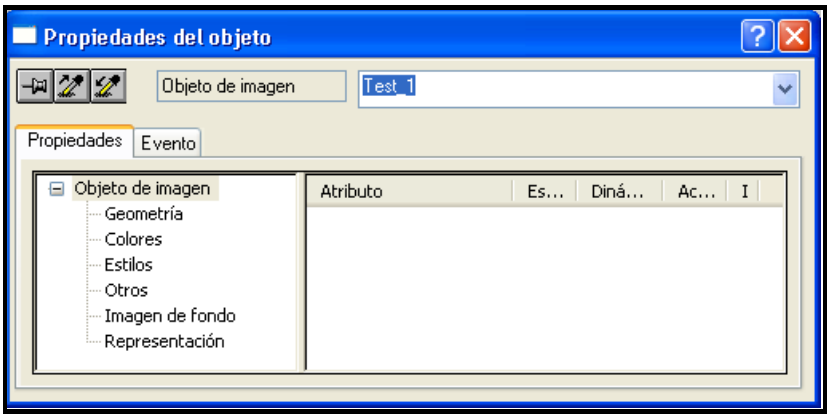


Figura 4.20: Box para seteo de las propiedades.

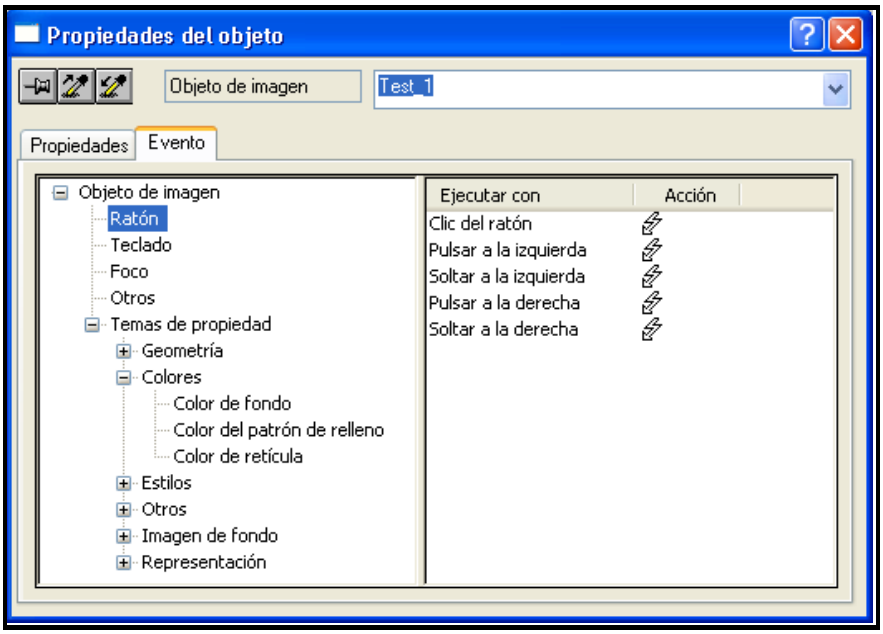


Figura 4.21: Box para seteos de los eventos.- acciones que se realizarán.

Muchas veces para manejar algún evento en especifico se requiere algo de código que ayude, éste puede ser hecho en script de VBA o C. El VBA es el Visual Basic, y el C corresponde al ANSI C. Es importante que se maneje las instrucciones dadas en el help del WinCC; y primero realizar el flujograma de cómo debe funcionar determinada acción, y luego plantear la codificación respectiva.

Para facilitar la **dinámica de los objetos** se puede realizar por medio de:

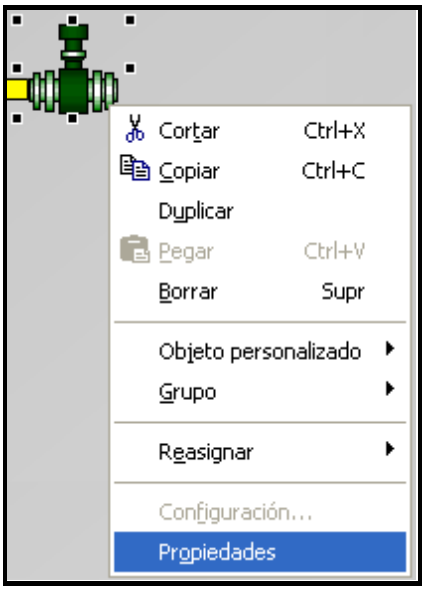


Figura 4.22: Señalando el objeto o ícono, clic derecho del mouse, se accede a las propiedades.

Figura 4.23: Los bombillos en blanco indican que se pueden dinamizar, el rayo indica que tiene un tipo de dinámica.

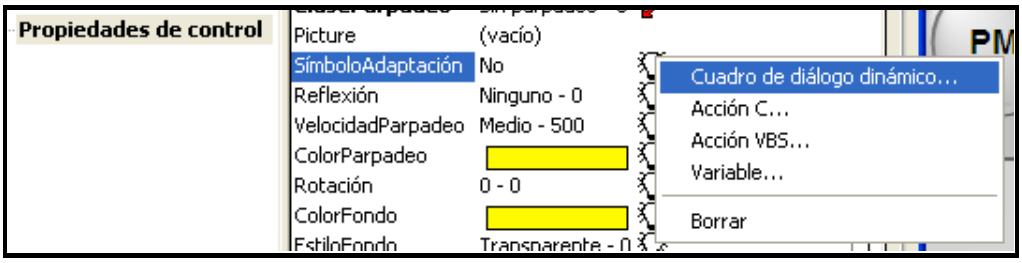


Figura 4.24: Clic derecho sobre la propiedad, el submenú da la opción de dinamizar en 4 formas. Las acciones en C y VBS son de programación. "Variable" es directo. "Borrar" quita cualquier acción.

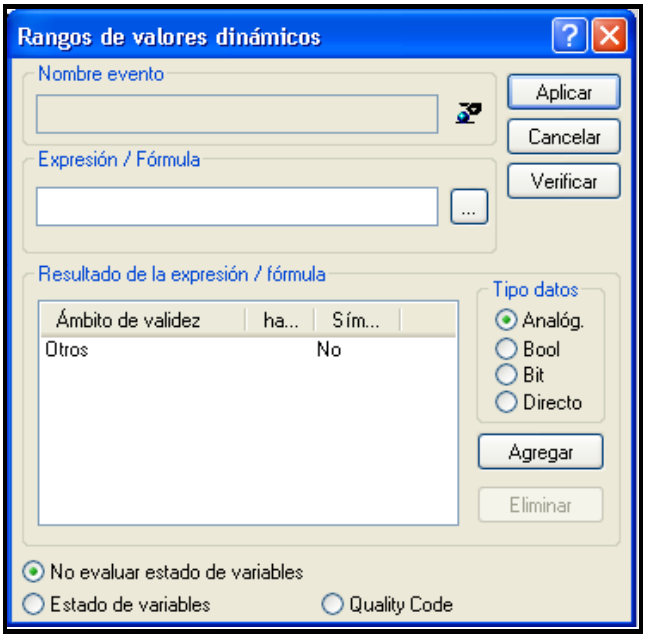


Figura 4.25: Este "Cuadro de dialogo dinámico" se puede emplear para los objetos y los íconos. Para los datos numéricos que puede manejar una variable

4.3 PCAcces

Es un tipo OPCServer que funciona perfectamente con el WinCC. Permite el acceso a cualquier cliente que maneje este protocolo. Para configurar una comunicación de éstas, se debe predefinir:

- El cable PC/PPI de comunicaciones.
- Un equipo con CPU 22X, en la cual esté corriendo un programa adecuadamente.
- Un ordenador portátil o de mesa, como también industrial.

Entonces:

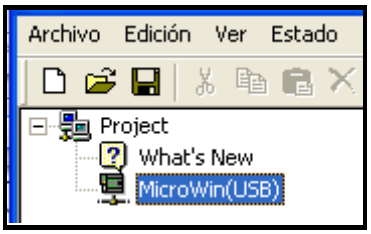


Figura 4.26: Al abrir o seleccionar nuevo, se crea un proyecto en blanco.

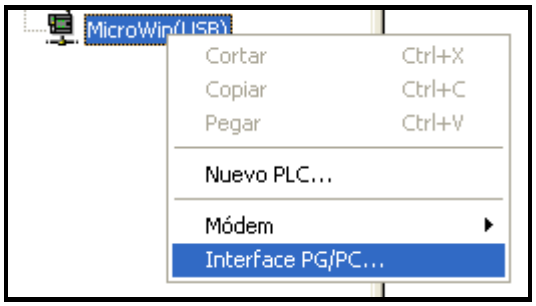


Figura 4.27: Se debe ajustar la interfase de comunicación entre el ordenador y el equipo externo.

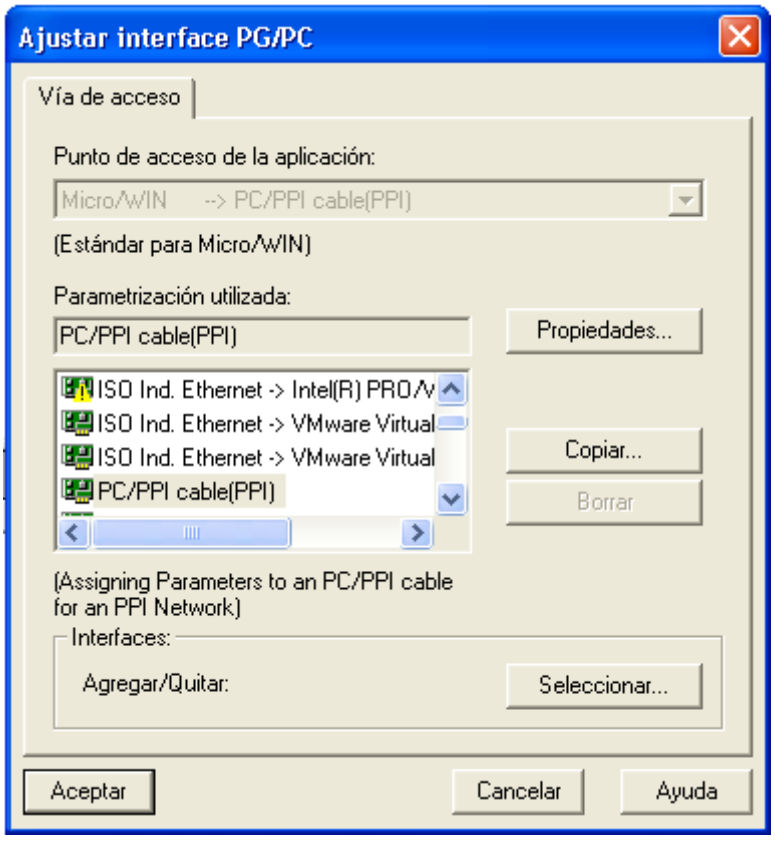


Figura 4.28: Box de ajuste de parámetros de operación de la interface.

Mediante las propiedades del nuevo PLC se configura como esclavo, manteniendo una dirección de estación superior o igual a 2.

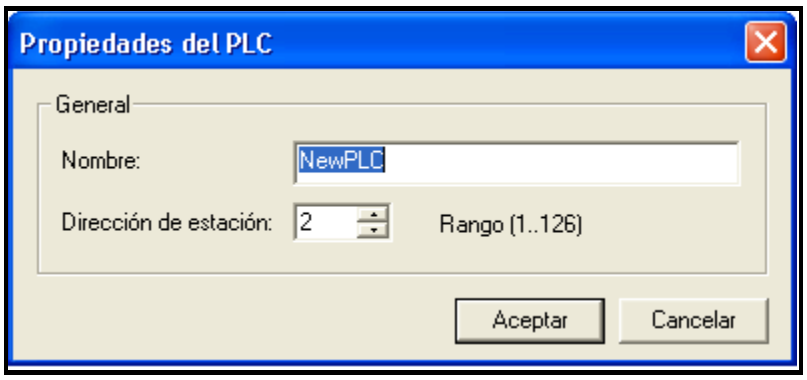


Figura 4.29: Configurar al PLC como esclavo.

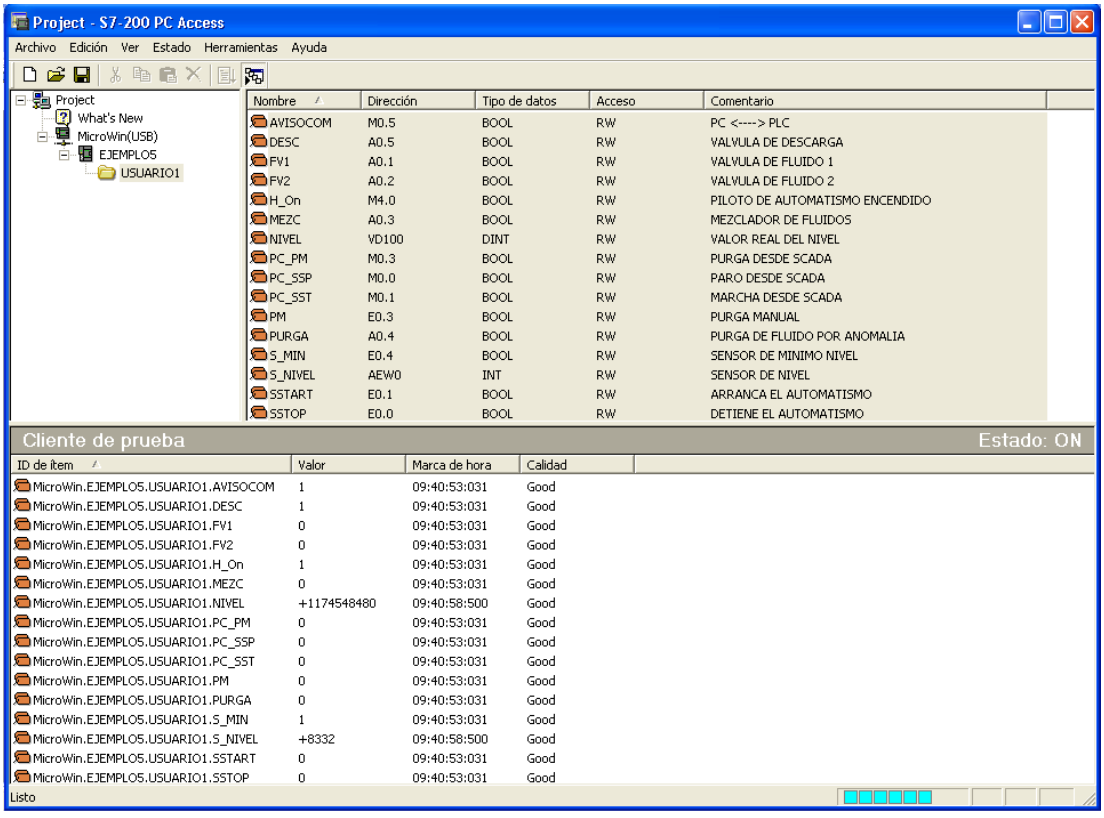


Figura 4.30: Una vez creada la conexión, se accede a las variables del proyecto. Las cuales se pueden pasar al cliente de prueba para verificar la interface. Si la calidad esta "Good", la configuración es correcta.

Sesión **5**

Tema **Ejemplo de aplicación.**

5.1 Definiciones

El procedimiento que se explicará no es el único que se puede emplear para resolver un automatismo, pero será un punto de referencia para afrontar en forma adecuada lo requerido. Para lo cual se tendrá los siguientes pasos:

5.1.1.- Definir el problema.

Se establece que es lo que se quiere automatizar, conociendo:

- Como opera el equipo en cuestión, para así plantear el diseño. Y determinar el tema del proyecto.
- Conocer las partes que integran el equipo, como: sensores, motores (AC o DC, etc), botoneras, etc.
- Establecer los elementos a controlar, para "a priori" idear la forma de control.
- Teniendo lo anterior, plantear el número de entradas/salidas que se requieren para el manejo completo del automatismo.

5.1.2.- Analizar por bloques de funcionamiento las señales necesarias.

Si el equipo lo amerita, en éste punto se hace una disección de la maquinaria objeto de control. Dividiéndola en controles parciales, que luego se unirán para el control

total. Procurando manejar y/o cotejar las señales que intervendrán desde y hacia el PLC.

5.1.3.- Efectuar los diagramas de control.

- Grafcet: secuencia lógica de funcionamiento según las condicionantes de operación de la maquinaria.
- Tabla de símbolos: es importante para manejar las variables que intervienen en el control.
- Diagrama de contactos: se pasa del grafcet al ladder.
- Tabla de estados.
- Pruebas del programa en el MicroWin conjuntamente con el PLC.

5.1.4.- Armar la planta en el WinCC.

Teniendo probado el programa de control, y conociendo las variables que intervienen, se arma la planta con los controles necesarios para que el ordenador también haga el comando.

5.1.5.- Pruebas finales.

Poner a operar la planta y el programa de control del MicroWin, interconectando el PLC a la PC y controlar la planta desde las señales provenientes de las entradas del PLC y desde el ordenador.

5.1.6.- Informe final.

Es recomendable que tome todos los apuntes necesarios para realizar un informe (memoria técnica). Y que siga éstos pasos cuando comience por primera vez a elaborar automatismos. Luego, según su experiencia podrá adoptar su propia metodología de resolución.

5.2 Ejemplo de aplicación

1.- Definir el problema: *Controlar automáticamente la mezcla de dos fluídos*. Para lo cual se conoce:

El fluído 1 (t_1 = 4.5 seg), y fluído 2 (t_2 = 4.3 seg), ingresan por tiempos determinados a un tanque (mix); se mezclan por 8 segundos. Luego la mezcla es descargada hacia otro proceso. Luego, la mezcla es descargada. Con esto se tiene:

- Para controlar el proceso se dispone de una botonera: "stop" y "start", con piloto de encendido.
- La entrada de los fluídos 1 y 2 son manejados por dos equipos, representados por válvulas eléctricas. Una para cada uno de los fluídos.
- La descarga es manipulada por una válvula eléctrica; y la purga del material defectuoso se lo hace por medio de una válvula accionada por un pistón neumático.
- El nivel mínimo es detectado por un sensor capacitivo. Inicia el proceso al detectar el tanque vacío.
- La descarga (purga) del material defectuoso se lo hace por medio de un pulsante. Solo activa al presionarlo.
- El nivel del tanque es chequeado y transmitido al ordenador, luego de leerse por el modulo analógico, por un sensor de radar. Este funciona continuamente.

Consideraciones:

* El piloto de sistema accionado debe de cambiar de color para indicar esto.

* Debe de proveerse de un piloto que indique que existe comunicación entre el ordenador y el PLC.

* Para terminar el modo RunTime se debe disponer de un comando desde el software del ordenador.

* El accionamiento de los equipos se visualiza por el cambio de color.

Con todo esto se elabora la tabla de variables que se requieren en el automatismo:

	Símbolo	Dirección	Comentario
1	SSTART	E0.1	ARRANCA EL AUTOMATISMO
2	SSTOP	E0.0	DETIENE EL AUTOMATISMO
3	PM	E0.3	PURGA MANUAL
4	S_MIN	E0.4	SENSOR DE MINIMO NIVEL
5	S_NIVEL	AEW0	SENSOR DE NIVEL
6	FV1	A0.1	VALVULA DE FLUIDO 1
7	FV2	A0.2	VALVULA DE FLUIDO 2
8	MEZC	A0.3	MEZCLADOR DE FLUIDOS
9	PURGA	A0.4	PURGA DE FLUIDO POR ANOMALIA
10	DESC	A0.5	VALVULA DE DESCARGA
11	H_On	M4.0	PILOTO DE AUTOMATISMO ENCENDIDO y ON DEL SENSOR DE NIVEL POR RADAR
12	NIVEL	VD100	VALOR REAL DEL NIVEL
13	PC_SST	M0.1	MARCHA DESDE SCADA
14	PC_SSP	M0.0	PARO DESDE SCADA
15	PC_PM	M0.3	PURGA DESDE SCADA
16	AVISOCOM	M0.5	PC <----> PLC

Figura 5.1: Tabla de variables de control del automatismo.

Fuente: Autor.- Elaborado en MicroWin V4.0.5.08.

2.- Analizar por bloques de funcionamiento las señales necesarias.

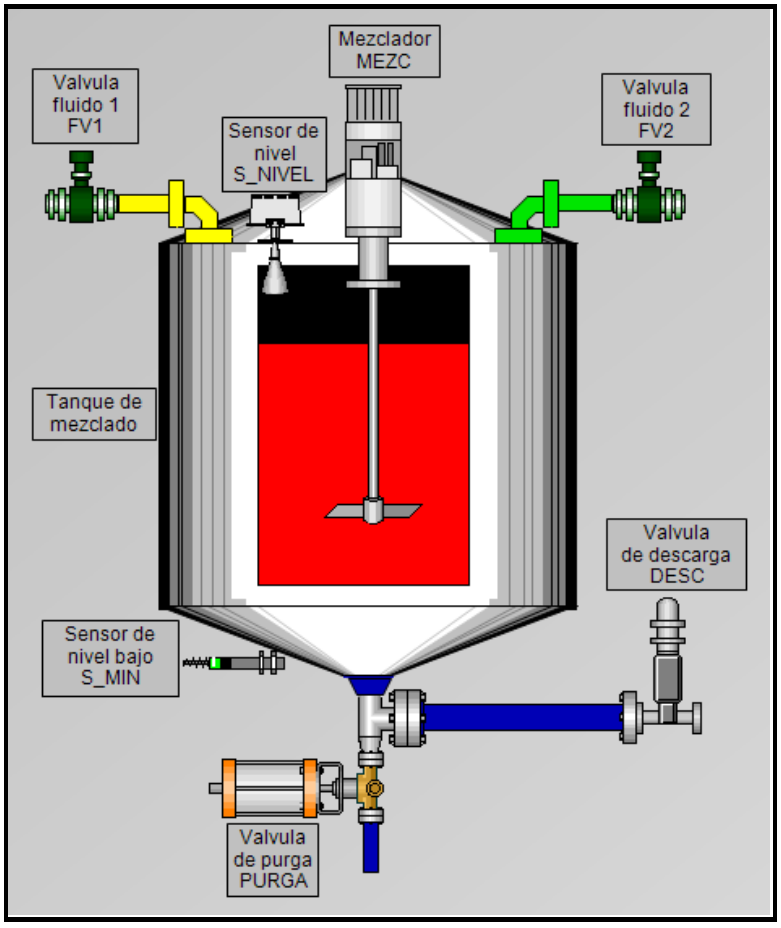


Figura 5.2: Bloque de planta.

Fuente: Autor.- Elaborado con Graphic Designer - WinCC V7.

Figura 5.3: Bloque de control de la planta.

Fuente: Autor.- Elaborado con Graphic Designer - WinCC

Cada uno de los bloques maneja un número determinado de señales para operar:

- Bloque 1: Válvula eléctrica FV_1 → 1 señal eléctrica = on/off.
  Válvula eléctrica FV_2 → 1 señal eléctrica = on/off.

  Sensor de nivel → 1 señal eléctrica analógica contínua.

  Motor agitador → 1 señal eléctrica = on/off.

  Sensor de nivel mínimo → 1 señal eléctrica = on/off.

  Válvula de purga → 1 señal eléctrica = on/off.

- Bloque 2: Control de start y stop → 2 señales.
  Comunicación entre el PLC y la PC → 1 señal.

  Pilotos de start → 1 señal.

  Señal de purga → 1 señal.

  Salir del RT → 1 señal.

3.- Efectuar los diagramas de control.

En función de las condicionantes de operación, y la información de las señales que se requieren; se elabora el grafcet. El mismo que resulta secuencial, dado por el tipo de operación a desarrollar. (*consultar la sesión 3*)

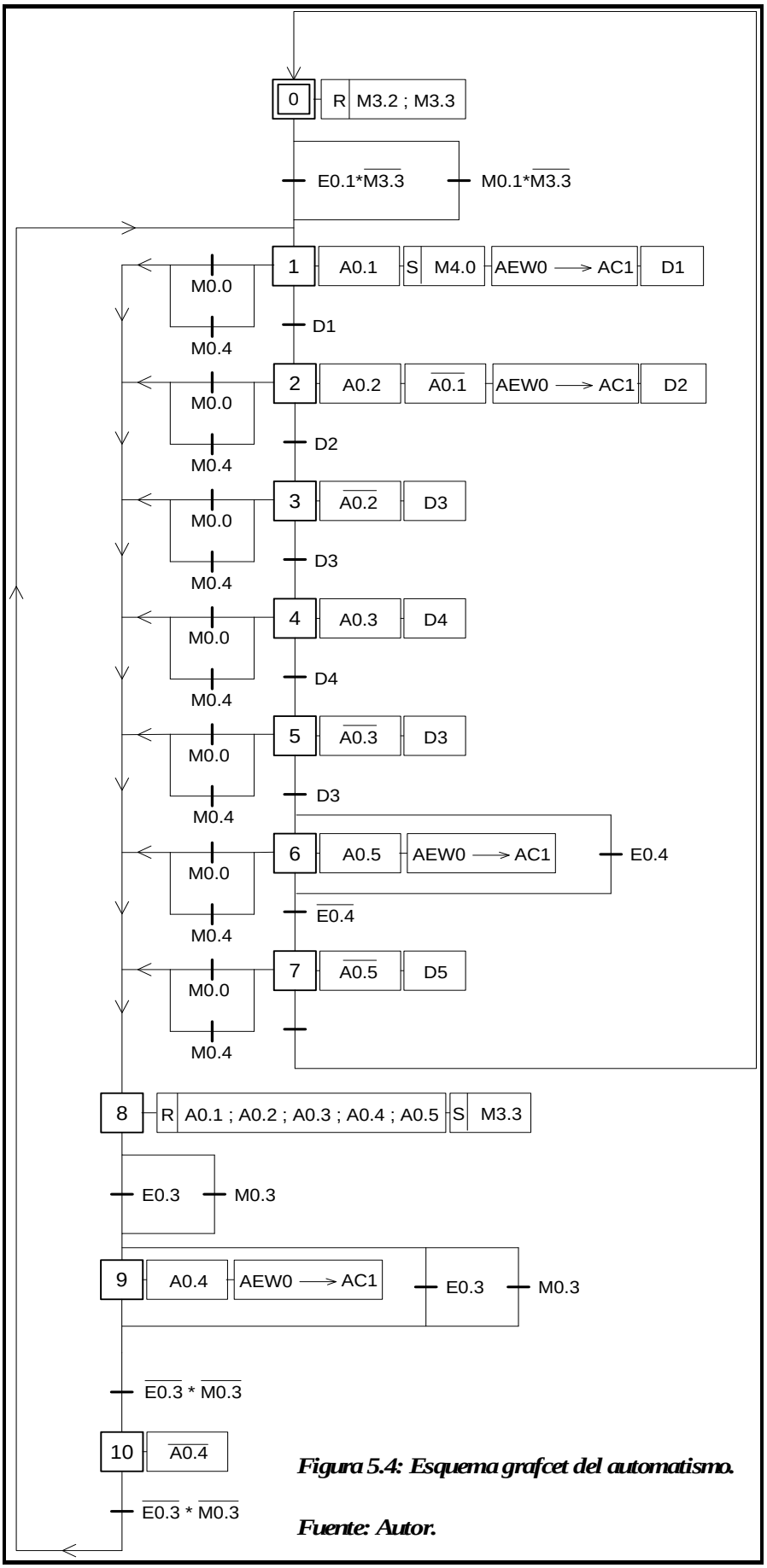
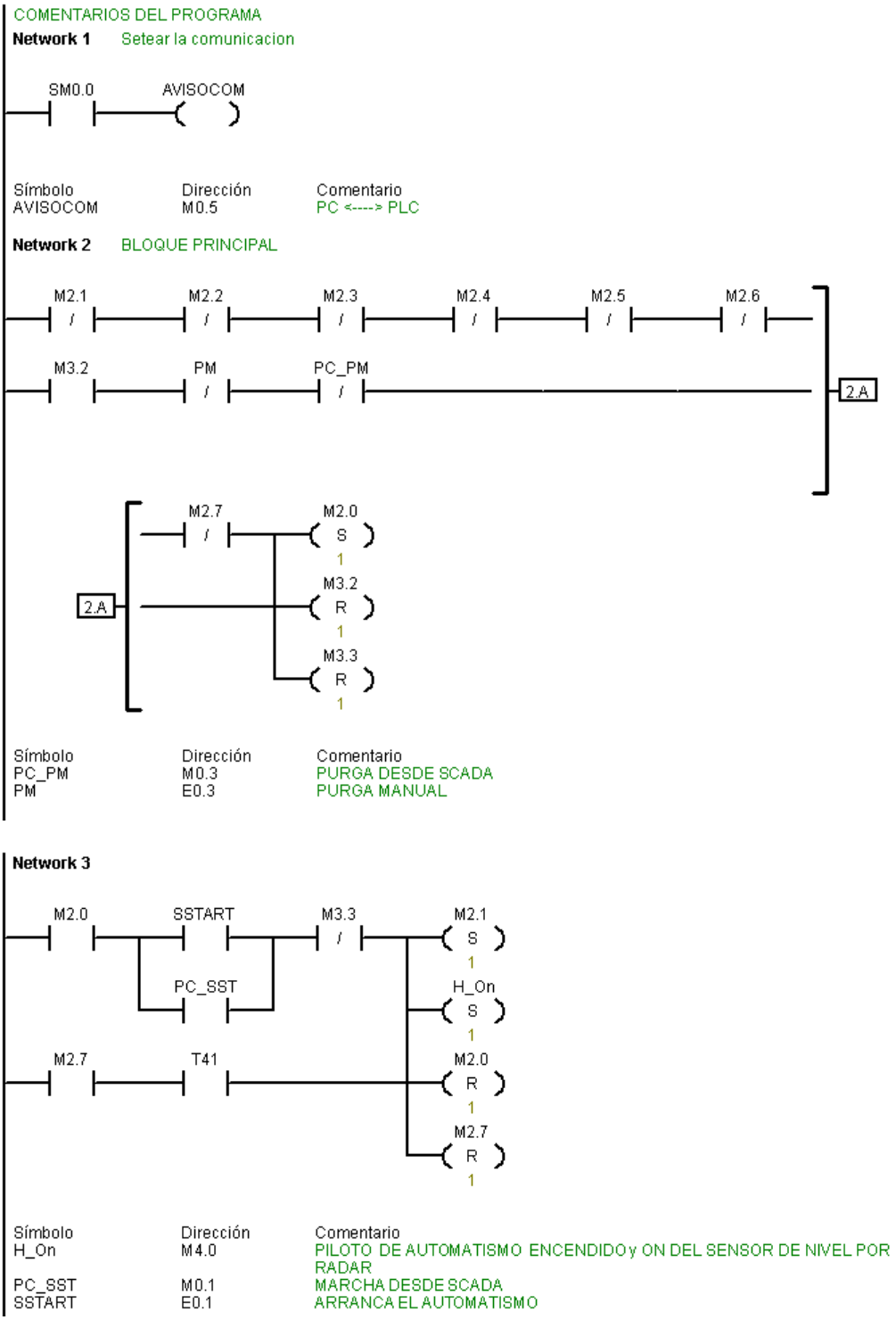


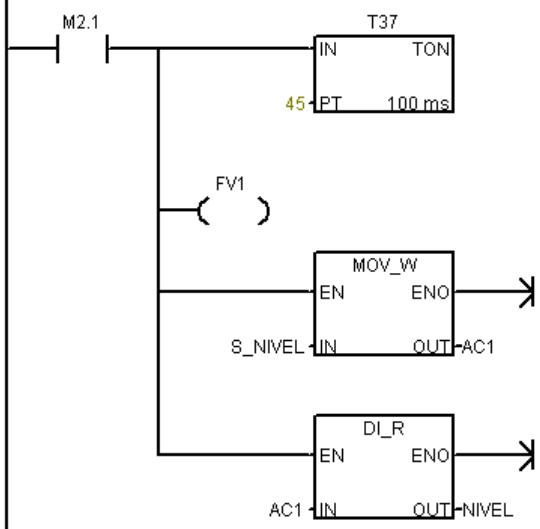
Figura 5.4: Esquema grafcet del automatismo.

Fuente: Autor.

Diagrama de contactos a programar en el MicroWin:

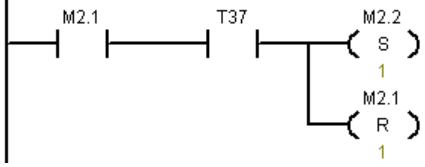


Network 4

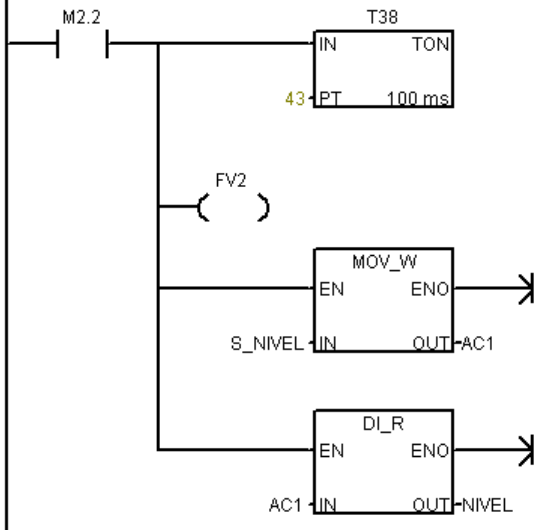


Símbolo	Dirección	Comentario
FV1	A0.1	VALVULA DE FLUIDO 1
NIVEL	VD100	VALOR REAL DEL NIVEL
S_NIVEL	AEW0	SENSOR DE NIVEL

Network 5

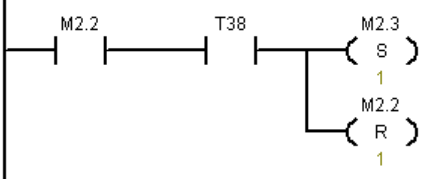


Network 6

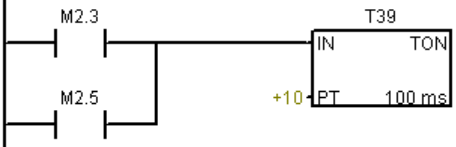


Símbolo	Dirección	Comentario
FV2	A0.2	VALVULA DE FLUIDO 2
NIVEL	VD100	VALOR REAL DEL NIVEL
S_NIVEL	AEW0	SENSOR DE NIVEL

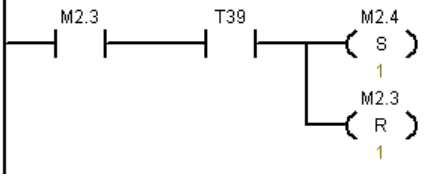
Network 7



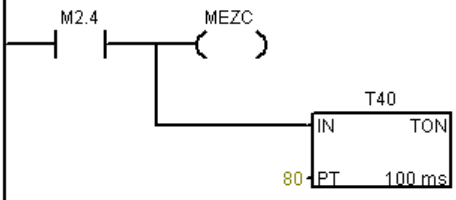
Network 8



Network 9

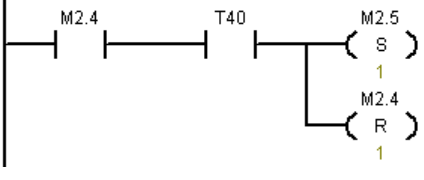


Network 10

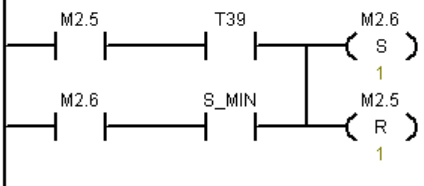


Símbolo	Dirección	Comentario
MEZC	A0.3	MEZCLADOR DE FLUIDOS

Network 11

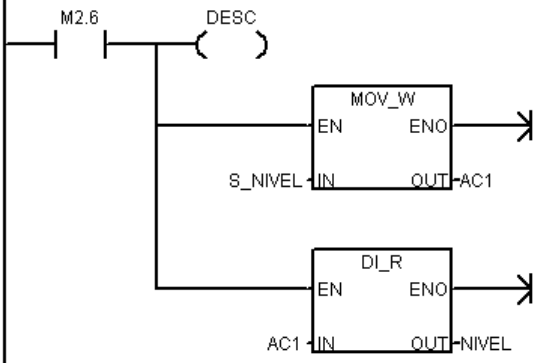


Network 12



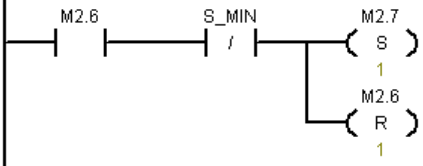
Símbolo	Dirección	Comentario
S_MIN	E0.4	SENSOR DE MINIMO NIVEL

Network 13



Símbolo	Dirección	Comentario
DESC	A0.5	VALVULA DE DESCARGA
NIVEL	VD100	VALOR REAL DEL NIVEL
S_NIVEL	AEW0	SENSOR DE NIVEL

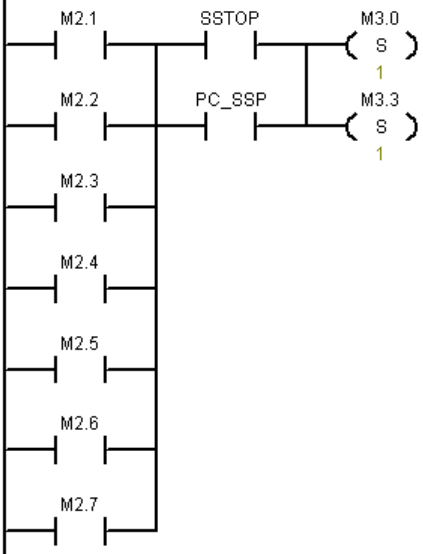
Network 14



Símbolo	Dirección	Comentario
S_MIN	E0.4	SENSOR DE MINIMO NIVEL

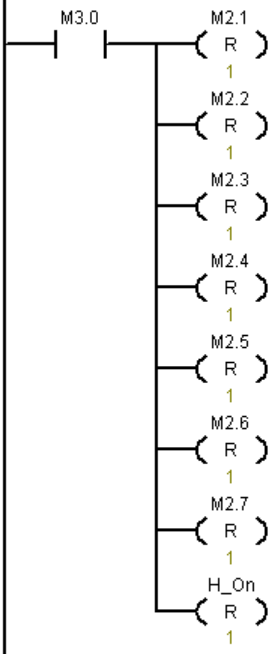
Network 15

Network 16 ACCION DEL PULSANTE DE PARO FISICO Y DEL SCADA



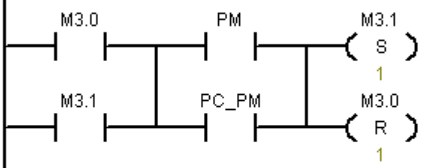
Símbolo	Dirección	Comentario
PC_SSP	M0.0	PARO DESDE SCADA
SSTOP	E0.0	DETIENE EL AUTOMATISMO

Network 17



Símbolo	Dirección	Comentario
H_On	M4.0	PILOTO DE AUTOMATISMO ENCENDIDO y ON DEL SENSOR DE NIVEL POR RADAR

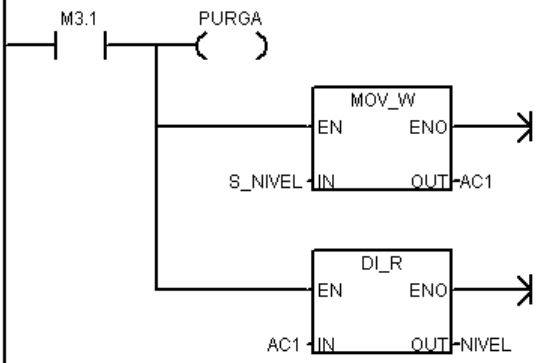
Network 18



Símbolo	Dirección	Comentario
PC_PM	M0.3	PURGA DESDE SCADA
PM	E0.3	PURGA MANUAL

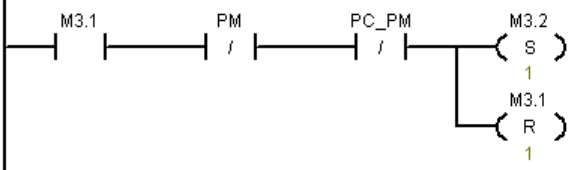
Network 19

Purgado de fluido



Símbolo	Dirección	Comentario
NIVEL	VD100	VALOR REAL DEL NIVEL
PURGA	A0.4	PURGA DE FLUIDO POR ANOMALIA
S_NIVEL	AEW0	SENSOR DE NIVEL

Network 20



Símbolo	Dirección	Comentario
PC_PM	M0.3	PURGA DESDE SCADA
PM	E0.3	PURGA MANUAL

La tabla de estados no se realizó dado que no es necesario para este automatismo.

Para las conexiones en el módulo analógico y el autómata, seguir:

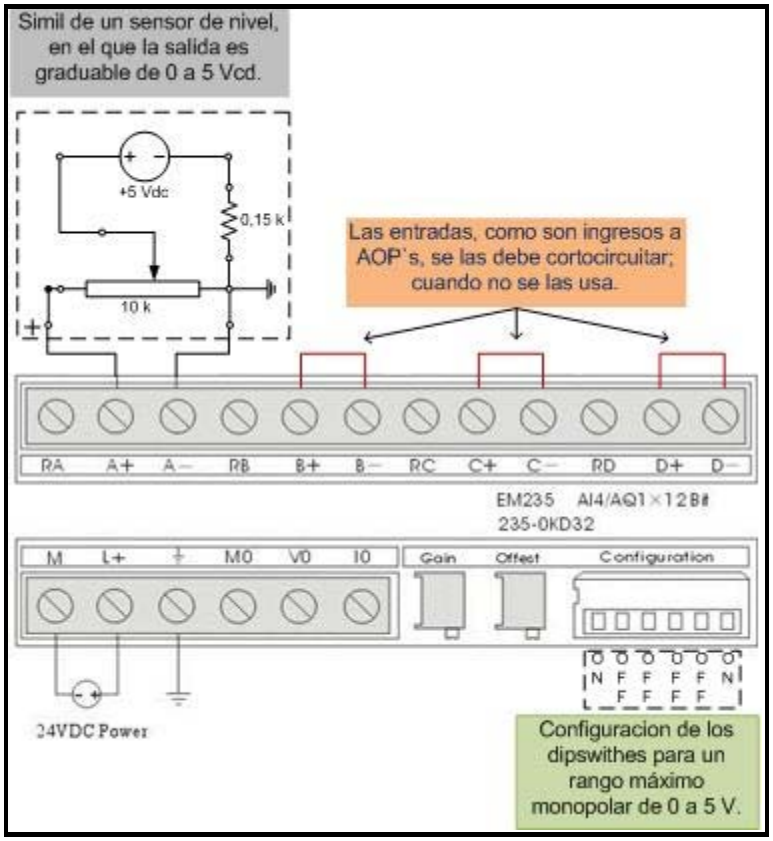


Figura 5.5: Esquema de conexionado en el módulo analógico.

Fuente: Autor.- Basado en: Siemens, Manual del sistema de automatización S7-200, Germany – 08/2005.

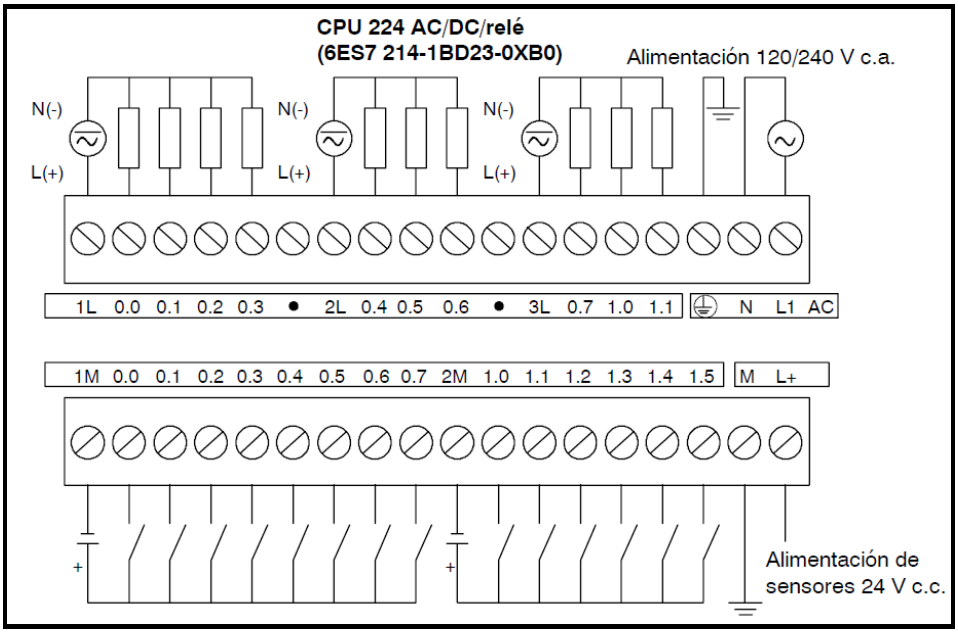


Figura 5.6: Esquema de conexionado en el autómata.

Fuente: Siemens, Manual del sistema de automatización S7-200, Germany – 08/2005.

4.- Armar la planta en el *WinCC*.

Para empezar, se debe de selectar un driver OPC para la conexión. A saber:

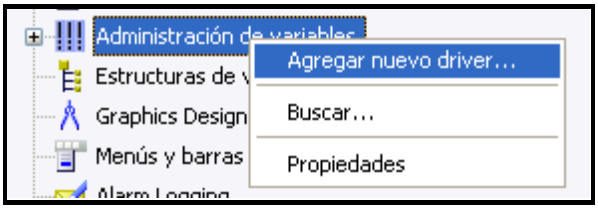


Figura 5.7: Path para escoger driver.

Luego se escoge:

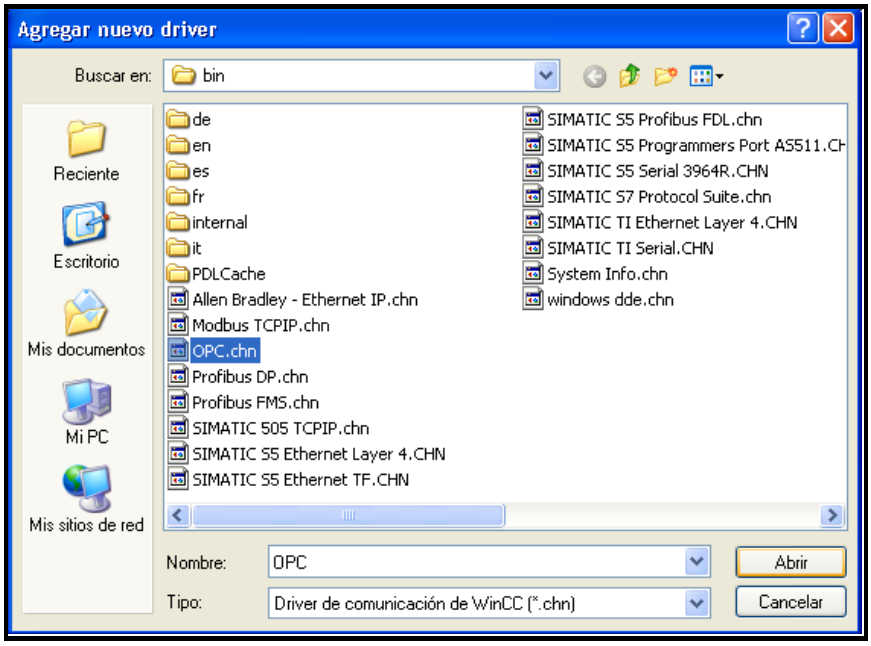
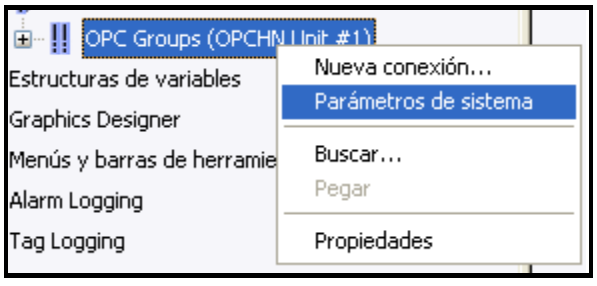


Figura 5.8: Box de drivers soportados por el WinCC.

Para agregar variables reconocidas desde el MicroWin por medio del OPCServer que se esté usando, seguir:



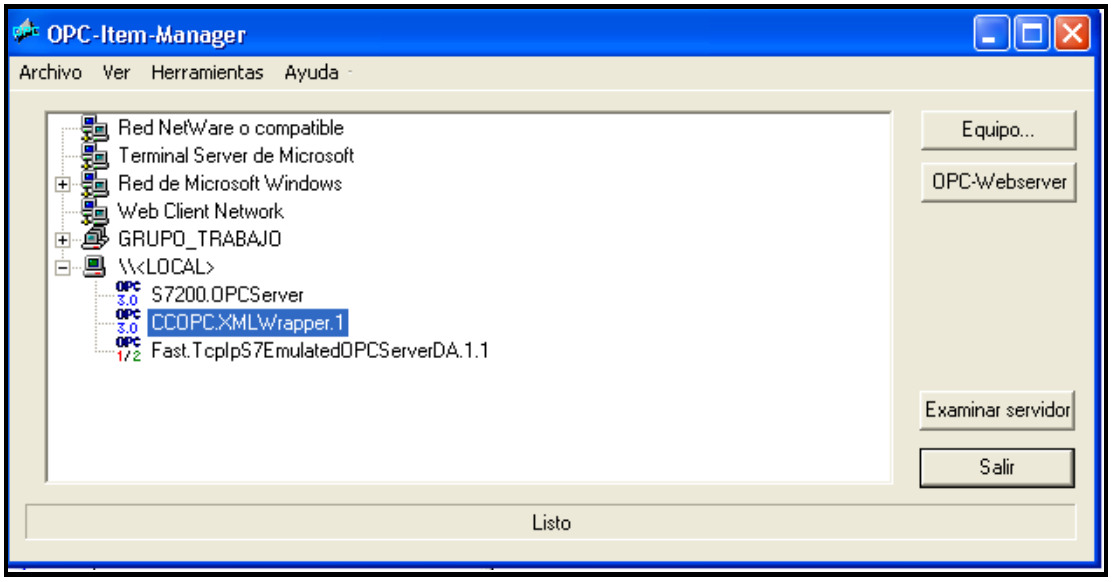


Figura 5.9: Selección del OPC.

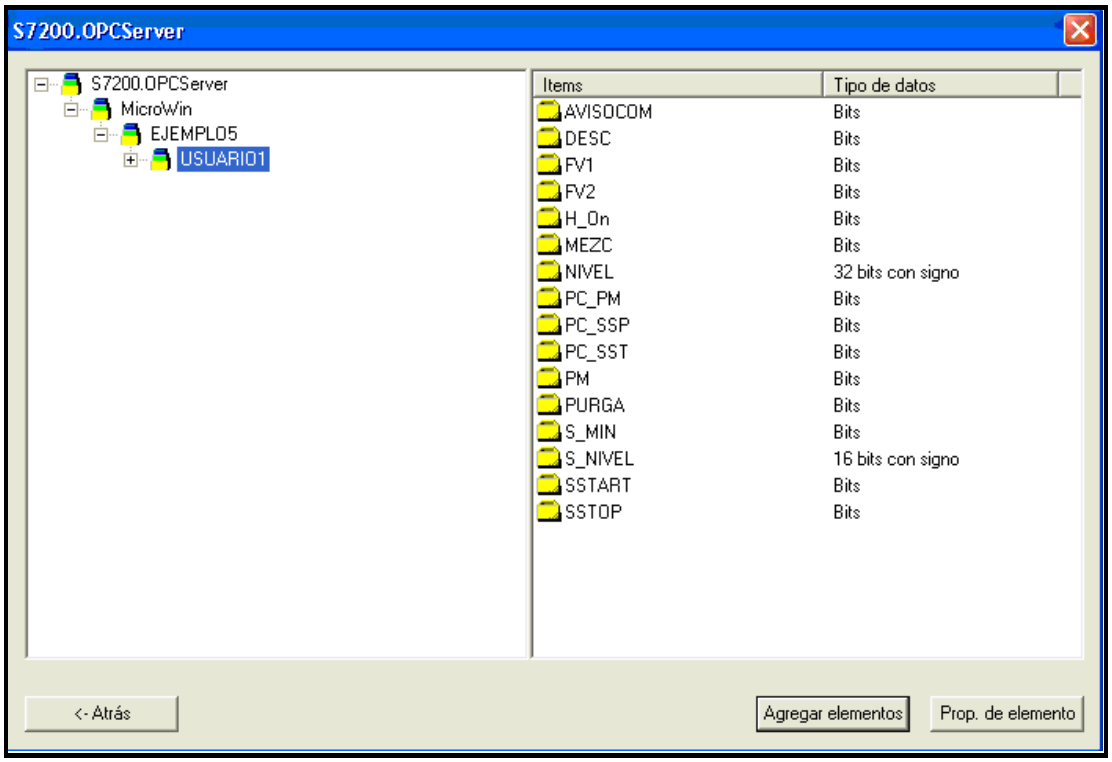


Figura 5.10: Escoger las variables para control y monitoreo y agregarlas.

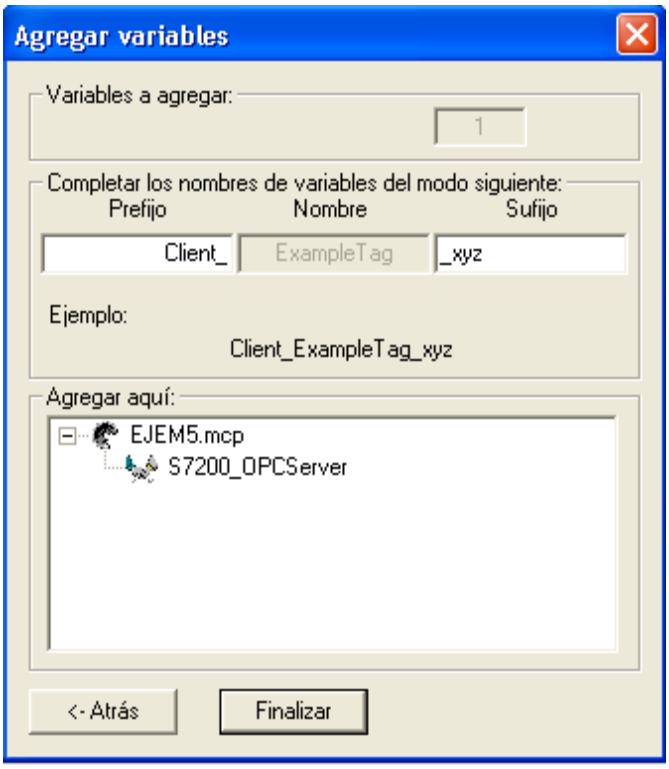


Figura 5.11: Se debe escribir "Client_" y "_xyz", luego pulsar "Finalizar".

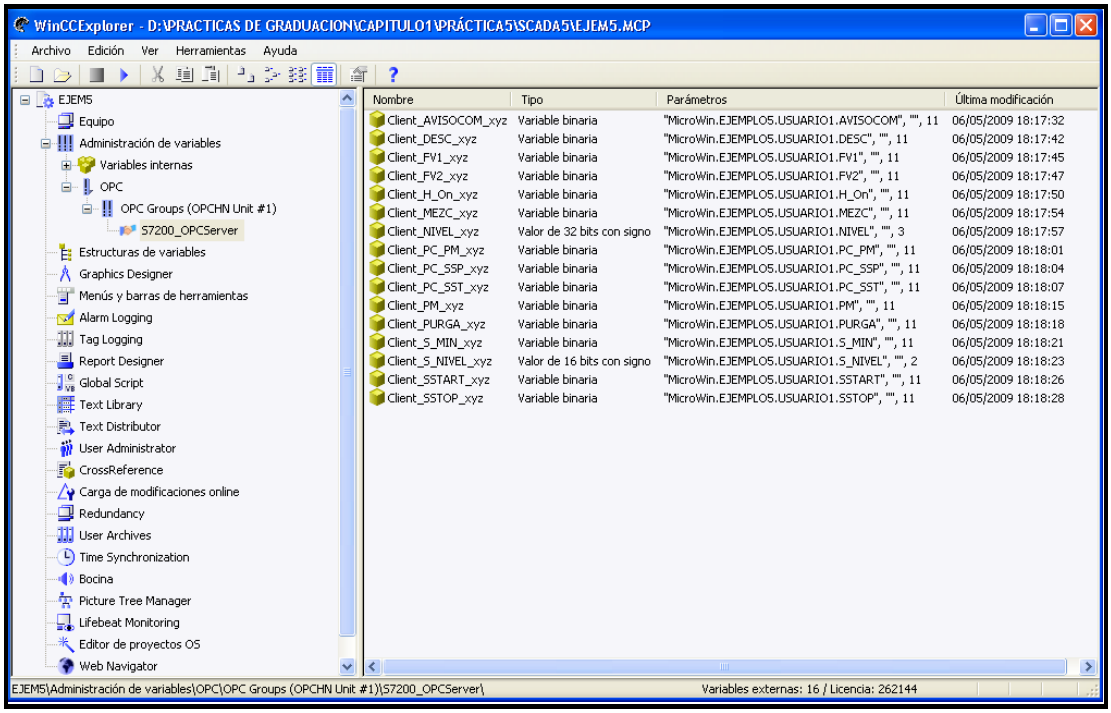


Figura 5.12: Ventana resultante de las variables ya reconocidas por el WinCC.

En el Graphic Designer se escoge nueva hoja para comenzar a dibujar.

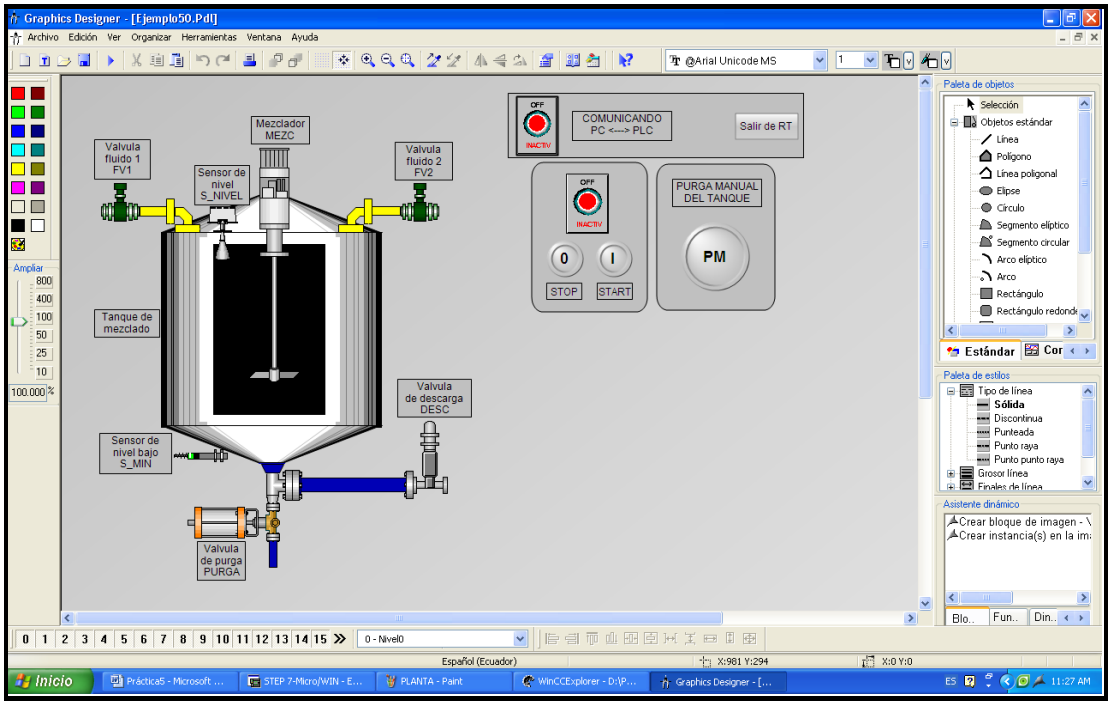


Figura 5.13: Pantalla del Graphics Designer de WinCC.

Para selectar los iconos para el armado, se debe de escogerlos de la librería HMI mostrada por la pulsación de  . Resultando:

Figura 5.14: Ejemplo de símbolos (iconos) en la librería HMI de WinCC.

Para un botón se debe de adicionar un poco de código C para lograr una operación similar a uno real:

```
BOOL pulsado;

{

  pulsado = GetPropBOOL("Ejemplo50","Botónredondo3","Pressed");

  if ( pulsado ) { SetTagBitWait("Client_PC_SST_xyz",1); }

  while ( pulsado )

  {

    pulsado=GetPropBOOL("Ejemplo50","Botónredondo3","Pressed");

    if ( pulsado == FALSE) { SetTagBitWait("Client_PC_SST_xyz",0);}

  }

}
```

Entre tanto para dinamizar los íconos se emplea:

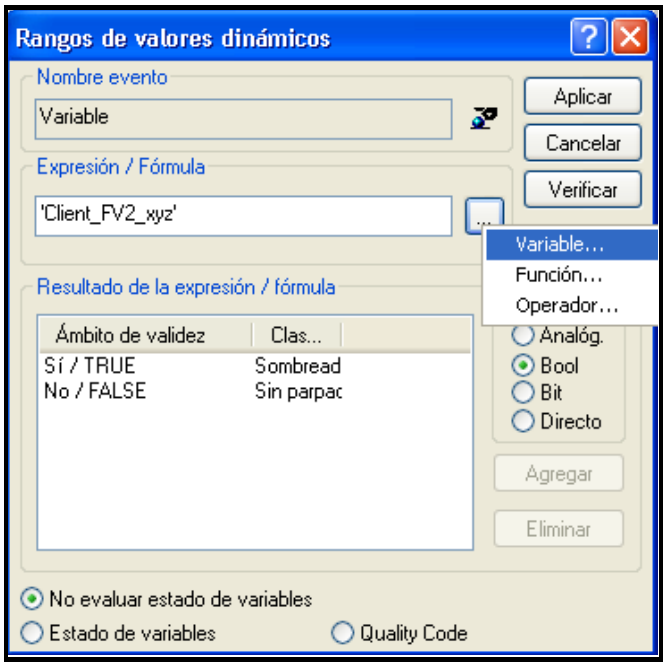


Figura 5.15: Box para dinamizar.

Permitiendo dinamizar por la acción de una variable, de alguna Función, o colocar algún operador matemático.

5.- Pruebas finales.

Para llegar a este punto se debe tener:

* Operando el servidor OPC para que reconozca las variables el WinCC.
* Implementada la planta.
* Probado el programa en el MicroWin.

El PLC debe estar en modo "RUN", adicionalmente el MicroWin se puede visualizar como trabajan los contactos al pulsar  .

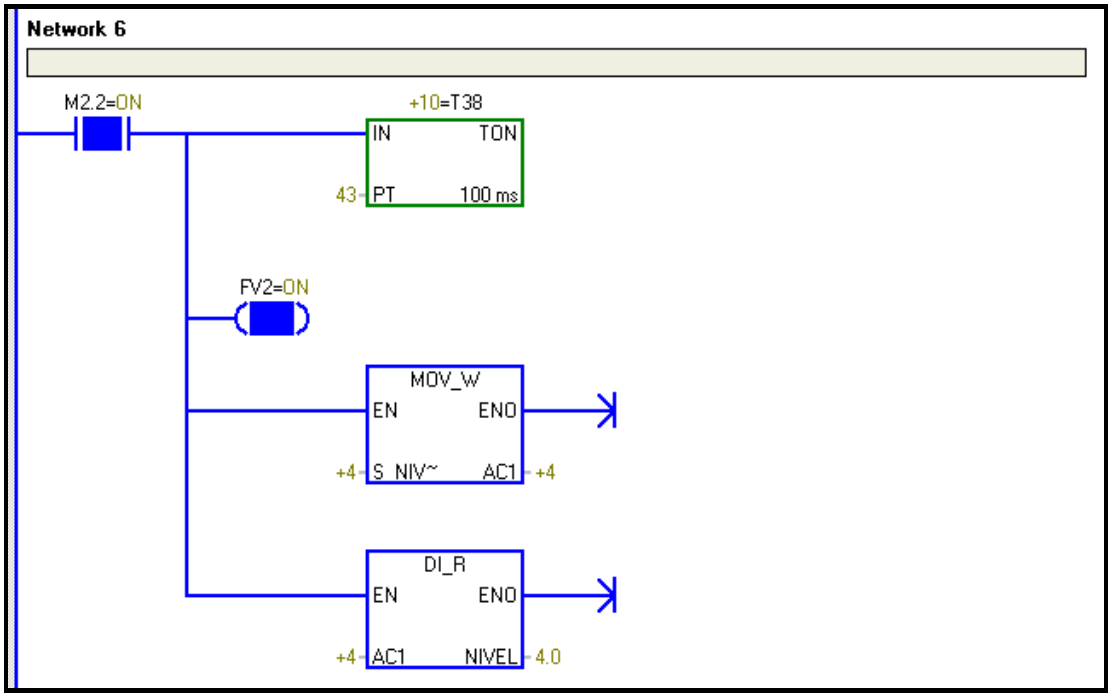


Figura 5.16: Operaciones del MicroWin.

Fuente: Autor.- Obtenido del MicroWin V4.0.5.08.

En esta fase es probable que surjan nuevos problemas de programación, por lo que se debe volver y corregir. Se recomienda que se tome todos los apuntes necesarios para facilitar la implementación posterior.

6.- Informe final.

Con sus apuntes de ésta práctica realice una memoria técnica, que incluya lo siguiente:

- Programa impreso correspondiente al que está corriendo en el autómata, y que está ejerciendo el control sobre la planta, acorde a las condicionantes.
- Diagrama impreso de conexiones físicas realizadas o proyectadas entre la planta y el autómata, y de los sensores si los hubiere.
- Hoja(s) impresa(s) con las consideraciones que encontró durante el diseño y puesta en funcionamiento del automatismo:
    - ¿Qué procedimiento efectuaría para representar en la salida analógica lo que está en la entrada analógica?
    - ¿Qué otro tipo de información necesitó para realizar el automatismo? Indíquela brevemente.
    - Indique si el automatismo es aplicable a la realidad, ¿requiere ajustes de software o hardware para aplicaciones específicas?

Anexo (*adjunte lo que considere necesario para el automatismo*)

Sesión **6**

Tema **Redes de trabajo.**

6.1 Introducción

El objetivo principal es el de proporcionar el intercambio de información entre dispositivos remotos. Este puede realizarse en base a distintas tecnologías:

- Comunicación punto a punto analógica.
- Comunicación punto a punto digital.
- Comunicación punto a punto híbrida.
- Comunicación digital con bus de campo.

Lo que debe brindar a los usuarios, según las necesidades previstas de la planta, de una Red Industrial:

- Reducción de la programación almacenable.
- Evitar el manejo de datos por el autómata en funciones de control.
- Evitar la programación de nodos existentes al añadir nuevos nodos.
- Aumentar las prestaciones del sistema.
- Determinismo en la información que maneja.
- Efectividad del ancho de banda.
- Reducción del cableado.
- Control, programación y diagnóstico sobre la misma red.
- Soluciones escalables.
- Elección del controlador adecuado para el control, y no para datos.

- Añadir o eliminar dispositivos sin influir en otros dispositivos del sistema.
- Reducción de los tiempos de paro.
- Diagnóstico de los dispositivos.
- Información predictiva.

6.2 Tipos de redes

- 6.2.1 *Red de Factoría*: redes de oficina. El volumen de información que se mueve por la red es muy alta, mientras que los tiempos de respuesta no suelen ser críticos.

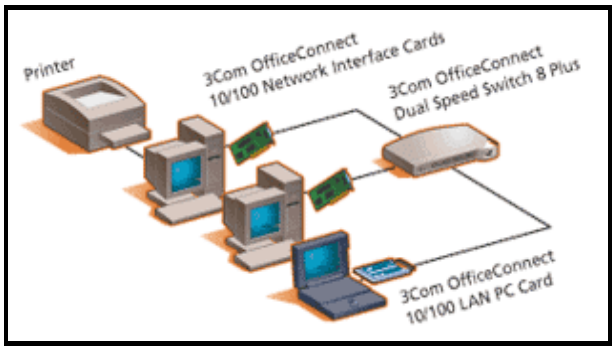


Figura 6.1: Símil de una red de oficinas.

Fuente: 3Com, Folleto pdf sobre redes.

- 6.2.2 *Red de Planta*: interconecta módulos y células de fabricación entre sí, y con los departamentos de diseño y planificación. Manejan mensajes de tamaño variable, gestionar eficazmente los errores de transmisión, cubrir extensas áreas, gestionar prioridades, y tener de un ancho de banda suficiente para manejar datos a subredes.
- 6.2.3 *Red de Célula*: interconecta dispositivos que actúan en modalidad secuencial. Gestiona mensajes cortos, capacidad de manejar trafico de eventos discretos, poseer mecanismos de control de errores, posibilidad de transmitir mensajes críticos, recuperación de eventos anormales y, tener una alta fiabilidad.
- 6.2.4 *Bus de Campo*: sustituye el cableado entre los sensores-actuadores y los correspondientes controladores. Gestiona mensajes cortos, posee mecanismos de control de errores, puede recuperarse de eventos anormales de la red.

Manejan a los elementos conectados pero no llegan a transmitir grandes cantidades de información. (Figura 6.2 y Figura 6.3).

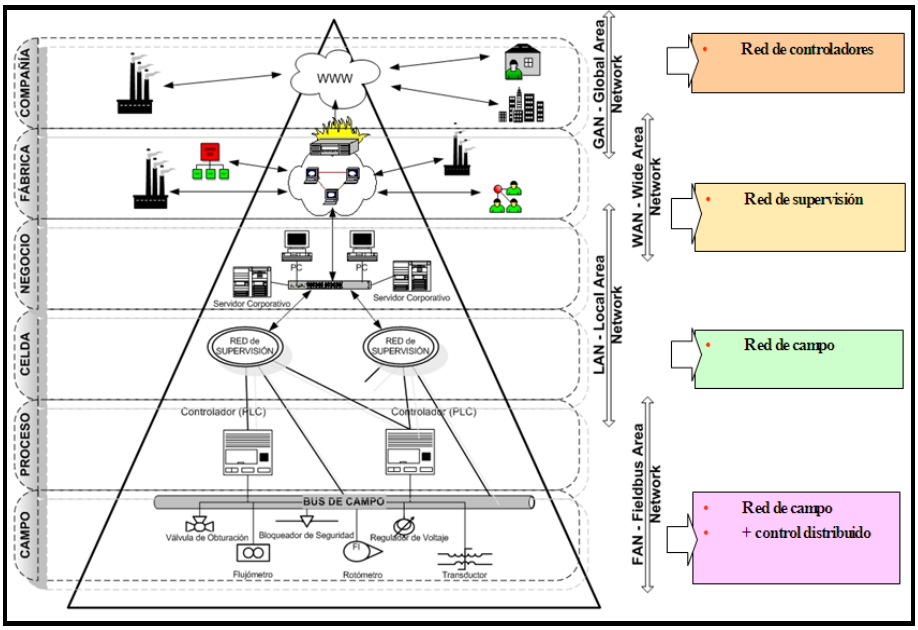


Figura 6.2: Niveles jerárquicos dentro de la pirámide de control. Así como también las diferentes redes que se emplean.

Fuente: Ferreira, Fabiana Ing, FIUBA, Introducción a los Buses de Campo, Buenos Aires – 2008.

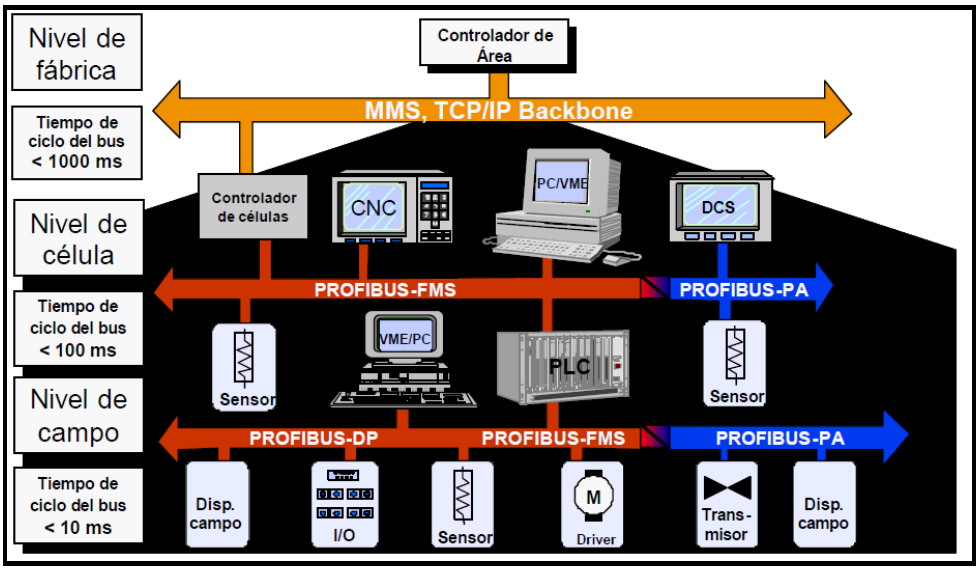


Figura 6.3: Topología de una pirámide de control según Simatic. Considerando los diferentes tipos de Profibus.

Fuente: Rosado Alfredo, Sistemas Industriales Distribuidos - Redes de Comunicación Industriales, Universidad de Valencia, Dpto. de Ingeniería Electrónica, texto pdf, España – 2003.

Entonces, "todas las *redes de comunicaciones industriales* tienen un orígen común, la fundación *FieldBus* (o Buses de campo). Esta desarrolló un nuevo *protocolo de comunicaciones* para la medición y control de procesos donde todos los instrumentos puedan comunicarse en una misma *plataforma*." (Marcos Peluso, 1994)

Dichas comunicaciones entre los instrumentos de proceso y el sistema de control se basan principalmente en señales analógicas (neumáticas de 3 a 15 psi en las válvulas de control; y, electrónicas de 4 a 20 mA, o de 0 a 5 / 10 V en CD).

Pero también se tiene instrumentos digitales capaces de manejar gran cantidad de datos y guardarlos en un histórico. Cuya precisión es diez veces mayor que la de la señal típica análoga. En vez de transmitir cada variable por un par de hilos, transmiten secuencialmente las variables por medio de un cable de comunicaciones llamado bus.

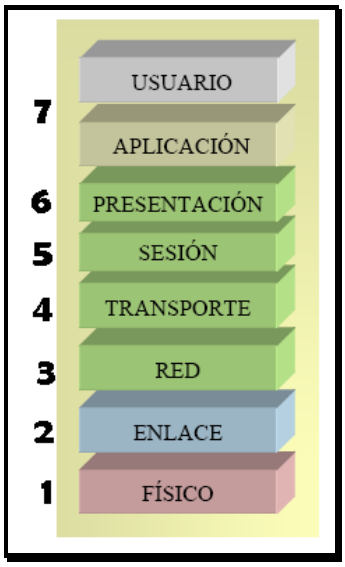
La *tecnología fieldbus* es un protocolo de comunicaciones digital de alta velocidad que esta creada para remplazar la clásica señal de 4-20 mA, que aún se utiliza en muchos de los sistemas DCS (Sistema de Control Distribuido) y PLC (Controladores Lógicos Programables), en los instrumentos de medida y transmisión y válvulas de control. La *arquitectura fieldbus* conecta estos instrumentos con ordenadores que se usan en diferentes niveles de coordinación y dirección de la planta."

Muchos de los protocolos patentados para aplicaciones propietarias se caracterizan por ser una "tecnología cerrada", lo que es un gran limitante. Lo que está tendiendo a desaparecer, con la "tecnología abierta", logrando dar mejoras al protocolo de comunicación para mejorar la transferencia de los datos, con lo que se asegura mejor de sincronismo de los elementos de la red, y el tiempo real de respuesta determinística de las aplicaciones.

6.3 EL MODELO OSI DE 7 NIVELES

Base fundamental de las comunicaciones abiertas, si el intercambio de datos entre sistemas de automatización se produce a través de un bus, es importante definir el sistema de transmisión y el procedimiento de acceso. Además, deben definirse

informaciones, por ejemplo, sobre el establecimiento de las comunicaciones. Por este motivo, la Organización de Normalización Internacional (ISO), definió un modelo de siete niveles o capas. Este modelo se subdivide en dos secciones (Siemens, Catálogo, IK 10, 1997):



a. Orientados al transporte

(Niveles 1-4)

b. Orientados al usuario(niveles 5-7)

Figura 6.4: Representación de los niveles OSI.

Fuente: Pérez Fede, Introducción a las Redes de Comunicaciones Industriales, España, 2007. (http://www.disa.bi.ehu.es/spanish/ftp/material_asignaturas/Fundamentos%20de%20Automatizaci%F3n%20Industrial/Comunicaciones%20y%20Supervisi%F3n/Introducci%F3n%20a%20las%20Comunicaciones%20Industriales.pdf)

6.4 Redes de "Tecnología OPEN"

6.4.1 *Redes Sensor – Actuador: ASi*

El bus "Actuator – Sensor Interface" es un bus destinado a la eliminación del cableado existente entre los sensores y los actuadores binarios, con la alimentación de alimentación y la transmisión de señales binarias sobre el mismo cable. Definido por el estándar europeo EN 50295 e IEC 62026-2. "Se considera uno de los sistemas más sencillo y con menos prestaciones, por lo que se lo emplea a nivel de campo en la parte más baja de la pirámide de automatización." (Univ. de Oviedo, 2002)

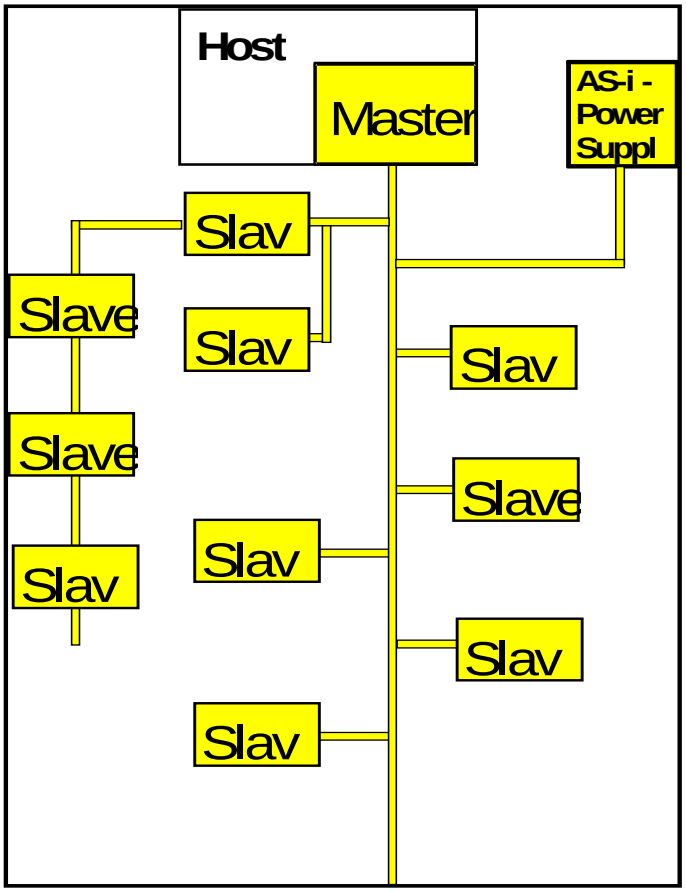


Figura 6.5: Configuración de una red ASi.

Fuente: Ferreira, Fabiana Ing, FIUBA, Introducción a los Buses de Campo, Buenos Aires – 2008.

Posee las características siguientes:

- El "master" utiliza la técnica de "polling" con sus esclavos.
- En 5 ms (máximo) se conoce el valor de todos los esclavos.
- Cada "master" puede manejar hasta 31 "slaves". Cada "slave" permite direccionar 4 "inputs" y 4 "outputs" digitales, y adicionalmente, 4 bits de parámetro por cada "slave", con un máximo de 248 input/output digitales.
- Posibilidad de I/O analógicas.
- Direccionamiento electrónico de los "slaves".
- Puede manejar cualquier topología de red, con una longitud máxima de 100 metros sin repetidores con caída de tensión de 3 V.
- Cada "slave" debe tener un voltaje de funcionamiento entre 26,5 y 31,6 V; la corriente de consumo típica es de 200 mA.

6.4.2 Redes DeviceNet

Es una red orientada a los niveles medio-bajos dentro de la pirámide de la automatización de una planta. Constituye un protocolo versátil en el área de buses de campo, utilizado en la comunicación a nivel de célula con: Fines de carrera, sensores fotoeléctricos, sensores inductivos, válvulas, arrancadores de motores, lectores de código de barras, y otros que posean certificados DeviceNet.

Además, es una solución simple de comunicación en red que reduce el costo y tiempo para cablear e instalar dispositivos de automatización industrial, al mismo tiempo que provee intercambiabilidad de componentes similares de distintos fabricantes**.**

Presenta algunas características:

- El número máximo de nodos está limitado a 64.
- Distancia máxima entre 100 y 500 metros.
- Corriente nominal de tronco: 8 A.
- Velocidad de transferencia de datos 125, 250 y 500 kbps, depende de la distancia admisible.(Tabla 6.1)

DATA RATES	125 KBPS	250 KBPS	500 KBPS
Thick Trunk Length	500 m (1,640 ft)	250 m (820 ft)	100 m (328 ft)
Thin Trunk Length	100 m (328 ft)	100 m (328 ft)	100 m (328 ft)
Flat Trunk Cable	380 m (1,250 ft)	200 m (656 ft)	75 m (246 ft)
Maximum Drop Length	6 m (20 ft)	6 m (20 ft)	6 m (20 ft)
Cumulative Drop Length	156 m (512 ft)	78 m (256 ft)	39 m (128 ft)

Tabla 6.1: Velocidades aproximadas dependiendo de la distancia lineal.

Fuente: Ferreira, Fabiana Ing, FIUBA, Introducción a los Buses de Campo, Buenos Aires – 2008.

- Corriente nominal de tronco: 8 A.
- En la terminación de línea se requiere de una impedancia de 120 ohmios.
- Tamaño del mensaje de 8 bytes máximo, para cada nodo.
- El sistema de transmisión está basado en "productor/consumidor", admitiendo el modelo "master/slave", "multimaster"; que es el envío de mensajes por "polling", envío cíclico, etc.
- Admite varias Topologías Básicas: Tronco (trunk)- rama (drop line -spurs). (Fig. 6.6)

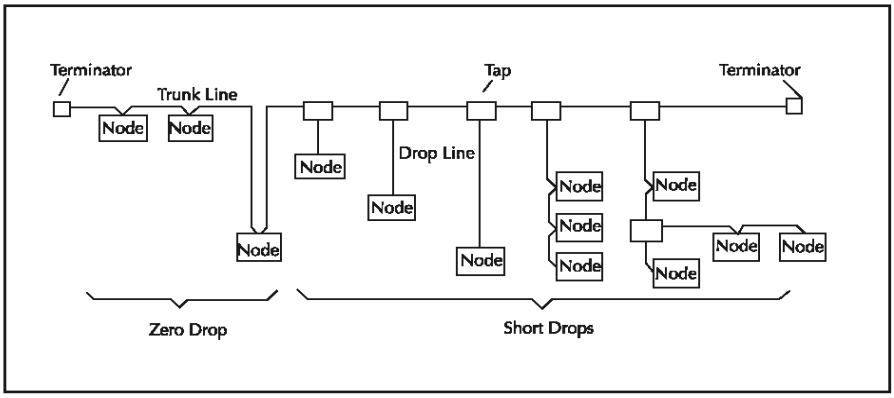


Figura 6.6: Configuración de una Red DeviceNet.

Fuente: Rosado Alfredo, Sistemas Industriales Distribuidos - Redes de Comunicación Industriales, Universidad de Valencia, Dpto. de Ingeniería Electrónica, publicación pdf, España – 2003.

Emplea las especificaciones del bus CAN (la capa 2 de DeviceNet es totalmente CAN), por lo que maneja la robustez de la CAN, añadiendo las especificaciones eléctricas de RS-485. Como CAN no incorpora la capa de aplicación, DeviceNet lo aprovecha para desarrollar sus propios circuitos integrados.

Por lo que al protocolo CAN se le añaden nuevas capas dentro de los niveles ISO/OSI. (Fig. 6.7)

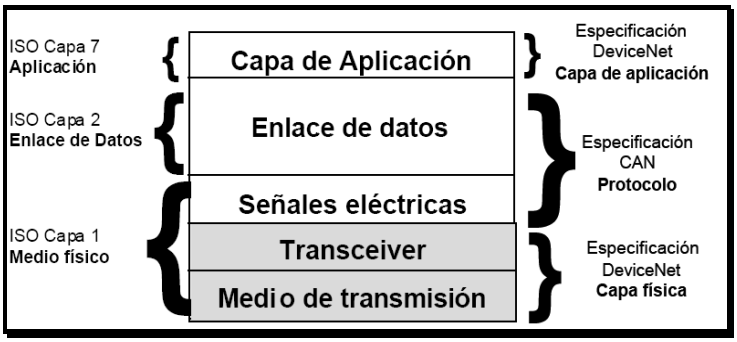


Figura 6.7: Ubicación de las capas ISO/OSI de Devicenet y empleo del protocolo CAN.

Fuente: Rosado Alfredo, Sistemas Industriales Distribuidos - Redes de Comunicación Industriales, Universidad de Valencia, Dpto. de Ingeniería Electrónica, publicación pdf, España – 2003.

Las funciones que agrega la capa de aplicación de DeviceNet son:

- Asigna identificadores a cada nodo CAN, establece niveles de prioridad.
- Gestión de mensajes a transmitir.
- Detección de duplicación de direcciones.
- Gestión de los datos que viajan a cada nodo.

6.5 Buses de Campo

Manejan protocolos abiertos pero que son impulsados por diversos fabricantes, entre ellos existen ciertas diferencias. Pero a pesar de eso, se pueden efectuar las mismas aplicaciones con cualquiera. Estos protocolos son:

6.5.1 Hart

Proporciona comunicación bidireccional con dispositivos de campo inteligentes mientras conserva la compatibilidad y familiaridad de los tradicionales “4 a 20 mA”. Consiste en el uso de la norma Bell 200, ésta permite la superposición simultánea a niveles bajos de una señal de comunicaciones digital en la parte superior de la señal análoga “4 a 20 mA”.

6.5.2 Foundation FieldBus

Constituye un subconjunto del estándar IEC/ISA. El objetivo principal es la sustitución del cableado asociado a los elementos aislados tales como los análogos "4 a 20 mA", por un bus capaz de proporcionar compatibilidad entre ellos con la inclusión de un dispositivo pequeño interfaz.

Emplea las capas 1,2 y 7 del modelo OSI. Las señales son transmitidas por un cable trenzado con una velocidad de 31,25 kbps. Se pueden conectar hasta 32 elementos en el bus, con una distancia de 1900 metros sin repetidores. Lo importante es que los dispositivos, por la cierta inteligencia incorporada, poseen funcionamiento autónomo a pesar de las interrupciones de la transmisión de datos, se dispone de un nivel intrínseco de seguridad.

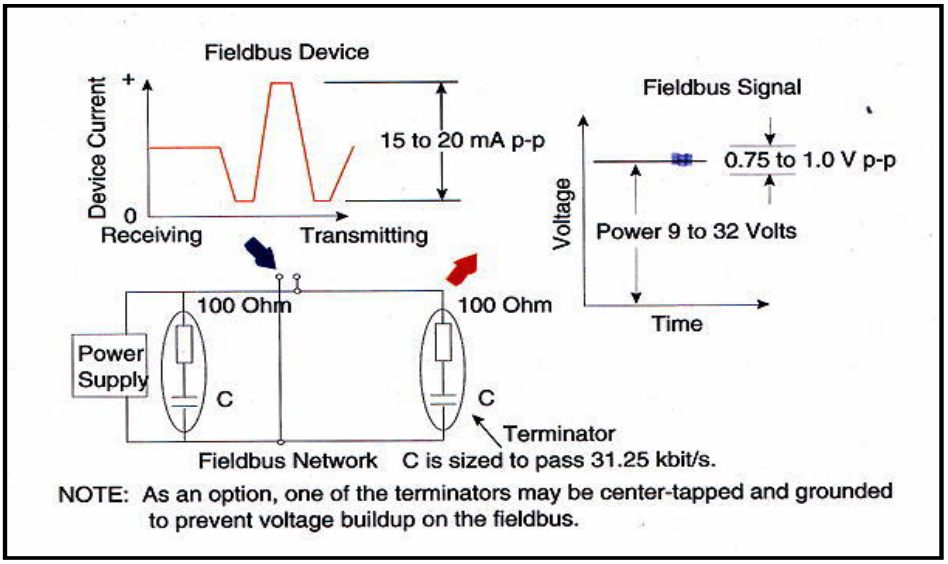


Figura 6.8: Características eléctricas del bus Foundation FieldBus.

Fuente: Ferreira, Fabiana Ing, FIUBA, Introducción a los Buses de Campo, Buenos Aires – 2008.

Presenta las topologías:

- Bus con derivaciones.
- Punto a punto.
- Daisy-Chain.
- Árbol.

6.5.3 ModBus

Es un protocolo para transmitir y recibir datos de control entre los controladores y los sensores, por medio del puerto RS-232, con un alcance máximo de 350 metros. Opera utilizando el sistema "master/slave", enviando dos caracteres (modo ASCII), espaciados hasta 1 segundo, para cada mensaje; como también enviando 4 caracteres hexadecimales de 4 bits cada uno (modo RTU). Algunos protocolos modbus se implementan en RS-485, cubriendo distancias de hasta 1500 metros con 32 nodos. La transmisión en por par trenzado apantallado, y la tensión de alimentación es independiente para cada dispositivo.

El ModBus sobre TCP/IP aprovecha la infraestructura de Internet, se empaqueta mensajes modbus dentro de lo paquetes TCP/IP.

6.5.4 ProfiBus

Es el estándar europeo en tecnología de buses, se encuentra jerárquicamente por encima de ASI y BITBUS, trabaja según procedimiento híbrido "token passing", dispone de 31 participantes hasta un máximo de 127. Su paquete puede transmitir un máximo de 246 Bytes, y el ciclo para 31 participantes es de aproximadamente 90 ms. Alcanza una distancia de hasta 22300 m.

Red abierta para procesos (Process Fieldbus); y, contiene 3 protocolos compatibles:

- Decentralized Peripheral (**DP**)
- Field Messaging Specification (**FMS**)
- Process Automation (**PA**)

Las características técnicas y funcionales son de un bus de campo serie, en el cual los dispositivos de control digital descentralizados pueden ser conectados entre sí desde el nivel de campo al nivel de control. Se tienen dos clases de dispositivos:

- **Master**, o estaciones activas, son los que manejan las comunicaciones de datos sobre el bus.
- **Slave**, o estaciones pasivas, suelen dispositivos I/O que solo requieren una parte del protocolo del bus.

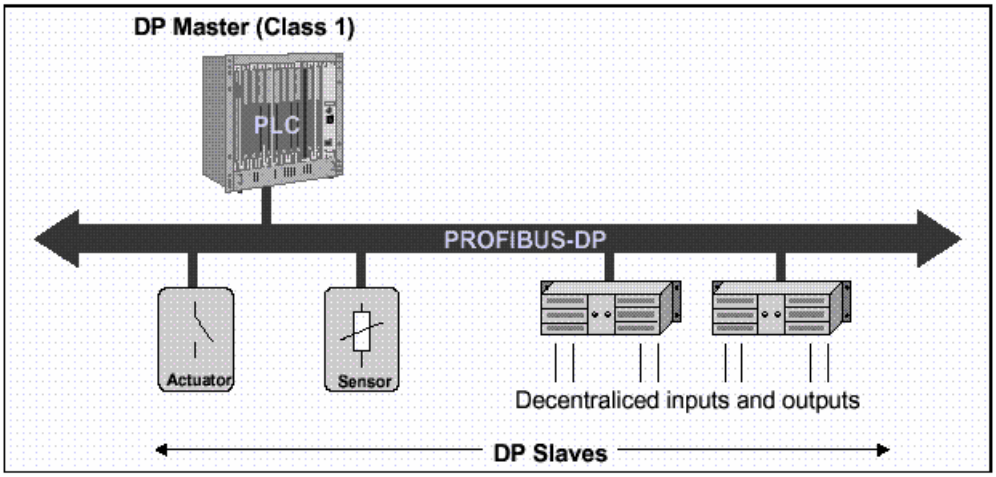


Figura 6.9: Configuración monomaestro de una red ProfiBus DP.

Fuente: Rosado Alfredo, Sistemas Industriales Distribuidos - Redes de Comunicación Industriales, Universidad de Valencia, Dpto. de Ingeniería Electrónica, texto pdf, España – 2003.

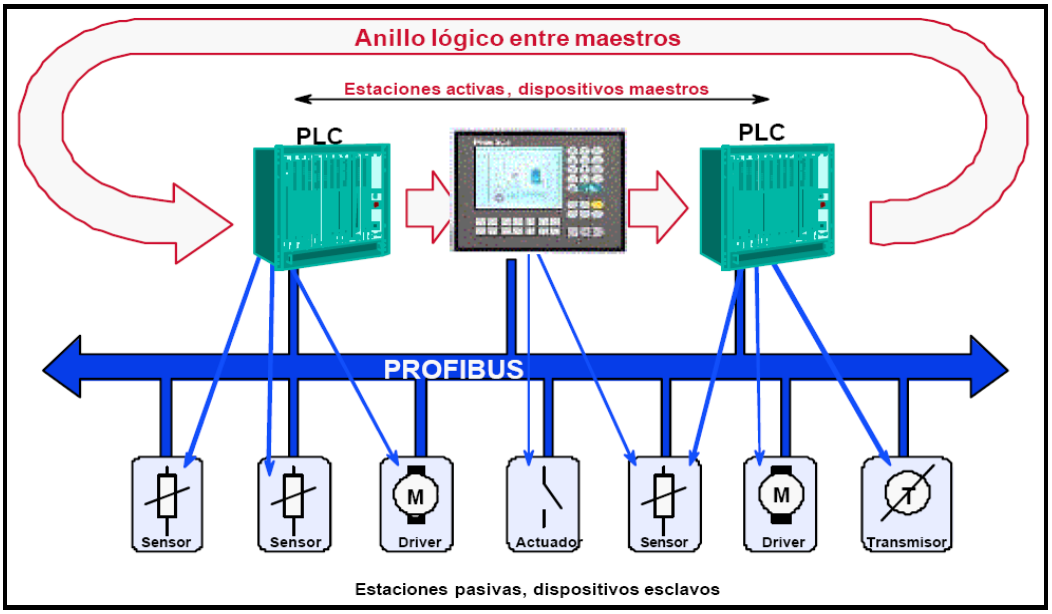
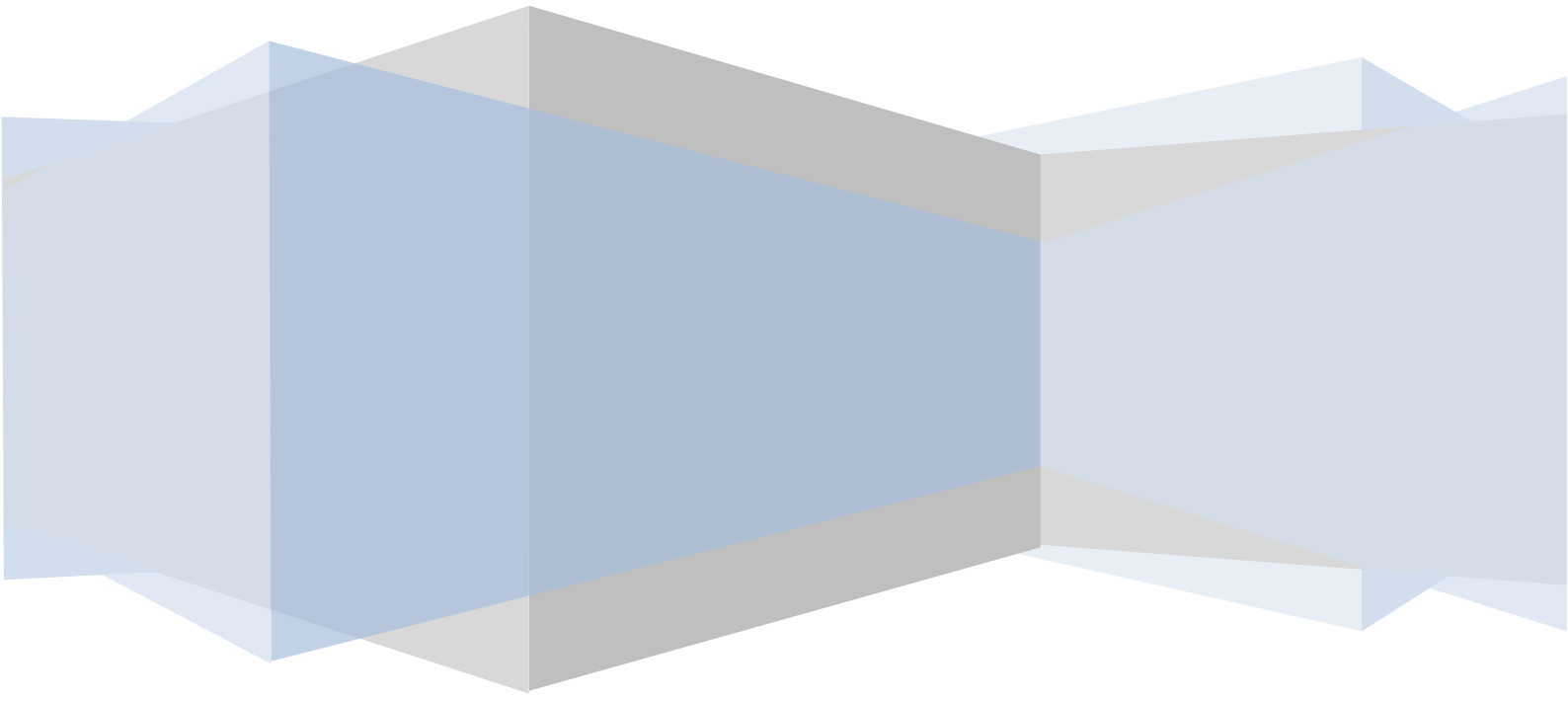


Figura 6.10: Configuración multimaestro de una red ProfiBus.

Fuente: Rosado Alfredo, Sistemas Industriales Distribuidos - Redes de Comunicación Industriales, Universidad de Valencia, Dpto. de Ingeniería Electrónica, texto pdf, España – 2003.

CAPITULO 2

Prácticas de Secuencias Básicas



ESCUELA DE INGENIERIA ELECTRONICA
LABORATORIO DE AUTOMATAS PROGRAMABLES

GUIA DE PRACTICAS PARA AUTOMATA PROGRAMABLE

Práctica **1**

Tema **Secuencia manual de dos actuadores**

Condicionantes de operación de la planta:

a) El circuito de control consiste en dos pulsantes de marcha S11 y S12, y paro S01 y S02 para cada actuador, un pulsante STOP para la desconexión general, y H1 con H2 que son las luces piloto que indican el funcionamiento de actuador1 y actuador2 respectivamente.(Figura 1)
b) Pulsando S11 funciona el actuador1 solo si éste se encuentra oprimido, dejando de presionar ya no puede volver a funcionar hasta que se haya activado y apagado el actuador2 mediante su botonera.
c) Si se inicia el funcionamiento con el actuador2, se mantiene las mismas condiciones del punto anterior.

Desarrollo del panel de control con WinCC V7:

Figura 1.1: Panel de control

Fuente: Autor.- Elaborado con el Graphic Designer del WinCC V7.

- El piloto de comunicación de interfase debe de cambiar de color para indicar la activación y desactivación, respectivamente.
- El botón de salir debe de activar la salida del Runtime.
- Los eventos relacionados a los botones de control deben de mantener las condiciones dadas, como si fueran pulsantes físicos.
- Los pilotos, actuadores, deben de cambiar de color para indicar on – off.

Material a adjuntar al Proyecto

- Programa impreso correspondiente al que está corriendo en el autómata, y que está ejerciendo el control sobre la planta, acorde a las condicionantes.
- Diagrama impreso de conexiones físicas realizadas o proyectadas entre la planta física y el autómata, y de los sensores si los hubiere.
- Hoja(s) impresa(s) con las consideraciones que encontró durante el diseño y puesta en funcionamiento del automatismo:
    - Con relación a la práctica anterior, ¿se encontraron rutinas similares?.
    - ¿Qué otro tipo de información necesitó para realizar el automatismo?. Indíquela brevemente.
    - Indique si el automatismo es aplicable a la realidad, ¿requiere ajustes de software o hardware para aplicaciones específicas?

Anexo (*adjunte lo que considere necesario para el automatismo*)

ESCUELA DE INGENIERIA ELECTRONICA
LABORATORIO DE AUTOMATAS PROGRAMABLES

GUIA DE PRACTICAS PARA AUTOMATA PROGRAMABLE

Práctica **2**

Tema **Secuencia de giro de un actuador**

Condicionantes de operación:

a) El control de arranque del actuador se efectúa con "Izquierda" y "Derecha", el paro con "STOP", y el cambio de giro con "Cambio". Tiene señalización luminosa que indica el sentido de rotación.
b) Se puede iniciar con "Izquierda" o "Derecha", pero para cambiar el sentido de rotación del actuador es necesario activar "Cambio" y esperar un tiempo "t" hasta que se active el sistema. "STOP" detiene el automatismo, pero si se mantiene presionado lo bloquea.
c) Disponga de las protecciones si es el caso.

Desarrollo del panel de control en WinCC

Figura 2.1: Panel de control del automatismo.

Fuente: Autor.- Elaborado con el Graphic Designer del WinCC V7.

Material a adjuntar al Proyecto

- Programa impreso correspondiente al que esta corriendo en el autómata, y que está ejerciendo el control sobre la planta, acorde a las condicionantes.
- Diagrama impreso de conexiones físicas realizadas o proyectadas entre la planta y el autómata, y de los sensores si los hubiere.
- Hoja(s) impresa(s) con las consideraciones que encontró durante el diseño y puesta en funcionamiento del automatismo:
    - Con relación a la práctica anterior, ¿se encontraron rutinas similares?.
    - ¿Qué otro tipo de información necesitó para realizar el automatismo?. Indíquela brevemente.
    - Indique si el automatismo es aplicable a la realidad, ¿requiere ajustes de software o hardware para aplicaciones específicas?

Anexo (*adjunte lo que considere necesario para el automatismo*)

ESCUELA DE INGENIERIA ELECTRONICA
LABORATORIO DE AUTOMATAS PROGRAMABLES

GUIA DE PRACTICAS PARA AUTOMATA PROGRAMABLE

Práctica **3**

Tema **Secuencia condicionada de giro de un actuador**

Condicionantes de operación:

a) El arranque del actuador se ejecuta con los pulsantes S1-S2-S3, el paro con S4. H1 señala la rotación a la derecha, H2 la rotación hacia la izquierda. En cambio H3 indica que el sistema está listo.
b) Para arrancar el actuador hacia la derecha se debe pulsar en secuencia S1 – S2 – S3. Para arrancar el actuador hacia la izquierda se pulsará en secuencia S3 – S2 – S1.
c) Para invertir la rotación es necesario parar el actuador con S4. Cualquier cambio en la secuencia no hacen posible que el actuador arranque.
d) Disponga de las protecciones si es el caso.

Figura 3.1: Panel de control de la planta.

Fuente: Autor.- Elaborado con el Graphic Designer del WinCC

Material a adjuntar al Proyecto

- Programa impreso correspondiente al que esta corriendo en el autómata, y que está ejerciendo el control sobre la planta, acorde a las condicionantes.
- Diagrama impreso de conexiones físicas realizadas o proyectadas entre la planta y el autómata, y de los sensores si los hubiere.
- Hoja(s) impresa(s) con las consideraciones que encontró durante el diseño y puesta en funcionamiento del automatismo:
    - Con relación a la práctica anterior, ¿se encontraron rutinas similares?.
    - ¿Qué otro tipo de información necesitó para realizar el automatismo?. Indíquela brevemente.
    - Indique si el automatismo es aplicable a la realidad, ¿requiere ajustes de software o hardware para aplicaciones específicas?

Anexo (*adjunte lo que considere necesario para el automatismo*)

ESCUELA DE INGENIERIA ELECTRONICA
LABORATORIO DE AUTOMATAS PROGRAMABLES

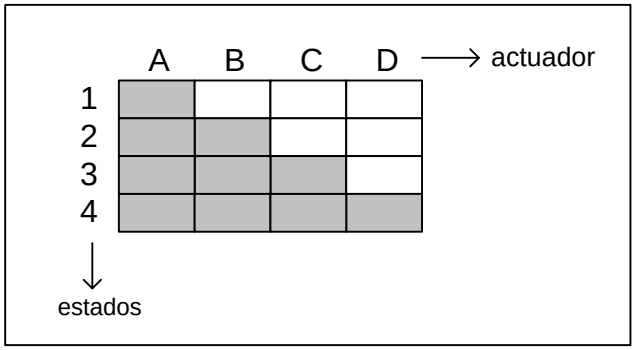
GUIA DE PRACTICAS PARA AUTOMATA PROGRAMABLE

Práctica **4**

Tema **Secuencia LIFO manual de 4 actuadores**

Condicionantes de operación:

a) Se cuenta con un botón de marcha y un botón de paro; además, un botón de paro total. Esto como controles únicos.
b) Se maneja la tabla de accionamientos:



c) Con cada pulso de "marcha" se va activando cada actuador, y con cada pulsación de "paro" se detiene cada actuador. Todo considerando que el ultimo en activar es el primero en salir.
d) "paro total" detiene todo el automatismo en cualquier momento, dejándolo listo para iniciar.
e) Disponga de las protecciones necesarias si es el caso.

Modelo de la planta en WinCC:

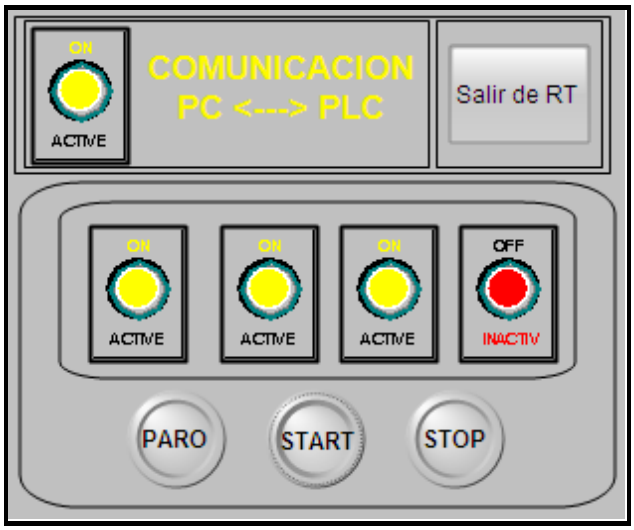


Figura 4.1: Panel de control.

Fuente: Autor.- Elaborado con el Graphic Designer del WinCC V7.

Material a adjuntar al Proyecto

- Programa impreso correspondiente al que esta corriendo en el autómata, y que está ejerciendo el control sobre la planta, acorde a las condicionantes.
- Diagrama impreso de conexiones físicas realizadas o proyectadas entre la planta y el autómata, y de los sensores si los hubiere.
- Hoja(s) impresa(s) con las consideraciones que encontró durante el diseño y puesta en funcionamiento del automatismo:
    - Con relación a la práctica anterior, ¿se encontraron rutinas similares?.
    - ¿Qué otro tipo de información necesitó para realizar el automatismo?. Indíquela brevemente.
    - Indique si el automatismo es aplicable a la realidad, ¿requiere ajustes de software o hardware para aplicaciones específicas?

Anexo (*adjunte lo que considere necesario para el automatismo*)

ESCUELA DE INGENIERIA ELECTRONICA
LABORATORIO DE AUTOMATAS PROGRAMABLES

GUIA DE PRACTICAS PARA AUTOMATA PROGRAMABLE

Práctica **5**

Tema **Secuencia FIFO de 4 actuadores neumáticos, con selección contínua de encendidos y apagados.**

Operación de la Planta

a) Se tiene un botón de "marcha", un botón de "paro", y un "paro total. Cada actuador posee un piloto que indica su estado.
b) Con el botón de "marcha" se activa cada actuador, y con el "paro" se lo detiene, considerando que el primero en activarse es el primero en desactivarse.
c) Con cada pulsación de "marcha" se activan los actuadores hasta "n-1" posición activa. Con la siguiente pulsación se activa la posición "n+1" desactivada.
d) Las imágenes que corresponde a las electroválvulas y cilindros cambian de posición al activarse, y desactivarse.

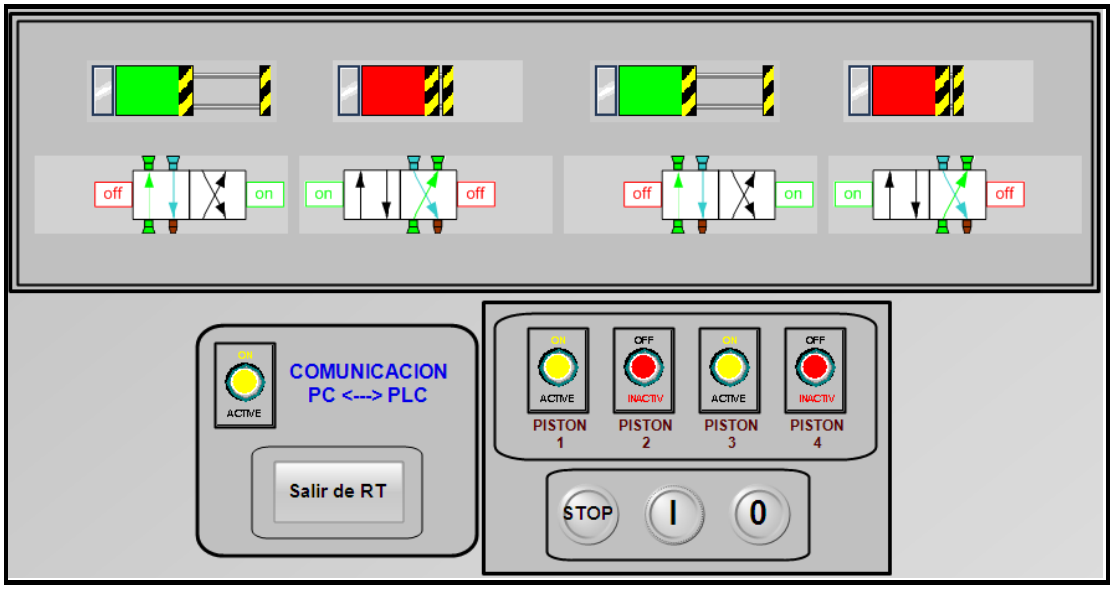


Figura 5.1: Visualización del panel de control y actuadores.

Fuente: Autor.- Elaborado con el Graphic Designer del WinCC V7.

Material a adjuntar al Proyecto

- Programa impreso correspondiente al que esta corriendo en el autómata, y que está ejerciendo el control sobre la planta, acorde a las condicionantes.
- Diagrama impreso de conexiones físicas realizadas o proyectadas entre la planta y el autómata, y de los sensores si los hubiere.
- Hoja(s) impresa(s) con las consideraciones que encontró durante el diseño y puesta en funcionamiento del automatismo:
    - Con relación a la práctica anterior, ¿se encontraron rutinas similares?.
    - ¿Qué otro tipo de información necesitó para realizar el automatismo?. Indíquela brevemente.
    - Indique si el automatismo es aplicable a la realidad, ¿requiere ajustes de software o hardware para aplicaciones específicas?

Anexo (*adjunte lo que considere necesario para el automatismo*)

Práctica **6**

Tema **Secuencia PWM con selección manual arbitraria para un actuador incandescente.**

Operación de la Planta

e) Se cuenta con 4 pulsantes, uno para cada secuencia.
f) Al pulsar, independientemente cada uno, se activa la secuencia dada. La manera de selectarlas es arbitraria.
g) Se utiliza la configuración para PWM que ofrece el PLC para implementar cada secuencia.
h) Además se tiene un piloto para la comunicación entre la PC y el PLC, y un botón para salir del RunTime.

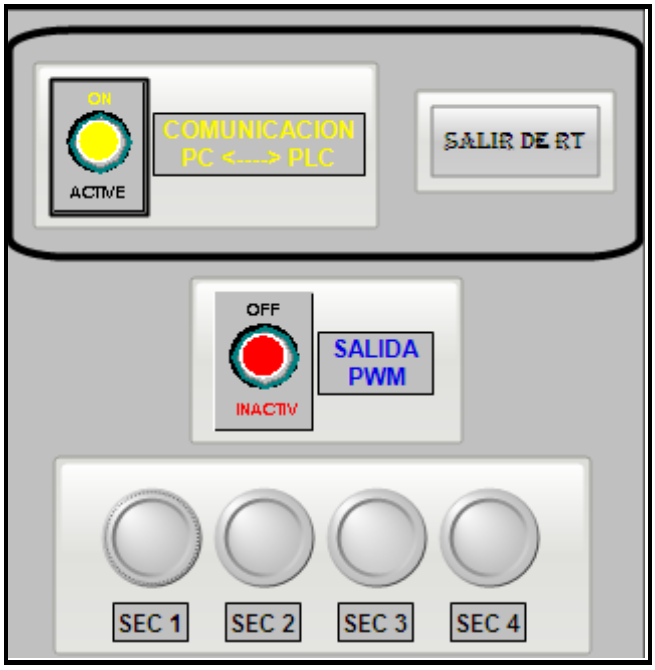


Figura 6.1: Visualización del panel de control y actuadores.

Fuente: Autor.- Elaborado con el Graphic Designer del WinCC V7.

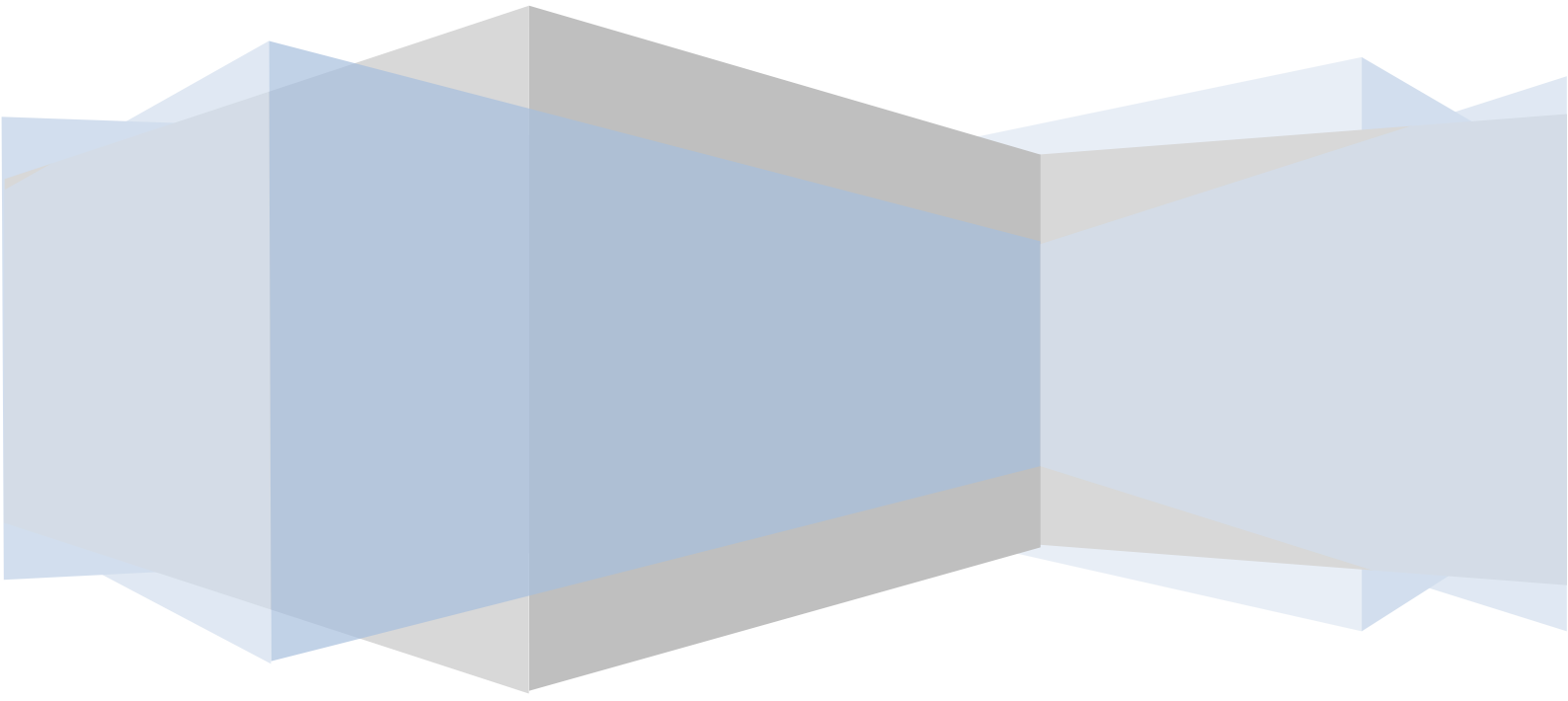
Material a adjuntar al Proyecto

- Programa impreso correspondiente al que esta corriendo en el autómata, y que está ejerciendo el control sobre la planta, acorde a las condicionantes.
- Diagrama impreso de conexiones físicas realizadas o proyectadas entre la planta y el autómata, y de los sensores si los hubiere.
- Hoja(s) impresa(s) con las consideraciones que encontró durante el diseño y puesta en funcionamiento del automatismo:
    - Con relación a la práctica anterior, ¿se encontraron rutinas similares?.
    - ¿Qué otro tipo de información necesitó para realizar el automatismo?. Indíquela brevemente.
    - Indique si el automatismo es aplicable a la realidad, ¿requiere ajustes de software o hardware para aplicaciones específicas?

Anexo (*adjunte lo que considere necesario para el automatismo*)

CAPITULO 3

Prácticas de Aplicaciones



ESCUELA DE INGENIERIA ELECTRONICA
LABORATORIO DE AUTOMATAS PROGRAMABLES

GUIA DE PRACTICAS PARA AUTOMATA PROGRAMABLE

Práctica **7**

Tema **Automatización de una escalera eléctrica.**

Condicionantes de operación:

a) Funciona en un sentido a la vez. No permitiendo, que en condiciones normales, las personas se queden a mitad del recorrido. Así:

Escalera off + [Se1 = on] ➔ *señal visual*

Escalera off + [Se2 = on] ➔ *señal visual*

b) Existe un botón de "start" y de "stop" que habilitan el funcionamiento de la planta.
c) Si "Se1" no detecta mas personas, la escalera se detiene luego de 7 segundos (tiempo total del recorrido). Pero si se sigue detectando personas, se resetea continuamente el tiempo, ocasionando un funcionamiento continuo de la escalera. Lo mismo para "Se2". Cada sensor tiene un piloto visual para la activación únicamente.
d) El motor posee un sensor térmico, que al activarse apaga al motor y bloquea al sistema, genera una alarma visual. Para volver a activar la planta, el sensor térmico debe desactivarse primero. Indicado por "pulse stop". Al hacerlo se rearma el sistema permitiendo otro ciclo de operación.

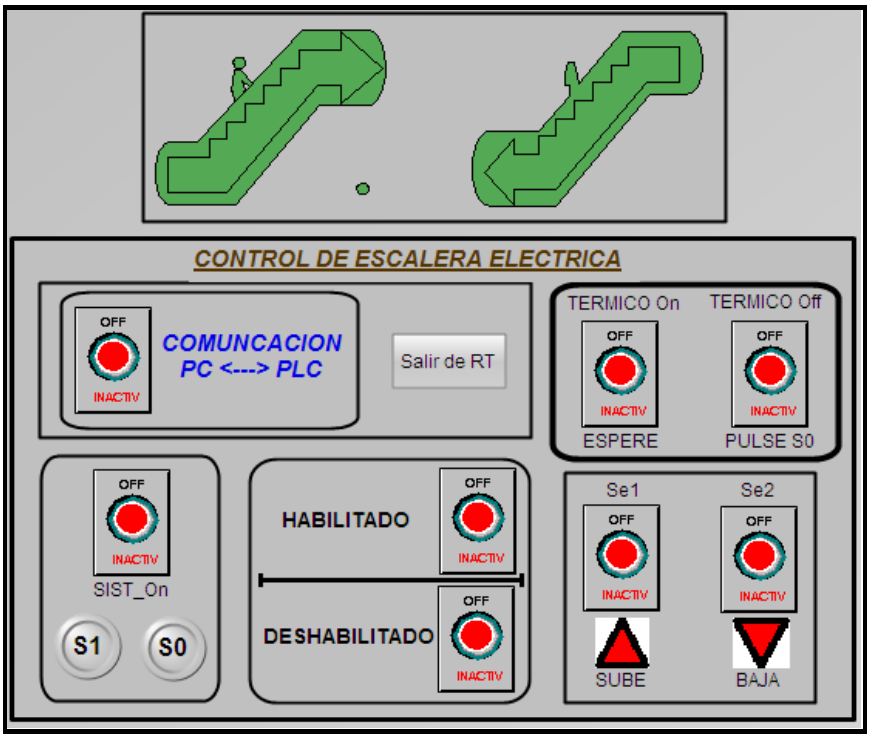


Figura 7.1: Panel de controles de la escalera.

Fuente: Autor.- Elaborado con el Graphic Designer del WinCC V7.

Material a adjuntar al Proyecto

- Programa impreso correspondiente al que esta corriendo en el autómata, y que está ejerciendo el control sobre la planta, acorde a las condicionantes.
- Diagrama impreso de conexiones físicas realizadas o proyectadas entre la planta y el autómata, y de los sensores si los hubiere.
- Hoja(s) impresa(s) con las consideraciones que encontró durante el diseño y puesta en funcionamiento del automatismo:
    - Con relación a la práctica anterior, ¿se encontraron rutinas similares?.
    - ¿Qué otro tipo de información necesitó para realizar el automatismo?. Indíquela brevemente.
    - Indique si el automatismo es aplicable a la realidad, ¿requiere ajustes de software o hardware para aplicaciones específicas?

Anexo (*adjunte lo que considere necesario para el automatismo*)

ESCUELA DE INGENIERIA ELECTRONICA
LABORATORIO DE AUTOMATAS PROGRAMABLES

GUIA DE PRACTICAS PARA AUTOMATA PROGRAMABLE

Práctica **8**

Tema **Control de un parqueadero.**

Condicionantes de operación:

1) Se tiene un habilitador de la planta, "Start" y "Stop" con un piloto. "Start" inicia la habilitación. "Stop" detiene todo, resetea el contador.
2) Se1 abre la puerta de ingreso y Se4 la cierra. Se2 abre la puerta de salida y Se3 la cierra. La capacidad máxima es de 30 autos, si no se rebasa la capacidad se indica con un piloto de "Libre". Al alcanzar el máximo se activa el piloto de "No Libre".
3) En el caso de avería de los sensores, los comandos manuales de "Entrada" y "Salida" permiten controlar cada portón respectivamente. Acorde a la selección "Man" / "Auto".
4) Los portones se controlan con motorreductores. Cada uno tiene protección de sobrecorriente.
5) Se tiene también un switch de "Emergencia" que bloquea, deshabilita y cierra los portones. Maneja una señal luminosa.

Consideraciones:

1) La salida es independiente del contador, pero la entrada depende de si existe o no espacio de parqueo.
2) En manual o automático los pulsantes de paro de las botoneras tienen la prioridad de cierre de los portones.
3) El cambio de giro es pasando por cero.

4) Entraría un carro a la vez, pero se debe disponer de una seguridad para ingreso contínuo.
5) Los semáforos para ingreso y salida indican a los vehículos que ingresen o salgan.
6) Las activaciones de las protecciones de sobrecorriente solo afectan al motor correspondiente. Y no al contador.
7) Para emergencia tiene la prioridad de cerrado de los portones, y no altera al contador.
8) El Se1 y Se2 pueden utilizar detector de flanco positivo.

Desarrollo de la planta en el WinCC:

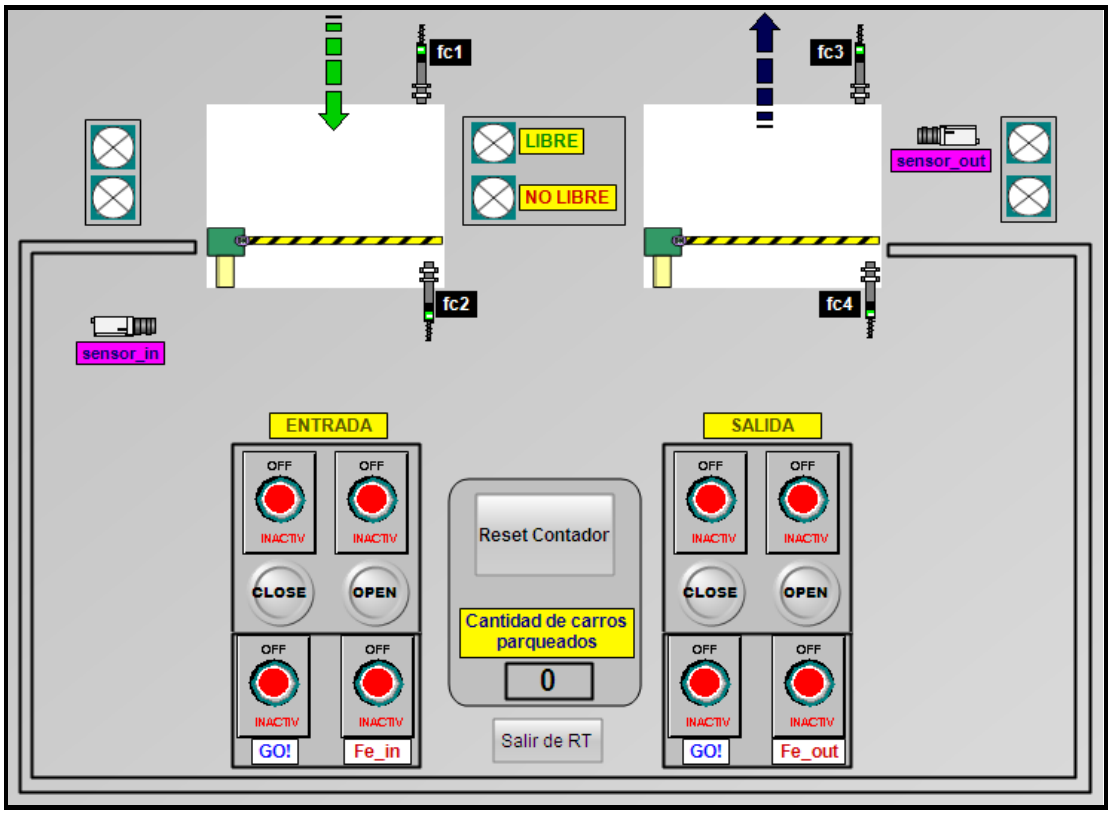


Figura 8.1: Planta a controlar.

Fuente: Autor.- Elaborado con el Graphic Designer del WinCC V7.

Material a adjuntar al Proyecto

- Programa impreso correspondiente al que está corriendo en el autómata, y que está ejerciendo el control sobre la planta, acorde a las condicionantes.
- Diagrama impreso de conexiones físicas realizadas o proyectadas entre la planta y el autómata, y de los sensores si los hubiere.
- Hoja(s) impresa(s) con las consideraciones que encontró durante el diseño y puesta en funcionamiento del automatismo:
    - Con relación a la práctica anterior, ¿se encontraron rutinas similares?.
    - ¿Qué otro tipo de información necesitó para realizar el automatismo?. Indíquela brevemente.
    - Indique si el automatismo es aplicable a la realidad, ¿requiere ajustes de software o hardware para aplicaciones específicas?

Anexo (*adjunte lo que considere necesario para el automatismo*)

ESCUELA DE INGENIERIA ELECTRONICA
LABORATORIO DE AUTOMATAS PROGRAMABLES

GUIA DE PRACTICAS PARA AUTOMATA PROGRAMABLE

Práctica **9**

Tema **Control semiautomático del transporte de paquetes.**

La planta a controlar es un sistema que transporta paquete de determinado peso hacia un lugar de recepción. El software en la PC solo sirve como visualizador del proceso. Constando de lo siguiente:

1. Los sensores de posición B1 – B2 – B3 – B4 – B5 son los encargados de indicar/detectar las posiciones del mecanismo asociado. Operan en secuencia on/off. Siendo:

    - B1 → detiene el avance del alimentador. Y provoca que el paquete caiga a la banda.
    - B2 → detiene el retroceso del alimentador.
    - B3 → activa el avance de la banda al detectar la presencia del paquete, y activa el retroceso del alimentador.
    - B4 → detiene avance de la banda. Luego el colocador se activa, trasladando el paquete a recepción.
    - B5 → detecta el paquete y manda al colocador a la posición inicial. Al desactivarse (por retiro del paquete), regresa al sistema a condiciones iniciales.

2. En el autómata se tiene los controles, S0 de paro general en cualquier momento; y, S1 de arranque del automatismo.

3. Los monitores lógicos "Sistema listo" y "Sistema operando" son los encargados de avisar el estado del proceso.
4. El "colocador" solo funciona con un pulso, de duración de 1 segundo, ya que es un sistema automático independiente del control de la planta (pero depende de ésta para operar).

Consideraciones generales:

- ♣ El proceso es secuencial, de un solo paquete a la vez.
- ♣ Considerar que los controles están centralizados en el autómata.
- ♣ Se debe de realizar el número de imágenes necesarias para representar coherentemente el movimiento de los mecanismos, sin sobrepasar el máximo permitido.
- ♣ Cada sensor debe de cambiar de color para indicar su activación.

Desarrollo de la planta en el WinCC:

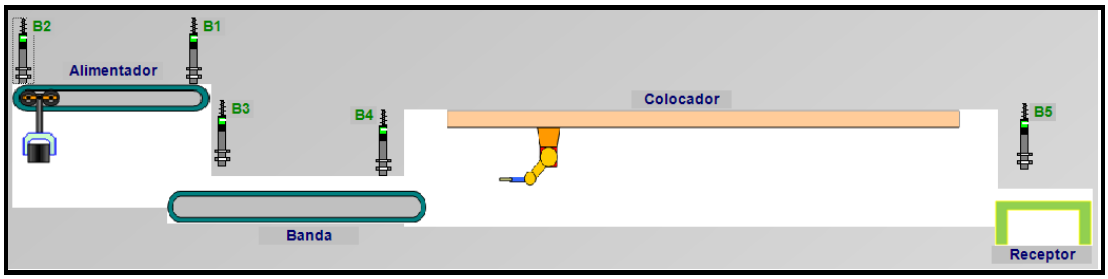


Figura 9.1: Planta a controlar.- Conformada por los cuatro procesos.

Fuente: Autor.- Elaborado con el Graphic Designer del WinCC V7.

Figura 9.2: Planta a controlar.- Conformada por el tablero de control.

Fuente: Autor.- Elaborado con el Graphic Designer del WinCC V7.

Material a adjuntar al Proyecto

- Programa impreso correspondiente al que está corriendo en el autómata, y que está ejerciendo el control sobre la planta, acorde a las condicionantes.
- Diagrama impreso de conexiones físicas realizadas o proyectadas entre la planta y el autómata, y de los sensores si los hubiere.
- Hoja(s) impresa(s) con las consideraciones que encontró durante el diseño y puesta en funcionamiento del automatismo:
    - Con relación a la práctica anterior, ¿se encontraron rutinas similares?.
    - ¿Qué otro tipo de información necesitó para realizar el automatismo?. Indíquela brevemente.
    - Indique si el automatismo es aplicable a la realidad, ¿requiere ajustes de software o hardware para aplicaciones específicas?

Anexo (*adjunte lo que considere necesario para el automatismo*)

ESCUELA DE INGENIERIA ELECTRONICA
LABORATORIO DE AUTOMATAS PROGRAMABLES

GUIA DE PRACTICAS PARA AUTOMATA PROGRAMABLE

Práctica **10**

Tema **Implementación de un semáforo de dos vías más el control peatonal, con una secuencia diurna y otra nocturna.**

En el autómata se tiene los siguientes controles:

- Se cuenta con un pulsante de arranque de la planta "start" y uno de paro total "stop". Este último detiene, en cualquier momento, la planta. Y una luz piloto que indica activación/desactivación.
- Un control por fotocelda es implementado, para que así se pueda simular la acción de operación durante el día y la noche. Posee dos pilotos, uno para el día y otro para la noche.

Si considera, Usted como técnico, necesaria la implementación de un control en el SCADA para la toma de alguna decisión sobre el automatismo, realícela sin alterar el funcionamiento de la planta.

Mientras tanto, el SCADA solo se considera como un visualizador de la operación del automatismo. Teniendo:

- Un piloto indicando la comunicación con el ordenador, si está encendido quiere decir que la comunicación está establecida.
- Un piloto indicando la activación del automatismo, si está activo, el automatismo está encendido.
- Un piloto de la activación de la fotocelda, solo indica cuando se activa, depende de la acción de la luz solar.

- Un croquis panorámico de la intersección de las avenidas, en las cuales se coloca el semáforo. La ubicación física de los semáforos peatonales.
- La indicación gráfica de la operación de la planta objeto del control. Sabiendo cuando cambia cada luminaria de los semáforos.
- Además, de un botón para "salir" del automatismo. Pero solo sirve para terminar la aplicación en el ordenador.

Operando así:

- La combinación inicial del automatismo, una vez que se ha pulsado el botón de "start", es:

  Rojo [vía 1] + Verde Peatonal [vía 1]

  Verde [vía 2] + Rojo Peatonal [vía 2]

- Luego para cambiar los colores del semáforo:
  - ☑ En la avenida en la cual están cruzando los vehículos:
    - ♣ Primero se detienen los vehículos, luego de 1500 milisegundos se permite el paso de los peatones. Este cruce dura "t" segundos. (estime el tiempo de "t").
  - ☑ En la avenida en la cual están cruzando los peatones:
    - ♣ Antes de permitir el paso de los vehículos el semáforo peatonal, que debe estar en "verde" comienza a alternar la imagen "rojo" y "verde" durante 5 veces. Para terminar en "rojo", deteniendo a los peatones. Y luego de 1500 milisegundos permitir el paso de los vehículos durante "t" segundos.

Para alternar el funcionamiento de las avenidas, se considera el sentido inverso de lo explicado. Es importante conocer lo que dice la ordenanza para el tránsito en éstos cruces.

- Entre tanto, por las noches la secuencia "on/off" es aplicada a los "Amarillos" y a los "Rojos peatonales". Controlado por una fotocelda.

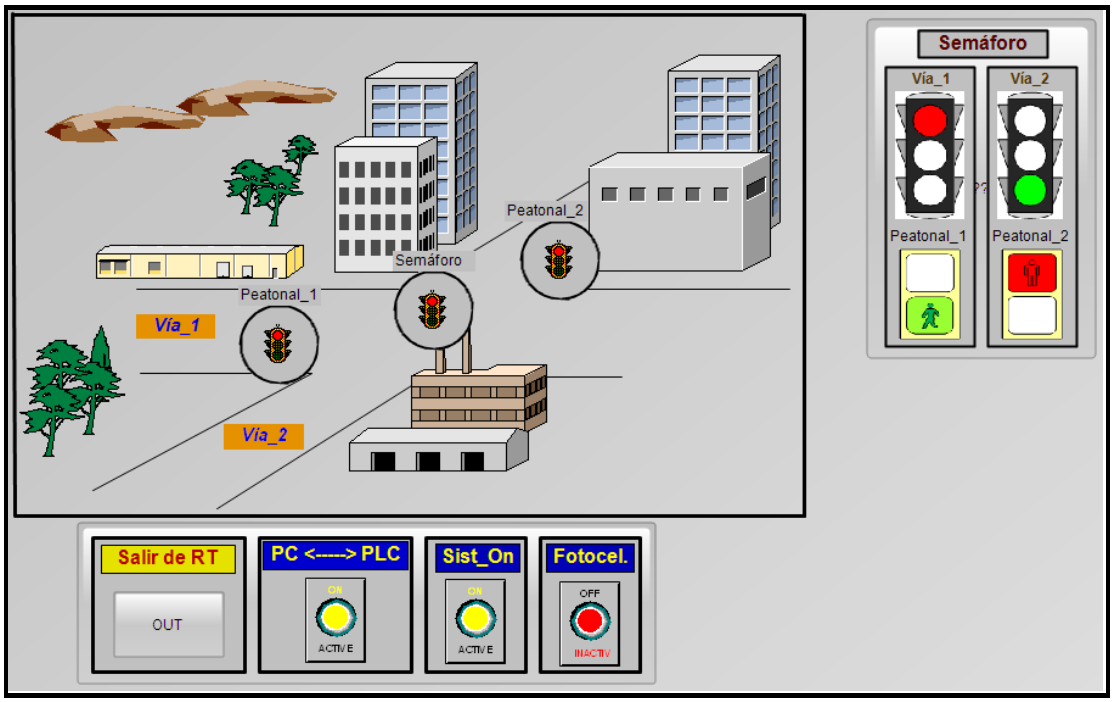


Figura 10.1: Planta que muestra al semáforo del programa.

Fuente: Autor.- Elaborado con el Graphic Designer del WinCC V7.

Material a adjuntar al Proyecto

- Programa impreso correspondiente al que está corriendo en el autómata, y que está ejerciendo el control sobre la planta, acorde a las condicionantes.
- Diagrama impreso de conexiones físicas realizadas o proyectadas entre la planta y el autómata, y de los sensores si los hubiere.
- Hoja(s) impresa(s) con las consideraciones que encontró durante el diseño y puesta en funcionamiento del automatismo:
    - Con relación a la práctica anterior, ¿se encontraron rutinas similares?.
    - ¿Qué otro tipo de información necesitó para realizar el automatismo?. Indíquela brevemente.
    - Indique si el automatismo es aplicable a la realidad, ¿requiere ajustes de software o hardware para aplicaciones específicas?

Anexo (*adjunte lo que considere necesario para el automatismo*)

ESCUELA DE INGENIERIA ELECTRONICA
LABORATORIO DE AUTOMATAS PROGRAMABLES

GUIA DE PRACTICAS PARA AUTOMATA PROGRAMABLE

Práctica **11**

Tema **Control de los cuatro motores de una máquina, de formación de piezas de madera, en secuencia fifo. Con un solo pulsante de arranque y uno de paro, en forma manual y automática.**

1) El "Selector" permite seleccionar entre un funcionamiento manual o automático. En "MAN" y en "AUTO" se tiene un pulsante de "start" y uno de "stop". Cada uno con su piloto de activación luminoso.
2) Se dispone de protecciones para cada proceso, éstas pueden ser de cualquier tipo.
3) Para conocer cual proceso se está dando en un momento determinado, se visualiza por la dinámica de cada figura.
4) Los sensores de detección "SeX" activan cada etapa de la planta en "AUTO".
5) El "SC1" permite contar las piezas de madera que provee el "Alimentador de Piezas".
6) El "SC2" cuenta las piezas de madera ya acabadas que se almacenan. Pudiendo observarse dicho valor en el cuadro adjunto al sensor.
7) El botón para terminar la aplicación en RunTime es "Salir de RT".

Secuencia de operación:

La planta posee cuatro motores. Montados en una máquina, procesan los maderos que le son alimentados. Posee dos modos de operación que pueden ser desde el autómata o desde el ordenador, así:

MANUAL: los motores se activan en una secuencia fifo. Esto es: con el botón de "start" se arranca cada motor, y con el "stop" se detiene los anteriores hasta el último que se activó.

AUTO: el botón de "start" inicia la secuencia, y "stop" la detiene. Unas ves iniciadas, los sensores de detección "SeX" son los encargados de controlar la activación y desactivación de las etapas de la planta.

Para ambos modos de operación, las protecciones de cualquier motor detienen la planta. Regresando todo al inicio, por lo que en la práctica se tiene que retirar la pieza para su descarte. Entre tanto el "alimentador de piezas" es detenido cuando se alcanza el número de piezas máximo (10) que procesa la máquina, entonces el sensor del almacén detiene la producción y se detiene el proceso con la última carga.

Las figuras de cada etapa se activan y desactivan acorde a la operación de cada una, al igual que la banda que al moverse representa el avance de la carga. La bodega del almacén es representado por un objeto tanque de recepción de piezas.

Desarrollo de la planta:

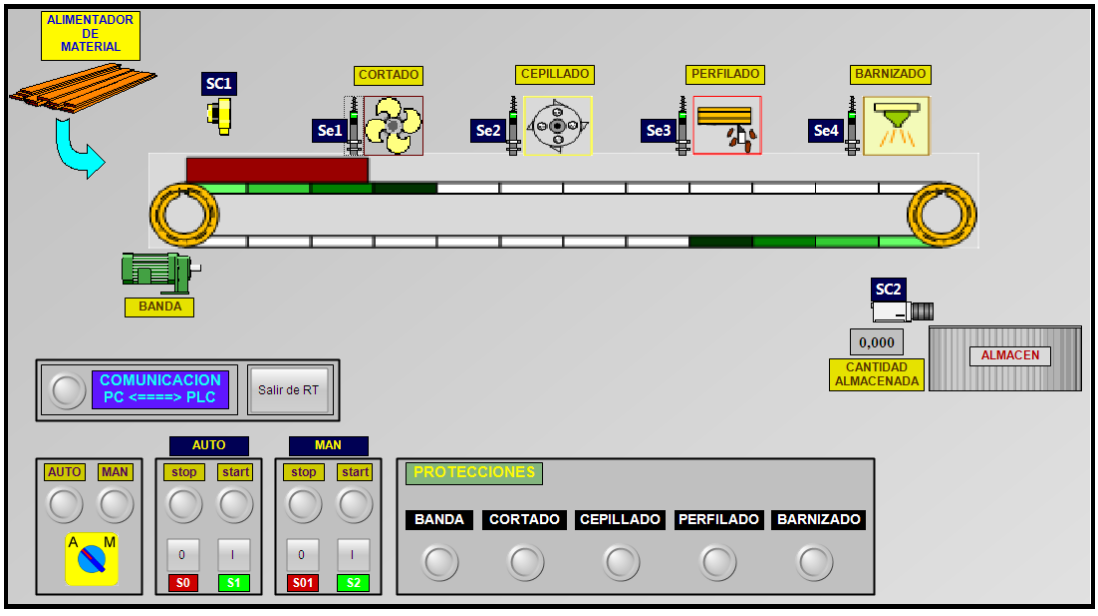


Figura 11.1: Planta a controlar.

Fuente: Autor.- Elaborado con el Graphic Designer del WinCC V7.

Material a adjuntar al Proyecto

- Programa impreso correspondiente al que está corriendo en el autómata, y que está ejerciendo el control sobre la planta, acorde a las condicionantes.
- Diagrama impreso de conexiones físicas realizadas o proyectadas entre la planta y el autómata, y de los sensores si los hubiere.
- Hoja(s) impresa(s) con las consideraciones que encontró durante el diseño y puesta en funcionamiento del automatismo:
    - Con relación a la práctica anterior, ¿se encontraron rutinas similares?.
    - ¿Qué otro tipo de información necesitó para realizar el automatismo?. Indíquela brevemente.
    - Indique si el automatismo es aplicable a la realidad, ¿requiere ajustes de software o hardware para aplicaciones específicas?

Anexo (*adjunte lo que considere necesario para el automatismo*)

ESCUELA DE INGENIERIA ELECTRONICA
LABORATORIO DE AUTOMATAS PROGRAMABLES

GUIA DE PRACTICAS PARA AUTOMATA PROGRAMABLE

Práctica **12**

Tema **Control sobre el llenado de un reservorio de agua, utilizado para proveer a una fábrica. (*El control se lo realiza exclusivamente desde la PC*)**

1) El selector de "Planta On" permite energizar el tablero de control. Luego se tiene que seleccionar la bomba que se desea emplear. Una vez hecho, los pulsantes "start" y "stop" encienden y apagan a la bomba correspondiente, en cualquier momento.
2) Para controlar el paso del agua se tiene unas válvulas que se las activa mediante los botones de "Accionamiento de válvulas". Tomar en cuenta que EV1 y EV2 son para la bomba 1, y EV3 y EV4 son para la bomba 2.
3) Para controlar los niveles del agua en el tanque se tiene un sensor de radar, que sensa "S.MAX" y "S.MIN" (*escoger los valores*). Para indicar el nivel del agua, del tanque, se lo hace en un visor. Maneja, el sensor, la condición: cuando el nivel está por debajo del "S.MIN" se activa la alarma "RSV. VACIO" en forma intermitente, y se cierra la válvula de paso "EVP". Cortándose el suministro a la fábrica.
4) Los pilotos "Fe1 y Fe2" son los encargados de mostrar las activaciones de las protecciones térmicas de cada motor de bomba. Al activarse, apagan el motor correspondiente y no se bombea más agua al reservorio. Si no se logra establecer el flujo normal, se activa la alarma de reservorio vacío y se desabastece la fábrica. En el instante en que se restaura la falla, "EVP" permite el paso a la fábrica. Trabajando todo en estado normal.
5) Así mismo, si las electroválvulas presentan fallas se las puede visualizar en las señales que están debajo de las activaciones. Estas comienzan a parpadear para indicarlo.

Desarrollo de la planta:

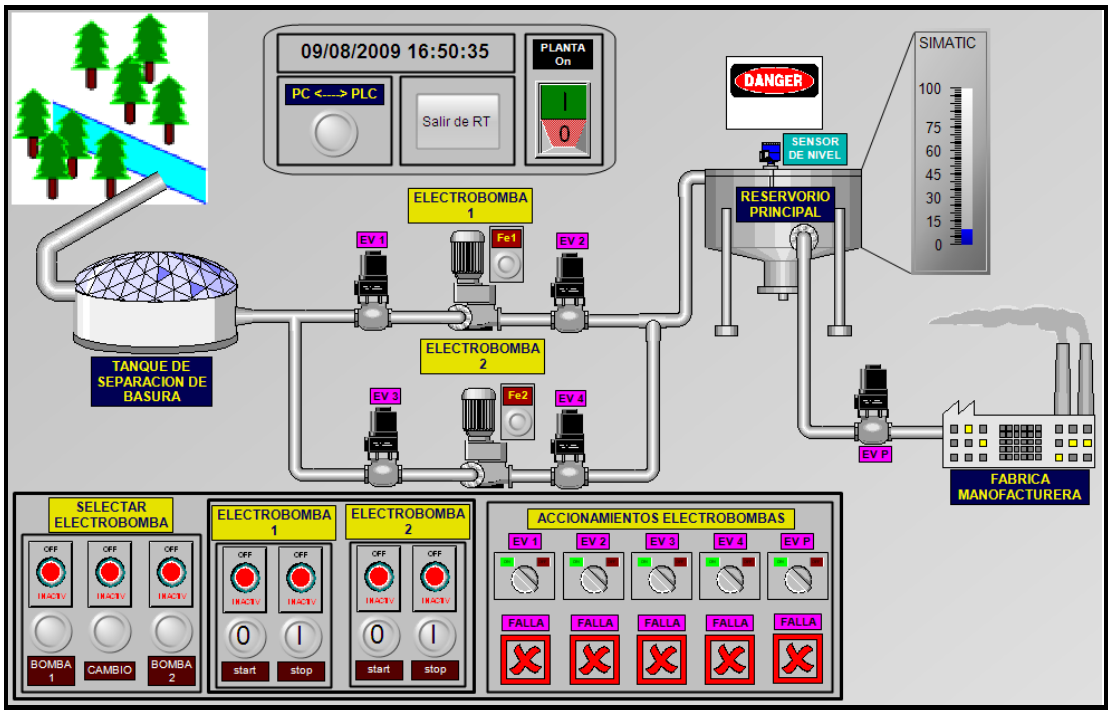


Figura 12.1: Planta del ingreso del fluído.

Fuente: Autor.- Elaborado con el Graphic Designer del WinCC V7.

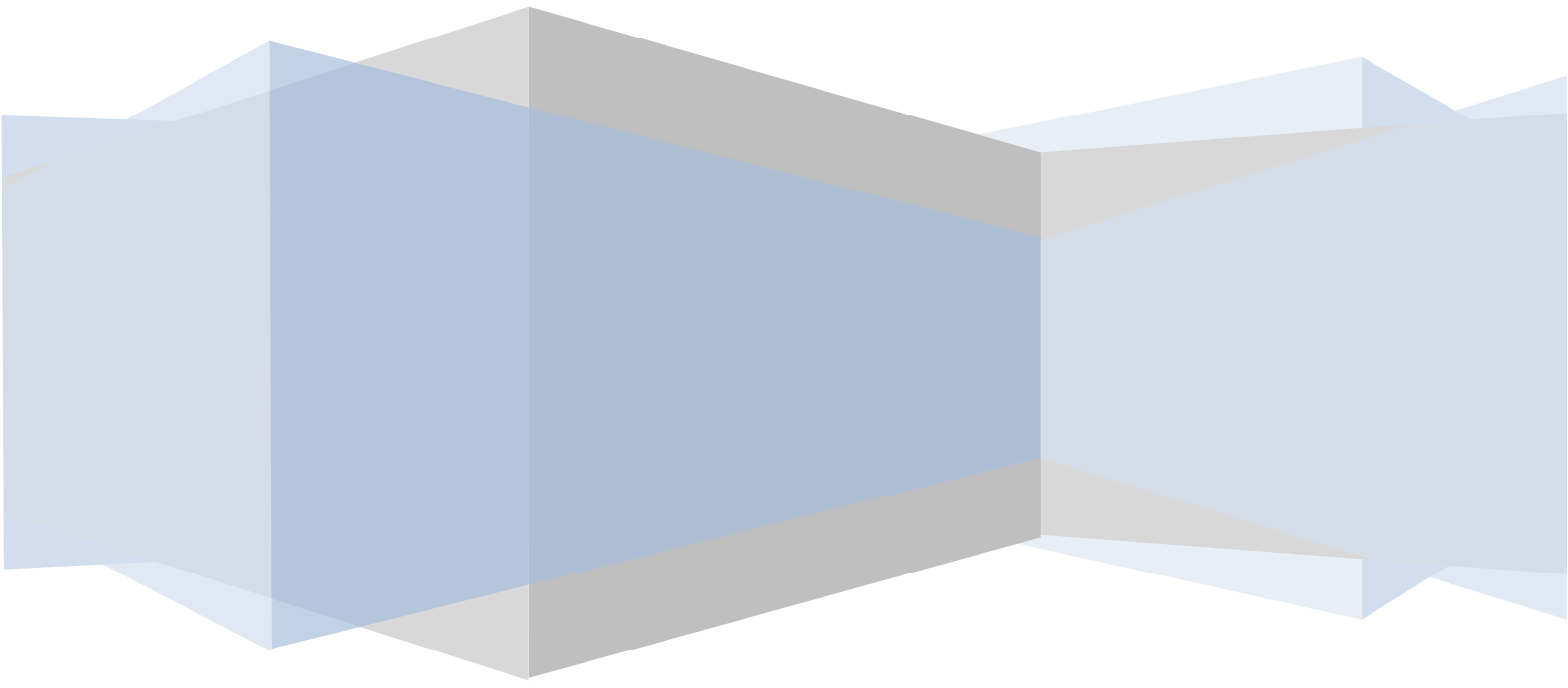
Material a adjuntar al Proyecto

- Programa impreso correspondiente al que está corriendo en el autómata, y que está ejerciendo el control sobre la planta, acorde a las condicionantes.
- Diagrama impreso de conexiones físicas realizadas o proyectadas entre la planta y el autómata, y de los sensores si los hubiere.
- Hoja(s) impresa(s) con las consideraciones que encontró durante el diseño y puesta en funcionamiento del automatismo:
    - Con relación a la práctica anterior, ¿se encontraron rutinas similares?.
    - ¿Qué otro tipo de información necesitó para realizar el automatismo?. Indíquela brevemente.
    - Indique si el automatismo es aplicable a la realidad, ¿requiere ajustes de software o hardware para aplicaciones específicas?

Anexo (*adjunte lo que considere necesario para el automatismo*)

CAPITULO 4

Proyectos



ESCUELA DE INGENIERIA ELECTRONICA
LABORATORIO DE AUTOMATAS PROGRAMABLES

GUIA DE PRACTICAS PARA AUTOMATA PROGRAMABLE

Práctica **13**

Tema **Control de un Puente Grúa.**

Se va a controlar un puente grúa que realiza el traslado de piezas desde el lugar de producción al lugar de almacenamiento. El cual tiene la siguiente construcción:

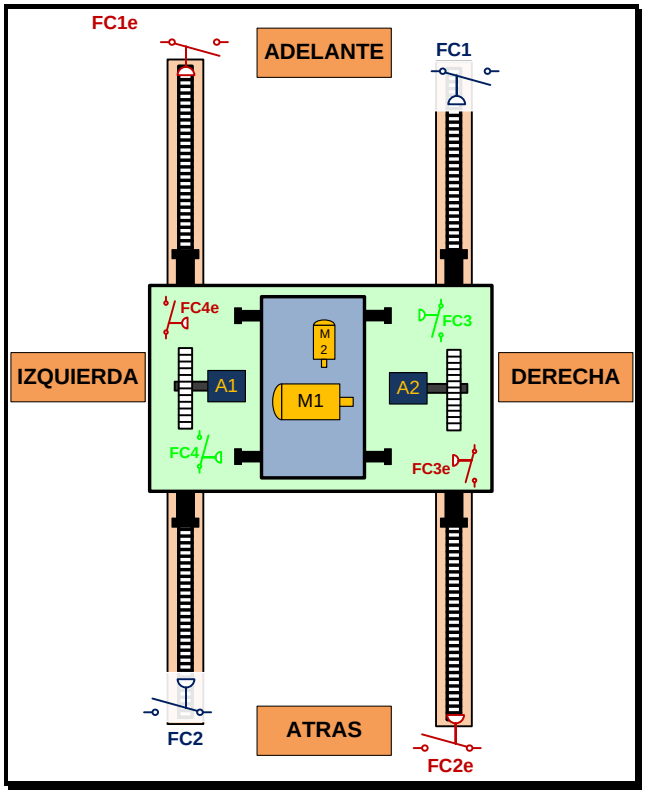


Figura 13.1: Topología constructiva del puente grúa a controlar.

Fuente: Autor.

El carro que transporta el gancho se mueve de izquierda a derecha. Siendo detenido, cada sentido de movimiento, por los fines carrera respectivos FC3 y FC4 (color verde). Cuando éstos fallan, se tiene a los FC3e y FC4e que detienen el carro del gancho (color rojo).

El carro principal que va adelante o atrás, también tiene al FC1 y FC2 para detención normal cuando llegan a activarse. Pero si fallan existen los FC1e y FC2e para la detención

El accionar de los motores es a impulsos:

- M1 es el encargado de subir o bajar el gancho. Se debe considerar que éste motor tiene acoplado en la parte posterior un sistema de frenado dinámico

para trabar el gancho en una posición determinada. Como también, un sistema de seguridad para detener el arrollamiento o desenrrollamiento del cable que sujeta el gancho (por seguridad), al llegar al máximo y mínimo permitido.

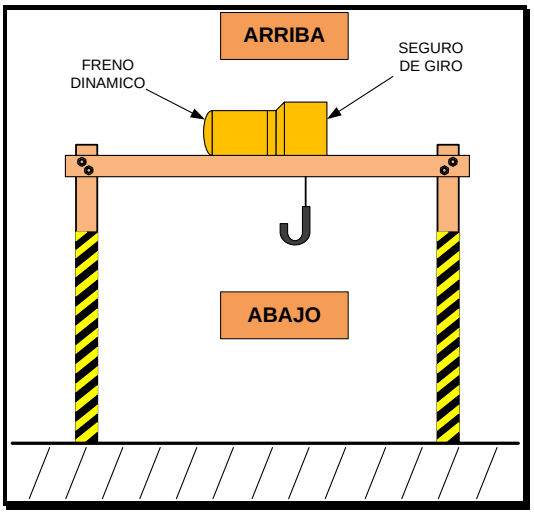
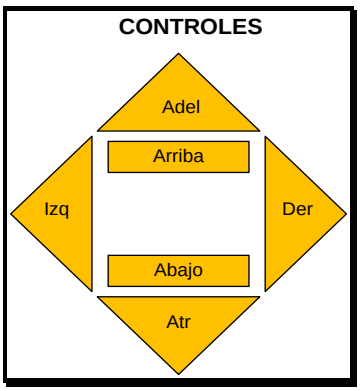


Figura 13.2: Accionar de M1.

Fuente: Autor.

- M2 es el encargado de mover el carro del gancho, hacia la izquierda o hacia la derecha.
- A1 y A2 se activan al mismo tiempo, y son los que mueven el carro principal hacia adelante o hacia atrás.

Para realizar el comando se tiene el panel:



Al darse la orden de movimiento, dando un click en el botón respectivo del SCADA o en el conectado al autómata, se activa la salida de mando del autómata y un piloto de aviso de acción.

Esto para todos los botones.

Figura13. 3: Bosquejo del arreglo de botones de control, que son físicos y virtuales.

Fuente: Autor.

Todos los motores son trifásicos de inducción tipo jaula de ardilla, y poseen cajas de engranajes, lo que los constituye en motorreductores. Por lo que el torque mecánico desarrollado en la parte del eje de servicio (salida de la caja de engranajes) es lo bastante fuerte para producir los movimientos necesarios en el puente grúa.

Dada esta característica, cada motor posee una protección térmica que al activarse (cualquiera de ellas) desactivan el puente grúa. Deteniendo los movimientos hasta que se rearme el térmico correspondiente al motor de falla.

Se pide:

- Qué tipo de autómata se debe usar para elaborar el control completo del automatismo. Explicar los criterios que debe manejar el técnico para la elección del autómata y de los equipos de apoyo al control.
- Planos eléctricos de las conexiones, y de posición arquitectónica del cuadro de control acorde a la distribución física del automatismo.
- Establecer los diagramas de flujo y/o grafcets necesarios para desarrollar el programa de control requerido.
- Desarrollar, si fuera el caso, una maqueta de aplicación; esto para la demostración práctica del automatismo.
- De que otra manera se pudiera implementar el control, explique como la haría.
- Establecer una memoria técnica con todo lo relacionado al desarrollo de ésta práctica.

Práctica **14**

Tema **Control de una caldera.**

El controlar una caldera que sirve para calentar un fluído, es un proceso en el cual se manejan múltiples variables físicas. Como son:

- Las temperaturas del combustible, del caldero en sí, del medio ambiente que rodea a la caldera, del aire que calienta el combustible (*el caso del proyecto*).
- La cantidad de combustible que se debe dar a la caldera para que llegue y mantenga una determinada temperatura.
- La cantidad del fluido que circula por la caldera, etc.

En este proyecto se hará un control sencillo de una caldera que debe de calentar un fluído líquido, el cual sirve para desarrollar una aplicación industrial. La representación de la figura 1 muestra las partes que están involucradas en este control. La operación de cada una, en el automatismo, es:

➢ *Alimentación de combustible*: es obtenido de un recipiente que lo almacena. La "válvula de combustible" es la que da el paso de éste al "intercambiador de calor" para que por medio del vapor caliente, asuma una temperatura alta. La "válvula dosificadora de combustible" mantiene un flujo contínuo regulado para mantener cierta temperatura en la caldera. Mientras que el "sensor de temperatura del combustible" muestra el valor de la temperatura del combustible. La presión constante es mantenida por la "bomba de combustible". Entonces, al ingresar el combustible caliente a la caldera,

permite que la combustión sea más rápida y por ende la caldera alcance un cierto grado de eficiencia al operar.

➢*Calentamiento del combustible*: el "compresor de aire" inyecta aire que al pasar por un calentador eleva su temperatura a cierto valor, que al circular por el "intercambiador de calor" produce que el combustible se caliente. Pero luego tiene que ser evacuado, y como tiende a condensarse entra en un sistema separador. Y es aquí que el líquido es desfogado por una válvula, y el gas es expulsado por medio de una chimenea.

➢*Actuación de la caldera*: ingresa fluído frío y sale fluído caliente, que es monitoreado por un sensor de lectura, que visualiza el valor. La bomba de fluído es la encargada de enviarlo.

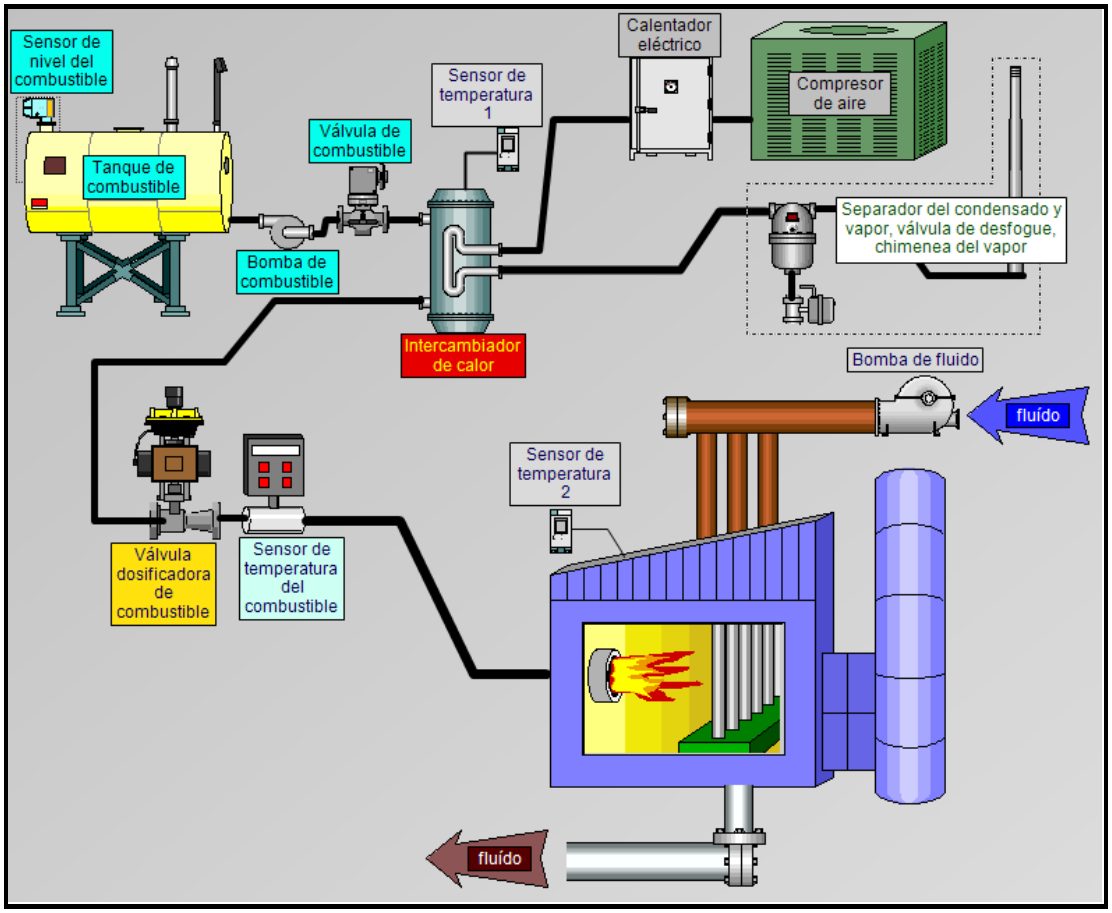


Figura 14.1: Partes que constituyen al automatismo de actuación de una caldera sencilla.

Fuente: Autor.- Elaborado en el Graphic Designer del WinCC.

Claramente se observa que existen tres procesos definidos, lo que facilita el elaborar un control.

Condiciones de operación

En el cuadro de control:

- Un sistema de control inicial para el compresor: la pulsar "marcha" se prende el compresor y el calentador, se comienza la inyección de aire al calentador y al intercambiador. Si se pulsa "paro" se apaga el compresor y el calentador.
- El "Sensor de temperatura 1" al detectar cierto valor de temperatura, da la orden de activación a la "válvula de combustible" y a la "bomba de combustible". Estos se activan si y solo si existe un nivel mínimo de combustible en el tanque (valores escogidos por el técnico).
- El "sensor de temperatura 2" al inicio deja pasar todo el combustible a la caldera, la cual cuenta con sistema de ignición al detectar éste. Pero como la temperatura está en frío, el sensor abre toda la "válvula dosificadora". A medida que la temperatura de la caldera empieza a subir, éste sensor comienza a regular el paso de combustible en forma inversamente proporcional entre el caudal de combustible y la temperatura de la caldera.
- Entre tanto, el paso del fluído líquido por la caldera es contínuo desde el momento de encendido de la caldera.
- Se cuenta con un apagado general de emergencia por medio de un pulsante seta (con enclavamiento mecánico). Además, se produciría un apagado total sí:
    - El nivel combustible está por debajo de un mínimo permitido.
    - La temperatura del combustible cae por debajo de cierto valor establecido en el sensor.
    - El fluído de ingreso a la caldera se detiene.
    - Si el compresor se apaga.

En el SCADA a realizar:

- Se deben visualizar los valores medidos de cada sensor.
- Establecer una dinámica adecuada de los elementos dibujados para indicar el on/off de cada uno.
- El cuadro de mando debe ser totalmente claro y fácil de maniobrar con el mouse.

Se pide:

- Qué tipo y cuantos autómatas se debe usar para elaborar el control completo de la planta. Explicar los criterios que debe manejar el técnico para la elección del autómata y de los equipos de apoyo al control.
- Planos eléctricos de las conexiones, y de posición arquitectónica del cuadro de control acorde a la distribución física de la planta.
- Establecer los diagramas de flujo y/o grafcets necesarios para desarrollar el programa de control requerido.
- Desarrollar, si fuera el caso, una maqueta de aplicación; esto para la demostración práctica del automatismo.
- De que otra manera se pudiera implementar el control, explique como la haría.
- Indicar que tipo de protecciones se tendría que implementar en la realidad, ¿es aplicable para una caldera real?
- Establecer la memoria técnica adecuada para mostrar el desarrollo del automatismo.

Práctica **15**

Tema **Proceso químico de piezas metálicas.**

En éste proceso se debe tratar químicamente a piezas metálicas, que son necesarias para la producción de ciertos electrodomésticos. Los químicos que se agregan a cada tanque son preparados en otros reservorios, y son colocados mediante bombeo independiente en cada tanque.

La planta a controlarse es:

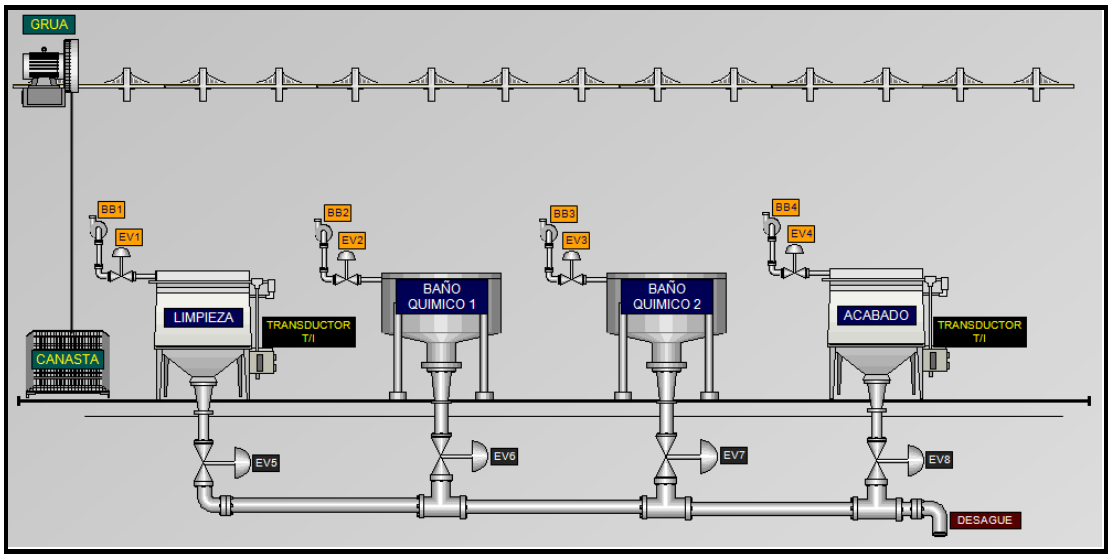


Figura 15.1: Disposición de la Planta a controlar.

Fuente: Autor.- Elaborado en el Graphic Designer del WinCC.

El funcionamiento de cada parte del automatismo es el siguiente:

Grúa: está constituido por un motor de inducción trifásico, con freno dinámico, reductor mecánico por engranajes. En la parte del eje del reductor tiene acoplado un

mecanismo para albergar un cable de acero que está unido a una canasta de transporte. Se desliza sobre un sistema de rieles.

Tanque de limpieza: contiene el primer químico. Se calienta por medio de niquelinas que están sumergidas. La temperatura, que va desde los 30ºC hasta los 50ºC, se controla por medio de termopares sumergidos, acoplados a un transductor T/V (*de temperatura a voltaje*) graduado de "0 a 5 V". El nivel del líquido es controlado por un sistema de vaso comunicante, en el cual se halla un sensor de nivel para "nivel mínimo" y "nivel máximo".

Baño químico 1: contiene el segundo químico. Está a temperatura ambiente, que para efectos del proceso se considera frío. El nivel del líquido es controlado por un sistema de vaso comunicante, en el cual se halla un sensor de nivel para "nivel mínimo" y "nivel máximo".

Baño químico 2: contiene el tercer químico. Está a temperatura ambiente, que para efectos de proceso se considera frío. El nivel del líquido es controlado por un sistema de vaso comunicante, en el cual se halla un sensor de nivel para "nivel mínimo" y "nivel máximo".

Acabado: contiene el último químico. Se calienta por medio de niquelinas que están sumergidas. La temperatura, que va desde los 65ºC hasta los 95ºC, se controla por medio de termopares sumergidos, acoplados a un transductor T/I (*de temperatura a corriente*) graduado de "4 a 20 mA". El nivel del líquido es controlado por un sistema de vaso comunicante, en el cual se halla un sensor de nivel para "nivel mínimo" y "nivel máximo".

Consideraciones generales:

- Los controles de nivel de cada tanque son independientes para cada uno.
- El fluído de cada tanque es bombeado, de los de preparación, por medio de bombas "BB1 a BB4"; el paso es permitido por las "EV1 a EV4."
- Se procesa una sola carga de la canasta por ciclo.

Condiciones de "paro" de la planta:

- Por la detección de un nivel inferior al "mínimo" permitido de cualquier tanque.
- Por la activación de la protección térmica del motor de la grúa.
- Por la activación del pulsante seta, de emergencia. Produciéndose:
  - Activación de una alarma sonora y visual.
  - La posibilidad de vaciar los tanques.
  - La posibilidad de regresar la canasta al punto de inicio.
- Al cumplirse 10 ciclos completos de operación de la planta.

Condiciones de marcha:

- La planta puede operar en "manual" o "automático".
- En manual solo se controla el avance de la canasta hacia cada tanque.
- En automático todo el proceso es automático desde el inicio, con solo la pulsación del botón de marcha.

El técnico deberá:

1. Establecer una consola de control que será ubicada al lado de la planta para ser controlada por el operario. Esta deberá tener todos los botones respectivos y pilotos necesarios.
2. Implementar un control SCADA adecuado para poder también controlar desde una PC la planta.
3. Establecer aquello que estime conveniente para el buen funcionamiento del automatismo.

Operación de la planta

Una vez que se haya preparado los químicos en los tanques primarios respectivos, se activan las bombas de cada tanque al mismo tiempo y empiezan trasladarse los fluídos hasta los tanques de proceso. Los sensores de cada tanque detienen los llenados al alcanzarse los niveles máximos permitidos.

Luego que se llenaron todos los tanques empieza el proceso con la selección del proceso, manual o automático. En general, la secuencia es:

I. La canasta va hacia la limpieza, en donde se retiraran los óxidos y polvo.
II. Luego va al tanque de "baño químico 1", en donde se coloca las piezas para que se adhiera éste químico.
III. En el siguiente se adiciona a las piezas una capa química que reacciona con la anterior para crear una superficie resistente a la corrosión.
IV. Por último, este acabado es para dotar de cierto color característico a las piezas metálicas.
V. El proceso en conjunto es repetido un número de ciclos completos, en condiciones normales.
VI. Luego de éstos ciclos se detiene, paraliza, para el cambio de fluídos químicos. Para luego comenzar de nuevo.

Se pide:

- Qué tipo y cuantos autómatas se debe usar para elaborar el control completo de la planta. Explicar los criterios que debe manejar el técnico para la elección del autómata y de los equipos de apoyo al control.
- Planos eléctricos de las conexiones, y de posición arquitectónica del cuadro de control acorde a la distribución física de la planta.
- Establecer los diagramas de flujo y/o grafcets necesarios para desarrollar el programa de control requerido.

- Desarrollar, si fuera el caso, una maqueta de aplicación; esto para la demostración práctica del automatismo.
- De que otra manera se pudiera implementar el control, explique como la haría.
- Indicar que tipo de protecciones se tendría que implementar en la realidad.
- Establecer un estudio de los elementos necesarios para implementarlo en la práctica, ¿Cómo lo haría?

ESCUELA DE INGENIERIA ELECTRONICA
LABORATORIO DE AUTOMATAS PROGRAMABLES

GUIA DE PRACTICAS PARA AUTOMATA PROGRAMABLE

Práctica **16**

Tema **Autolavado de vehículos.**

El control es ejercido en una planta que consiste en una banda que transporta vehículos para ser lavados. Los pasos son definidos en cuatro:

1) MOJADO: el vehículo es sometido a unos chorros de agua que lo mojan completamente por todos los ángulos.
2) ENJABONADO: en el vehículo se coloca detergente en forma líquida, por medio de chorros que salen de las tomas de la base metálica.
3) CEPILLADO: en base a unos cepillos grandes que cuelgan de una base metálica, se cepilla al vehículo por la parte superior.
4) SECADO: unas corrientes de aire se encargan de secar la superficie del vehículo.

Cada proceso es activado y desactivado por los sensores S1 a S5. Los cuales tienen el siguiente funcionamiento:

- Las condiciones iniciales son barrera arriba para permitir el paso de los vehículos, y el semáforo en verde.
- Al activarse el S1 se acciona el mecanismo que permite el paso del fluido a la estructura y se moja el vehículo. Pero al activarse el S2, y, si S1 no detecta otro vehículo, se desactiva el mojado.
- S2 al detectar activa el mecanismo del fluido jabonoso hacia el vehículo. Pero a activarse S3, y, si S2 no detecta otro vehículo, se desactiva el enjabonado.

- Al activarse S3 se pone en marcha los cepillos que actúan sobre el vehículo. Pero al activarse S4, y, si S3 no detecta otro vehículo, se desactiva el cepillado.
- S4 se encarga de activar el sistema de secado. Fluidos de aire caliente que secan al vehículo. El S5 apaga el secado siempre y cuando el S4 no detecte otro vehículo.
- Si están en la cinta transportadora cuatro vehículos, la barrera impide el paso de otro vehículo y el semáforo se pone en rojo. Y cuando esté ya vacía, se puede habilitar el ingreso de otros vehículos.

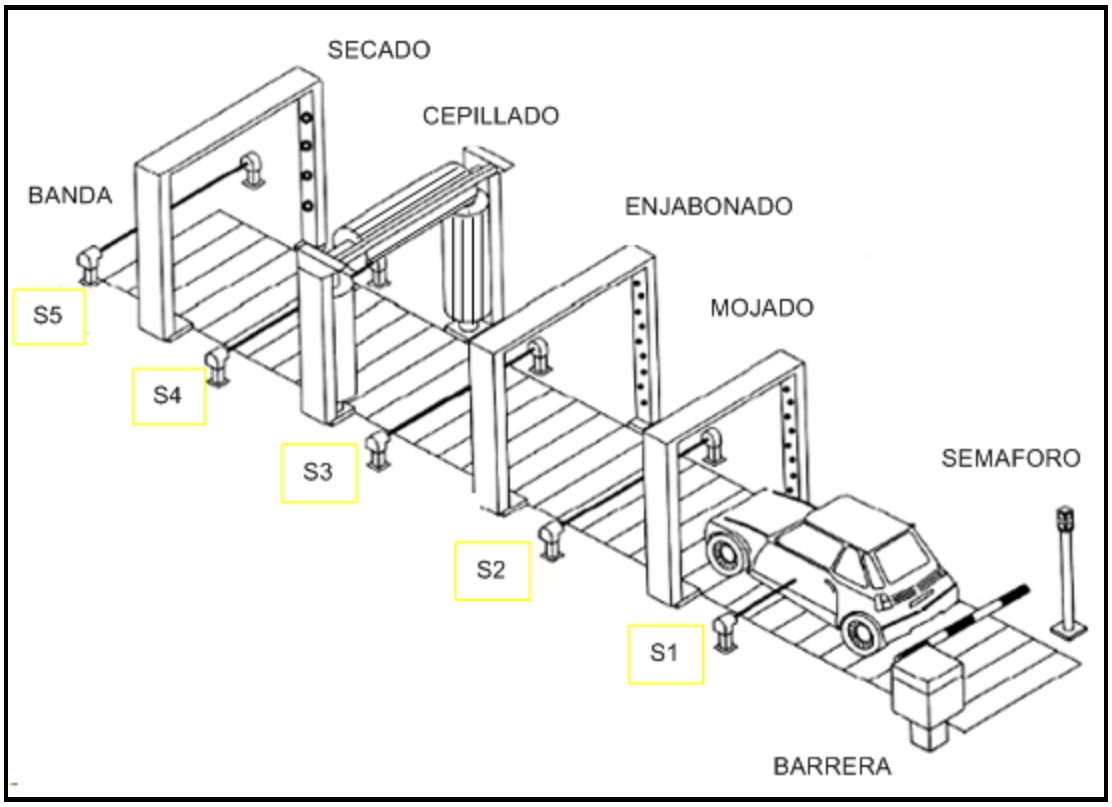


Figura 16.1: Planta a controlar. Se muestra los procesos que se debe controlar.

Fuente: García J – González J, El Autómata Programable CPM1A – Problemas, Universidad de Oviedo, España – 2001.

Se debe considerar:

- ☑ Si el motorreductor de la banda sufre un elevado consumo de corriente, se debe bloquear y desconectar toda la planta.

- ☑ Cada proceso cuenta con un motor de inducción trifásico de anillos rosantes, que sirve de impulsor para los fluídos. Si se dispara la protección térmica de éstos, se detiene el proceso en cuestión. La cinta transportadora también se detiene, y si después un tiempo determinado no se soluciona la falla, se detiene toda la planta.

- ☑ Por cada sensor se debe contar con un comando manual, para que en caso de falla del sensor se pueda efectuar un control manual de los procesos.

- ☑ Para una situación de emergencia se tiene un pulsante seta, es el encargado de apagar toda la planta, todos los procesos. Y además los bloquea y activa un sistema de señalización óptica – auditiva.

Se pide:

- Qué tipo y cuantos autómatas se debe usar para elaborar el control completo de la planta. Explicar los criterios que debe manejar el técnico para la elección del autómata y de los equipos de apoyo al control.
- Planos eléctricos de las conexiones, y de posición arquitectónica del cuadro de control acorde a la distribución física de la planta.
- Establecer los diagramas de flujo y/o grafcets necesarios para desarrollar el programa de control requerido.
- Desarrollar, si fuera el caso, una maqueta de aplicación; esto para la demostración práctica del automatismo.
- De que otra manera se pudiera implementar el control, explique como la haría.
- Indicar que tipo de protecciones se tendría que implementar en la realidad.
- Elaborar el informe respectivo.

ESCUELA DE INGENIERIA ELECTRONICA
LABORATORIO DE AUTOMATAS PROGRAMABLES

GUIA DE PRACTICAS PARA AUTOMATA PROGRAMABLE

Práctica **17**

Tema **Ascensor de 3 pisos.**

NOTA: Esta práctica está basada en la de "Máquinas Secuenciales", del Instituto Tecnológico de Costa Rica – Departamento de Electrónica, I Semestre 2004.

El equipo que se tiene que controlar es un ascensor que sirve de transporte en una casa de huéspedes. Este se mueve en tres pisos. Cada piso se tiene 4 departamentos, que pueden albergar 5 personas, la distancia entre el piso y el techo de cada departamento es de 3 m.

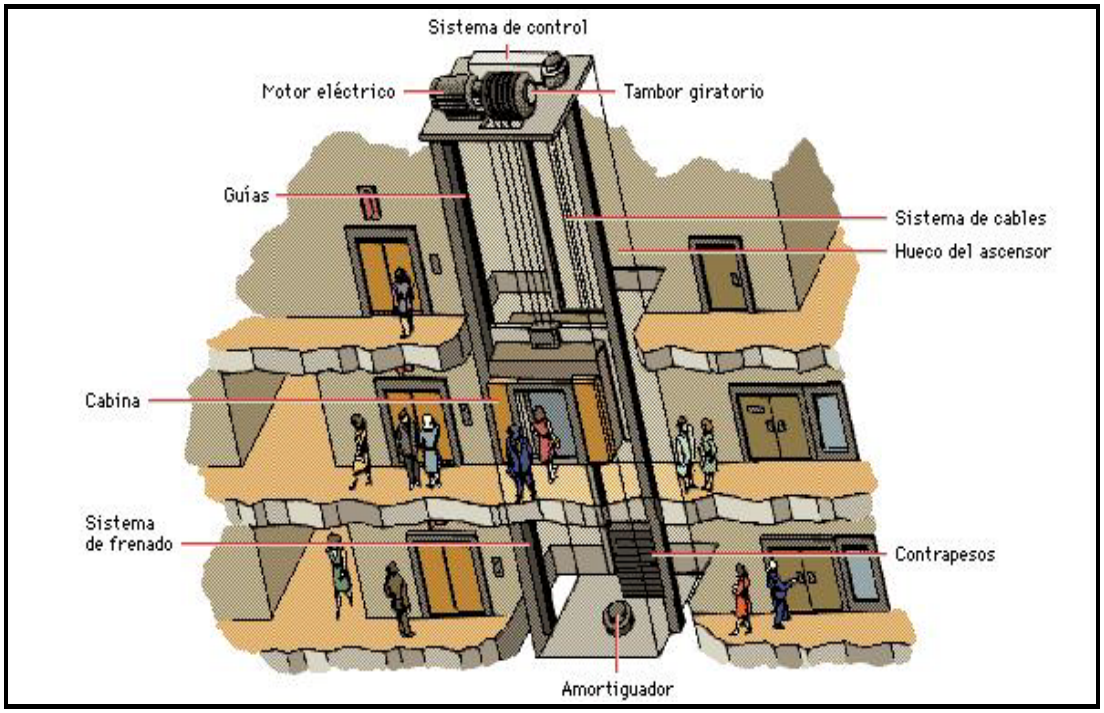


Figura 17.1: Esquemático de un ascensor de 3 pisos.

Fuente: Fundación Renault – Instituto Técnico, Trabajo final de Máquinas y Elementos de Transporte, Córdova – Argentina, 2004.

Considerar:

- Que el número máximo de personas que puede soportar el ascensor es de 6, un peso aproximado de 700lb.
- El peso de la cabina es de 250 lb.
- El motor de accionamiento del ascensor baja la velocidad al llegar al piso que se llamó.
- El movimiento de la cabina lo realiza, verticalmente, un motor de inducción trifásico acoplado a un mecanismo de freno dinámico, y de tracción.

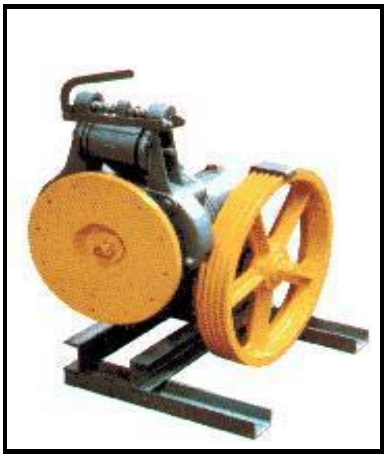


Figura17. 2: Motor de tracción de cabina.

Fuente: Fundación Renault – Instituto Técnico, Trabajo final de Máquinas y Elementos de Transporte, Córdova – Argentina, 2004.

Dicha cabina se mueve sobre unos rieles laterales.

- El peso de la cabina es controlado por un sensor de tensión, que está instalado en los cables que sujetan a la cabina. Este sensor está basado en un circuito puente de galgas extensiométricas. Al detectar la deformación de la estructura por acción del peso, el circuito electrónico asociado emite una señal de voltaje. Esta señal es proporcional al peso detectado.

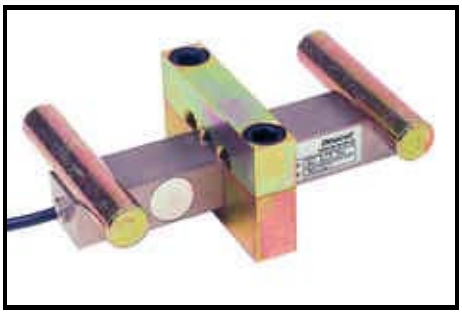


Figura 17.3: Sensor de tracción, accionado por la suspensión de una carga, que se coloca en los cables que sujetan a un objeto suspendido.

Fuente: García A – Sanclemente A, Equipos de Pesaje en Aparatos Elevadores, Universidad de Zaragoza, Depto. de Ingeniería de Diseño y Fabricación, España – 2007.

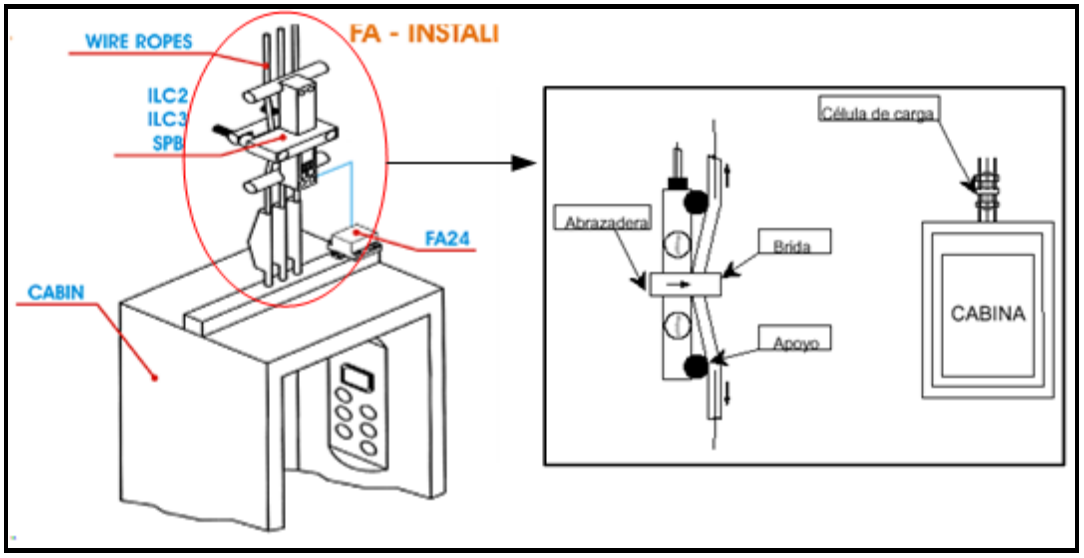


Figura 17.4: Sensor de tracción: Forma de montaje del sensor de tracción. Visualizar que la tracción de los cables tienden a deformar la celda, lo que origina una señal variable en el tiempo.

Fuente: García A – Sanclemente A, Equipos de Pesaje en Aparatos Elevadores, Universidad de Zaragoza, Depto. de Ingeniería de Diseño y Fabricación, España – 2007.

Si se ha sobrepasado el límite permitido, se bloquea el ascensor en el piso que se encuentre. Se emite una alarma sonora y visual indicando a las personas que se hallan en el interior de la cabina sobre la falla. Manteniendo las puertas abiertas y paralizada la cabina.

- La puerta de la cabina es controlada por un motor monofásico acoplado a un motorreductor, que la abre y la cierra. Al estar cerrando, si se detecta un bloqueo se vuelve a abrir, y se mantiene abierta hasta que se pulse nuevamente el cerrado de la puerta. Ahora, si se está cerrando y se detecta que existe un ingreso a la cabina; la puerta detiene su cerrado y se abre. Manteniéndose abierta hasta pulsar el botón de cerrado.

- Al interior de la cabina se tiene un tablero de control con botones. Constando con un botón de accionamiento para indicar el piso de llegada, uno por cada piso. Con un botón para activar una alarma en caso de emergencia, alarma sonora. Un botón para dar la orden de apertura y otro para el cierre de la puerta. Un sistema visual para indicar el piso en que se halla o pasa la cabina.

- En la parte de afuera, en cada piso se tiene un botón de llamada al ascensor. Al mismo tiempo, se cuenta con puertas de protección. Estas funcionan casi al mismo tiempo que las de cabina. Al llegar la cabina al piso correspondiente, primero se abre la puerta de cabina y luego de unos milisegundos se abre la puerta de piso. Al cerrarse, primero se cierra la puerta de piso, y luego de unos milisegundos se cierra la puerta del ascensor. Esto por seguridad.

  Además, se tiene un display que indica el piso de tránsito de la cabina.

- Si durante el tránsito de la cabina por el túnel de ascensor se suscita un corte de electricidad, se activa un sistema de frenado que la detiene.

- El sistema de control debe responder a la primera orden que detecte, luego tendrá que ir evacuando los pedidos conforme se vayan registrando. Pero al no recibir ninguna orden de traslado, debe de permanecer en el último piso que llegue, con las puertas cerradas (de piso y de ascensor).

Entonces, el estado de reposo del sistema es el piso 1 con la puerta del piso cerrada y la de la cabina. El sistema se encuentra en este estado al inicio de la operación (después de un master reset), o una vez que el sistema quede ocioso.

El criterio lógico en el cual se basa el funcionamiento del ascensor, es el de *"discriminación de subida y bajada"*, el cual consiste en que si el ascensor está subiendo, este dará prioridad a las instrucciones destinadas a subir desde un cierto piso o desde la misma cabina, mientras que si el ascensor está bajando, se dará prioridad a las instrucciones destinadas a bajar desde un cierto piso o desde la cabina. Por ejemplo: si el ascensor va subiendo del primero al cuarto piso y se recibe una solicitud de bajar del tercer piso, se completa el servicio al cuarto piso y luego se ejecuta el servicio del tercer piso. Si no hay más solicitudes se va a reposo.

El sistema cuenta con dos modos de solicitud de servicio:

a. ***Solicitud de servicio desde un piso.*** La solicitud de servicio desde un piso se da cuando algún usuario ubicado en alguno de los cinco pisos solicita el servicio del ascensor, ya sea para subir o bajar si se encuentra en los pisos 2 o 3, o solo para subir si se encuentra en el piso 1, y solo para bajar si se encuentra en el piso 3.
b. ***Solicitud de servicio desde la cabina.*** La solicitud de servicio desde la cabina se da cuando el usuario que se encuentra dentro de ella selecciona el piso al cual desea ser transportado.

Independientemente del servicio que se realice debe operar bajo las siguientes consideraciones:

CARGA DE PASAJEROS

La carga de pasajeros se realiza en cualquier piso y se debe cumplir con las siguientes consideraciones:

1. El sensor de puerta de piso debe indicar que la puerta está abierta.
2. El sensor de la puerta de la cabina debe indicar que está abierta.
3. El sensor de llegada de la cabina activado.
4. Durante este proceso la señal del PISO X debe estar encendida.
5. Se debe verificar el estado del peso. Si este sensor indica que hay sobrepeso, se debe encender la señal interna de la cabina que indica que hay PESO MAXIMO, y debe permanecer en esta condición hasta que el sensor de peso se libere.

OPERACIÓN HACIA ARRIBA

Una vez cargada la cabina, se debe esperar por la solicitud de servicio desde la cabina. Aquí los usuarios deben seleccionar el piso al cual se debe realizar el transporte. Inmediatamente después se deben realizar las siguientes funciones:

1. Encender en el panel de la cabina la o los pisos solicitados.
2. Consultar el estado de la puerta.
3. Cerrar la puerta del piso,
4. Cerrar la puerta de la cabina.
5. En caso de que el sensor de puerta cerrada no se active, se debe encender el piloto de cerrar puerta (en el panel de control de la cabina) para que el usuario realice la operación de cerrar la puerta de modo manual.
6. Revisar si en alguno de los pisos hay solicitud de servicio hacia arriba.
7. Activar la señal SUBIR.
8. Revisar la señal de LLEGADA DE LA CABINA, en cada uno de los pisos para visualizar la señal de piso que se encuentra dentro de la cabina.

Si la cabina se encuentra en el estado de reposo permanecer hasta que se presente una solicitud de servicio de algún piso, en este caso se deberán realizar las funciones 5, 6 y 7.

OPERACIÓN HACIA ABAJO

Una vez cargada la cabina, se debe esperar por la solicitud de servicio desde la cabina. Aquí los usuarios deben seleccionar el piso al cual se debe realizar el transporte. Inmediatamente después se deben realizar las siguientes funciones:

1. Encender en el panel de la cabina el o los pisos solicitados.
2. Consultar el estado de la puerta.
3. Cerrar la puerta del piso.
4. Cerrar la puerta de la cabina.
5. En caso de que el sensor de puerta cerrada no se active, se debe encender la señal de cerrar puerta (en el panel de control de la cabina) para que el usuario realice la operación de cerrar la puerta de modo manual.
6. Revisar si en alguno de los pisos hay solicitud de servicio hacia abajo.
7. Activar la señal BAJAR.
8. Revisar la señal de LLEGADA DE LA CABINA, en cada uno de los pisos para visualizar la señal de piso que se encuentra dentro de la cabina.

LLEGADA DE LA CABINA A PISO (PARQUEAR)

Una vez que la cabina llega al piso debe se debe cumplir con las siguientes condiciones:

1. Revisar el sensor de LLEGADA DE LA CABINA, al piso correspondiente.
2. Activar la señal de detenerse.
3. Encender la señal del piso.
4. Abrir la puerta del piso.
5. Abrir la puerta de la cabina.
6. En caso de que el sensor de puerta abierta no se active, se debe encender el piloto de abrir puerta (en el panel de control de la

cabina) para que el usuario realice la operación de abrir la puerta de modo manual.

SOLICITUD DESDE PISO

1. Si el usuario desea subir, encender señal de subir.
2. Si el usuario desea bajar, encender señal de bajar.

SOLICITUD DE SERVICIO DURANTE EL TRANSPORTE

Independientemente del transporte que se esté realizando (hacia arriba o hacia abajo), si se presenta una solicitud de servicio, este se realizará bajo las siguientes condiciones:

1. Si la solicitud de servicio tiene el mismo sentido del transporte, y no se ha pasado por ese piso, el transporte se realiza, si ya se paso por ese piso deberá esperarse hasta que se vuelva a realizar un transporte en donde no se haya pasado por ese piso, o no se presenten más solicitudes en el sentido del transporte.
2. Si la solicitud de servicio es hacia abajo y la cabina se encuentra en un piso superior y no hay más solicitudes de servicio hacia arriba, se le dará servicio.

CONSIDERACIONES GENERALES

1. Existen condiciones que no pueden darse, y es necesario bloquearlas.
2. El sistema queda ocioso si cuando en alguna de las rutas hacia arriba o hacia abajo no hay solicitud de servicio, después de 16 ciclos de reloj.
3. Estando el sistema en reposo, se revisa constantemente por las solicitudes de servicio hacia arriba, si no hay debe revisar las solicitudes de servicio hacia abajo, cualquiera que se encuentre debe de dársele el servicio, de acuerdo con las prioridades.

4. Obviar las solicitudes de algunos pisos en casos especiales (horario, día de la semana, etc.).

Para desarrollar éste automatismo, se tiene que:

1. Desarrollar un recetario con todos los procesos que se dan, y cada uno de ellos detallarlos.
2. Revisar si se requieren más de un autómata para realizar este tipo de control, y que módulos adicionales se deben usar.
3. Plantear un grafcet considerando el recetario. Englobando las "condiciones no"; aquellas que no pueden darse.
4. Qué clase de sensores se deben emplear, y porque.
5. Como se lograría minimizar el cableado del automatismo. Explíquelo.
6. Que otras protecciones se requieren para implementarlo en la realidad.
7. Mediante los planos correspondientes visualice conexiones eléctricas, consideraciones mecánicas, el programa que resulta del entorno de programación.
8. Desarrollar un control SCADA que permita tomar decisiones sobre el automatismo.
9. Elaborar el informe respectivo adecuado para éste automatismo.

ESCUELA DE INGENIERIA ELECTRONICA
LABORATORIO DE AUTOMATAS PROGRAMABLES

GUIA DE PRACTICAS PARA AUTOMATA PROGRAMABLE

Práctica **18**

Tema **Control de la velocidad de un ventilador de un sistema de calefacción de un invernadero.**

Se tiene un invernadero, el cual se debe de controlar la temperatura interior aplicando un equipo S7-200 + EM-235. El tipo de trabajo del automatismo es:

Operación	*Tipo 1*	*Tipo 2*	*Tipo 3*	*Tipo 4*
Temperatura	< 14°	15° - 25°	26° - 28°	>29°
Accionamiento ventilador	Velocidad máxima	Velocidad mínima	Velocidad mínima	Velocidad máxima
Resistores de calor	full	low	apagadas	apagadas

El funcionamiento del ventilador es mediante un motor Dahlander construido con un solo devanado trifásico, pero conectado internamente de tal manera que se tiene acceso a los devanados por los bornes de conexión de la placa. Entonces, según se conecten los bornes exteriores a la red, el motor tenga un número de polos distinto siempre, siendo uno el doble del otro. Teniéndose dos velocidades de rotación, una rápida y otra lenta.

La conexión de sus devanados, se realiza en triángulo para la velocidad menor y en doble estrella para la mayor.

Figura 18.1: Bornera de conexiones externas de un motor Dahlander.

Fuente: ET_AL, I.E.S "La Merced", Departamento de Electricidad, Elaboración de Prácticas Automatizadas Mediante el Modelo S7-200, Período Enero – Marzo 2001, Valladolid – España.

Para el iniciar en la velocidad mínima, hay que aplicar el voltaje de la red a los bornes U1 - V1 - W1 de la bornera de conexiones del motor; debido a que ya está realizada la "conexión triángulo" de sus tres fases en los varios devanados internos del motor. Entre tanto, para la velocidad máxima, se deben realizar dos acciones:

1. Se debe cortocircuitar los bornes U1 - V1 - W1.

2. Aplicar la tensión de alimentación a los bornes U2 - V2 - W2 de la bornera de conexiones.

La calefacción se plantea mediante tres lámparas incandescentes de 100 W a 120 VCA, conectadas en triángulo. Estas representan a una unidad de calor, que contiene resistores que se calientan al ser atravesados por un corriente eléctrica; conciderar que la intensidad luminosa para cada luminaria refleja el grado de calor emitido por las resistencias.

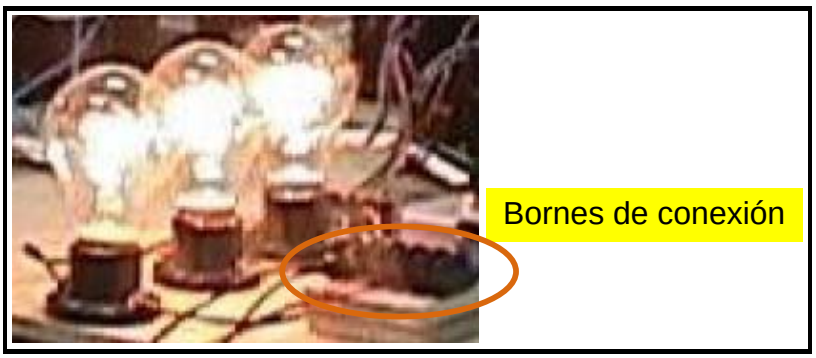


Figura 18.2: Símil de un equipo calefactor.

Fuente: ET_AL, I.E.S "La Merced", Departamento de Electricidad, Elaboración de Prácticas Automatizadas Mediante el Modelo S7-200, Período Enero – Marzo 2001, Valladolid – España.

Para el termostato, detector de la temperatura del invernadero, se utilizó un partidor de tensión con base en un potenciómetro, con salida variable entre 0 y 10 V conectado a través del módulo analógico al autómata.

La correspondencia entre temperatura del invernadero y voltaje de salida del sensor es la siguiente:

Temperatura (°C)	0	14	15	25	26	29
Voltaje (V)	0	2,3	4,2	6,9	8	8,4

Nota: el termostato es de acción mecánica, con un bulbo para el sensado de la temperatura. Los valores presentados pueden cambiar en función del sensor que se use.

Para el desarrollo del automatismo, se pide:

- Establecer ¿cómo se puede variar la velocidad de un motor de CA asíncrono tipo jaula de ardilla?
- Establecer ¿cómo funciona la variación de velocidad de un motor Dahlander?
- Plantear un circuito tentativo para diseñar un sensor de temperatura con elementos que se hallan en el comercio. ¿Sería viable un sensor electrónico?, y ¿cómo se lo desarrollaría?, ¿Qué ventajas y desventajas tendrá un termostato electrónico con uno de mecánico de bulbo?
- ¿Se podría utilizar la salida del módulo analógico EM235 para controlar el calor que emite la unidad calefactora?. Explique.
- Si se alimenta con una tensión línea de 220 VCA trifásica al motor Dahlander, ¿Cuánto es la tensión de cada devanado interno?. Demostrar.
- Plantear todos los esquemas que se requiere para efectuar las conexiones de los equipos involucrados en el automatismo.
- Elaborar el grafcet correspondiente para el control del automatismo.
- Diseñar el algoritmo de programación en el MicroWin.
- Diseñar el interfaz HMI, que indique el funcionamiento y pueda controlar el automatismo.
- Indicar toda la información, que se considere pertinente, para el desarrollo de éste proyecto.

Práctica **19**

Tema **Comunicación Ethernet "Cliente – Servidor" entre dos S7-200, controlando una carga -acoplada al Servidor- desde ambos puestos indistintamente.**

Esta práctica es fundamental para entender cómo opera el intercambio de datos de equipos en una red Ethernet (*Industrial Ethernet*). Es imprescindible conocer el funcionamiento de una unidad de comunicación Ethernet CP243-1 y la CP243-1 IT, para lo cual se recomienda revisar los catálogos correspondientes. Como también, tener claro lo que es una red Ethernet, y como montarla.

Se plantea una red **"cliente – servidor"**, configurada así:

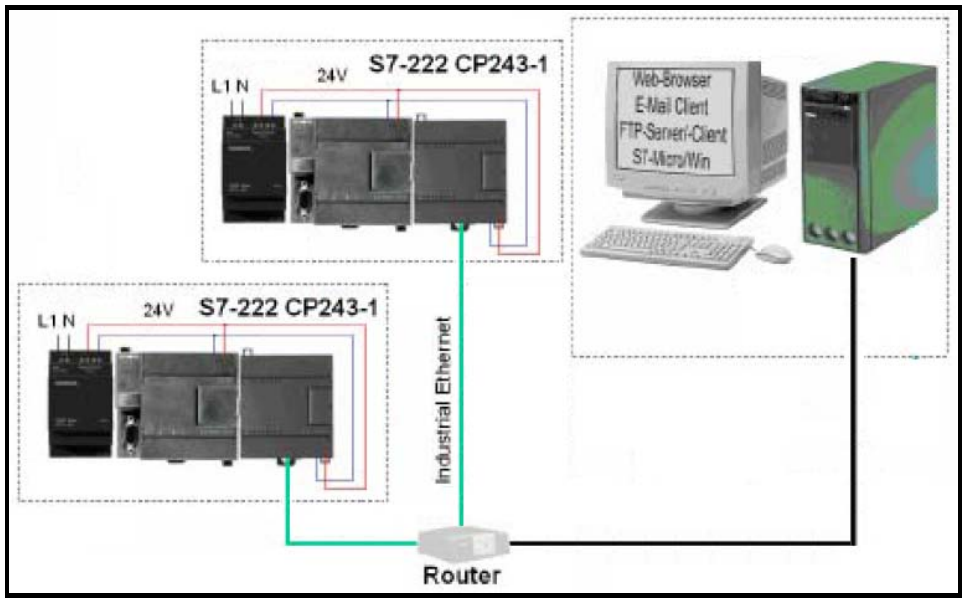


Figura 19.1: Esquemático de una red Ethernet con dos unidades S7-200 + CP243-1, una PC/PG.

Fuente: Siemens, Set 15: Comunicación de Procesos Sencilla basada en Tecnología Web (CP243-1 y CP243-1 IT), Germany – 2006.

En donde una estación S7 sea el "cliente" y la otra el "servidor", mientras que la estación de monitoreo sea la PC, que siempre será "cliente". Considerando:

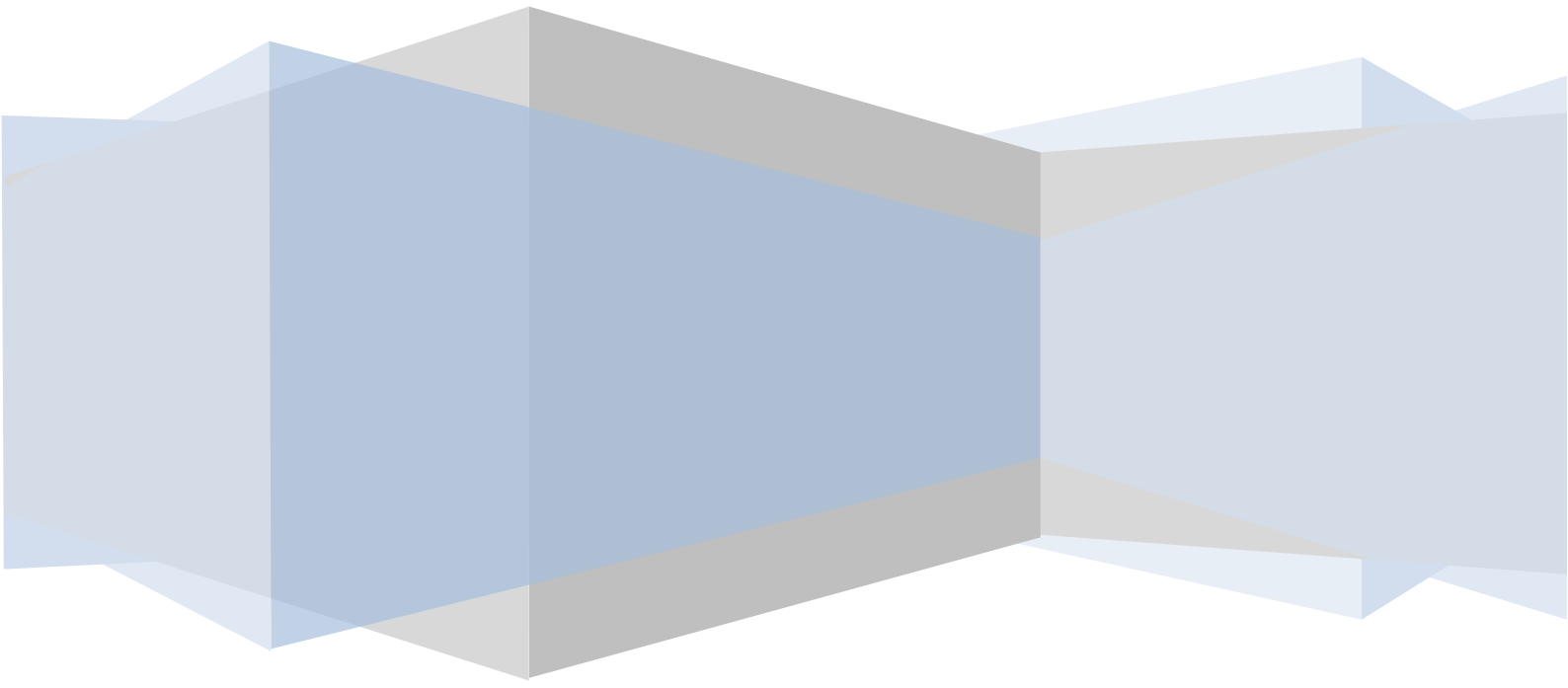
1. En la unidad servidor se tiene dos pulsantes, S1 y S2, para activar o desactivar la carga. Ésta está acoplada en ésta unidad.
2. En la unidad cliente se tiene dos pulsantes, S3 y S4, para activar o desactivar la carga.
3. La carga es un motor trifásico de jaula de ardilla, siendo controlado por un telerruptor trifásico con bobina a 220 VCA.
4. La interfaz HMI es solo de monitoreo, es decir, no puede tomar acción alguna de comando. Y debe indicar, además que existe comunicación entre las unidades periféricas, y entre éstas y la PC.
5. También, si se activa la carga desde una unidad S7 cualquiera, se la pueda apagar desde la otra unidad S7.

Para desarrollar éste automatismo, se tiene que:

- Delimitar y explicar cuáles serían las configuraciones de las unidades S7, por medio del microwin, para el cliente y el servidor.
- ¿Que se colocaría para establecer la comunicación TCP/IP, un "switch" o un "router"?. Explique cada uno.
- Explique brevemente que funciones (subrutinas) se emplean en la comunicación por medio de los módulos CP243-1 y CP243-1 IT.
- Determinar cómo se lograría la monitorización de las variables en el servidor, por medio de la interface HMI.
- Realizar los grafcets correspondientes, tanto para el servidor como para el cliente.
- Diseñar la interfaz gráfica con el WinCC.
- Diseñar el algoritmo de programación en el MicroWin.
- Diseñar los diagramas de conexiones para los dispositivos que intervienen en el automatismo.
- ¿Qué variaría si se pusiese una unidad más S7?
- Adjunte toda la información que considere pertinente para el desarrollo de este automatismo.

CAPITULO 5

Prácticas de desafío



Práctica **20**

Tema **Control del nivel de un tanque de fluído, para mantenerlo constante; con la aplicación de la función PID.**

Esta práctica está basada en el Trabajo de Graduación de Coronel Delgado Pedro Xavier y Peralta Espinoza Paúl Andrés; que consistió en: "Diseño e Implementación de un sistema PID para el control de nivel de un tanque desarrollado en el PLC Siemens S7-200".

Implementado en laboratorios de la UDA, se pretende controlar el nivel de un reservorio de fluído líquido con la configuración siguiente:

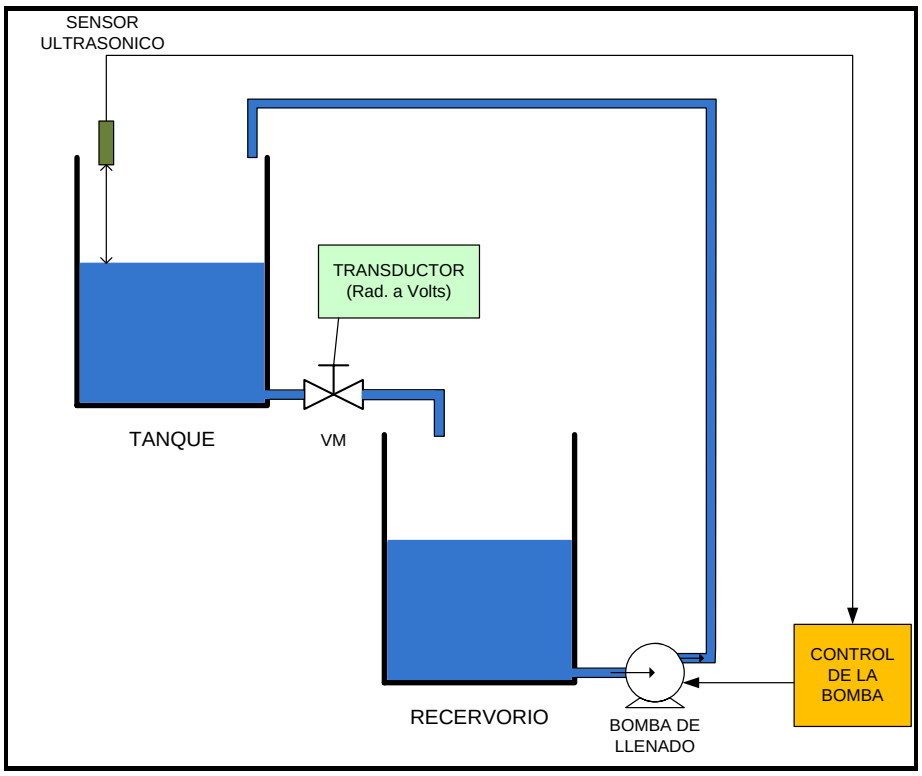


Figura 20.1: Descripción de las partes del proyecto.

Fuente: Autor.

Tanque: Contiene un líquido que se debe de mantener a cierto nivel.

VM: Válvula manual de palanca, que tiene un transductor de radianes a voltaje. Diámetro de ¾ de pulgada.

Figura 20.2: Válvula de descarga VM.

Fuente: Peralta P. y Coronel P, Diseño e Implementación de un sistema PID para el control de nivel de un tanque desarrollado en el PLC Siemens S7-200, UDA, Cuenca, Octubre – 20009.

Reservorio: Recipiente de almacenamiento del líquido.

Bomba de llenado: Encargada de trasladar el líquido. Consta de un control para la velocidad. Sus tomas son de diámetro de ½ pulgada.

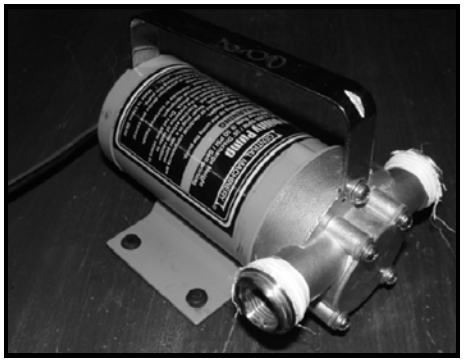


Figura 20.3: Bomba para el llenado del tanque.

Fuente: Peralta P. y Coronel P, Diseño e Implementación de un sistema PID para el control de nivel de un tanque desarrollado en el PLC Siemens S7-200, UDA, Cuenca, Octubre – 20009.

Sensor ultrasónico: Sensa el nivel y emite una señal de voltaje acorde a esa medición.

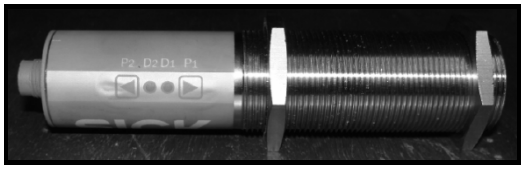


Figura 20.4: Sensor ultrasónico SICK UM30-13113; que está montado en la planta de los laboratorios de la Universidad del Azuay.

Fuente: Peralta P. y Coronel P, Diseño e Implementación de un sistema PID para el control de nivel de un tanque desarrollado en el PLC Siemens S7-200, UDA, Cuenca, Octubre – 2009.

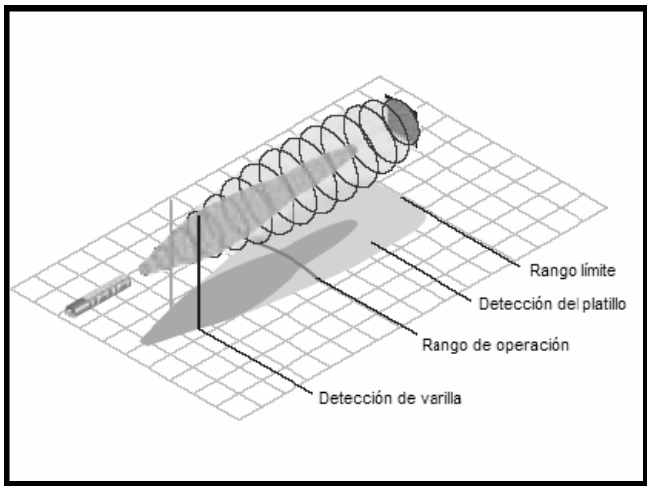


Figura 20.5: Funcionamiento del sensor.

Fuente: SICK AG, Manual del sensor ultrasónico D18 D30 pag.4.

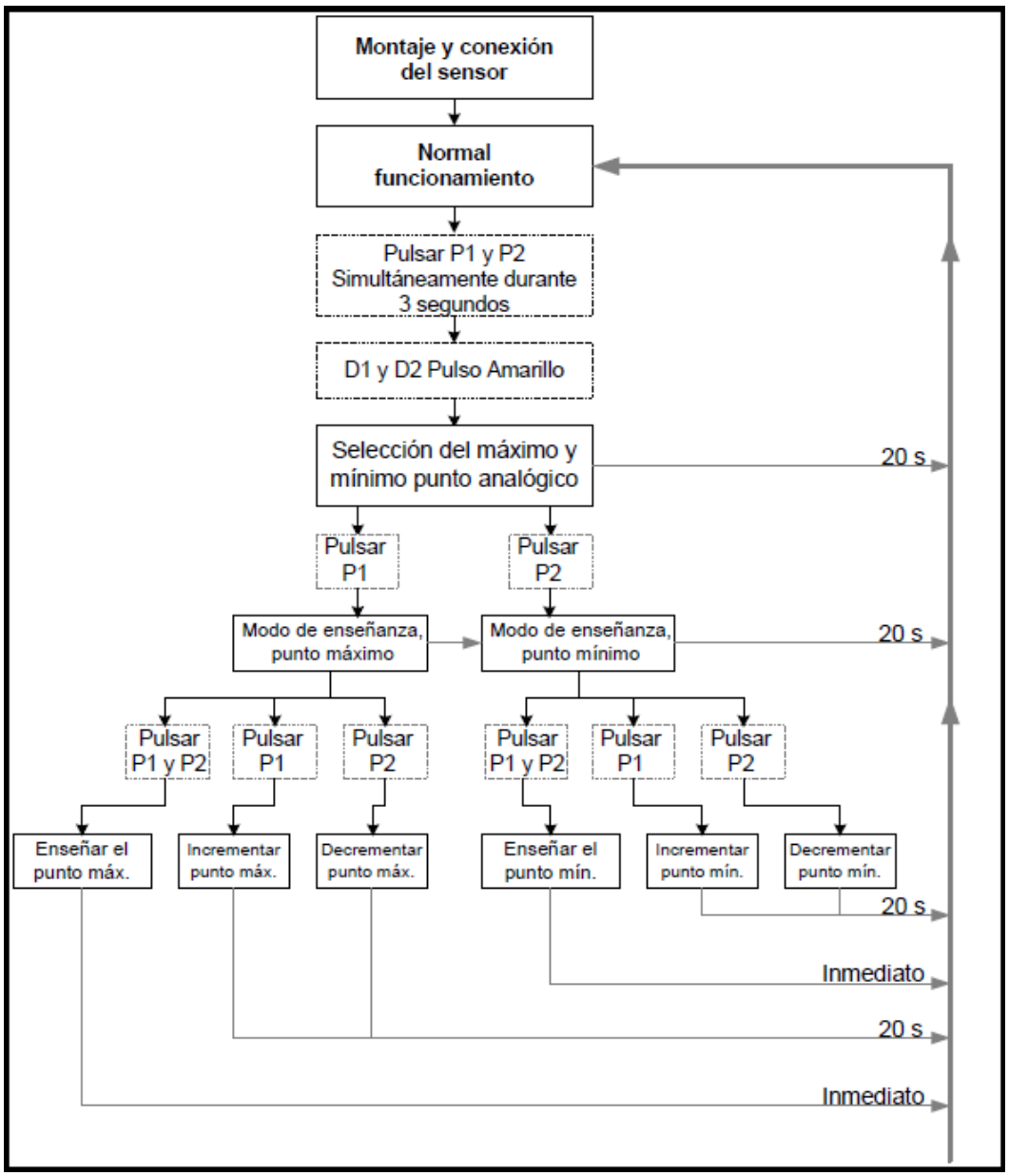


Figura 20.6: Modalidad de configuración del sensor ultrasónico.

Fuente: SICK AG, Manual del sensor ultrasónico D18 D30, Germany.

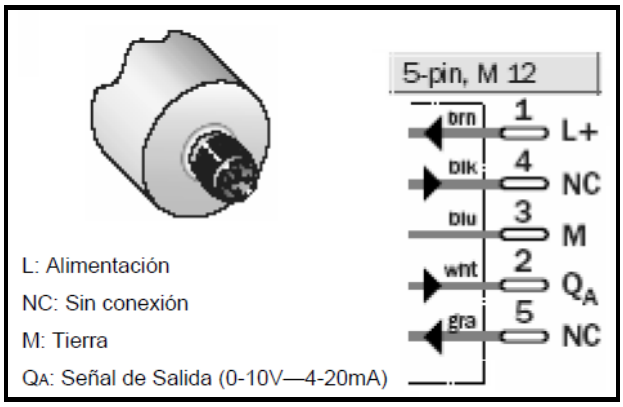


Figura 20.7: Conexiones del sensor ultrasónico.

Fuente: SICK AG, Manual del sensor ultrasónico D18 D30.

Entonces, la operación de llenado del tanque es controlada por la apertura de la válvula de descarga [VM]. En donde:

1. Si por medio de la válvula se da mayor caudal de salida, la bomba de llenado debe de aumentar el caudal de ingreso al tanque.
2. Si por medio de la válvula se da menor caudal de salida, la bomba de llenado debe disminuir el caudal de ingreso.
3. Y si se ha llegado al nivel adecuado, la bomba debe detenerse. Con la finalidad de mantener siempre el mismo nivel de líquido en el tanque.

Antes de desarrollar e implementar este sistema, se debe tener conocimientos de:

- Electrónica digital y de potencia.
- Nociones de Control Automático.
- Manejo de "Matlab y CStation".
- Y, de las nociones de programación vistas en las prácticas anteriores.

Para implementar éste desafío, se requiere:

- Determinar cómo se lograría obtener una señal analógica que relacione la apertura de la válvula en radianes, con un voltaje eléctrico que vaya de "0 a 10 Volts".

- Determinar cómo se lograría controlar la velocidad de giro del motor eléctrico de la bomba de llenado, sabiendo que debe variar en forma proporcional al caudal de salida del tanque.
- Establecer cuál es la planta a controlar, realizando un análisis de cómo trabaja el automatismo. Y a ésta planta aplicarle el control PID, en el caso de que sea factible.
- Demostrar por medio de un diagrama de bloques como opera el "Control PID", analizando si es posible un control en "Lazo Abierto" o "Lazo Cerrado", y cuál de los dos lazos recomienda. Además, que señales requiere manejar para implementar el algoritmo de control en el S7-200.
- Establecer el tipo de trabajo del sensor de ultrasonido.
- Para hacer el programa interactivo, cual serían las variables que se pueden manipular para hacer variar al PID; y como se podrían manipular desde el MicroWin y/o WinCC.

Una vez efectuado los puntos anteriores, desarrollar:

1. El diagrama, grafcet o flujograma, que servirá de soporte para el programa del S7-200.
2. La configuración con el asistente en el MicroWin.
3. El diagrama de conexiones eléctricas del autómata con los equipos y/o elementos usados en la práctica.
4. Una aplicación demostrativa teórica de un control PID con "matlab". De la cual se obtendrá los valores teóricos que servirán de punto de partida para efectuar ajustes al PID real.
5. Si es posible, una aplicación demostrativa teórica de un control PID con "CStation". Y así contrastar los resultados obtenidos en el matlab.
6. Dibujar en el WinCC una "picture" que muestre el funcionamiento de la planta que se está controlando.
7. Armar una memoria técnica sobre toda la información que se necesitó para ésta práctica, incluyendo sus conclusiones y que recomienda.

ESCUELA DE INGENIERIA ELECTRONICA
LABORATORIO DE AUTOMATAS PROGRAMABLES

GUIA DE PRACTICAS PARA AUTOMATA PROGRAMABLE

Práctica **21**

Tema **Configuración de una red Ethernet aplicada al monitoreo y control de procesos independientes.**

Antes de empezar a desarrollar esta práctica, se recomienda establecer la siguiente información:

- Redes informáticas.
- Tipos de redes industriales.
- Como trabaja el CP243-1 y CP243-1 IT.

Bien, con esos preliminares, se presenta el siguiente enunciado:

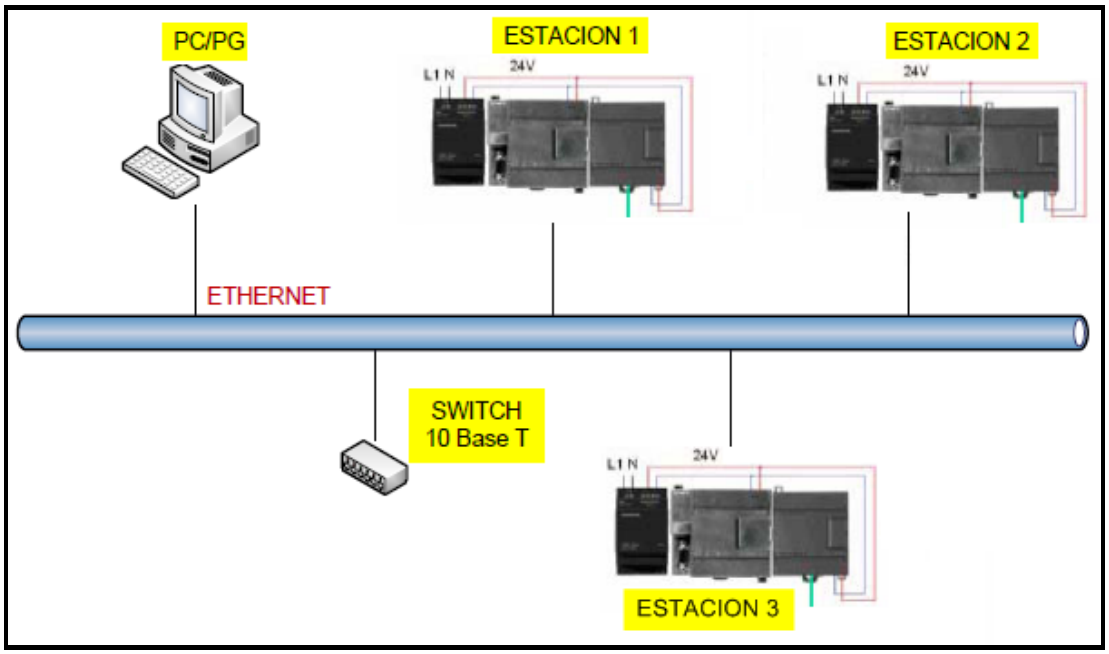


Figura 21.1: Planteamiento de la red Ethernet con PCU S7-200 y CP243-1 y/o CP243-1 IT.

Fuente: Autor.

1. La red Ethernet está conformada por tres equipos S7-200, uno en modo "Servidor" y los otros dos en modo "Cliente".
2. En el "Servidor" estará corriendo un programa que permita conocer el estado de los programas que están corriendo en cada uno de los "Clientes". Si se produce un evento cualquiera, se podrá tomar la decisión de bloquear al "Cliente" correspondiente.
3. La interface gráfica HMI visualizará los procesos que estén corriendo en los "Clientes" y en el "Servidor", pero no podrá interferir en su funcionamiento.
4. En cada "Cliente" se puede implementar cualquiera de las prácticas anteriores, sin perjuicio de que se las pueda modificar. Como también cualquier otro programa.
5. Los autómatas se interconectarán por medio de los módulos CP243-1 o CP243-1 IT, mediante cables estándar de red Cat. 5e [mínimo], y con conectores RJ-45. Formando una red "estrella".

Para lo cual se planteará y resolverá:

- Como se tiene que ponchar los terminales RJ-45, indicando gráficamente.
- Determinar las direcciones IP que tendrán, así como sus máscaras de red y gateway.
- Que algoritmos se tendrán que proyectar para colocar un "Servidor" y dos "Clientes".
- Establecer un programa "test" para conocer si la red esta operativa.
- Elaborar una interface gráfica que permita visualizar los eventos que ocurran en cada "Cliente", así como también en el "Servidor".
- ¿Qué relación existe entre una Red ProfiBus y una Red Ethernet?
- Plantear y depurar un diagrama de conexiones externas que represente, en forma verás, todas las conexiones hechas en cada CPU y su equipo y/o dispositivos.
- Plantear los grafcets respectivos para el correcto funcionamiento del automatismo.
- Diseñar y depurar al máximo posible el programa en MicroWin.

- Al final elaborar una memoria técnico-descriptiva sobre la práctica, adjuntando toda la información que sea pertinente para el proyecto.

Práctica **22**

Tema **Emplear el software LabView para efectuar el monitoreo de variables en un servidor S7.**

Esta práctica es abierta al diseño e implementación, pero se tiene que mantener que se va a montar una red Ethernet con dos unidades S7, y para monitorear y/o controlar éstas unidades se empleará el LabView de National Instruments como componente SCADA. Para lo cual se deberá:

- Establecer los dispositivos necesarios para montar una red Ethernet con dos unidades S7-200 con sus respectivos CP243-1 o CP243-1 IT, y la PC. Considerando que ahora el SCADA que se va a usar el LabView.
- Establecer las normativas correspondientes que regirán la comunicación entre éstas, y con la PC.
- Determinar que software o librería se empleará para efectuar el enlace entre el LabView y las unidades S7.

Para los programas:

- En la estación cliente se tendrá corriendo cualquier programa.
- En la estación servidor se tendrá las variables que se desea monitorear y/o controlar.
- En la PC se tendrá el enlace entre los autómatas y el LabView. En éste último se desarrollará una programación tal que permita establecer un control sobre las variables que se desee.

- Los programas en los autómatas deberán tener relación entre sí, que justifique el monitoreo SCADA. Pueden ser tomados de las prácticas anteriores, o se puede diseñar otro algoritmo.

Por lo tanto, para el desarrollo del automatismo se requiere:

- Diseñar los diagramas correspondientes para las conexiones de los varios dispositivos que se emplearán. Y si es necesario para el hardware de interfaz del S7 con el SCADA.
- Establecer cuál va a ser la dinámica de comunicación del SCADA para con las variables en el PLC.
- Establecer los diseños más óptimos para los algoritmos de los programas, tanto para los autómatas como para el SCADA.
- Desarrollar una memoria técnica descriptiva de la totalidad del proyecto, adjuntando toda la información que se considere pertinente para la correcta realización.

NOTA: *Para poder desarrollar ésta práctica se debe consultar la página de web www.ni.com, en relación a:*

- *¿Cómo se puede interconectar el LabView con cualquier otro dispositivo comercial de otra marca y/o fabricante?*
- *¿Cómo se logra manipular una "shared variable" en una programación?*
- *¿Cómo funciona un OPCServer?*
- *¿Qué procedimiento se tiene que realizar para enlazar una posición de memoria del autómata S7-200 con el LabView?*

UNIVERSIDAD DEL AZUAY

FACULTAD DE CIENCIA Y TECNOLOGÍA

ESCUELA DE INGENIERÍA ELECTRÓNICA

GUÍA DE PRÁCTICAS PARA AUTÓMATA PROGRAMABLE BASADO EN EL S7-200 Y EL EM-235

TRABAJO DE GRADUACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA

Autor

Francisco Eduardo Alvarado Correa

Director

Ing. Francisco Eugenio Vázquez Calero

TOMO 2

Cuenca - Ecuador

2010

RESPONSABILIDAD

Las ideas y opiniones vertidas, sin perder la óptica y finalidad de los fabricantes [*propietarios de las marcas comerciales Siemens y LabView, y de los equipos aquí usados, respectivamente*], en éste trabajo de graduación son de exclusiva responsabilidad del Autor. Además, cada práctica se debe tomar como una posibilidad; y no como una solución definitiva.

Esto porque cada tema puede resolverse de otra manera, que optimice aun más el automatismo.

INDICE DE CONTENIDO

TOMO 2

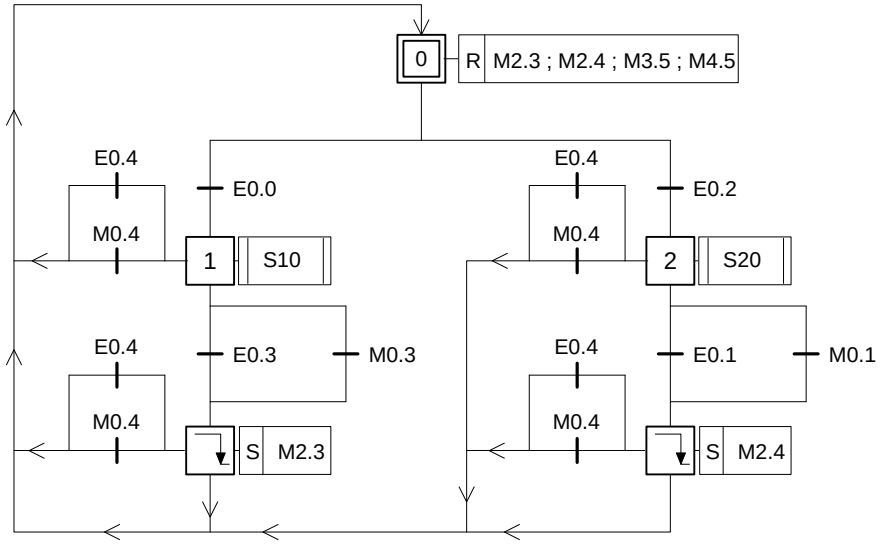
INDICE DE FIGURAS

INDICE DE TABLAS

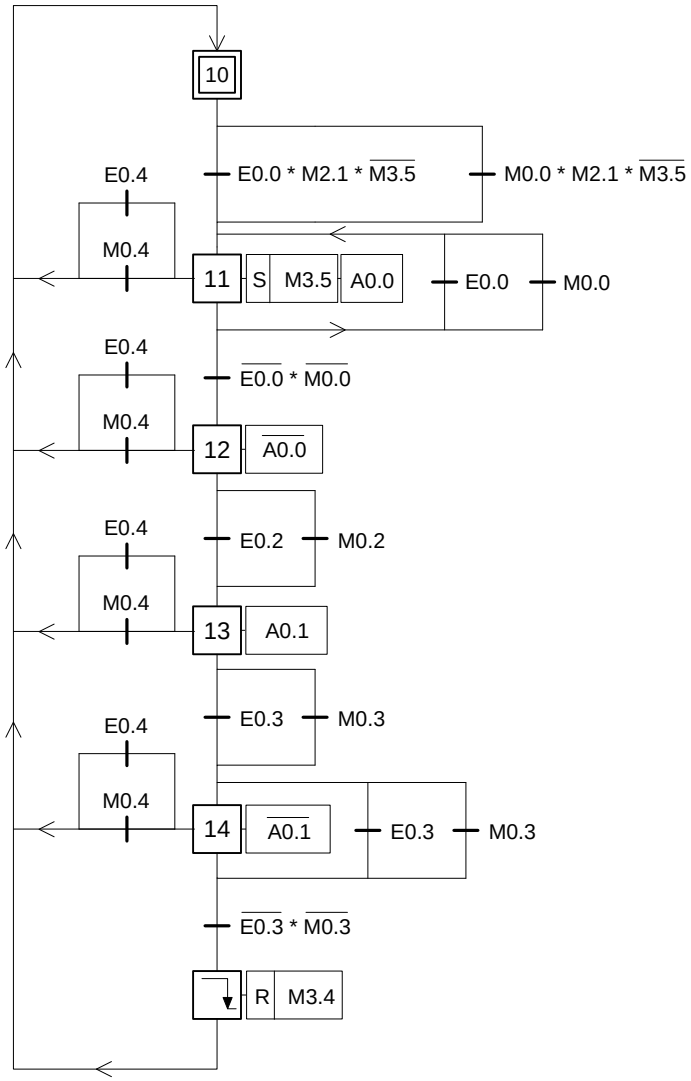
INDICE DE ANEXOS

RESPUESTAS

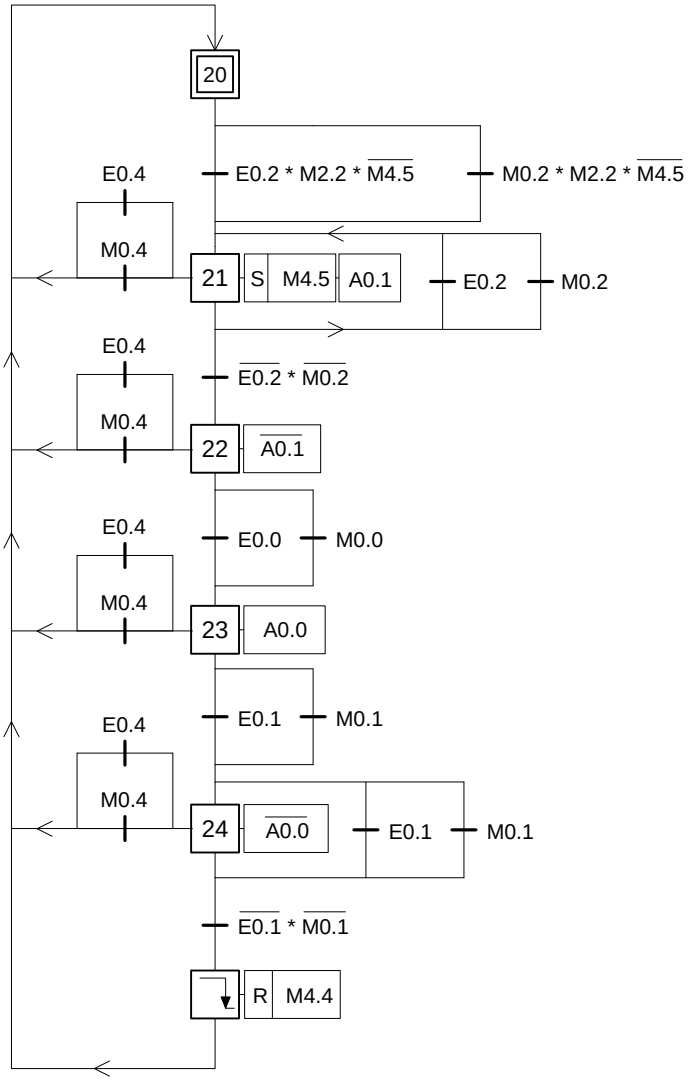
Programa principal:



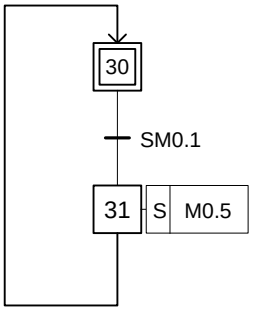
Subrutina 1:



Subrutina 2:



Comunicación:



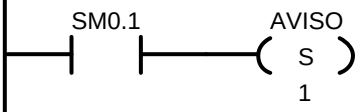
Bloque: PRINCIPAL
Autor:
Fecha de creación: 25.04.2009 9:31:03
Fecha de modificación: 27.04.2009 22:06:25

Símbolo	Tipo var.	Tipo de datos	Comentario
	TEMP		
	TEMP		
	TEMP		
	TEMP		

Programa Principal

Network 1 <<<<<<<< Comunicacion entre PLC y la PC >>>>>>>>>>>>>>>

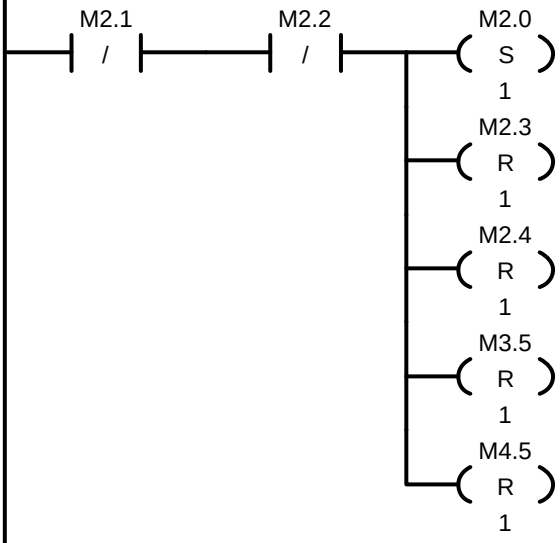
Solo se verifica una vez al inicio del programa.



Símbolo	Dirección	Comentario
AVISO	M0.5	VIRTUAL

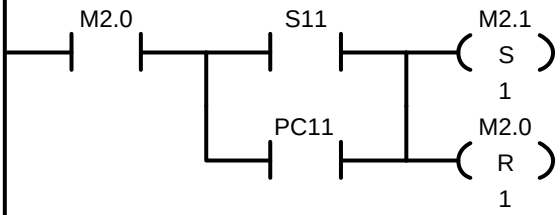
Network 2 ++++++++Bloque principal del programa de control++++++++++

Comentario de segmento



Network 3

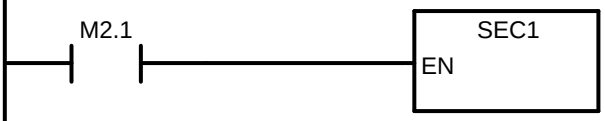
Ingreso a la primera secuencia



Símbolo	Dirección	Comentario
PC11	M0.0	SCADA
S11	E0.0	FISICO

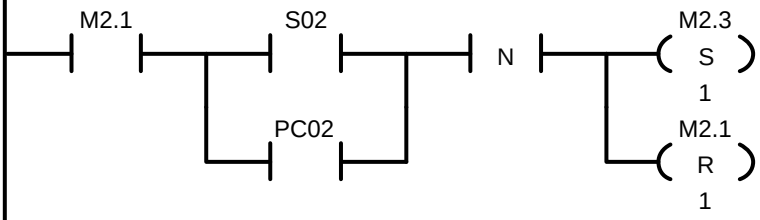
Network 4

Llamado a subrutina 1==> si se desactiva M2.1, se desactiva la subrutina y el programa se bloquea.



Network 5

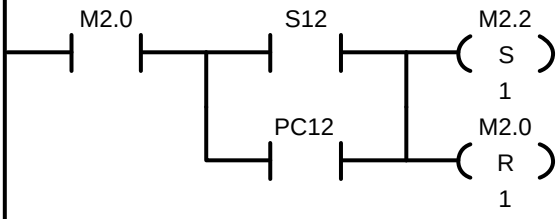
Una ves que sale de la subrutina, se deshabilita la llamada a esta con el flanco negativo del pulsante de paro del "actuador 2".



Símbolo	Dirección	Comentario
PC02	M0.3	SCADA
S02	E0.3	FISICO

Network 6

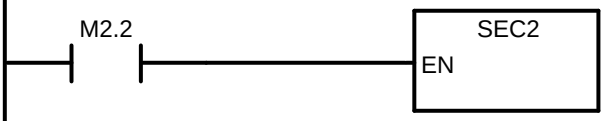
Ingreso a la segunda secuencia



Símbolo	Dirección	Comentario
PC12	M0.2	SCADA
S12	E0.2	FISICO

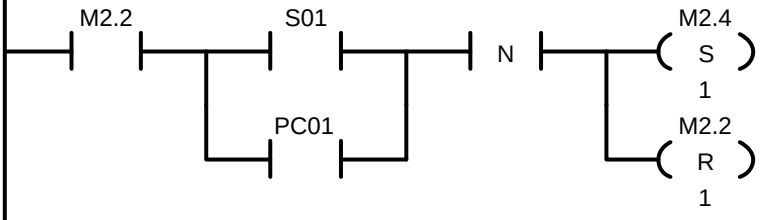
Network 7

Llamado a la subrutina 2



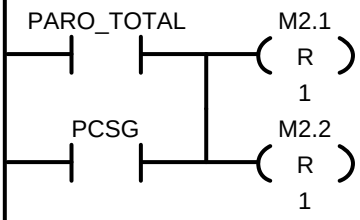
Network 8

Una que sale de la subrutina, se deshabilita la llamada a esta con el flanco negativo del pulsante de paro del "actuador 1".



Símbolo	Dirección	Comentario
PC01	M0.1	SCADA
S01	E0.1	FISICO

Network 9



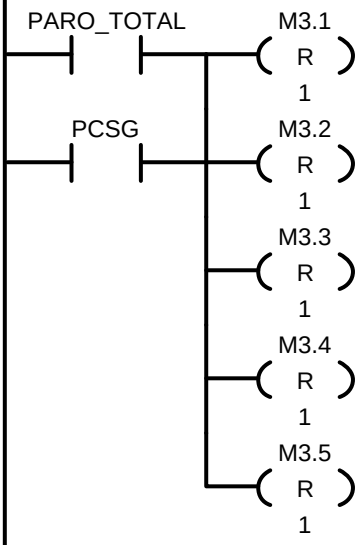
Símbolo	Dirección	Comentario
PARO_TOTAL	E0.4	FISICO
PCSG	M0.4	SCADA

Bloque: SEC1
Autor:
Fecha de creación: 25.04.2009 9:31:03
Fecha de modificación: 27.04.2009 22:06:25

Símbolo	Tipo var.	Tipo de datos	Comentario
EN	IN	BOOL	
	IN	BOOL	
	IN		
	IN_OUT		
	OUT		
	TEMP		

Subrutina de la primera secuencia

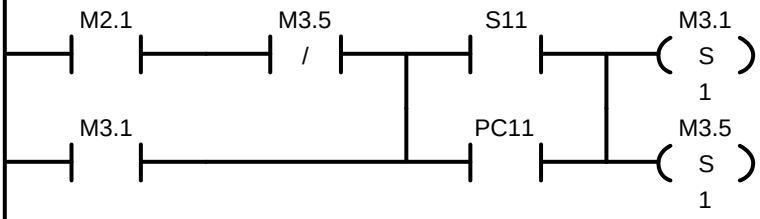
Network 1 Resets de las marcas



Símbolo	Dirección	Comentario
PARO_TOTAL	E0.4	FISICO
PCSG	M0.4	SCADA

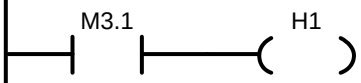
Network 2 Título de segmento

Comentario de segmento



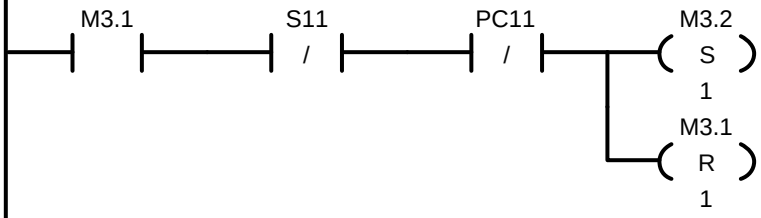
Símbolo	Dirección	Comentario
PC11	M0.0	SCADA
S11	E0.0	FISICO

Network 3



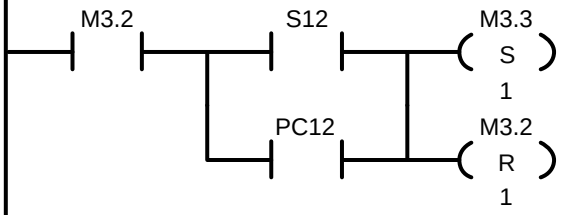
Símbolo	Dirección	Comentario
H1	A0.0	FISICO

Network 4



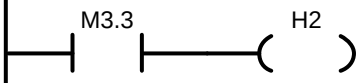
Símbolo	Dirección	Comentario
PC11	M0.0	SCADA
S11	E0.0	FISICO

Network 5



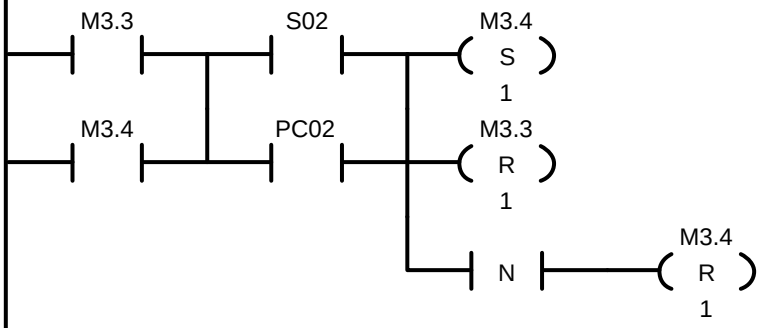
Símbolo	Dirección	Comentario
PC12	M0.2	SCADA
S12	E0.2	FISICO

Network 6



Símbolo	Dirección	Comentario
H2	A0.1	FISICO

Network 7



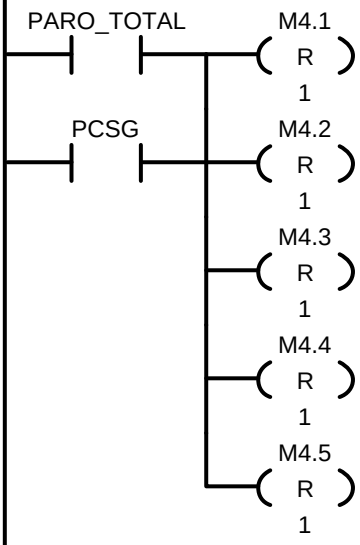
Símbolo	Dirección	Comentario
PC02	M0.3	SCADA
S02	E0.3	FISICO

Bloque: SEC2
Autor:
Fecha de creación: 25.04.2009 9:58:49
Fecha de modificación: 27.04.2009 18:04:16

Símbolo	Tipo var.	Tipo de datos	Comentario
EN	IN	BOOL	
	IN	BOOL	
	IN		
	IN_OUT		
	OUT		
	TEMP		

Subrutina de la segunda secuencia

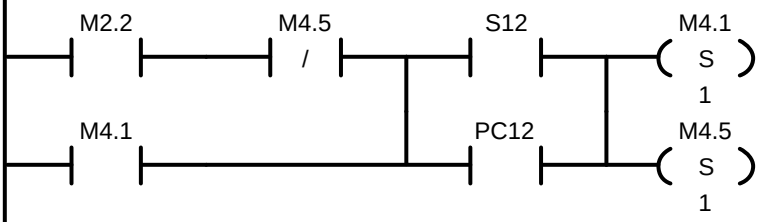
Network 1 Resets de las marcas



Símbolo	Dirección	Comentario
PARO_TOTAL	E0.4	FISICO
PCSG	M0.4	SCADA

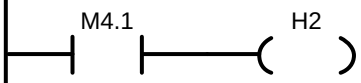
Network 2 Título de segmento

Comentario de segmento



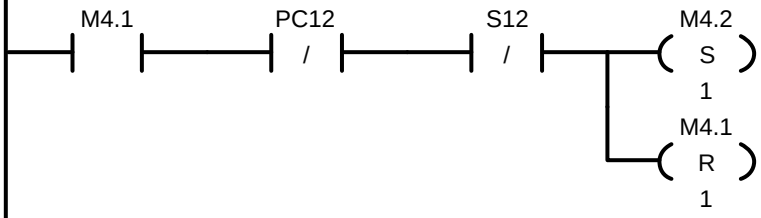
Símbolo	Dirección	Comentario
PC12	M0.2	SCADA
S12	E0.2	FISICO

Network 3



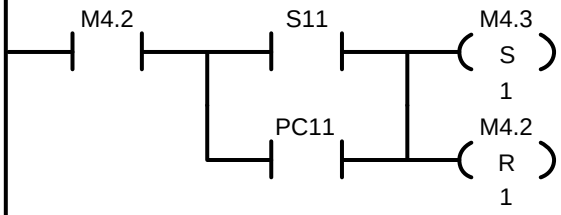
Símbolo	Dirección	Comentario
H2	A0.1	FISICO

Network 4



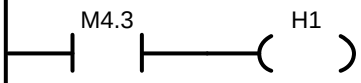
Símbolo	Dirección	Comentario
PC12	M0.2	SCADA
S12	E0.2	FISICO

Network 5



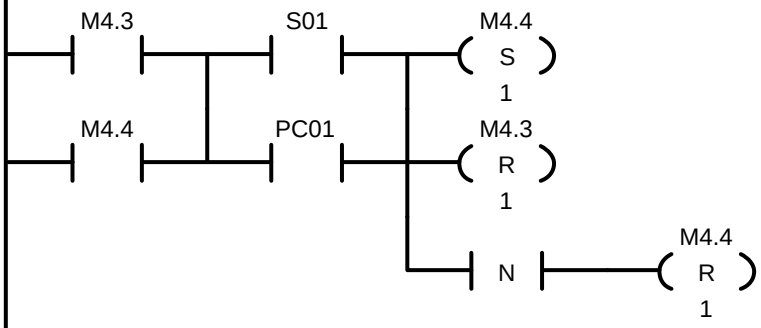
Símbolo	Dirección	Comentario
PC11	M0.0	SCADA
S11	E0.0	FISICO

Network 6



Símbolo	Dirección	Comentario
H1	A0.0	FISICO

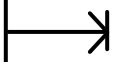
Network 7



Símbolo	Dirección	Comentario
PC01	M0.1	SCADA
S01	E0.1	FISICO

Network 8 Título de segmento

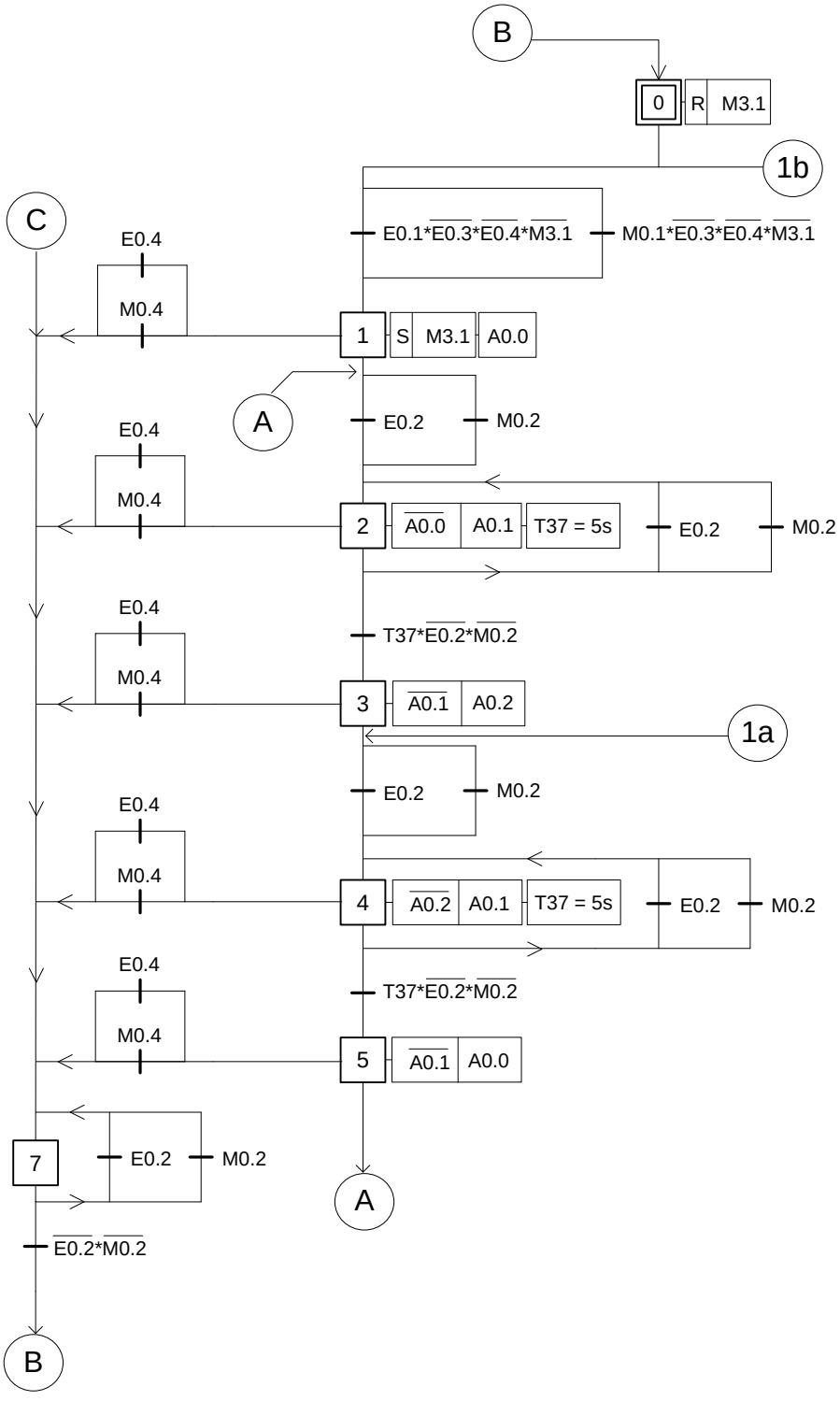
Comentario de segmento

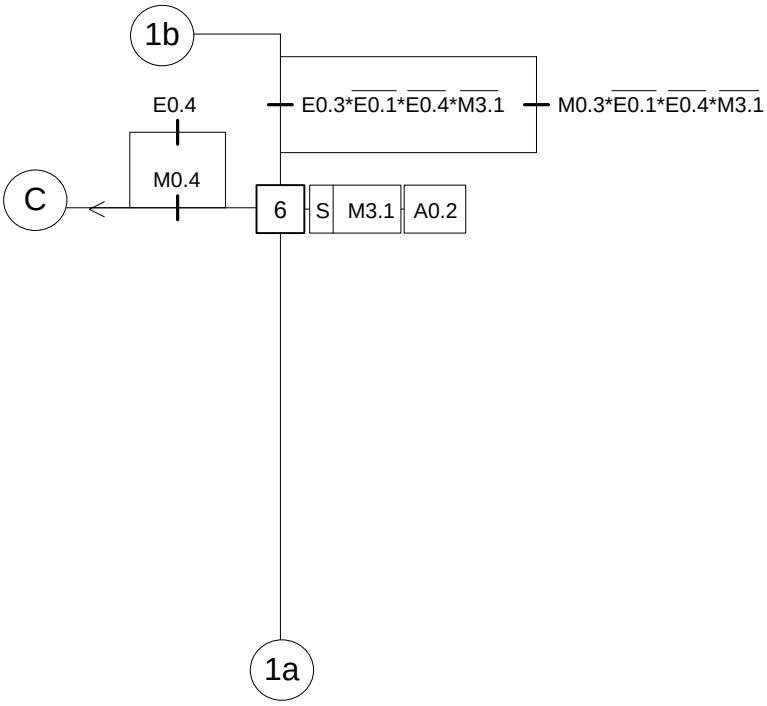


Símbolo	Dirección	Comentario
PC11	M0.0	SCADA
PC01	M0.1	SCADA
PC12	M0.2	SCADA
PC02	M0.3	SCADA
PCSG	M0.4	SCADA
AVISO	M0.5	VIRTUAL
H1	A0.0	FISICO
H2	A0.1	FISICO
S11	E0.0	FISICO
S01	E0.1	FISICO
S12	E0.2	FISICO
S02	E0.3	FISICO
PARO_TOTAL	E0.4	FISICO

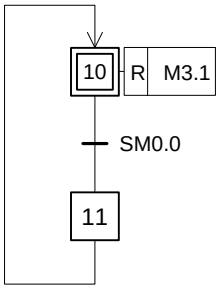
Símbolo	Dirección	Comentario
SEC1	SBR0	Subrutina de la primera secuencia
SEC2	SBR1	Subrutina de la segunda secuencia
PRINCIPAL	OB1	Programa Principal

Programa principal:





Comunicación:

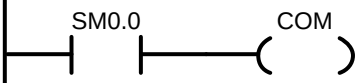


Bloque: PRINCIPAL
Autor:
Fecha de creación: 09.05.2009 17:34:19
Fecha de modificación: 30.05.2010 21:16:43

Símbolo	Tipo var.	Tipo de datos	Comentario
	TEMP		
	TEMP		
	TEMP		
	TEMP		

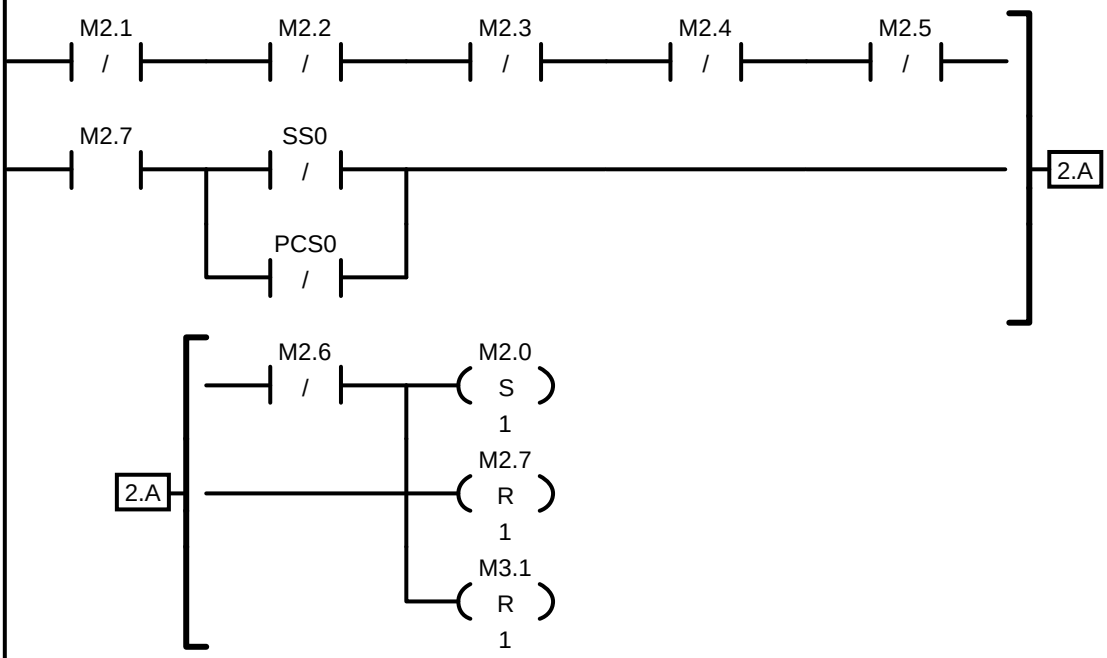
PROGRAMA PARA INVERSION DE GIRO TEMPORIZADA

Network 1 COMUNICACION CON LA PC



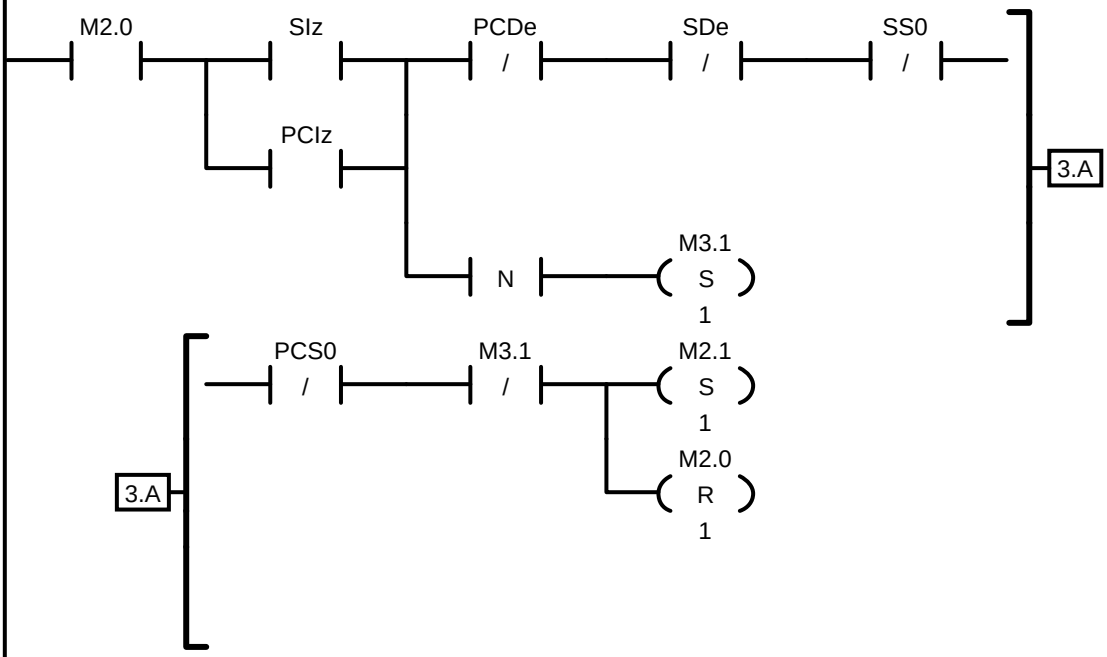
Símbolo	Dirección	Comentario
COM	M0.0	COMUNICACION PC <---> PLC

Network 2 BLOQUE PRINCIPAL



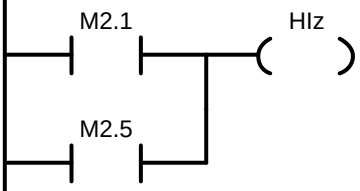
Símbolo	Dirección	Comentario	
PCS0	M0.4	PARO VIRTUAL	<====
SS0	E0.4	PARO FISICO	

Network 3



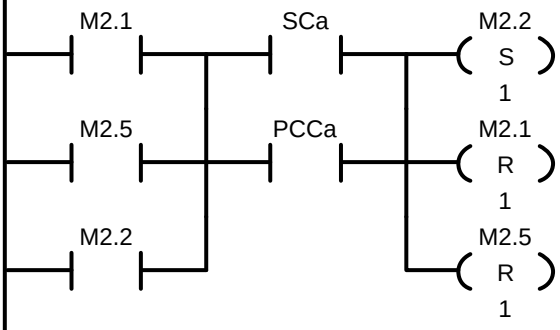
Símbolo	Dirección	Comentario		
PCDe	M0.3	DER VIRTUAL	=	SCADA
PCIz	M0.1	IZQ VIRTUAL	<====	
PCS0	M0.4	PARO VIRTUAL	<====	
SDe	E0.3	GIRO DERECHA FISICO		
SIz	E0.1	GIRO IZQUIERDA FISICO		
SS0	E0.4	PARO FISICO		

Network 4



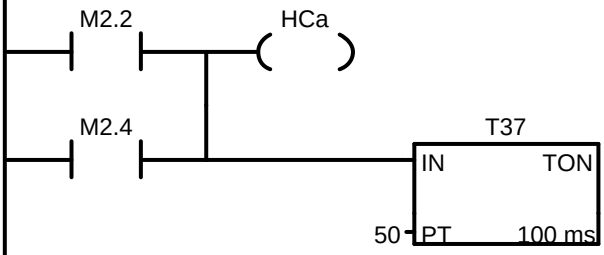
Símbolo	Dirección	Comentario
HIz	A0.0	SALIDA IZQ

Network 5



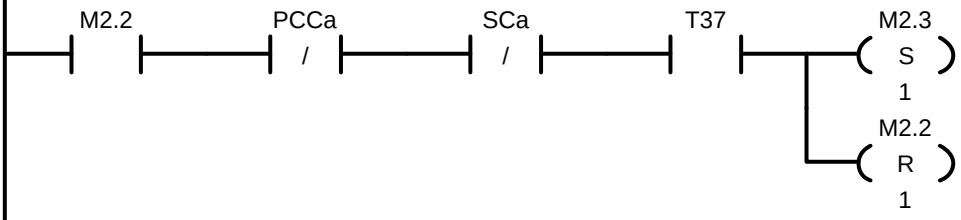
Símbolo	Dirección	Comentario		
PCCa	M0.2	CAMBIO VIRTUAL	=	DESDE
SCa	E0.2	CAMBIO DE GIRO FISICO		

Network 6



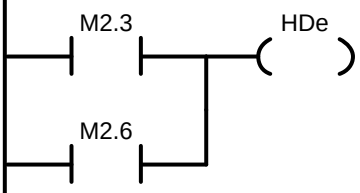
Símbolo	Dirección	Comentario
HCa	A0.1	PILOTO DE CAMBIO

Network 7



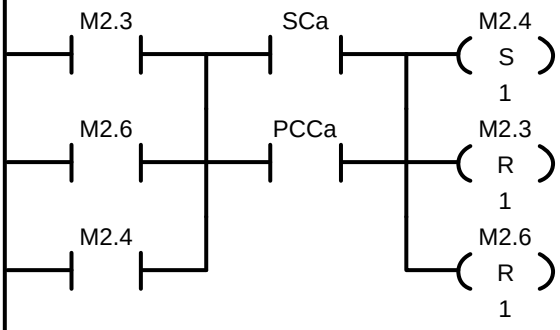
Símbolo	Dirección	Comentario		
PCCa	M0.2	CAMBIO VIRTUAL	=	DESDE
SCa	E0.2	CAMBIO DE GIRO FISICO		

Network 8



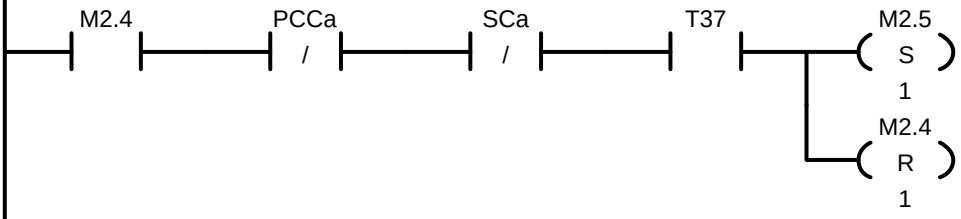
Símbolo	Dirección	Comentario
HDe	A0.2	SALIDA DER

Network 9



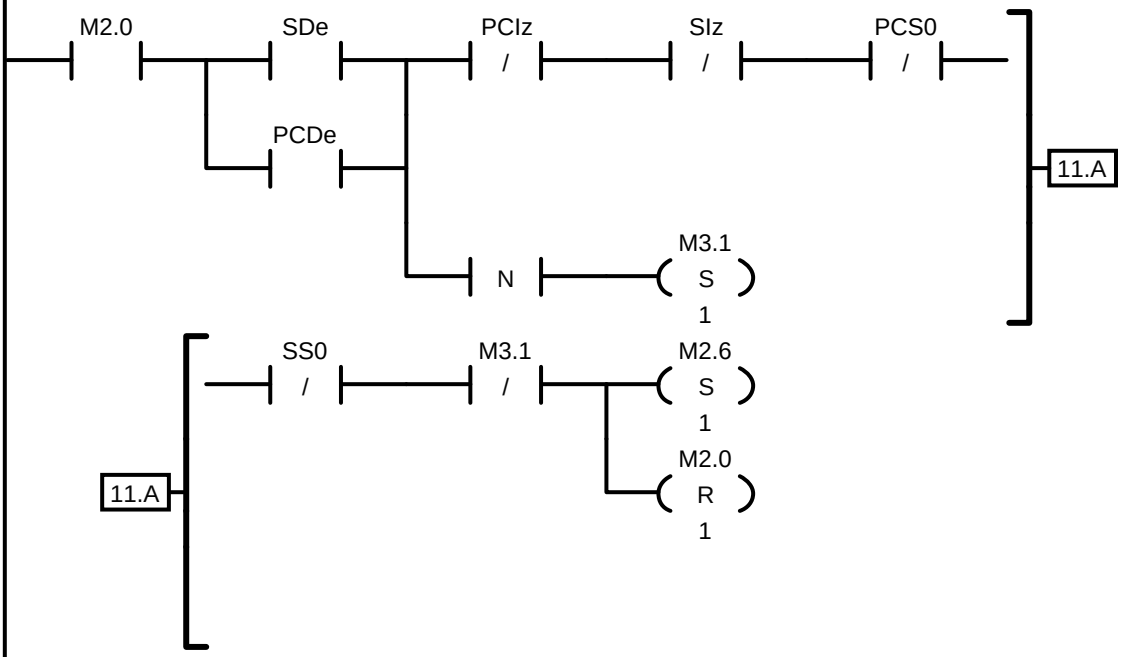
Símbolo	Dirección	Comentario		
PCCa	M0.2	CAMBIO VIRTUAL	=	DESDE
SCa	E0.2	CAMBIO DE GIRO FISICO		

Network 10



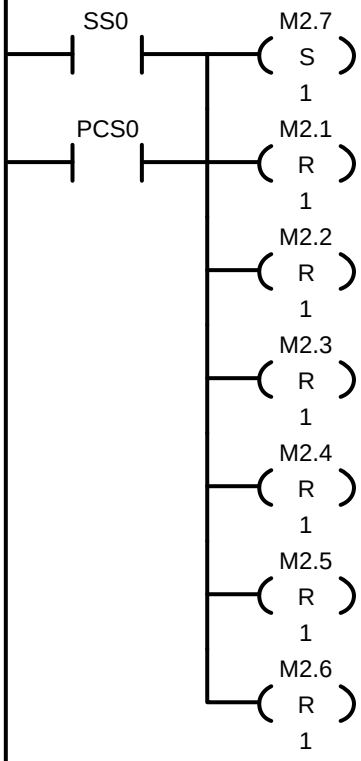
Símbolo	Dirección	Comentario		
PCCa	M0.2	CAMBIO VIRTUAL	=	DESDE
SCa	E0.2	CAMBIO DE GIRO FISICO		

Network 11



Símbolo	Dirección	Comentario		
PCDe	M0.3	DER VIRTUAL	=	SCADA
PCIz	M0.1	IZQ VIRTUAL	<====	
PCS0	M0.4	PARO VIRTUAL	<====	
SDe	E0.3	GIRO DERECHA FISICO		
SIz	E0.1	GIRO IZQUIERDA FISICO		
SS0	E0.4	PARO FISICO		

Network 12 PARO TOTAL

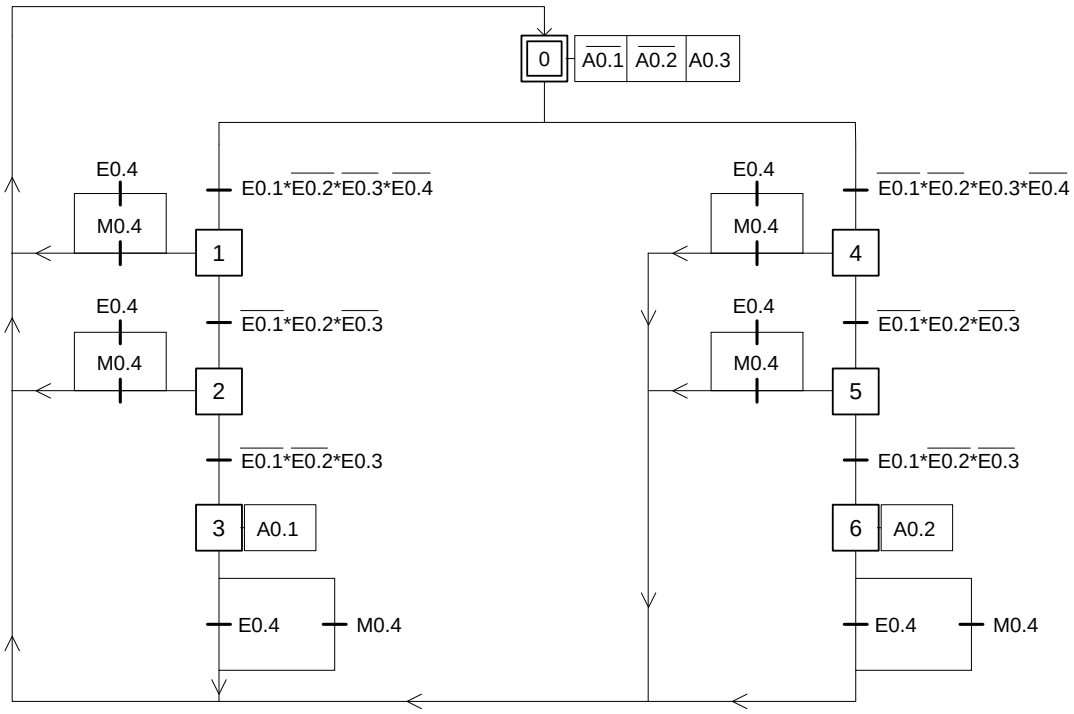


Símbolo	Dirección	Comentario	
PCS0	M0.4	PARO VIRTUAL	<====
SS0	E0.4	PARO FISICO	

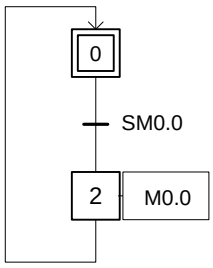
Símbolo	Dirección	Comentario
COM	M0.0	COMUNICACION PC <---> PLC
SIz	E0.1	GIRO IZQUIERDA FISICO
SCa	E0.2	CAMBIO DE GIRO FISICO
SDe	E0.3	GIRO DERECHA FISICO
SS0	E0.4	PARO FISICO
HIz	A0.0	SALIDA IZQ
HCa	A0.1	PILOTO DE CAMBIO
HDe	A0.2	SALIDA DER
PCIz	M0.1	IZQ VIRTUAL <====
PCCa	M0.2	CAMBIO VIRTUAL = DESDE
PCDe	M0.3	DER VIRTUAL = SCADA
PCS0	M0.4	PARO VIRTUAL <====

Símbolo	Dirección	Comentario
PRINCIPAL	OB1	PROGRAMA PARA INVERSION DE GIRO TEMPORIZADA

Programa principal:



Comunicación:

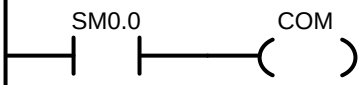


Bloque: PRINCIPAL
Autor:
Fecha de creación: 21.01.2009 17:13:12
Fecha de modificación: 30.05.2010 21:14:12

Símbolo	Tipo var.	Tipo de datos	Comentario
	TEMP		
	TEMP		
	TEMP		
	TEMP		

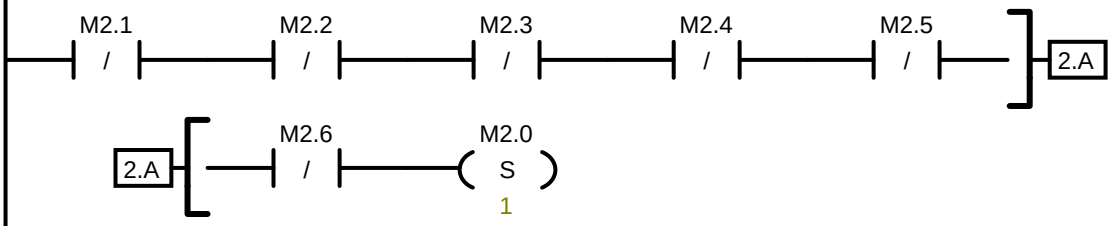
PROGRAMA DE INVERSION CONDICIONADA

Network 1 COMUNICACION PC <---> PLC

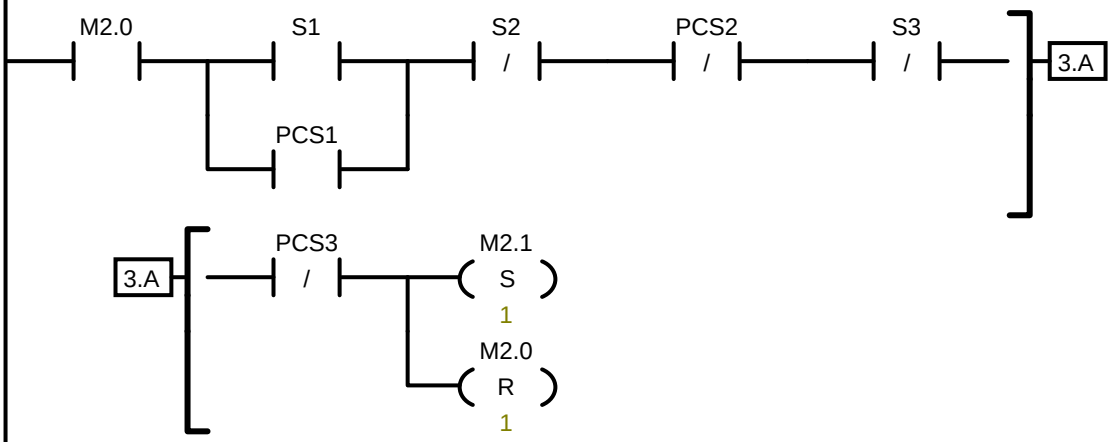


Símbolo	Dirección	Comentario
COM	M0.0	PC <----> PLC

Network 2

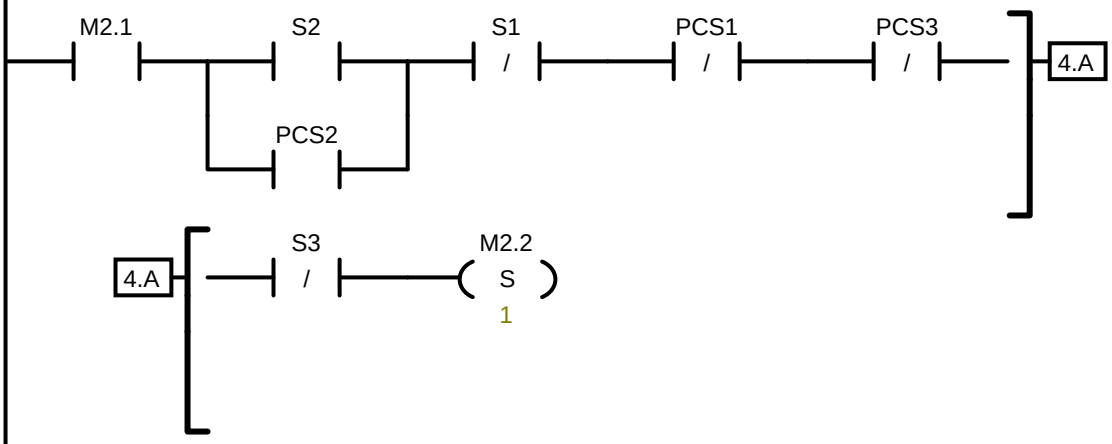


Network 3 SECUENCIA 1



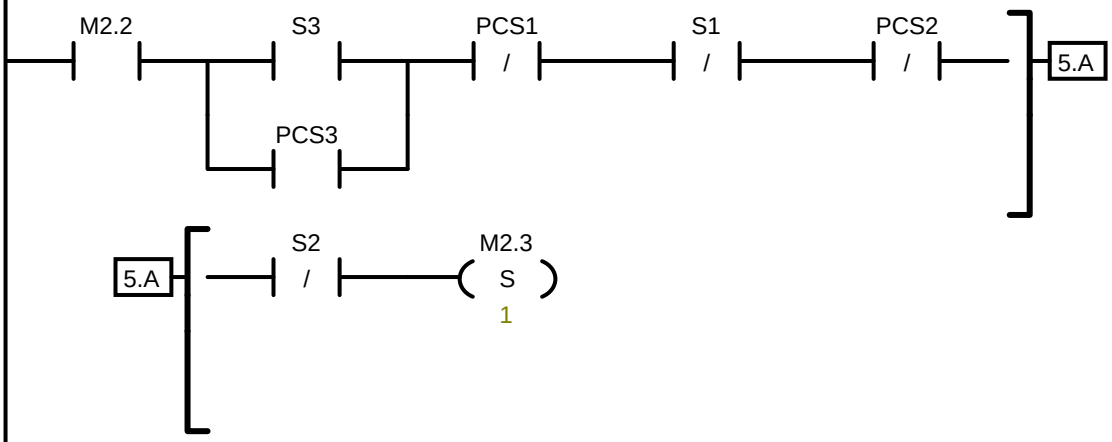
Símbolo	Dirección	Comentario
PCS1	M0.1	PULSANTE VIRTUAL 1
PCS2	M0.2	PULSANTE VIRTUAL 2
PCS3	M0.3	PULSANTE VIRTUAL 3
S1	E0.1	PULSANTE 1
S2	E0.2	PULSANTE 2
S3	E0.3	PULSANTE 3

Network 4



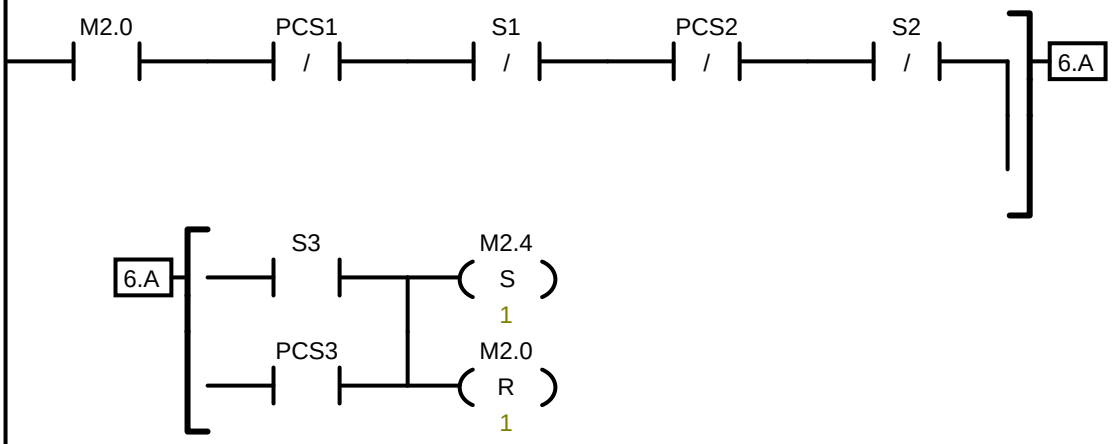
Símbolo	Dirección	Comentario
PCS1	M0.1	PULSANTE VIRTUAL 1
PCS2	M0.2	PULSANTE VIRTUAL 2
PCS3	M0.3	PULSANTE VIRTUAL 3
S1	E0.1	PULSANTE 1
S2	E0.2	PULSANTE 2
S3	E0.3	PULSANTE 3

Network 5



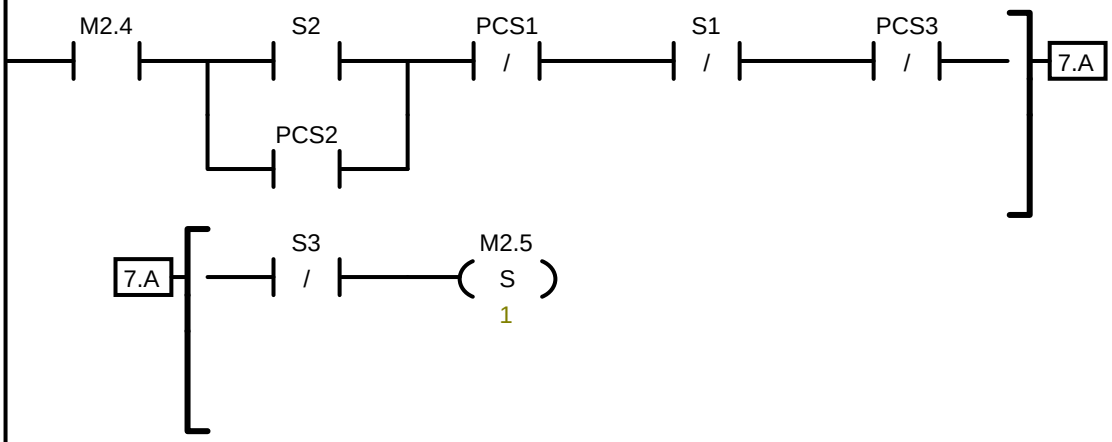
Símbolo	Dirección	Comentario
PCS1	M0.1	PULSANTE VIRTUAL 1
PCS2	M0.2	PULSANTE VIRTUAL 2
PCS3	M0.3	PULSANTE VIRTUAL 3
S1	E0.1	PULSANTE 1
S2	E0.2	PULSANTE 2
S3	E0.3	PULSANTE 3

Network 6 SECUENCIA 2



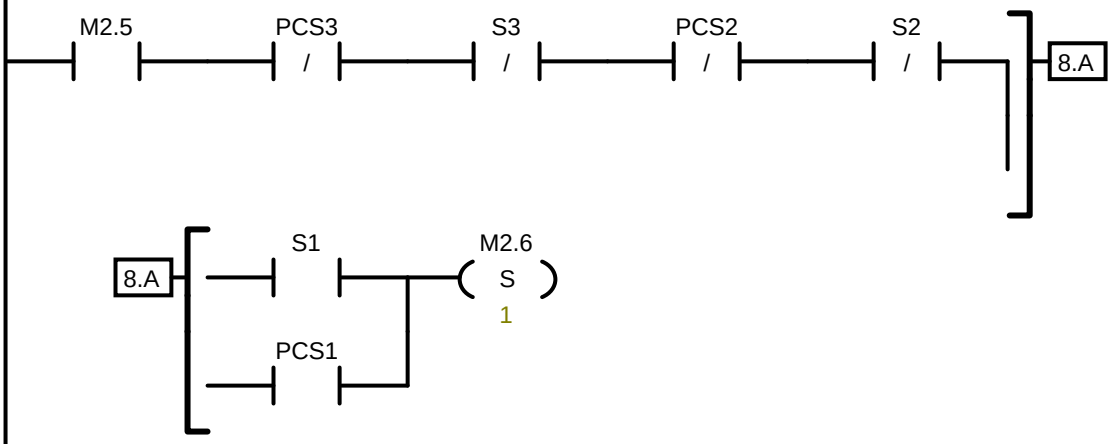
Símbolo	Dirección	Comentario
PCS1	M0.1	PULSANTE VIRTUAL 1
PCS2	M0.2	PULSANTE VIRTUAL 2
PCS3	M0.3	PULSANTE VIRTUAL 3
S1	E0.1	PULSANTE 1
S2	E0.2	PULSANTE 2
S3	E0.3	PULSANTE 3

Network 7



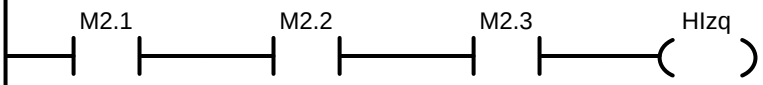
Símbolo	Dirección	Comentario
PCS1	M0.1	PULSANTE VIRTUAL 1
PCS2	M0.2	PULSANTE VIRTUAL 2
PCS3	M0.3	PULSANTE VIRTUAL 3
S1	E0.1	PULSANTE 1
S2	E0.2	PULSANTE 2
S3	E0.3	PULSANTE 3

Network 8



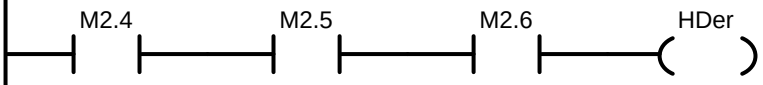
Símbolo	Dirección	Comentario
PCS1	M0.1	PULSANTE VIRTUAL 1
PCS2	M0.2	PULSANTE VIRTUAL 2
PCS3	M0.3	PULSANTE VIRTUAL 3
S1	E0.1	PULSANTE 1
S2	E0.2	PULSANTE 2
S3	E0.3	PULSANTE 3

Network 9 activaciones a q0.1



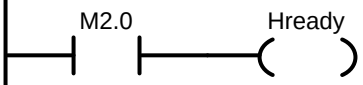
Símbolo	Dirección	Comentario
HIzq	A0.1	ACTUADOR IZQ

Network 10 activaciones a q0.2



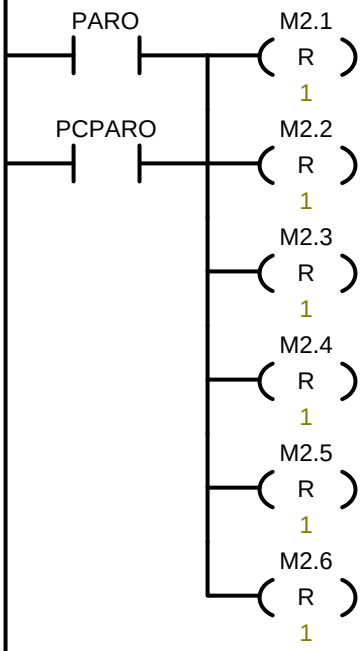
Símbolo	Dirección	Comentario
HDer	A0.2	ACTUADOR DER

Network 11 SISTEMA LISTO



Símbolo	Dirección	Comentario
Hready	A0.3	SISTEMA LISTO

Network 12 PARO TOTAL

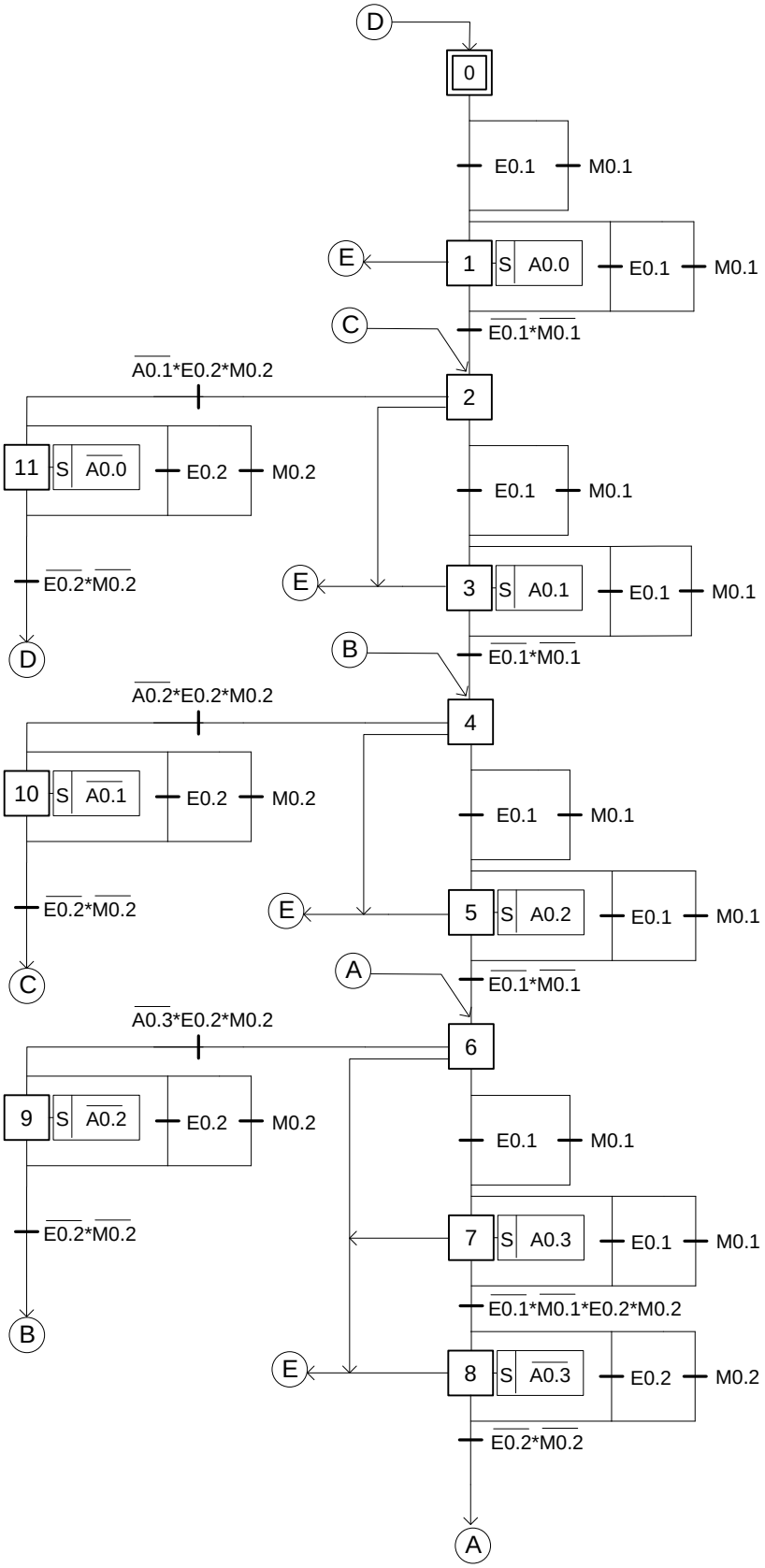


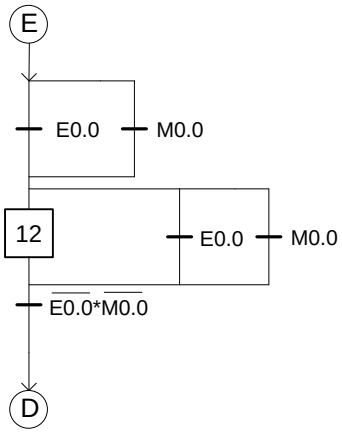
Símbolo	Dirección	Comentario
PARO	E0.4	PARO TOTAL
PCPARO	M0.4	PARO VIRTUAL

Símbolo	Dirección	Comentario
S1	E0.1	PULSANTE 1
S2	E0.2	PULSANTE 2
S3	E0.3	PULSANTE 3
PARO	E0.4	PARO TOTAL
HIzq	A0.1	ACTUADOR IZQ
HDer	A0.2	ACTUADOR DER
Hready	A0.3	SISTEMA LISTO
PCS1	M0.1	PULSANTE VIRTUAL 1
PCS2	M0.2	PULSANTE VIRTUAL 2
PCS3	M0.3	PULSANTE VIRTUAL 3
PCPARO	M0.4	PARO VIRTUAL
COM	M0.0	PC <----> PLC

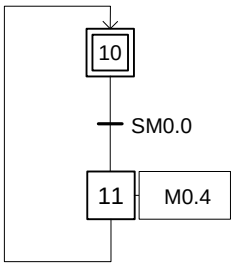
Símbolo	Dirección	Comentario
PRINCIPAL	OB1	PROGRAMA DE INVERSION CONDICIONADA

Programa principal:





Comunicación:

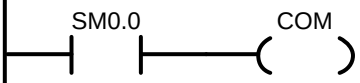


Bloque: PRINCIPAL
Autor:
Fecha de creación: 11.05.2009 9:07:17
Fecha de modificación: 30.05.2010 21:21:08

Símbolo	Tipo var.	Tipo de datos	Comentario
	TEMP		
	TEMP		
	TEMP		
	TEMP		

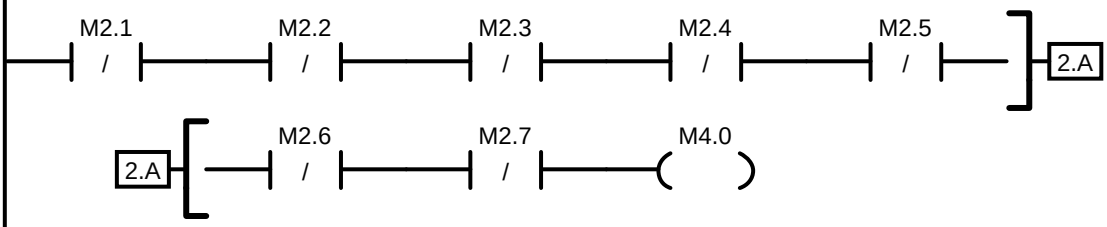
COMENTARIOS DEL PROGRAMA

Network 1 COM PC <----> PLC

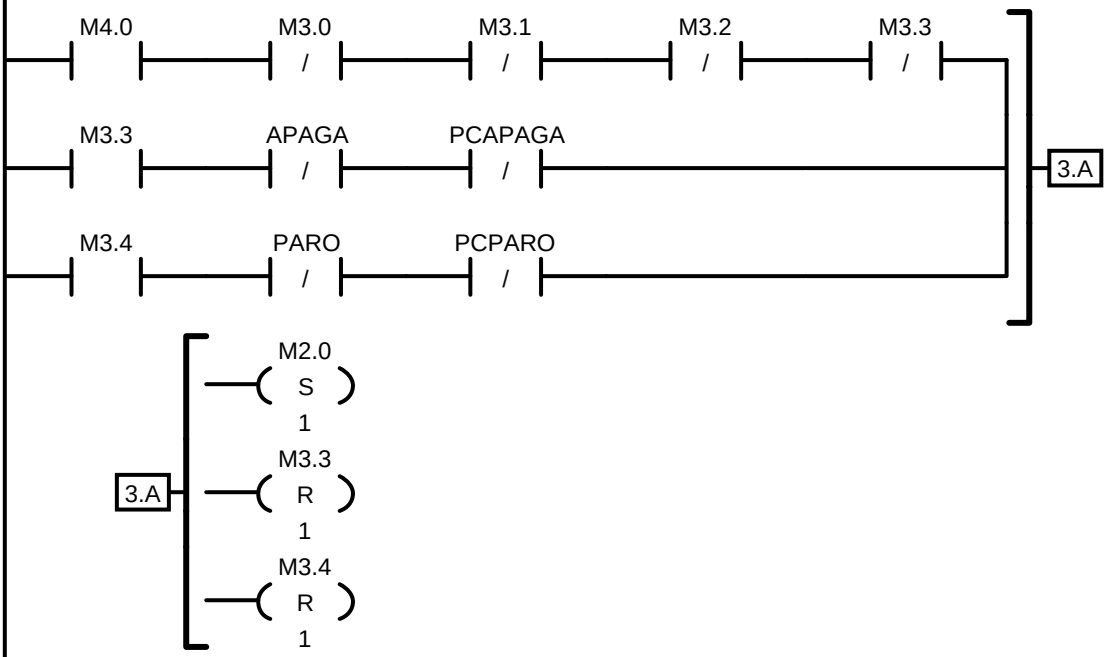


Símbolo	Dirección	Comentario
COM	M0.4	PC <----> PLC

Network 2 BLOQUE DE CONTROL

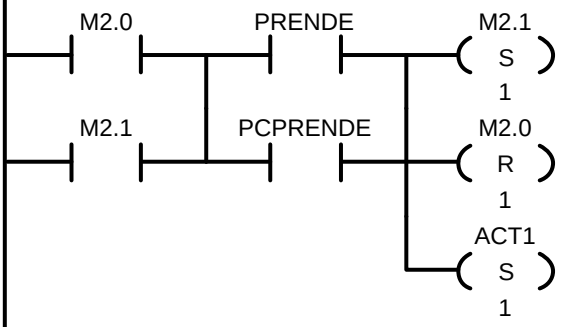


Network 3



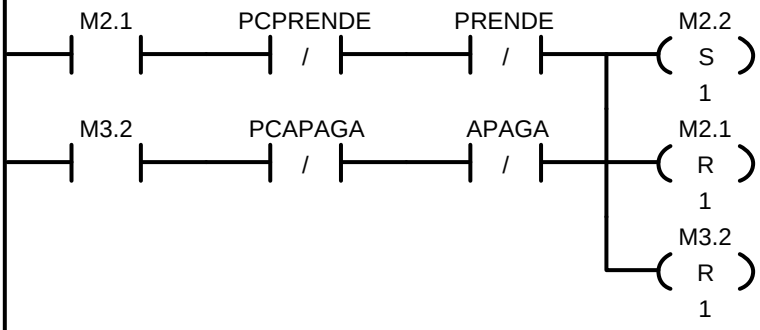
Símbolo	Dirección	Comentario
APAGA	E0.2	DESDE PLC
PARO	E0.0	DESDE PLC
PCAPAGA	M0.2	DESDE PC
PCPARO	M0.0	DESDE PC

Network 4



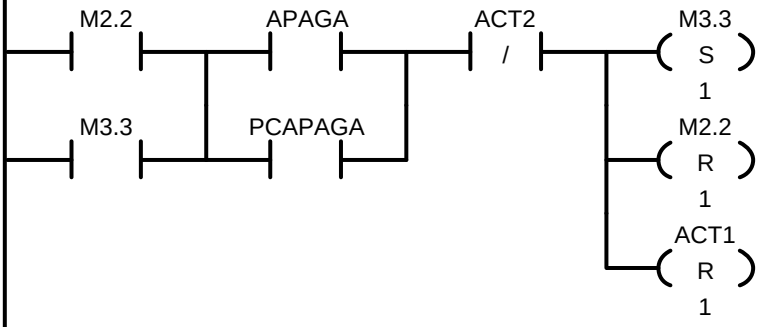
Símbolo	Dirección	Comentario
ACT1	A0.0	
PCPRENDE	M0.1	DESDE PC
PRENDE	E0.1	DESDE PLC

Network 5



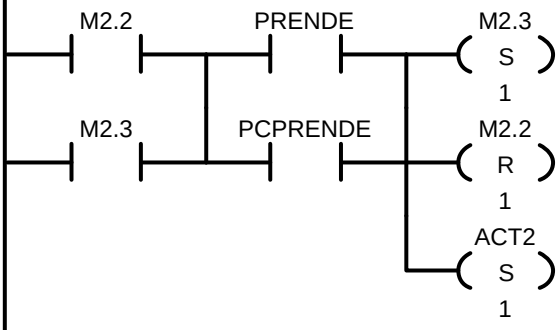
Símbolo	Dirección	Comentario
APAGA	E0.2	DESDE PLC
PCAPAGA	M0.2	DESDE PC
PCPRENDE	M0.1	DESDE PC
PRENDE	E0.1	DESDE PLC

Network 6



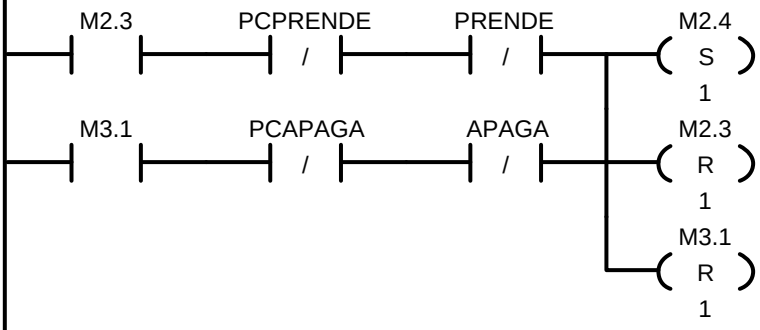
Símbolo	Dirección	Comentario
ACT1	A0.0	
ACT2	A0.1	
APAGA	E0.2	DESDE PLC
PCAPAGA	M0.2	DESDE PC

Network 7



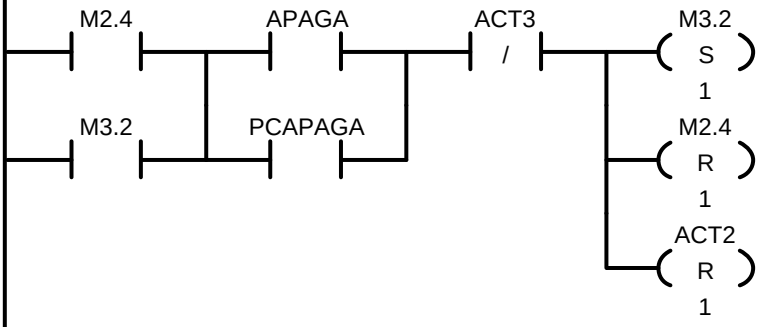
Símbolo	Dirección	Comentario
ACT2	A0.1	
PCPRENDE	M0.1	DESDE PC
PRENDE	E0.1	DESDE PLC

Network 8



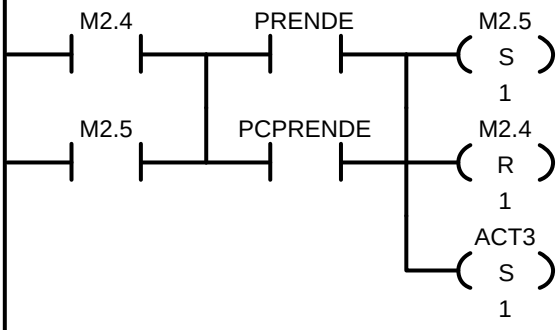
Símbolo	Dirección	Comentario
APAGA	E0.2	DESDE PLC
PCAPAGA	M0.2	DESDE PC
PCPRENDE	M0.1	DESDE PC
PRENDE	E0.1	DESDE PLC

Network 9



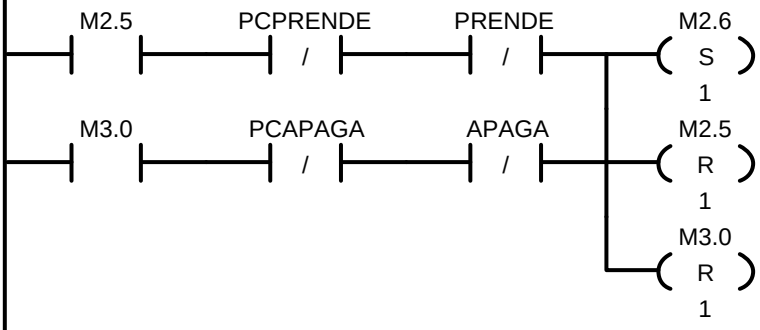
Símbolo	Dirección	Comentario
ACT2	A0.1	
ACT3	A0.2	
APAGA	E0.2	DESDE PLC
PCAPAGA	M0.2	DESDE PC

Network 10



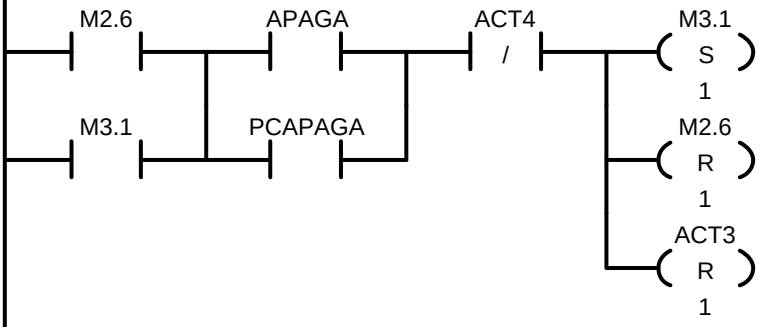
Símbolo	Dirección	Comentario
ACT3	A0.2	
PCPRENDE	M0.1	DESDE PC
PRENDE	E0.1	DESDE PLC

Network 11



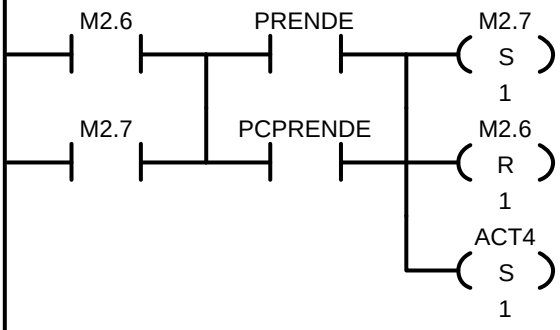
Símbolo	Dirección	Comentario
APAGA	E0.2	DESDE PLC
PCAPAGA	M0.2	DESDE PC
PCPRENDE	M0.1	DESDE PC
PRENDE	E0.1	DESDE PLC

Network 12



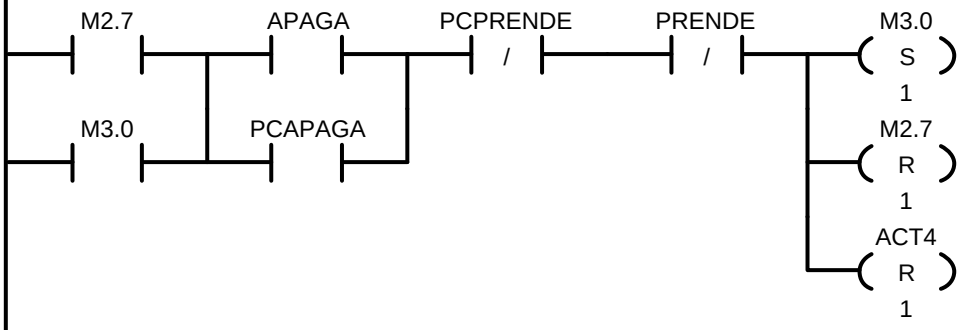
Símbolo	Dirección	Comentario
ACT3	A0.2	
ACT4	A0.3	
APAGA	E0.2	DESDE PLC
PCAPAGA	M0.2	DESDE PC

Network 13



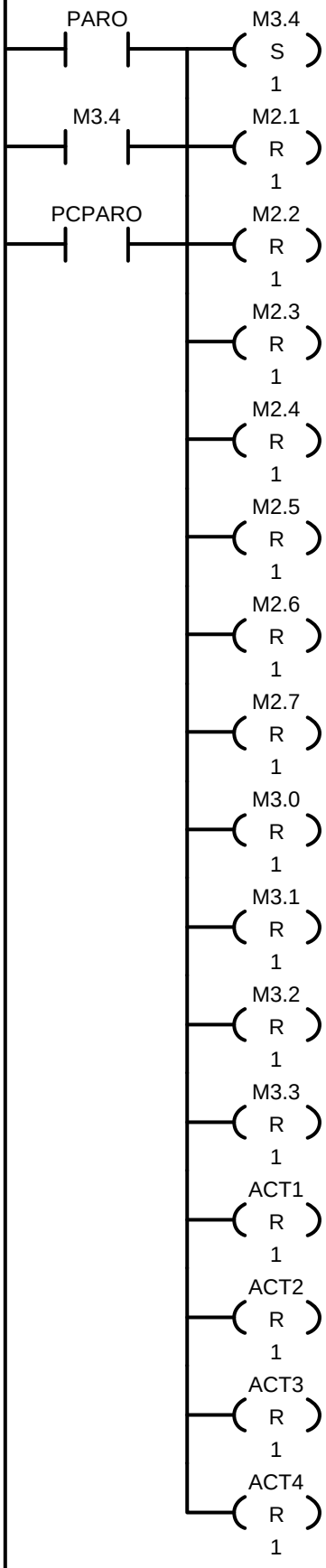
Símbolo	Dirección	Comentario
ACT4	A0.3	
PCPRENDE	M0.1	DESDE PC
PRENDE	E0.1	DESDE PLC

Network 14



Símbolo	Dirección	Comentario
ACT4	A0.3	
APAGA	E0.2	DESDE PLC
PCAPAGA	M0.2	DESDE PC
PCPRENDE	M0.1	DESDE PC
PRENDE	E0.1	DESDE PLC

Network 15 PARO TOTAL



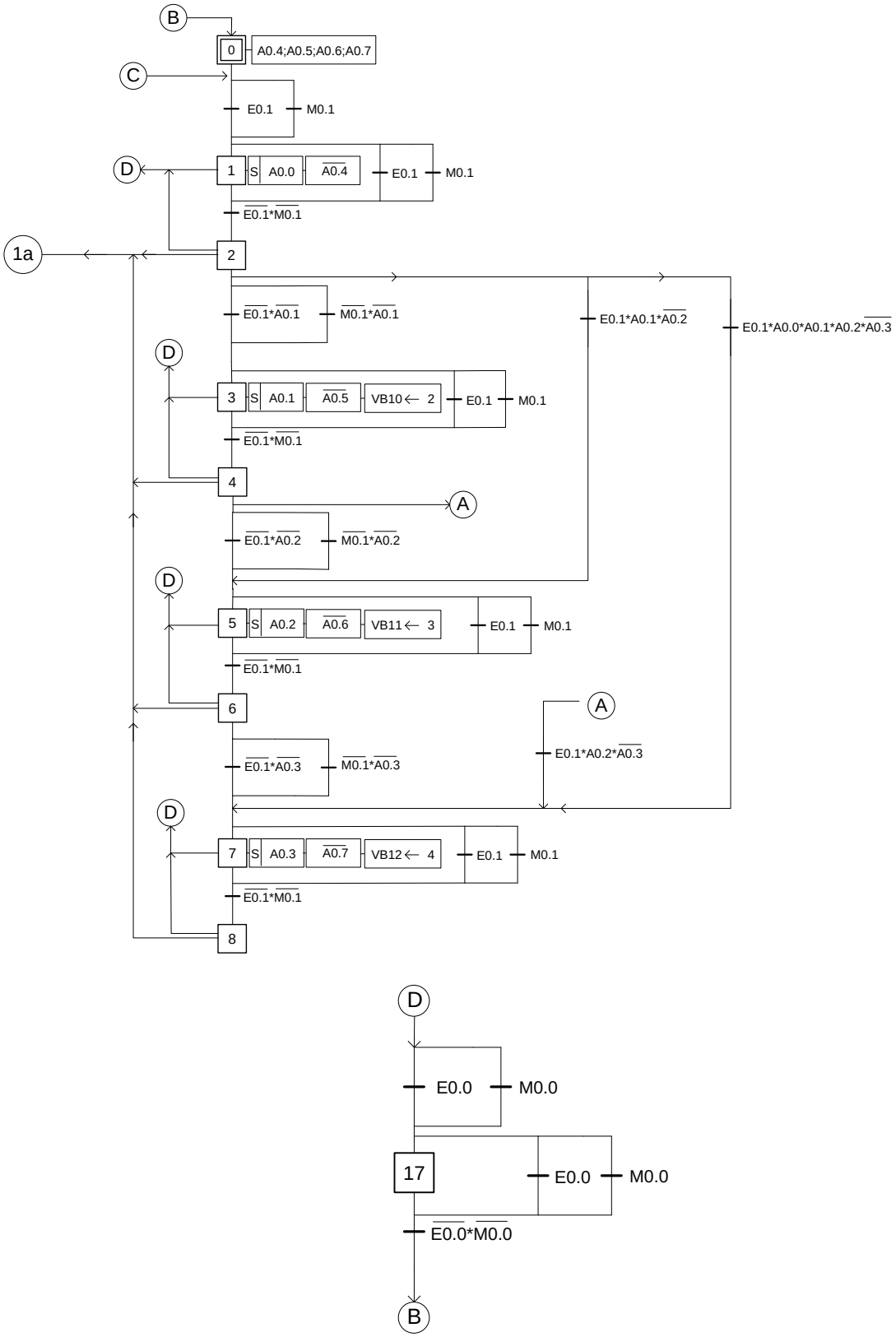
Símbolo	Dirección	Comentario
ACT1	A0.0	
ACT2	A0.1	

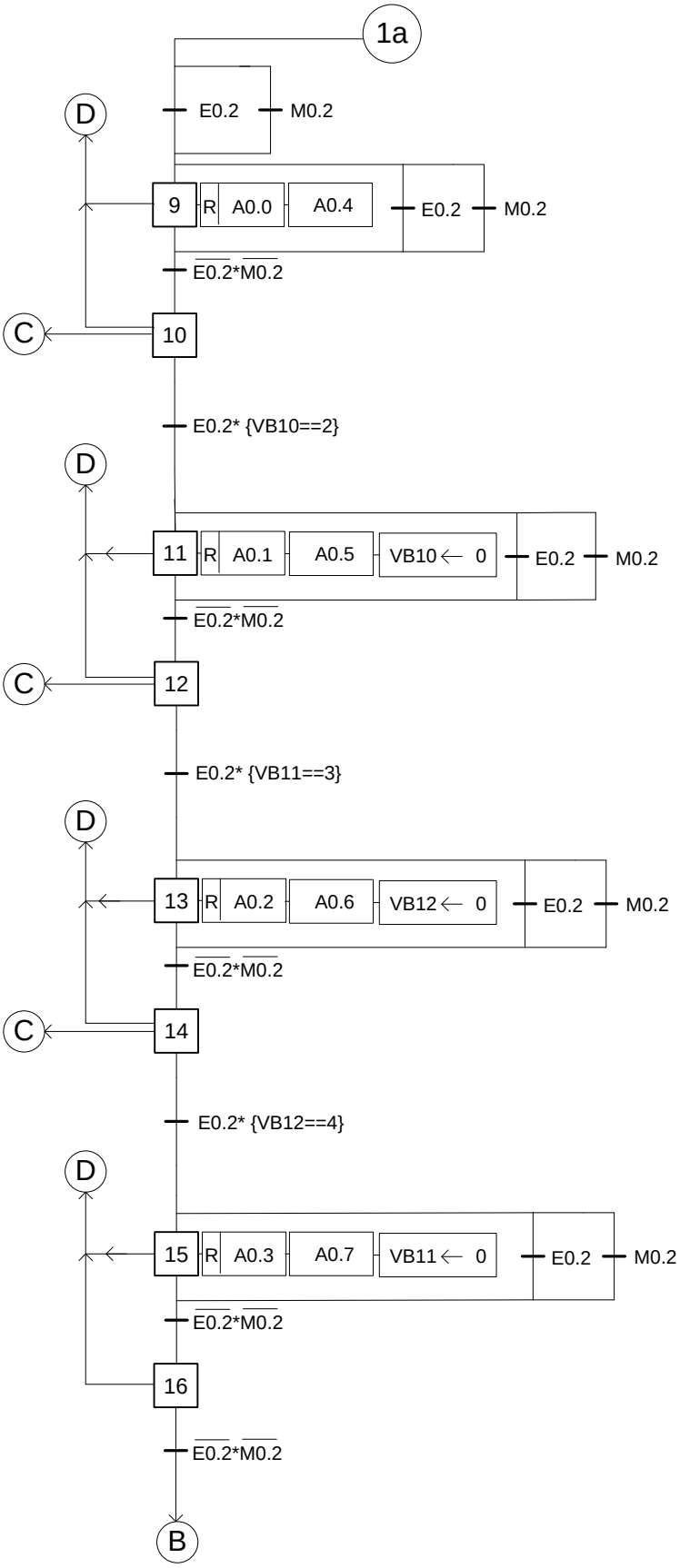
ACT3	A0.2	
ACT4	A0.3	
PARO	E0.0	DESDE PLC
PCPARO	M0.0	DESDE PC

Símbolo	Dirección	Comentario
PCPARO	M0.0	DESDE PC
PCPRENDE	M0.1	DESDE PC
PCAPAGA	M0.2	DESDE PC
COM	M0.4	PC <----> PLC
PRENDE	E0.1	DESDE PLC
APAGA	E0.2	DESDE PLC
PARO	E0.0	DESDE PLC
ACT1	A0.0	
ACT2	A0.1	
ACT3	A0.2	
ACT4	A0.3	

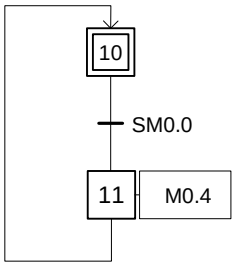
Símbolo	Dirección	Comentario
PRINCIPAL	OB1	COMENTARIOS DEL PROGRAMA

Programa principal:




1a
D
E0.2
M0.2
9
R
A0.0
A0.4
E0.2
M0.2
E0.2*M0.2
C
10
E0.2* {VB10==2}
D
11
R
A0.1
A0.5
VB10 ← 0
E0.2
M0.2
E0.2*M0.2
C
12
E0.2* {VB11==3}
D
13
R
A0.2
A0.6
VB12 ← 0
E0.2
M0.2
E0.2*M0.2
C
14
E0.2* {VB12==4}
D
15
R
A0.3
A0.7
VB11 ← 0
E0.2
M0.2
E0.2*M0.2
16
E0.2*M0.2
B

Comunicación:

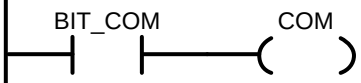


Bloque: PRINCIPAL
Autor:
Fecha de creación: 14.05.2009 14:41:43
Fecha de modificación: 14.05.2009 19:25:46

Símbolo	Tipo var.	Tipo de datos	Comentario
	TEMP		
	TEMP		
	TEMP		
	TEMP		

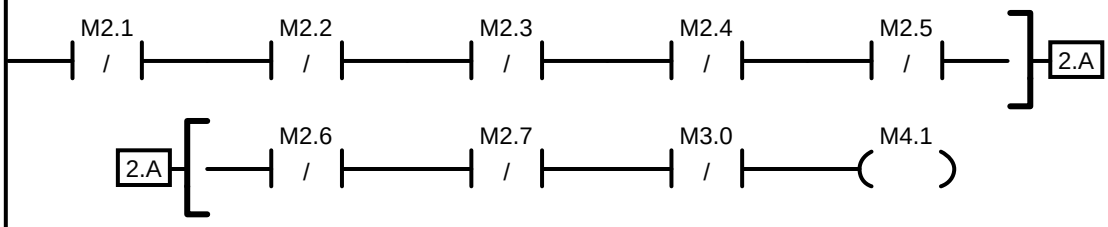
COMENTARIOS DEL PROGRAMA

Network 1 COMUNICACION PC <-----> PLC

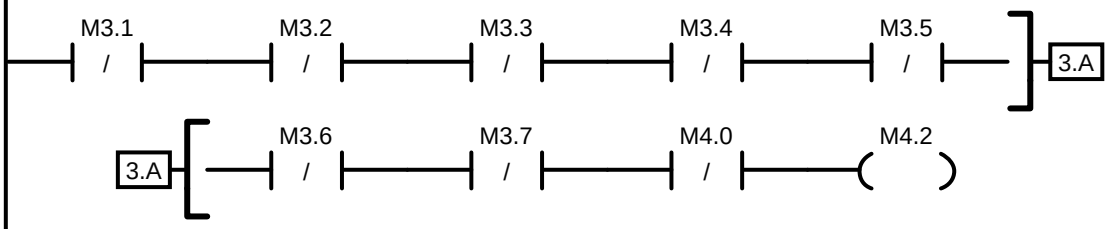


Símbolo	Dirección	Comentario
BIT_COM	SM0.0	
COM	M0.4	

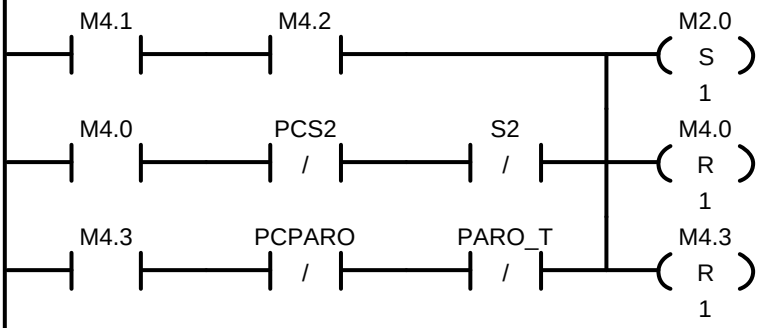
Network 2 BLOQUE DE CONTROL



Network 3



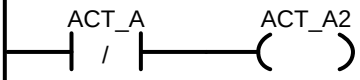
Network 4



Símbolo	Dirección	Comentario
PARO_T	E0.0	
PCPARO	M0.0	
PCS2	M0.2	
S2	E0.2	

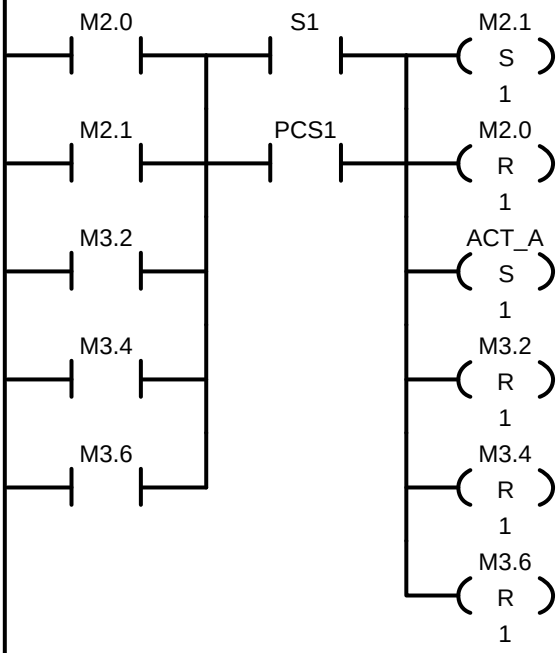
Network 5

ACTUADOR 1 EN REPOSO



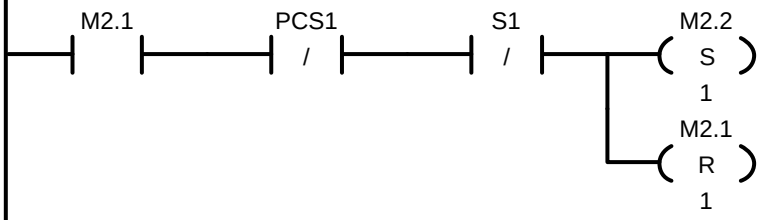
Símbolo	Dirección	Comentario
ACT_A	A0.0	
ACT_A2	A0.4	

Network 6



Símbolo	Dirección	Comentario
ACT_A	A0.0	
PCS1	M0.1	
S1	E0.1	

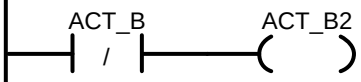
Network 7



Símbolo	Dirección	Comentario
PCS1	M0.1	
S1	E0.1	

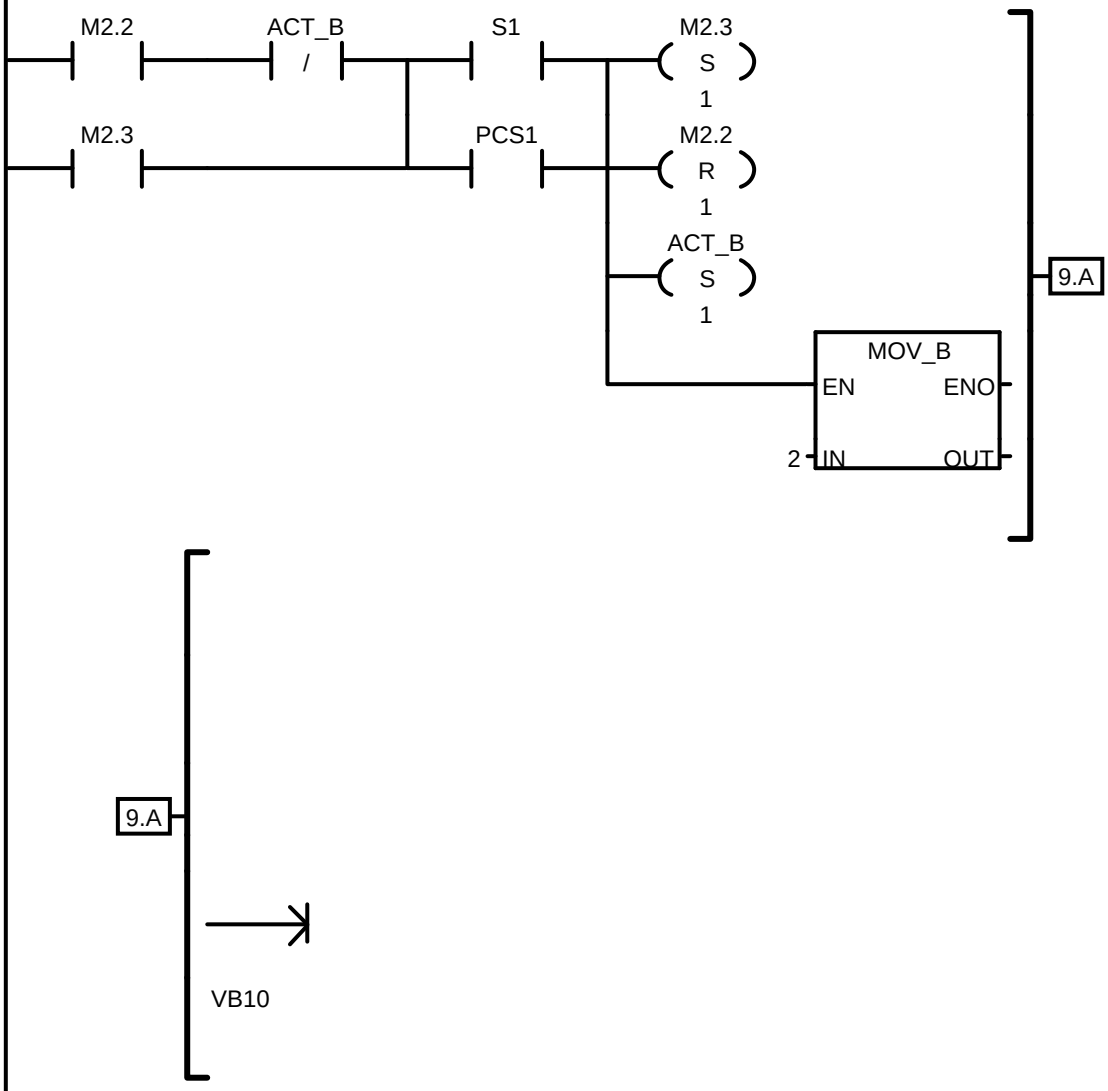
Network 8

ACTUADOR 2 EN REPOSO



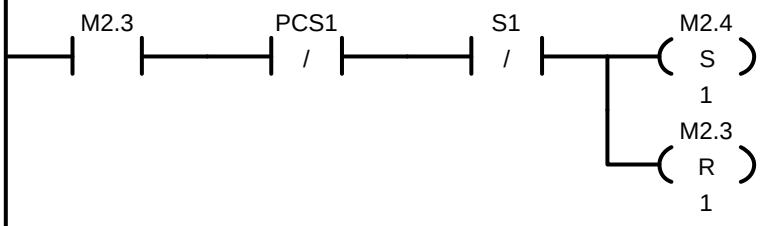
Símbolo	Dirección	Comentario
ACT_B	A0.1	
ACT_B2	A0.5	

Network 9



Símbolo	Dirección	Comentario
ACT_B	A0.1	
PCS1	M0.1	
S1	E0.1	

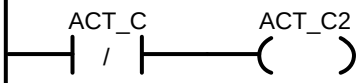
Network 10



Símbolo	Dirección	Comentario
PCS1	M0.1	
S1	E0.1	

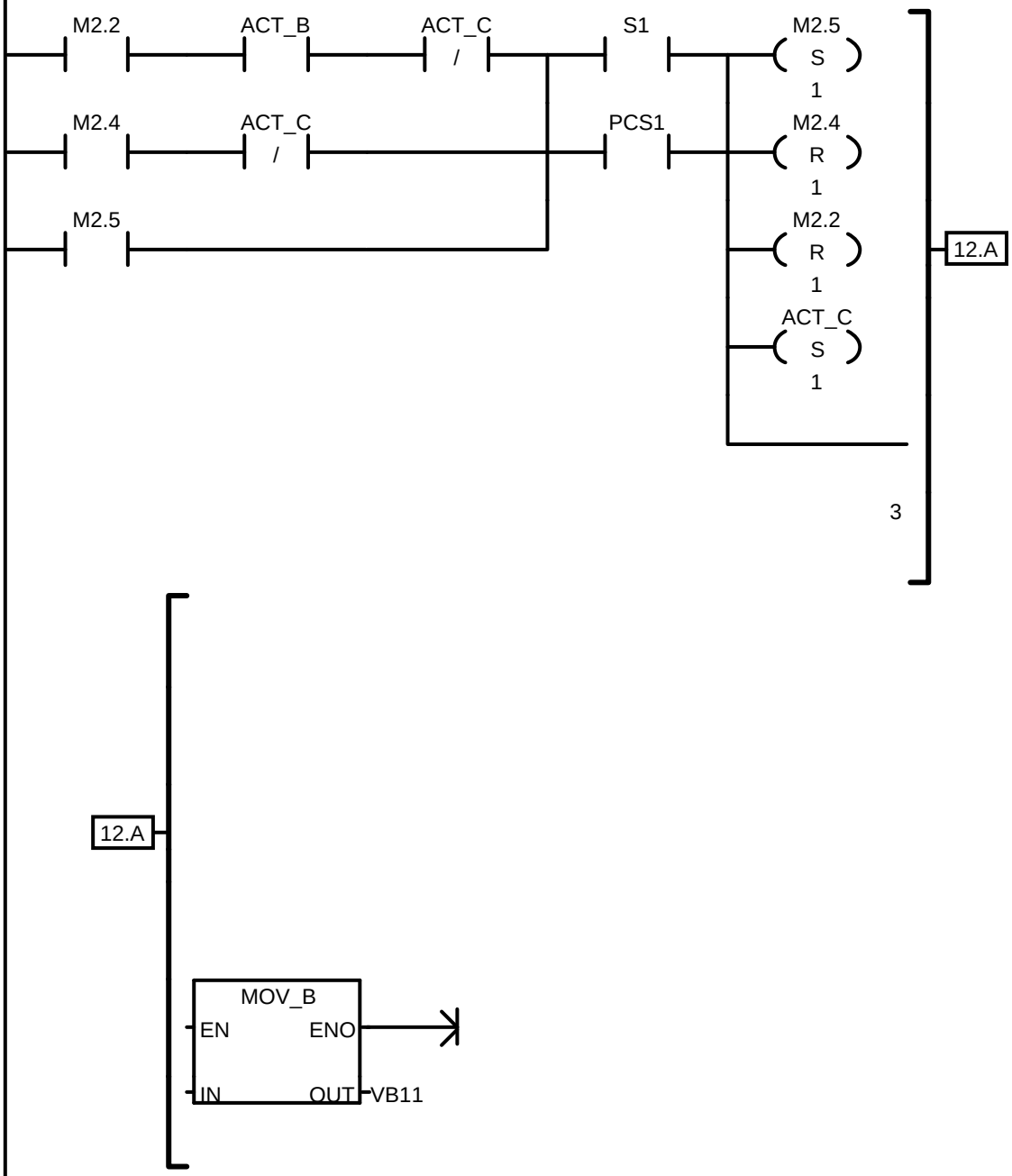
Network 11

ACTUADOR 3 EN REPOSO



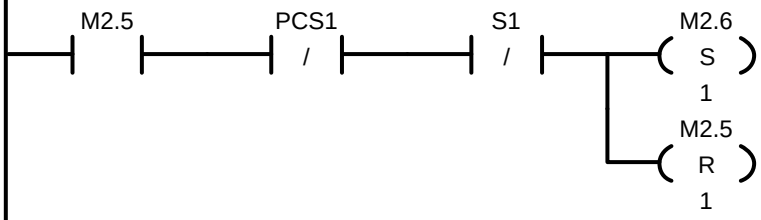
Símbolo	Dirección	Comentario
ACT_C	A0.2	
ACT_C2	A0.6	

Network 12



Símbolo	Dirección	Comentario
ACT_B	A0.1	
ACT_C	A0.2	
PCS1	M0.1	
S1	E0.1	

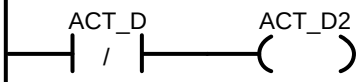
Network 13



Símbolo	Dirección	Comentario
PCS1	M0.1	
S1	E0.1	

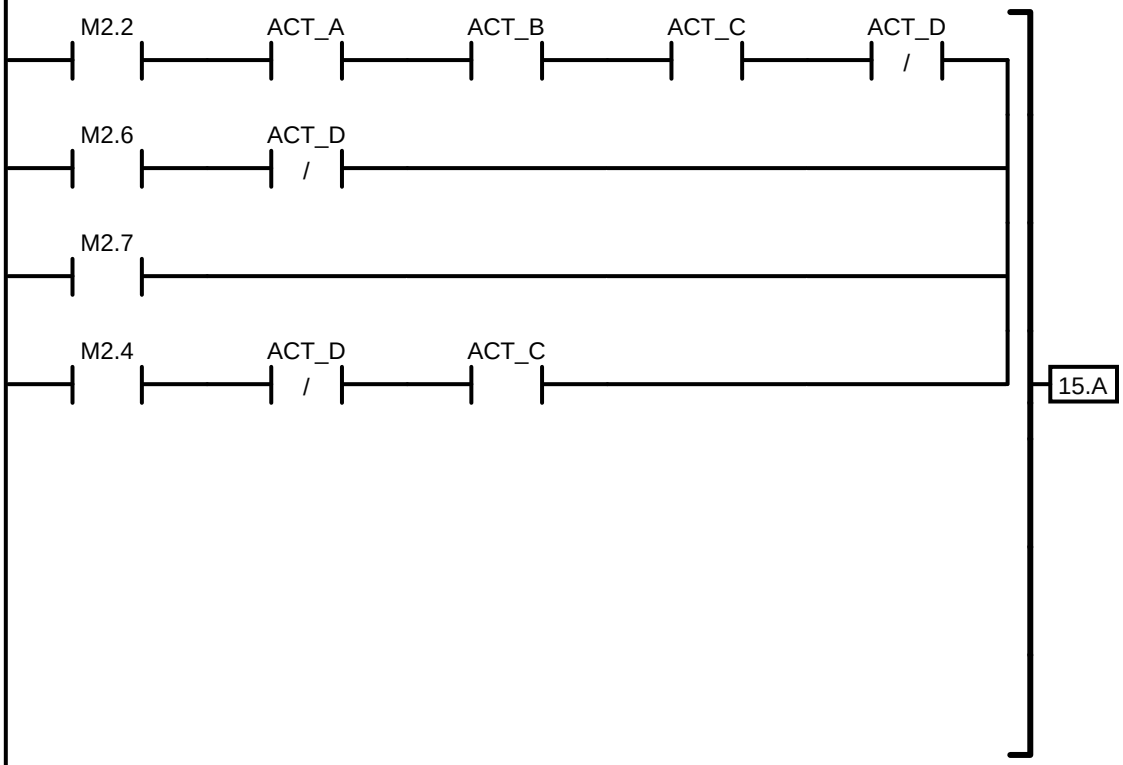
Network 14

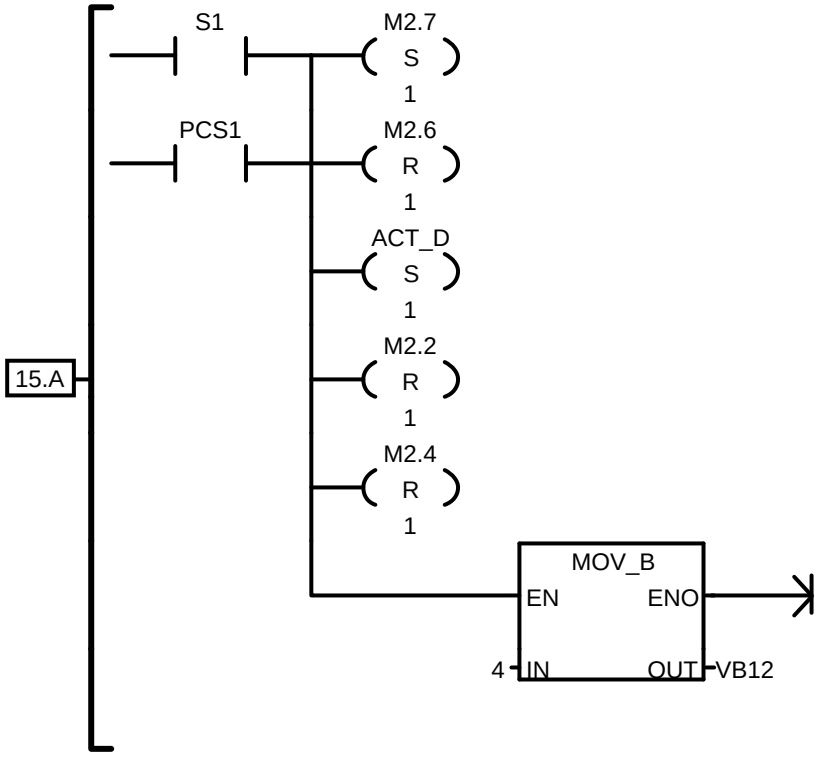
ACTUADOR 4 EN REPOSO



Símbolo	Dirección	Comentario
ACT_D	A0.3	
ACT_D2	A0.7	

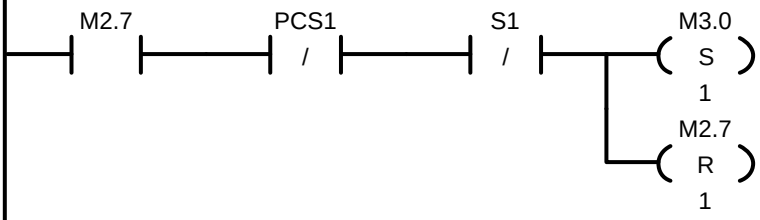
Network 15





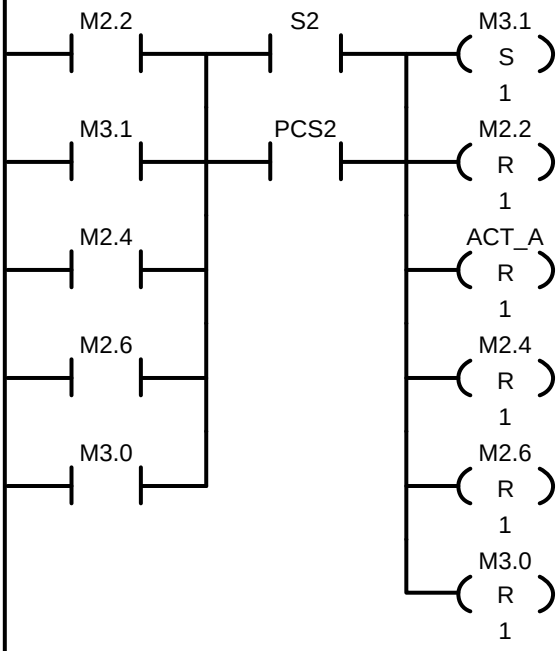
Símbolo	Dirección	Comentario
ACT_A	A0.0	
ACT_B	A0.1	
ACT_C	A0.2	
ACT_D	A0.3	
PCS1	M0.1	
S1	E0.1	

Network 16



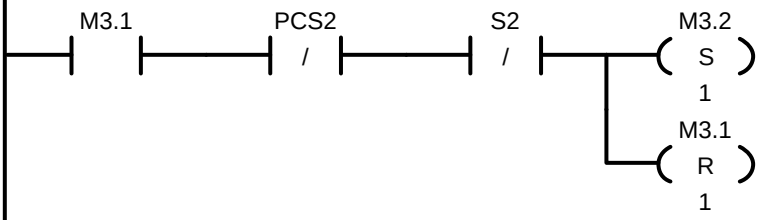
Símbolo	Dirección	Comentario
PCS1	M0.1	
S1	E0.1	

Network 17



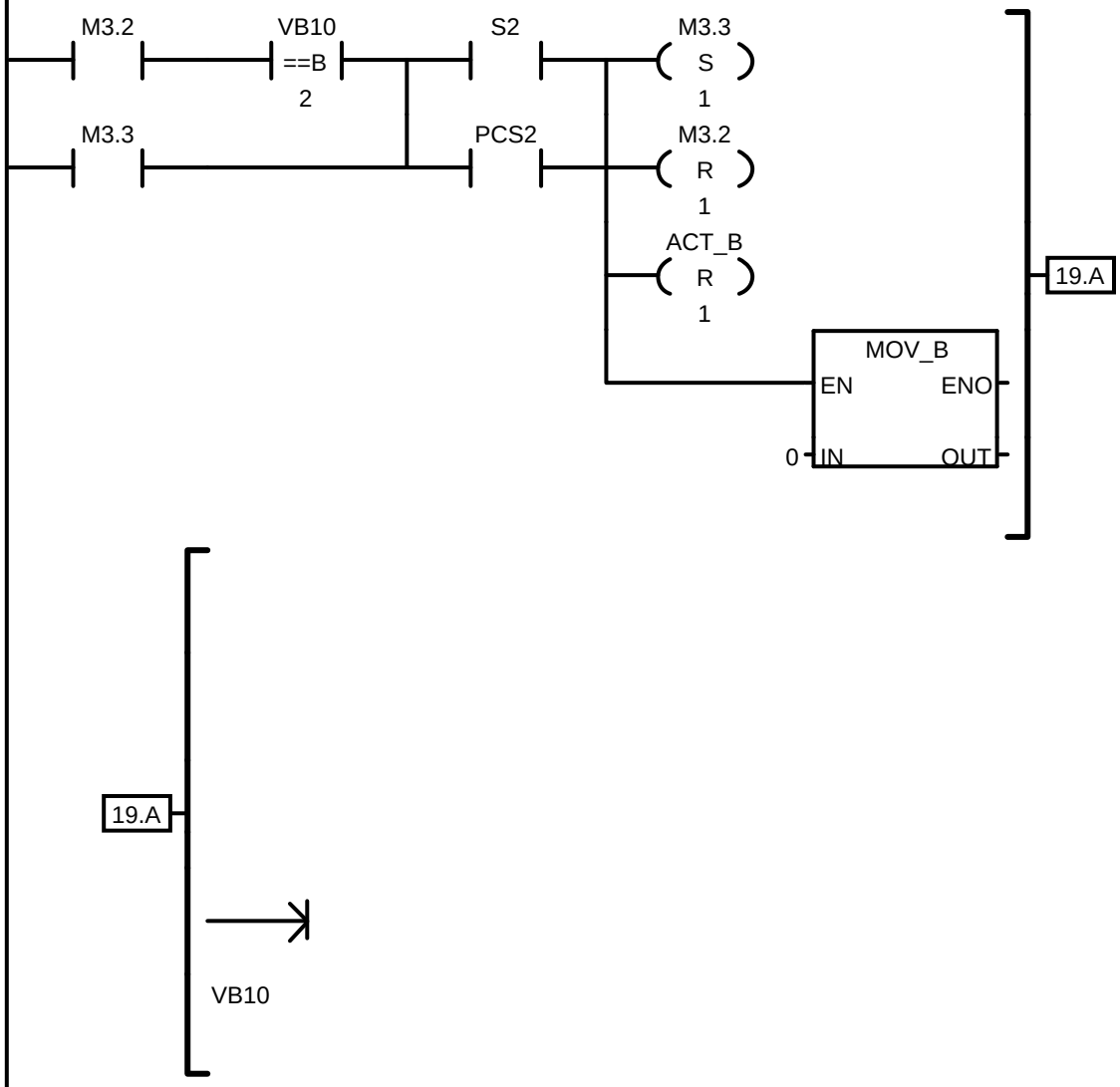
Símbolo	Dirección	Comentario
ACT_A	A0.0	
PCS2	M0.2	
S2	E0.2	

Network 18



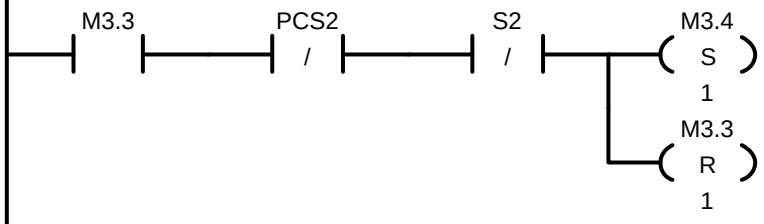
Símbolo	Dirección	Comentario
PCS2	M0.2	
S2	E0.2	

Network 19



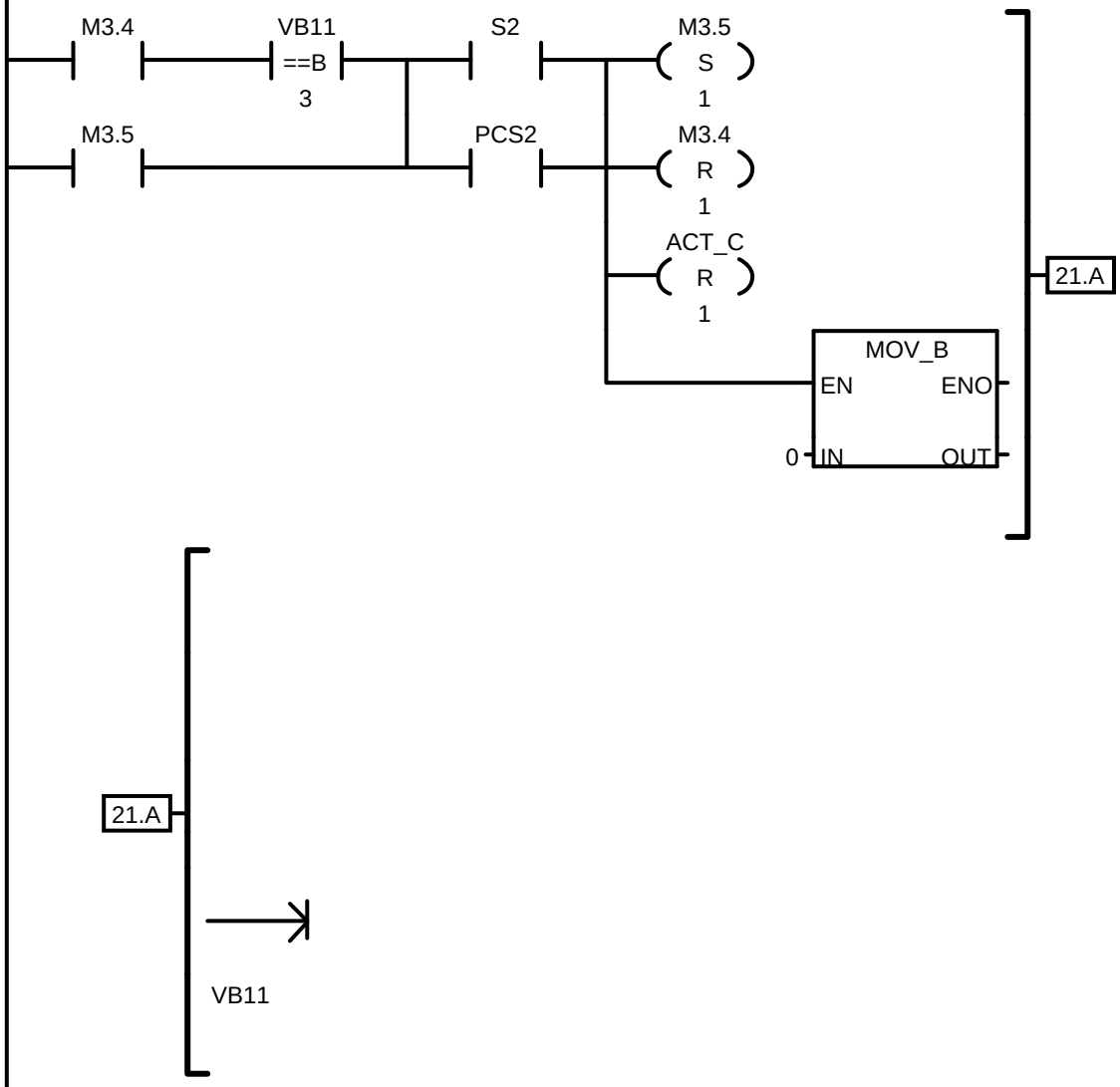
Símbolo	Dirección	Comentario
ACT_B	A0.1	
PCS2	M0.2	
S2	E0.2	

Network 20



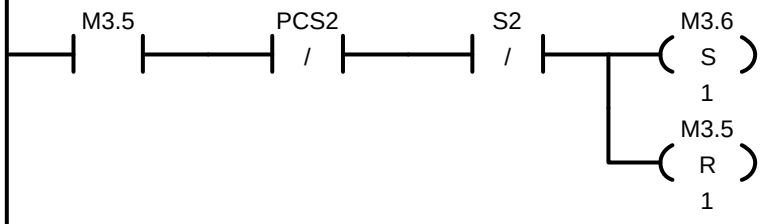
Símbolo	Dirección	Comentario
PCS2	M0.2	
S2	E0.2	

Network 21



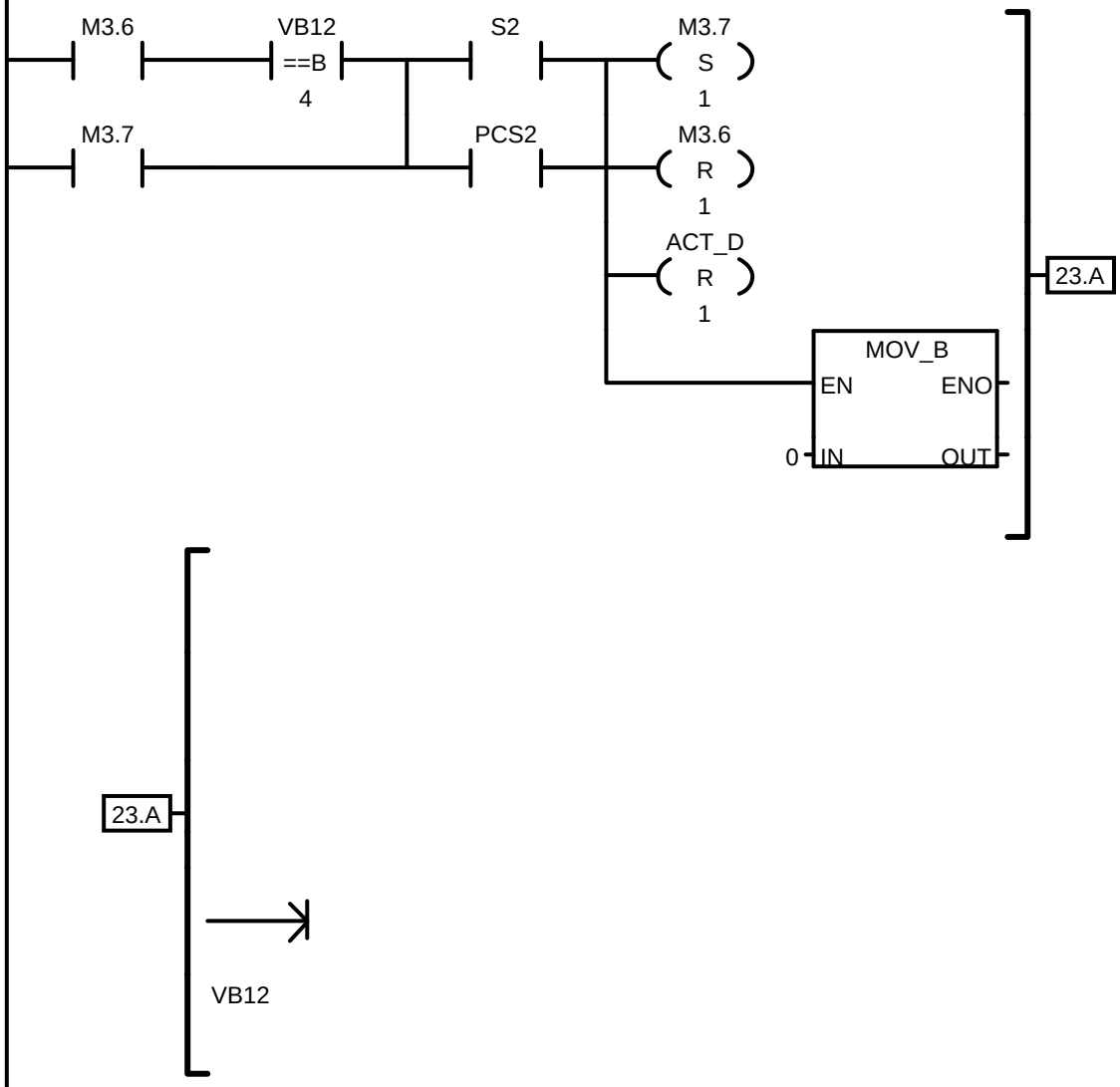
Símbolo	Dirección	Comentario
ACT_C	A0.2	
PCS2	M0.2	
S2	E0.2	

Network 22



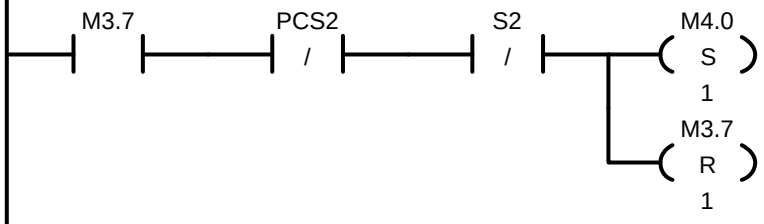
Símbolo	Dirección	Comentario
PCS2	M0.2	
S2	E0.2	

Network 23



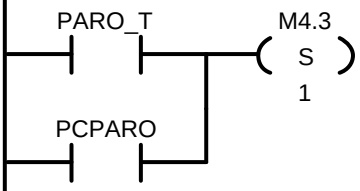
Símbolo	Dirección	Comentario
ACT_D	A0.3	
PCS2	M0.2	
S2	E0.2	

Network 24



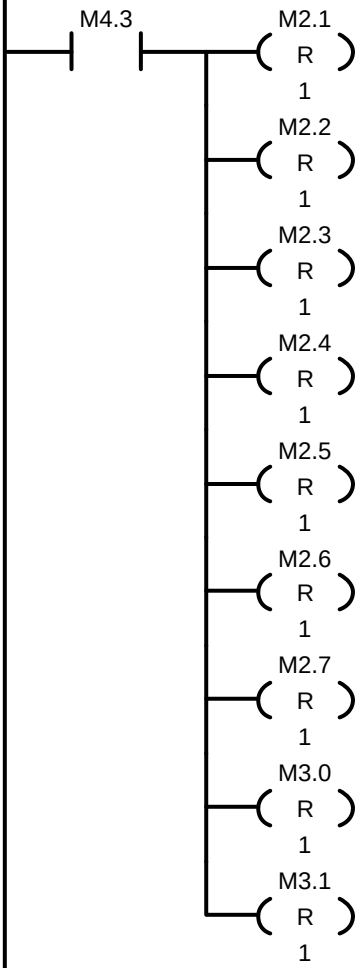
Símbolo	Dirección	Comentario
PCS2	M0.2	
S2	E0.2	

Network 25 ACCION DEL PARO TOTAL

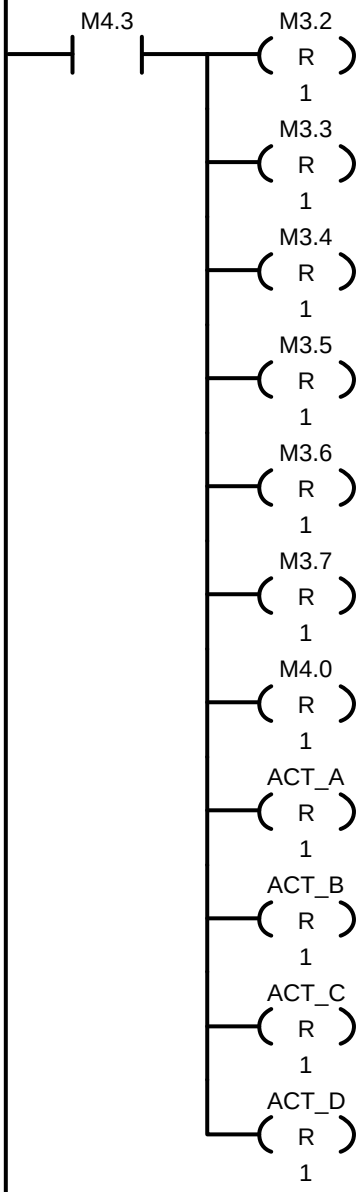


Símbolo	Dirección	Comentario
PARO_T	E0.0	
PCPARO	M0.0	

Network 26



Network 27

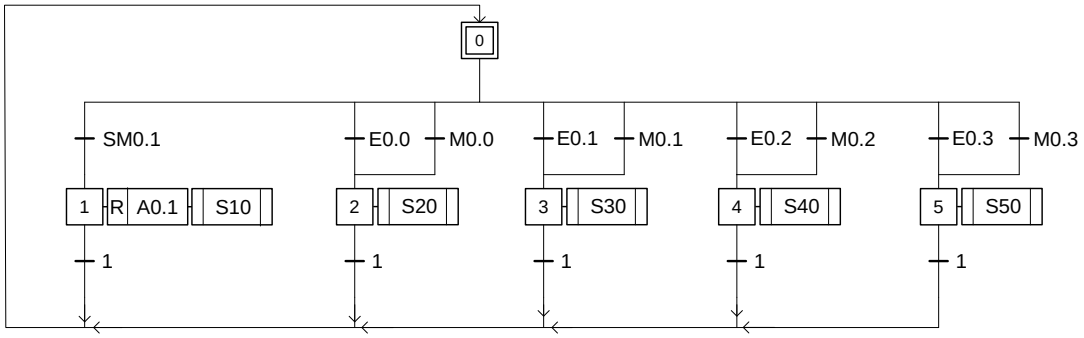


Símbolo	Dirección	Comentario
ACT_A	A0.0	
ACT_B	A0.1	
ACT_C	A0.2	
ACT_D	A0.3	

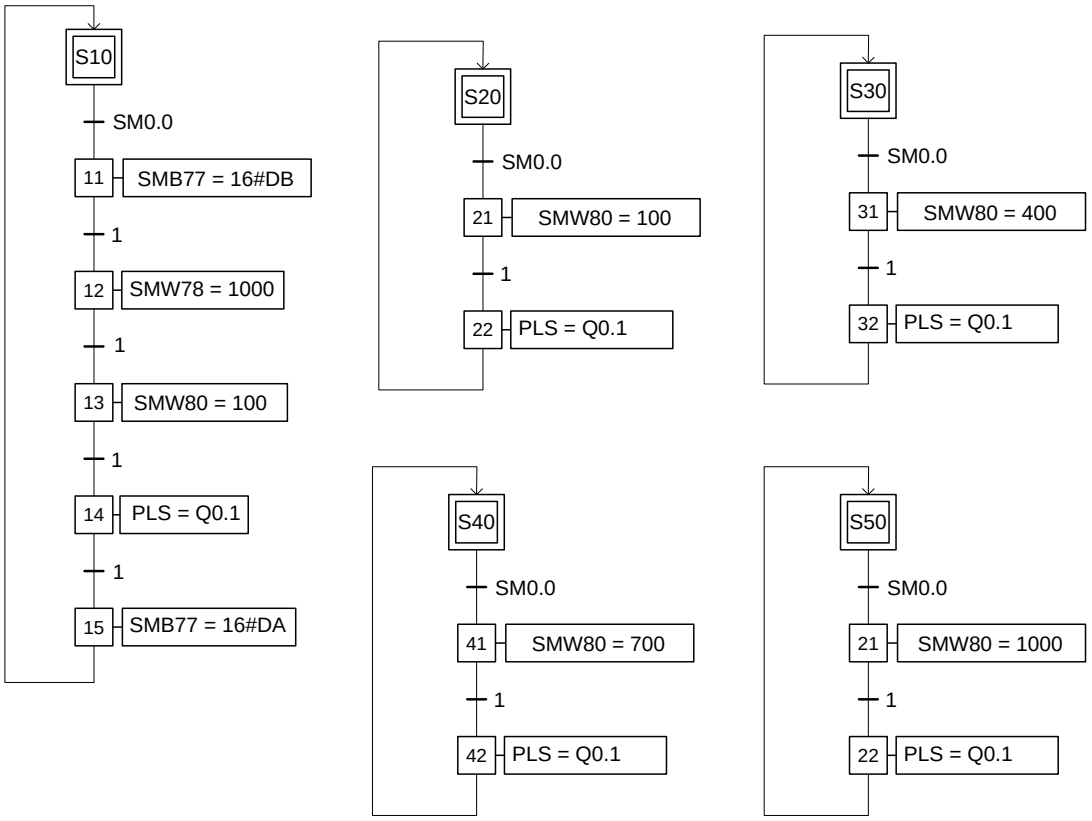
Símbolo	Dirección	Comentario
ACT_A	A0.0	
ACT_B	A0.1	
ACT_C	A0.2	
ACT_D	A0.3	
PARO_T	E0.0	
S1	E0.1	
S2	E0.2	
COM	M0.4	
PCS1	M0.1	
PCS2	M0.2	
PCPARO	M0.0	
BIT_COM	SM0.0	
ACT_A2	A0.4	
ACT_B2	A0.5	
ACT_C2	A0.6	
ACT_D2	A0.7	

Símbolo	Dirección	Comentario
PRINCIPAL	OB1	COMENTARIOS DEL PROGRAMA

Programa principal:



Subrutinas



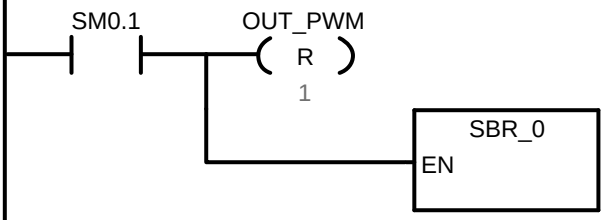
Bloque: PRINCIPAL
Autor:
Fecha de creación: 23.05.2009 14:08:11
Fecha de modificación: 31.05.2009 20:13:36

Símbolo	Tipo var.	Tipo de datos	Comentario
	TEMP		
	TEMP		
	TEMP		
	TEMP		

PROGRAMA PARA GENERAR UN PWM

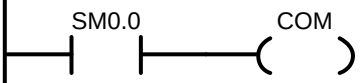
Network 1

Reseteo de la salida a emplear, y llamado a la subrutina de inicialización.



Símbolo	Dirección	Comentario
OUT_PWM	A0.1	SALIDA PWM

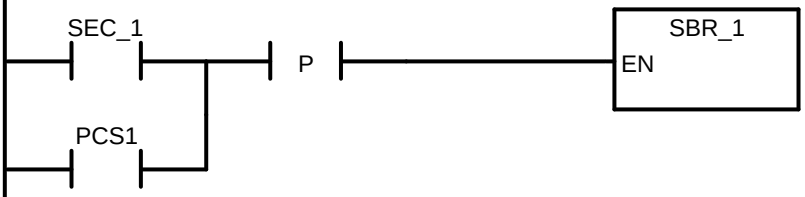
Network 2 COMUNICACION CON PC



Símbolo	Dirección	Comentario
COM	M0.0	PC <----> PLC

Network 3

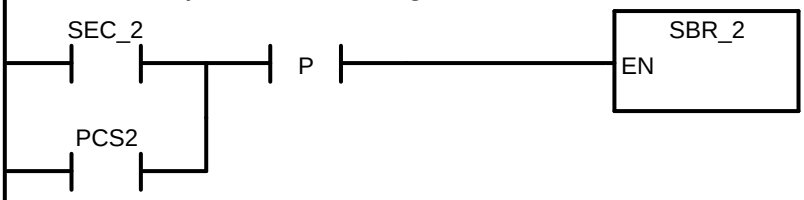
Secuencia de Ajuste 1 --> T = 0,1 seg.



Símbolo	Dirección	Comentario
PCS1	M0.1	
SEC_1	E0.0	ACTIVA SECUENCIA DE T=100ms

Network 4

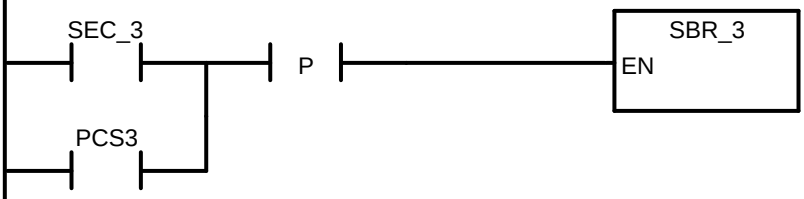
Secuencia de Ajuste 2 --> T = 0,4 seg.



Símbolo	Dirección	Comentario
PCS2	M0.2	
SEC_2	E0.1	ACTIVA SECUENCIA DE T=400ms

Network 5

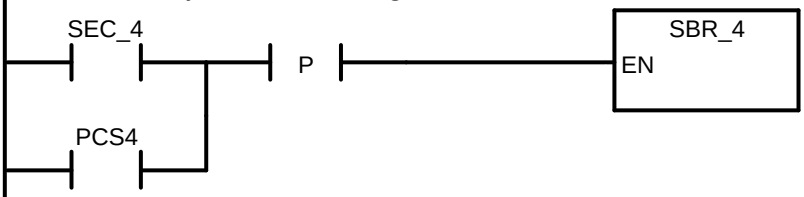
Secuencia de Ajuste 3 --> T = 0,7 seg.



Símbolo	Dirección	Comentario
PCS3	M0.3	
SEC_3	E0.2	ACTIVA SECUENCIA DE T=700ms

Network 6

Secuencia de Ajuste 4 --> T = 1 seg.



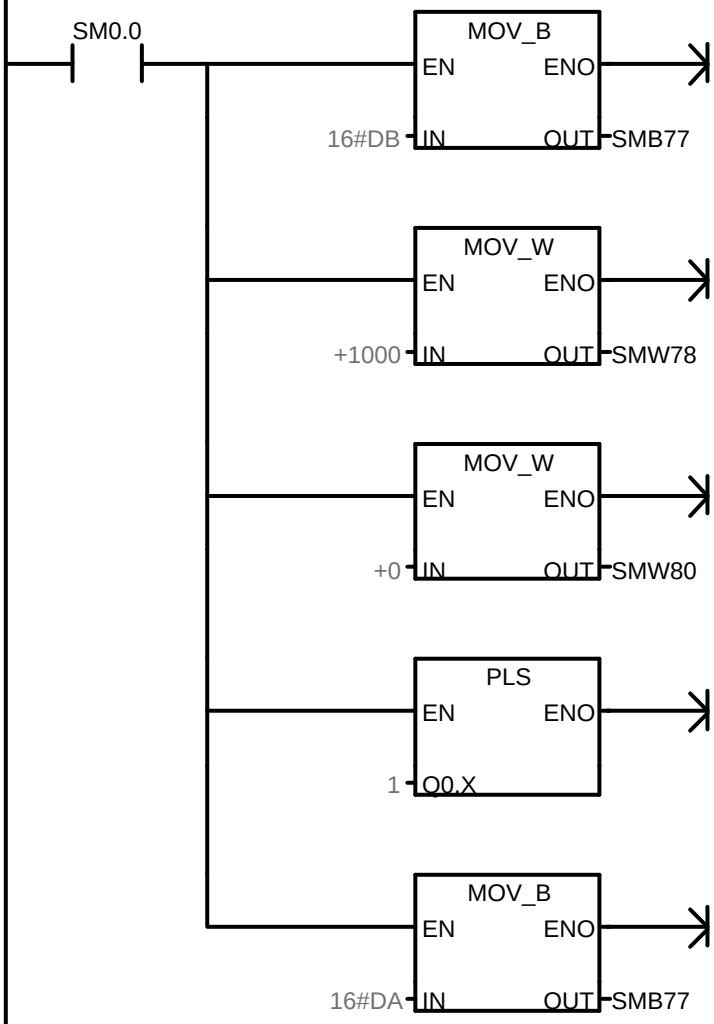
Símbolo	Dirección	Comentario
PCS4	M0.4	
SEC_4	E0.3	ACTIVA SECUENCIA DE T=1000ms

Bloque: SBR_0
Autor:
Fecha de creación: 23.05.2009 14:08:11
Fecha de modificación: 23.05.2009 15:29:19

Símbolo	Tipo var.	Tipo de datos	Comentario
EN	IN	BOOL	
	IN		
	IN_OUT		
	OUT		
	TEMP		

INICIALIZACION DE PARAMETROS

Network 1

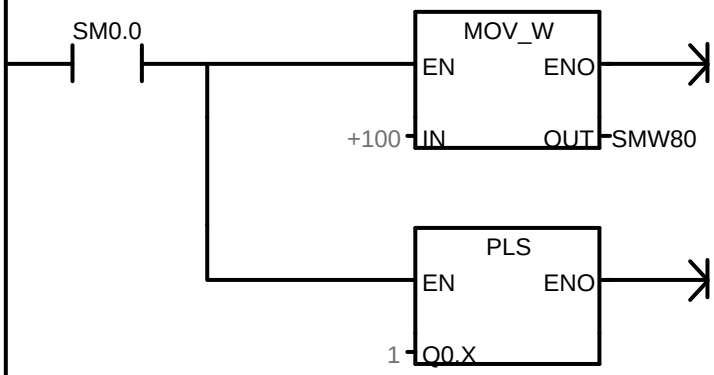


Bloque: SBR_1
Autor:
Fecha de creación: 23.05.2009 14:10:31
Fecha de modificación: 23.05.2009 15:31:19

Símbolo	Tipo var.	Tipo de datos	Comentario
EN	IN	BOOL	
	IN		
	IN_OUT		
	OUT		
	TEMP		

TIEMPO 1

Network 1

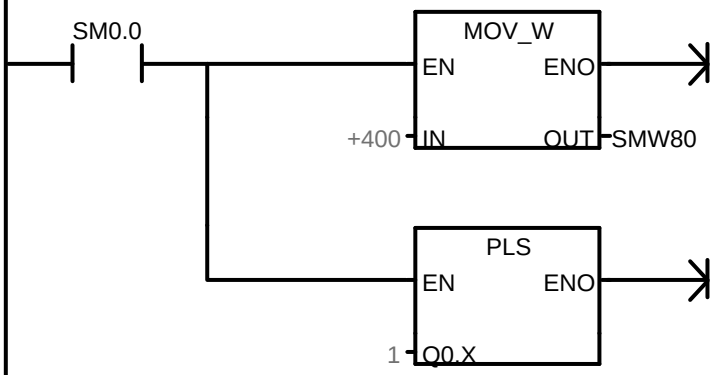


Bloque:	SBR_2	
Autor:		
Fecha de creación:	23.05.2009	15:01:53
Fecha de modificación:	23.05.2009	15:31:30

Símbolo	Tipo var.	Tipo de datos	Comentario
EN	IN	BOOL	
	IN		
	IN_OUT		
	OUT		
	TEMP		

TIEMPO 2

Network 1

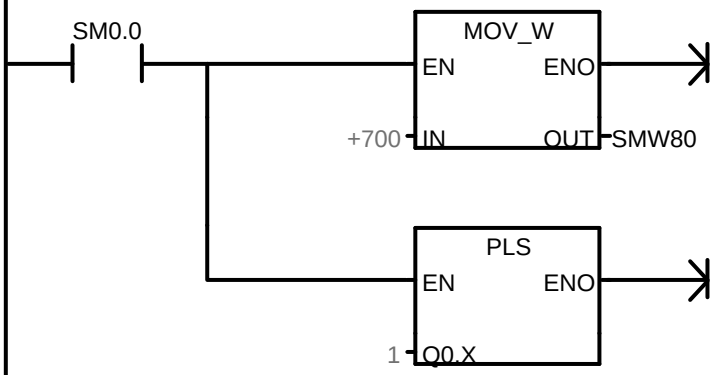


Bloque: SBR_3
Autor:
Fecha de creación: 23.05.2009 15:01:56
Fecha de modificación: 23.05.2009 15:31:42

Símbolo	Tipo var.	Tipo de datos	Comentario
EN	IN	BOOL	
	IN		
	IN_OUT		
	OUT		
	TEMP		

TIEMPO 3

Network 1

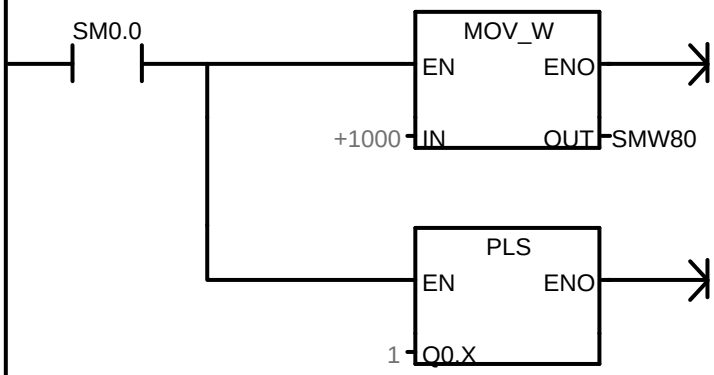


Bloque: SBR_4
Autor:
Fecha de creación: 23.05.2009 15:31:48
Fecha de modificación: 23.05.2009 15:34:34

Símbolo	Tipo var.	Tipo de datos	Comentario
EN	IN	BOOL	
	IN		
	IN_OUT		
	OUT		
	TEMP		

TIEMPO 4

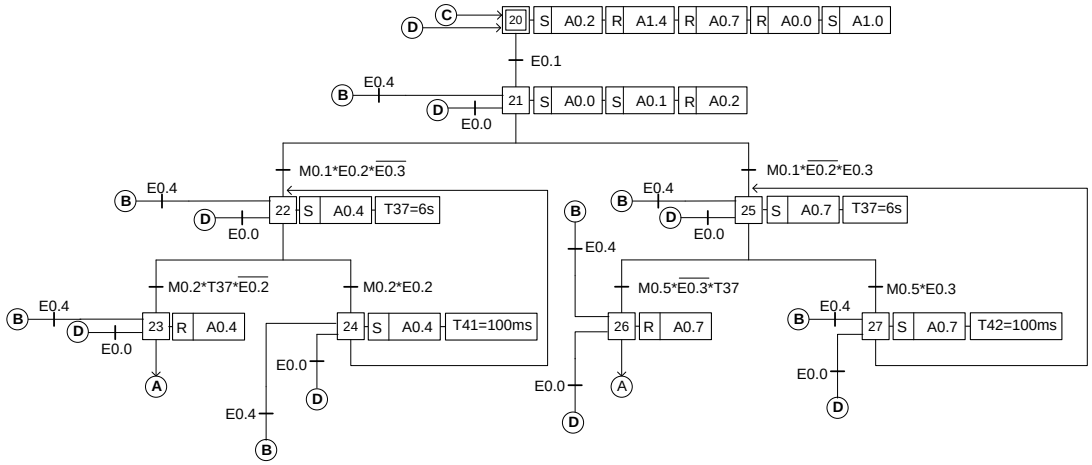
Network 1

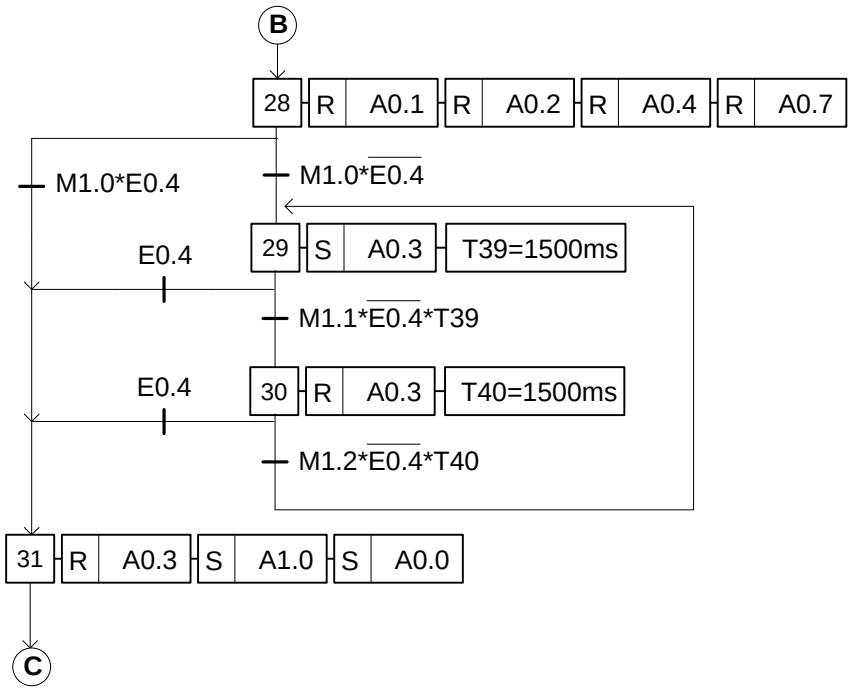


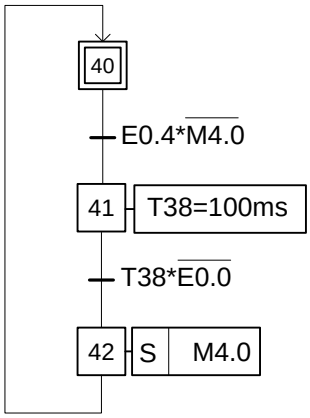
Símbolo	Dirección	Comentario
SEC_1	E0.0	ACTIVA SECUENCIA DE T=100ms
SEC_2	E0.1	ACTIVA SECUENCIA DE T=400ms
SEC_3	E0.2	ACTIVA SECUENCIA DE T=700ms
SEC_4	E0.3	ACTIVA SECUENCIA DE T=1000ms
OUT_PWM	A0.1	SALIDA PWM
COM	M0.0	PC <----> PLC
PCS1	M0.1	
PCS2	M0.2	
PCS3	M0.3	
PCS4	M0.4	

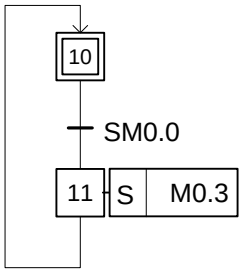
Símbolo	Dirección	Comentario
SBR_0	SBR0	INICIALIZACION DE PARAMETROS
SBR_1	SBR1	TIEMPO 1
SBR_2	SBR2	TIEMPO 2
SBR_3	SBR3	TIEMPO 3
SBR_4	SBR4	TIEMPO 4
PRINCIPAL	OB1	PROGRAMA PARA GENERAR UN PWM

Programa principal:







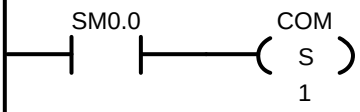


Bloque: PRINCIPAL
Autor:
Fecha de creación: 31.05.2009 19:49:31
Fecha de modificación: 30.05.2010 21:41:38

Símbolo	Tipo var.	Tipo de datos	Comentario
	TEMP		
	TEMP		
	TEMP		
	TEMP		

CONTROL DE UNA ESCALERA ELECTRICA

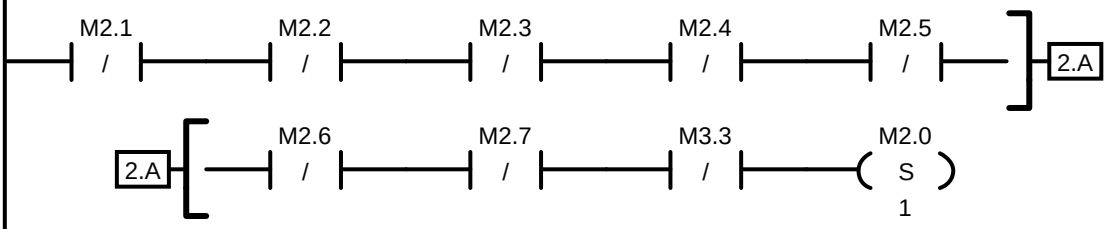
Network 1 COMUNICACION PC <------> PLC



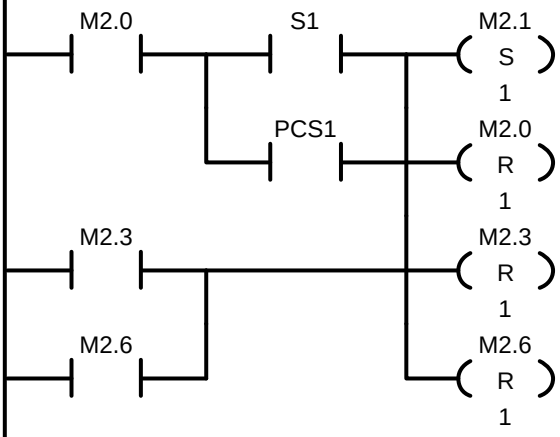
Símbolo	Dirección	Comentario
COM	M0.3	

Network 2 BLOQUE DE CONTROL

Comentario de segmento

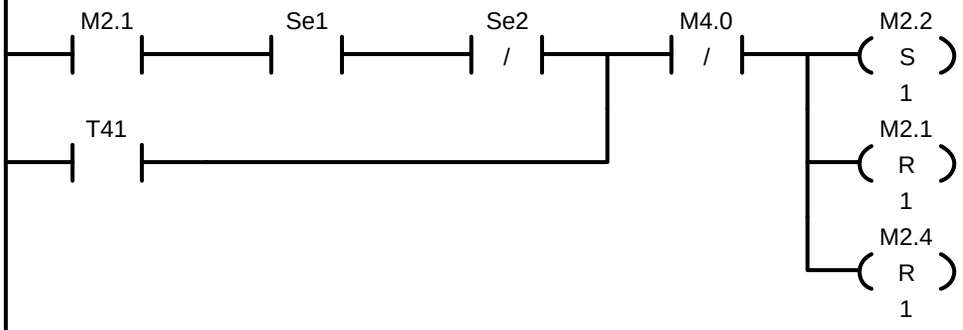


Network 3



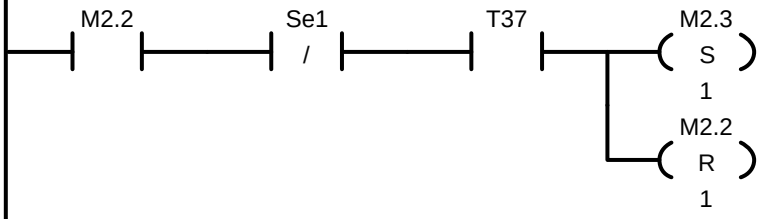
Símbolo	Dirección	Comentario
PCS1	M0.1	MARCHA DESDE PC
S1	E0.1	DESDE PLC

Network 4



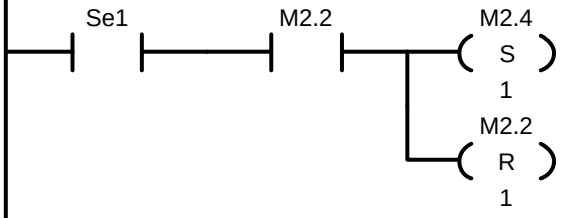
Símbolo	Dirección	Comentario
Se1	E0.2	SENSOR PLANTA BAJA
Se2	E0.3	SENSOR PLANTA ALTA

Network 5



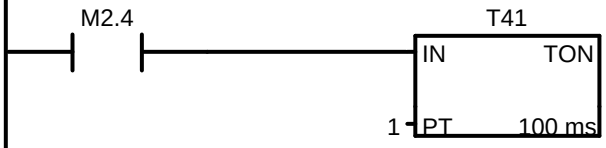
Símbolo	Dirección	Comentario
Se1	E0.2	SENSOR PLANTA BAJA

Network 6

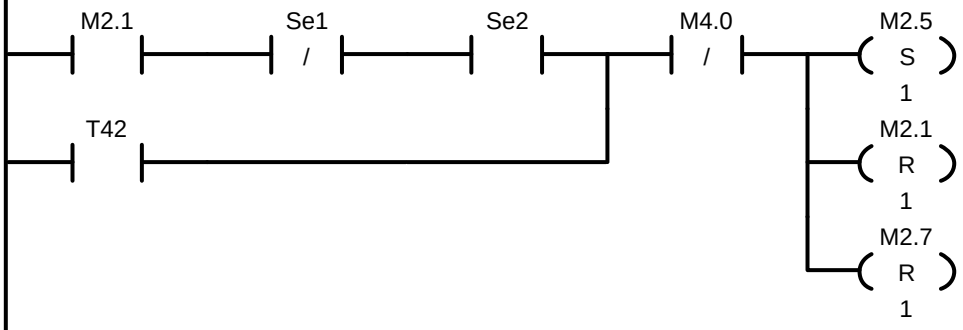


Símbolo	Dirección	Comentario
Se1	E0.2	SENSOR PLANTA BAJA

Network 7

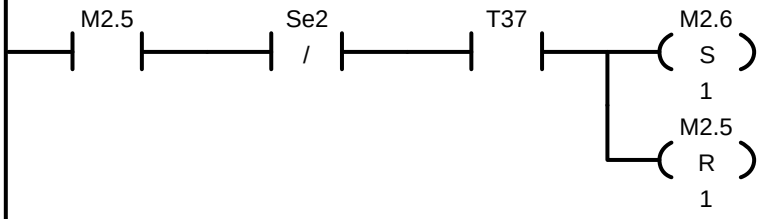


Network 8



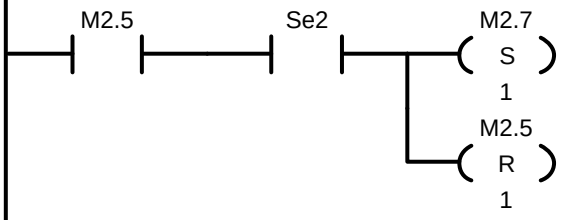
Símbolo	Dirección	Comentario
Se1	E0.2	SENSOR PLANTA BAJA
Se2	E0.3	SENSOR PLANTA ALTA

Network 9



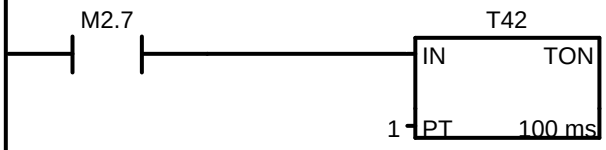
Símbolo	Dirección	Comentario
Se2	E0.3	SENSOR PLANTA ALTA

Network 10



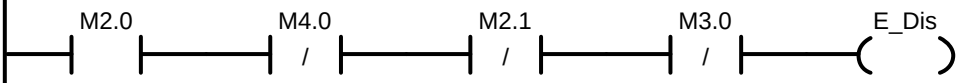
Símbolo	Dirección	Comentario
Se2	E0.3	SENSOR PLANTA ALTA

Network 11



Network 12 BLOQUE DE ACTIVACIONES

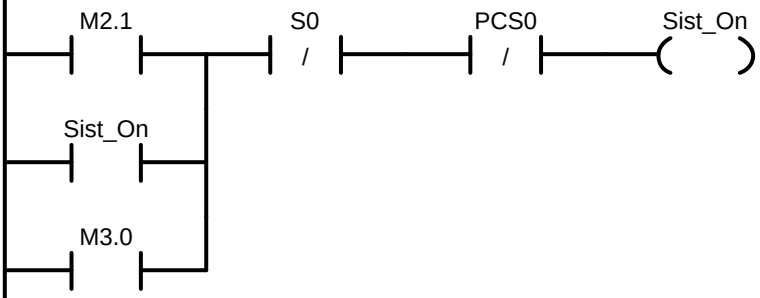
PILOTO DE "DESHABILITADO"



Símbolo	Dirección	Comentario
E_Dis	A0.2	CONTROL DESHABILITADO

Network 13

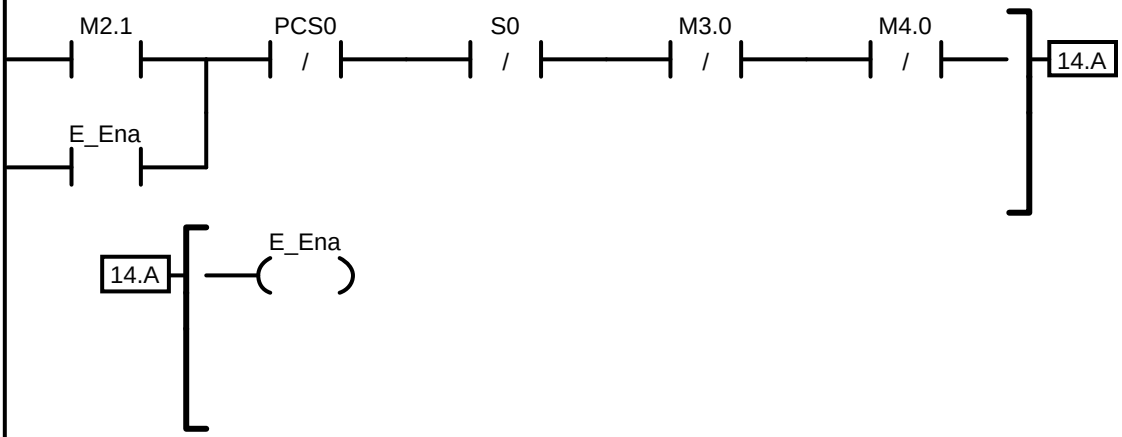
PILOTO DE "ENABLE"



Símbolo	Dirección	Comentario
PCS0	M0.0	PARO DESDE PC
S0	E0.0	DESDE PLC
Sist_On	A0.0	SISTEMA ACTIVO

Network 14

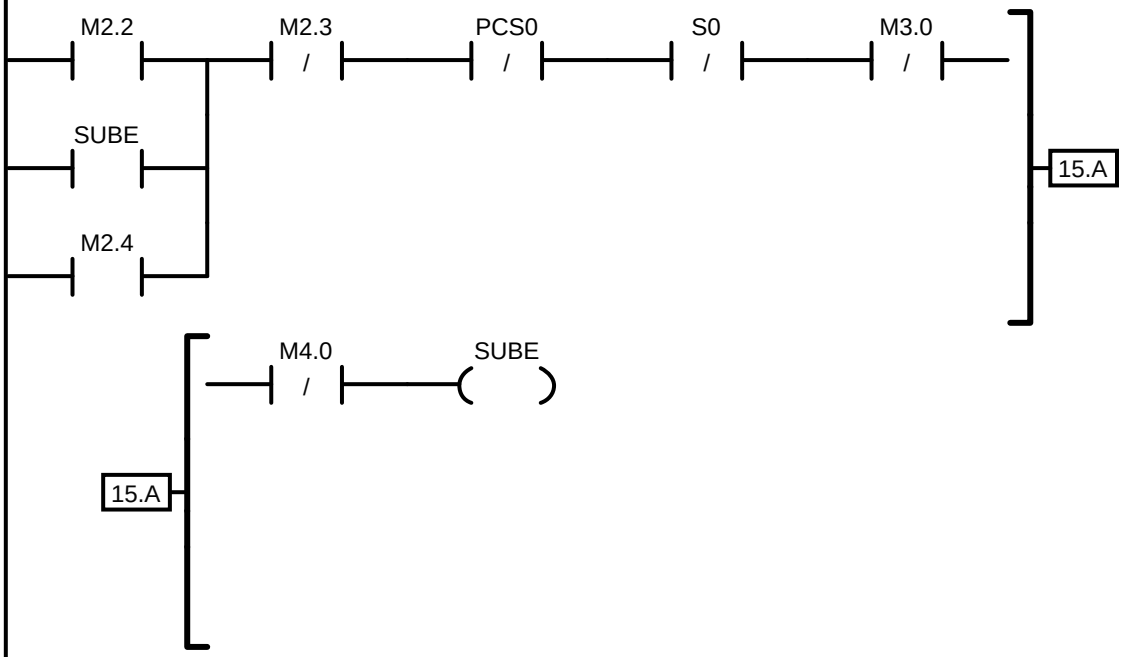
PILOTO DE "SISTEMA HABILITADO"



Símbolo	Dirección	Comentario
E_Ena	A0.1	CONTROL HABILITADO
PCS0	M0.0	PARO DESDE PC
S0	E0.0	DESDE PLC

Network 15

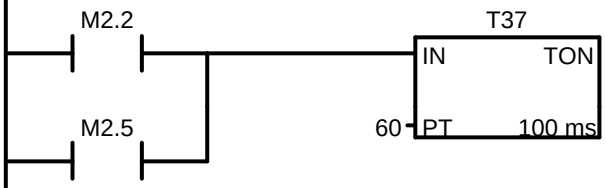
PILOTO DE "SUBE"



Símbolo	Dirección	Comentario
PCS0	M0.0	PARO DESDE PC
S0	E0.0	DESDE PLC
SUBE	A0.4	ESCALERA SUBIENDO

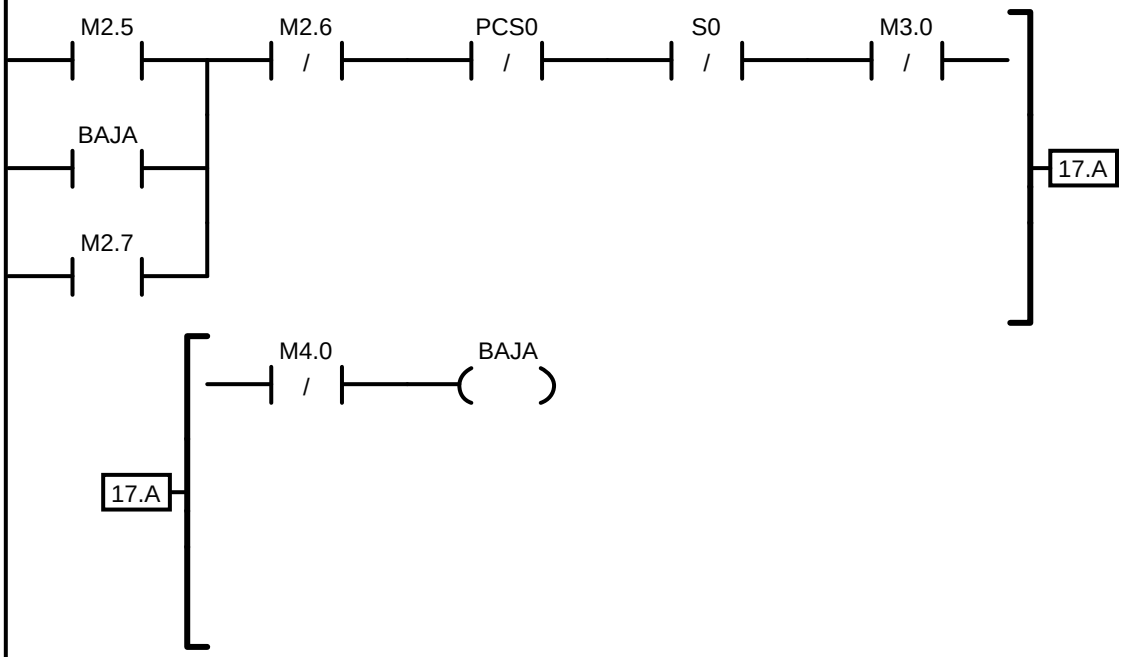
Network 16

DA EL TIEMPO QUE TARDA EN RECORRER UN ESCALON DESDE UN EXTREMO A OTRO DE LA ESCALERA.



Network 17

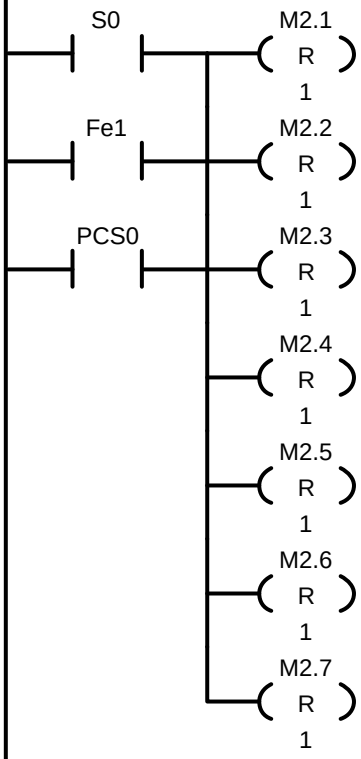
PILOTO DE "BAJA"



Símbolo	Dirección	Comentario
BAJA	A0.7	ESCALERA BAJANDO
PCS0	M0.0	PARO DESDE PC
S0	E0.0	DESDE PLC

Network 18 BLOQUE INDEPENDIENTE

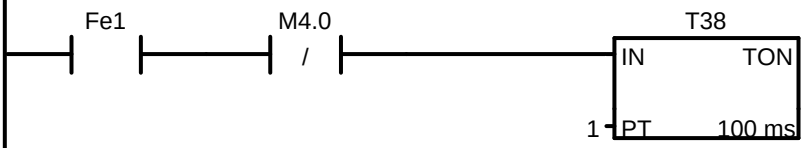
ACTIVACION DE S0 Y DE EL SWITCH DE EMERGENCIA



Símbolo	Dirección	Comentario
Fe1	E0.4	TERMICO DEL MOTOR
PCS0	M0.0	PARO DESDE PC
S0	E0.0	DESDE PLC

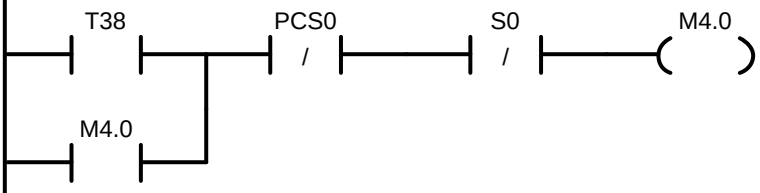
Network 19

BLOQUEO DEL SWITCH DE EMERGENCIA



Símbolo	Dirección	Comentario
Fe1	E0.4	TERMICO DEL MOTOR

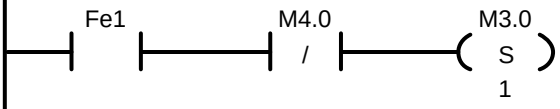
Network 20



Símbolo	Dirección	Comentario
PCS0	M0.0	PARO DESDE PC
S0	E0.0	DESDE PLC

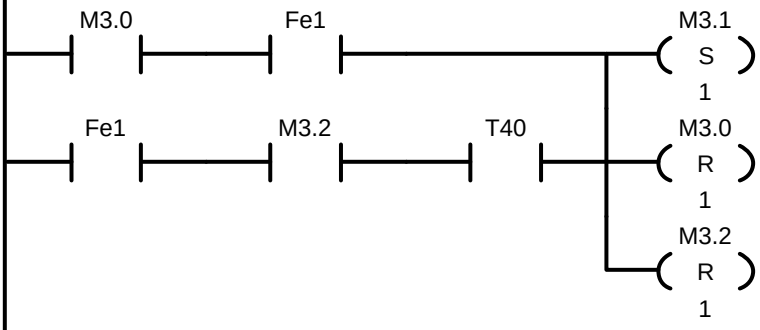
Network 21

BLOQUE DE EMERGENCIA



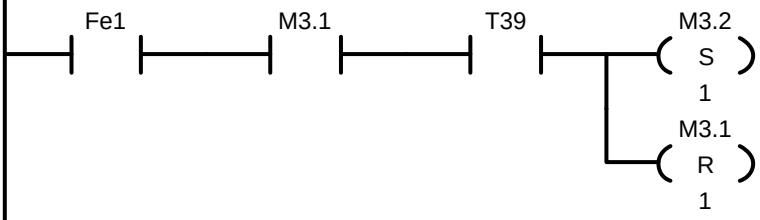
Símbolo	Dirección	Comentario
Fe1	E0.4	TERMICO DEL MOTOR

Network 22



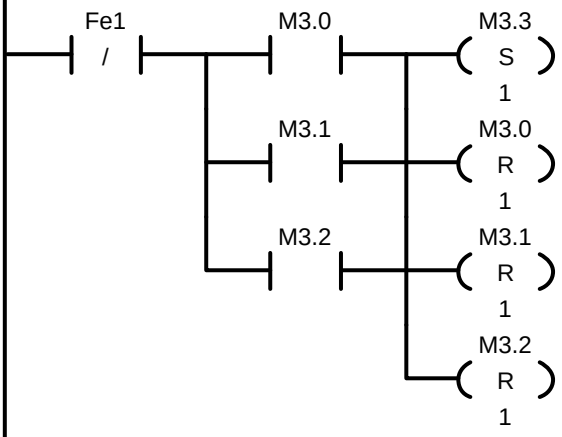
Símbolo	Dirección	Comentario
Fe1	E0.4	TERMICO DEL MOTOR

Network 23



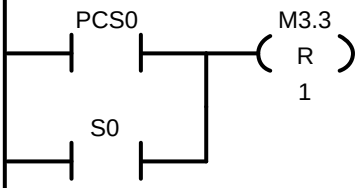
Símbolo	Dirección	Comentario
Fe1	E0.4	TERMICO DEL MOTOR

Network 24



Símbolo	Dirección	Comentario
Fe1	E0.4	TERMICO DEL MOTOR

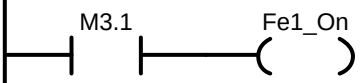
Network 25



Símbolo	Dirección	Comentario
PCS0	M0.0	PARO DESDE PC
S0	E0.0	DESDE PLC

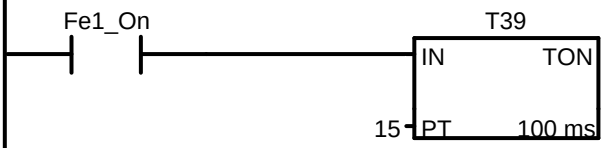
Network 26

PILOTO DE EMERGENCIA ON



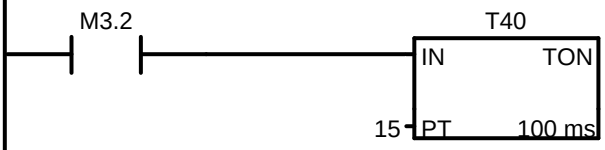
Símbolo	Dirección	Comentario
Fe1_On	A0.3	TERNICO ACTIVO

Network 27



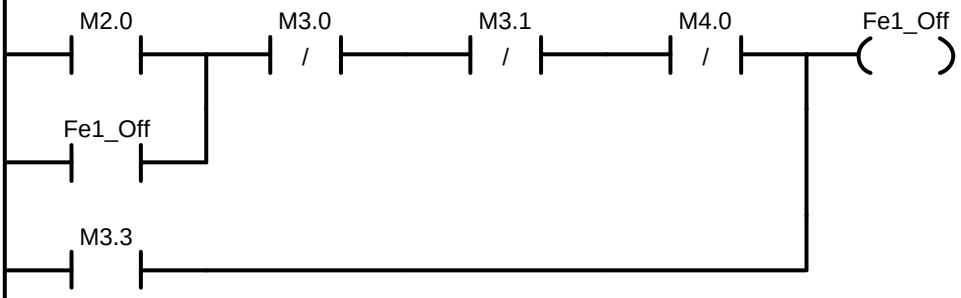
Símbolo	Dirección	Comentario
Fe1_On	A0.3	TERNICO ACTIVO

Network 28



Network 29

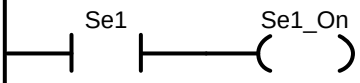
PILOTO DE EMERGENCIA OFF



Símbolo	Dirección	Comentario
Fe1_Off	A1.0	TERMICO NO ACTIVO

Network 30 PILOTO DE LOS SENSORES DE ACTIVACION

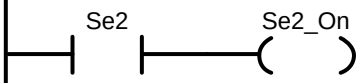
SENSOR DE SUBIDA



Símbolo	Dirección	Comentario
Se1	E0.2	SENSOR PLANTA BAJA
Se1_On	A0.5	SENSOR DE ABAJO ACTIVO

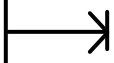
Network 31

SENSOR DE BAJADA



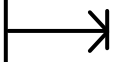
Símbolo	Dirección	Comentario
Se2	E0.3	SENSOR PLANTA ALTA
Se2_On	A0.6	SENSOR DE ARRIBA ACTIVO

Network 32



Network 33 Título de segmento

Comentario de segmento



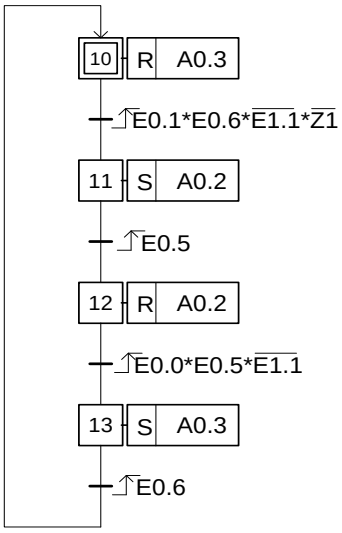
Símbolo	Dirección	Comentario
S0	E0.0	DESDE PLC
S1	E0.1	DESDE PLC
Se1	E0.2	SENSOR PLANTA BAJA
Se2	E0.3	SENSOR PLANTA ALTA
Fe1	E0.4	TERMICO DEL MOTOR
Sist_On	A0.0	SISTEMA ACTIVO
E_Ena	A0.1	CONTROL HABILITADO
E_Dis	A0.2	CONTROL DESHABILITADO
Fe1_Off	A1.0	TERMICO NO ACTIVO
Fe1_On	A0.3	TERNICO ACTIVO
SUBE	A0.4	ESCALERA SUBIENDO
Se1_On	A0.5	SENSOR DE ABAJO ACTIVO
Se2_On	A0.6	SENSOR DE ARRIBA ACTIVO
BAJA	A0.7	ESCALERA BAJANDO
PCS0	M0.0	PARO DESDE PC
PCS1	M0.1	MARCHA DESDE PC
COM	M0.3	

Símbolo	Dirección	Comentario
PRINCIPAL	OB1	CONTROL DE UNA ESCALERA ELECTRICA

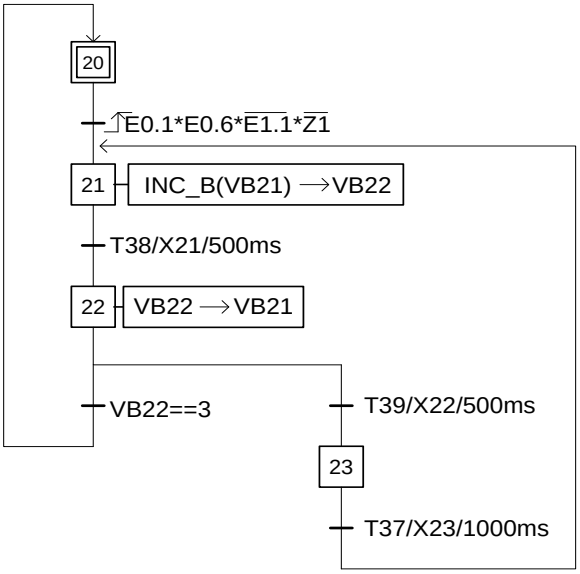
Programa principal:

CONTROL DE LA BARANDA DE INGRESO

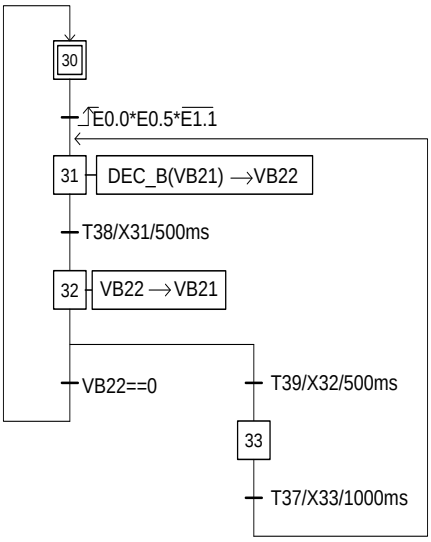
APERTURA



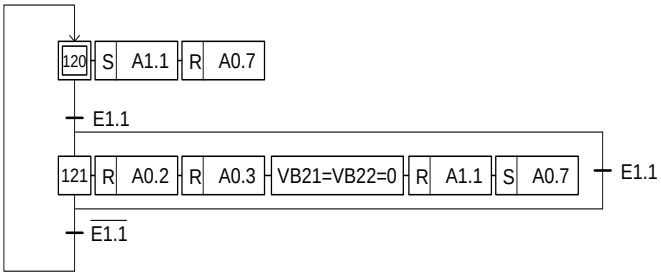
IMÁGENES DE APERTURA



IMÁGENES DE CIERRE

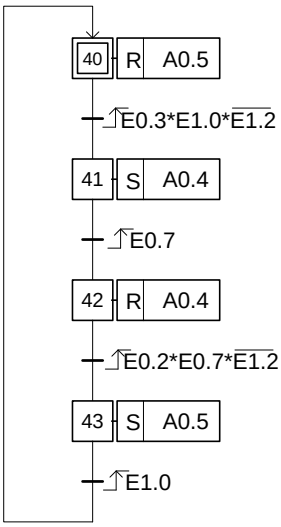


ACCIÓN DEL TÉRMICO

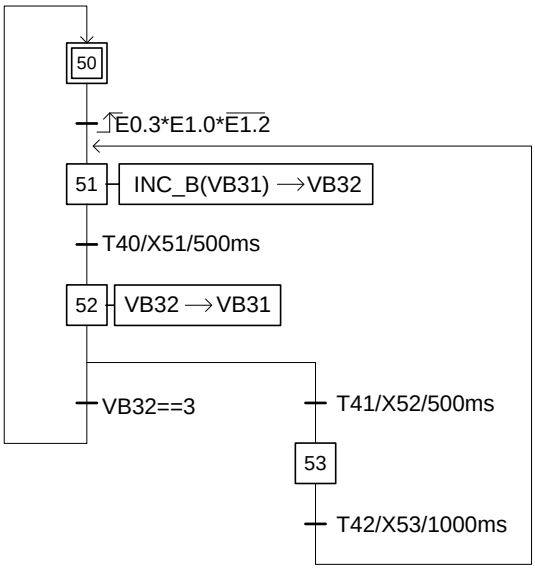


CONTROL DE LA BARANDA DE SALIDA

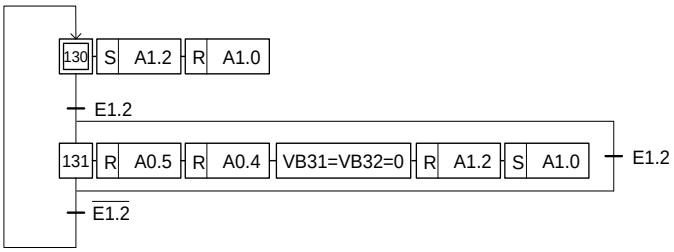
APERTURA



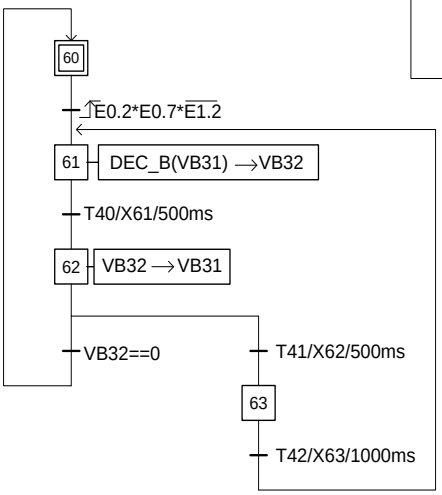
IMÁGENES DE APERTURA



ACCIÓN DEL TÉRMICO

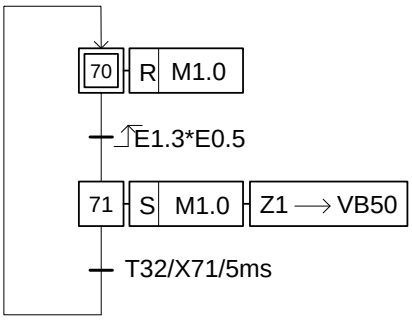


IMÁGENES DE CIERRE

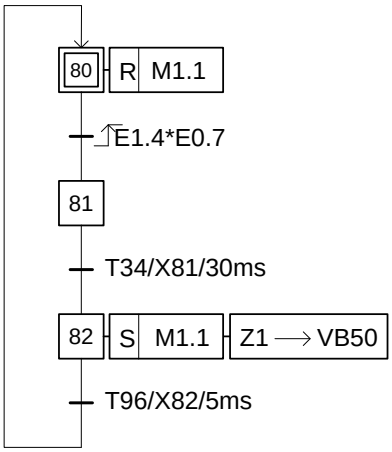


CONTADOR

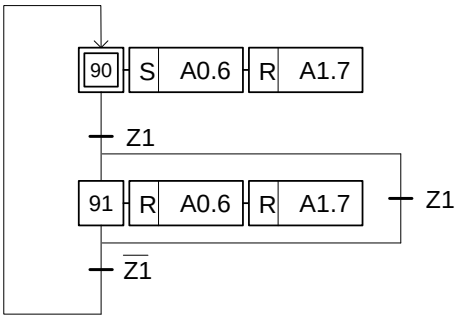
CUENTA ASCENDENTE



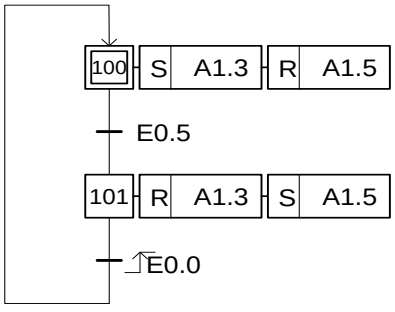
CUENTA DESCENDENTE



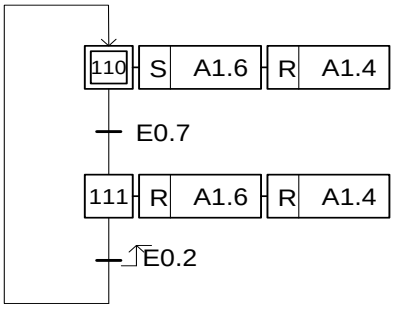
CONTROL DE INDICADORES



SEMÁFORO DE INGRESO



SEMÁFORO DE SALIDA



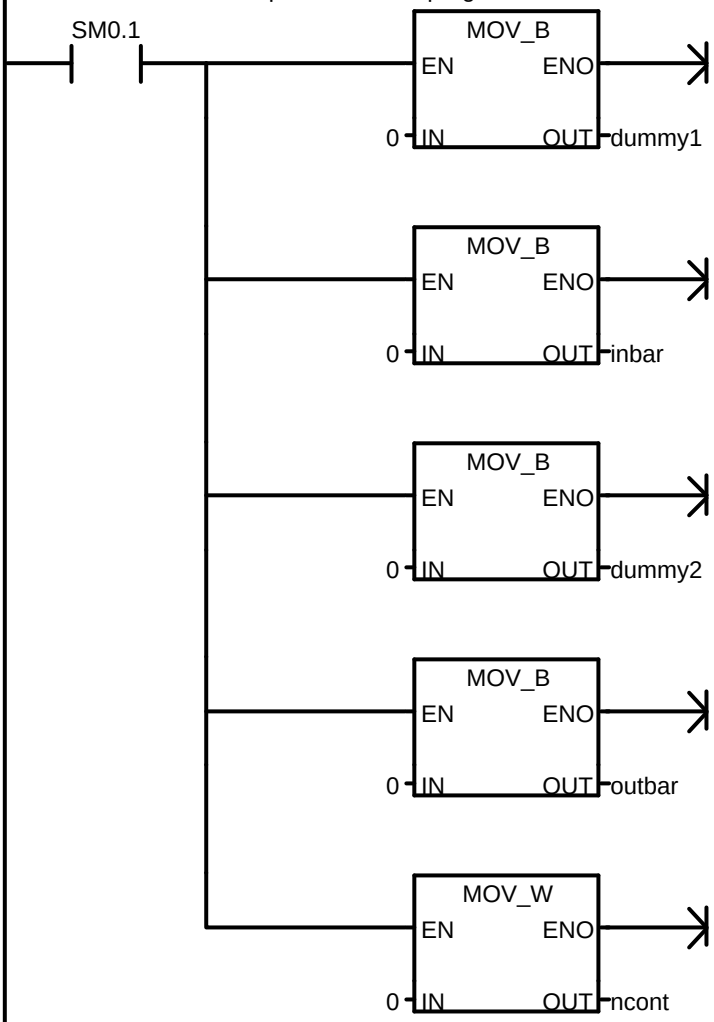
Bloque: PRINCIPAL
Autor:
Fecha de creación: 09.06.2009 19:50:00
Fecha de modificación: 11.06.2009 20:19:09

Símbolo	Tipo var.	Tipo de datos	Comentario
	TEMP		
	TEMP		
	TEMP		
	TEMP		

CONTROL DE UN PARQUEADERO

Network 1

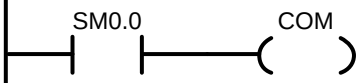
Esta rutina solo se cumple al incio del programa.



Símbolo	Dirección	Comentario
dummy1	VB21	Mover imagen de ingreso
dummy2	VB31	Mover imagen de salida
inbar	VB22	Mover imagen de ingreso
ncont	VW50	Valor de conteo
outbar	VB32	Mover imagen de salida

Network 2

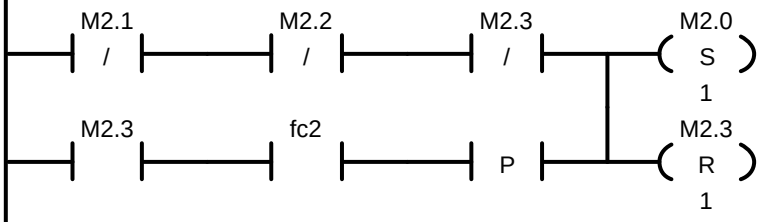
COMUNICACION



Símbolo	Dirección	Comentario
COM	M0.5	pc ---- plc

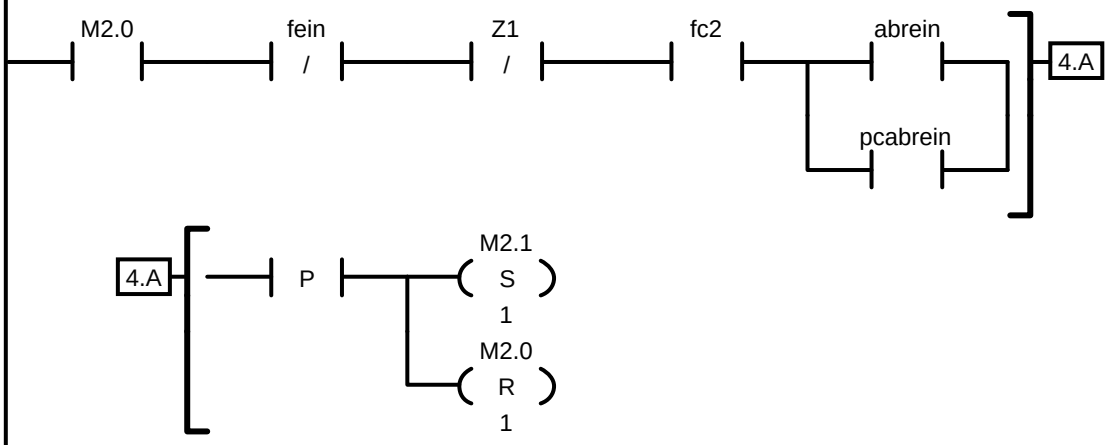
Network 3 ********** PROGRAMA DE CONTROL

Bloque de control para el motor de apertura/cierre de ingreso al parqueadero. Abrir y cerrar se controla con un motor en inversion de giro, consta de sistema de frenado por zapata.



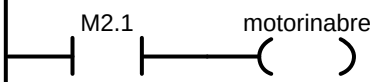
Símbolo	Dirección	Comentario
fc2	E0.6	plc

Network 4



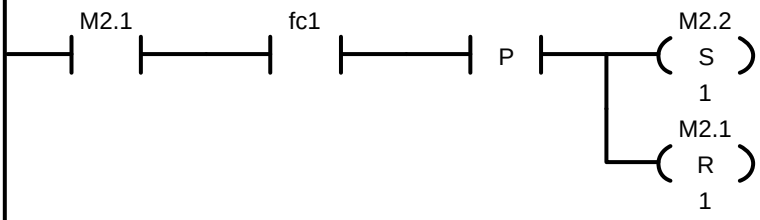
Símbolo	Dirección	Comentario
abrein	E0.1	plc
fc2	E0.6	plc
fein	E1.1	plc
pcabrein	M0.1	pc

Network 5



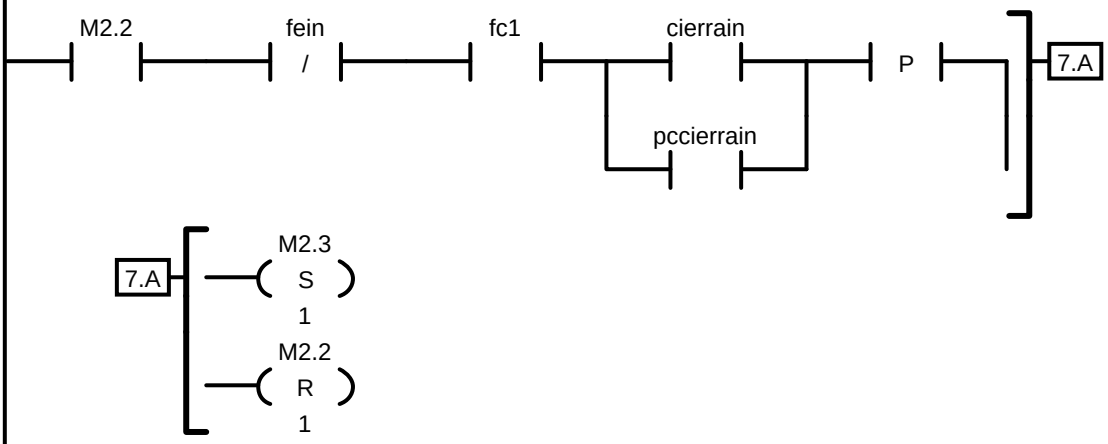
Símbolo	Dirección	Comentario
motorinabre	A0.2	plc

Network 6



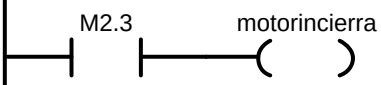
Símbolo	Dirección	Comentario
fc1	E0.5	plc

Network 7



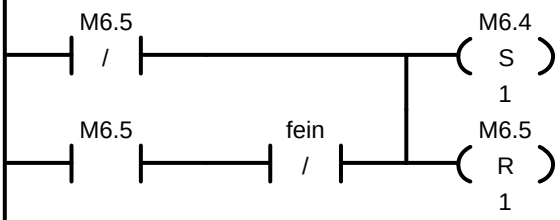
Símbolo	Dirección	Comentario
cierrain	E0.0	plc
fc1	E0.5	plc
fein	E1.1	plc
pccierrain	M0.0	pc

Network 8



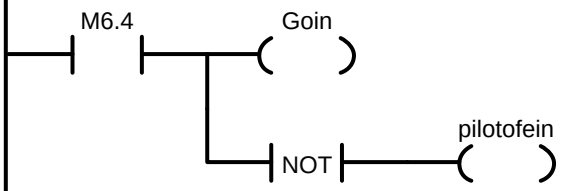
Símbolo	Dirección	Comentario
motorincierra	A0.3	plc

Network 9 ********** ********** Bloque de operacion del termico de ingreso



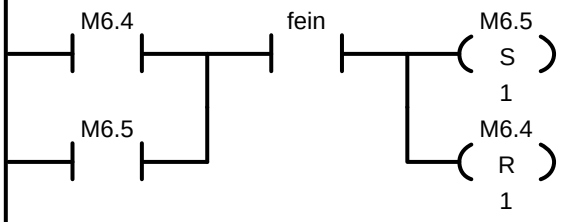
Símbolo	Dirección	Comentario
fein	E1.1	plc

Network 10



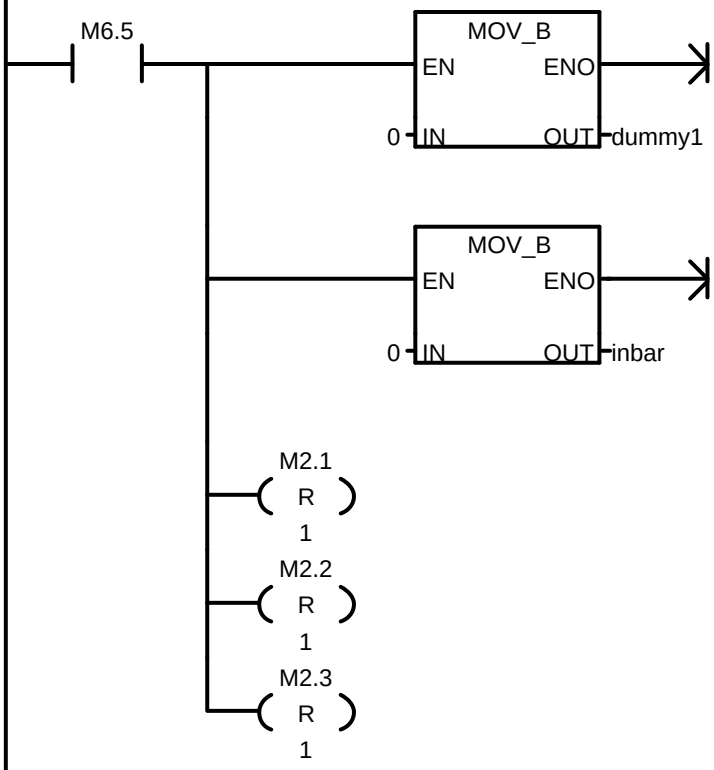
Símbolo	Dirección	Comentario
Goin	A1.1	plc
pilotofein	A0.7	plc

Network 11



Símbolo	Dirección	Comentario
fein	E1.1	plc

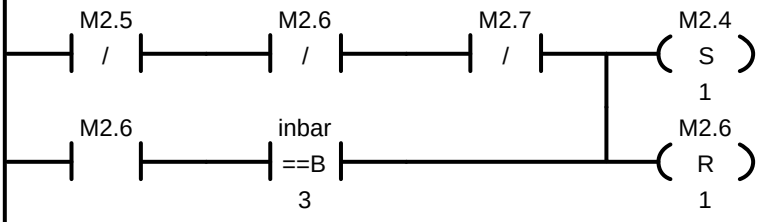
Network 12



Símbolo	Dirección	Comentario
dummy1	VB21	Mover imagen de ingreso
inbar	VB22	Mover imagen de ingreso

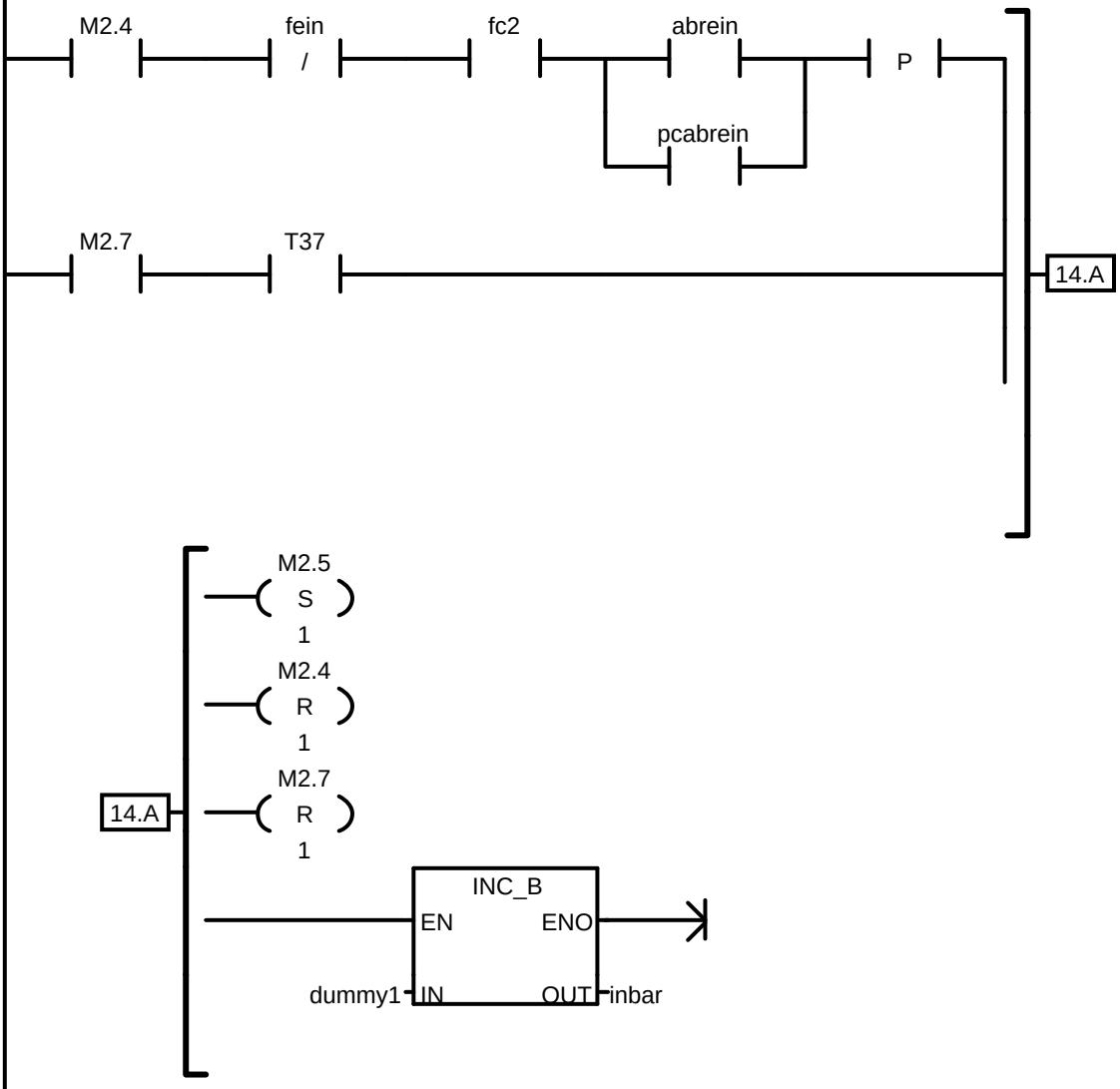
Network 13 ==========

Secuencia que controla la imagen de apertura de la baranda de ingreso. Es decir, la imagen del scada empieza a moverse en apertura.



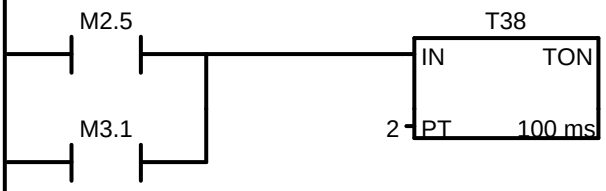
Símbolo	Dirección	Comentario
inbar	VB22	Mover imagen de ingreso

Network 14

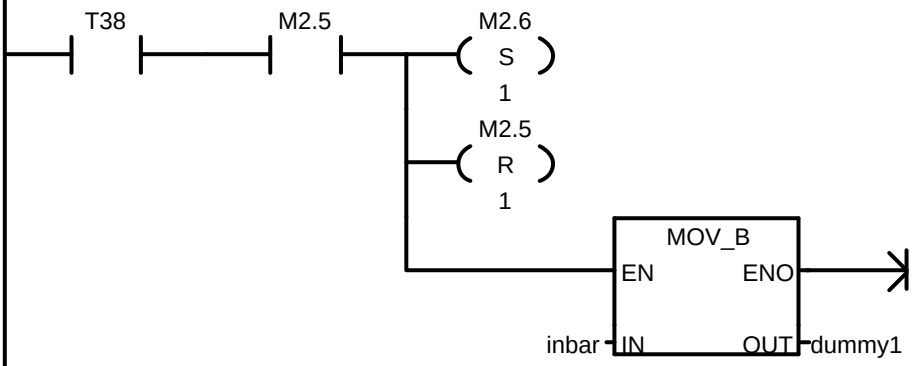


Símbolo	Dirección	Comentario
abrein	E0.1	plc
dummy1	VB21	Mover imagen de ingreso
fc2	E0.6	plc
fein	E1.1	plc
inbar	VB22	Mover imagen de ingreso
pcabrein	M0.1	pc

Network 15

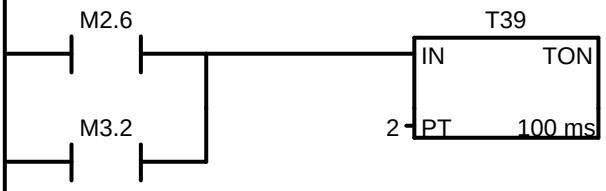


Network 16

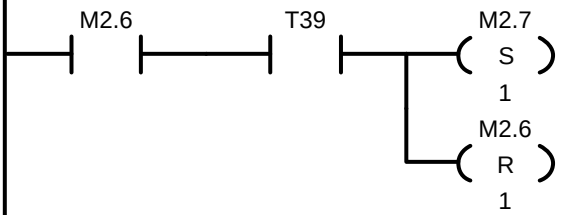


Símbolo	Dirección	Comentario
dummy1	VB21	Mover imagen de ingreso
inbar	VB22	Mover imagen de ingreso

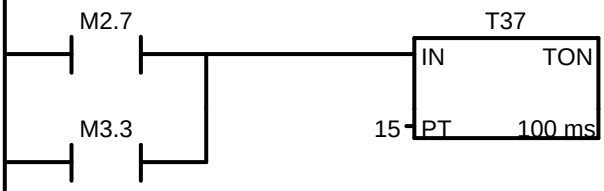
Network 17



Network 18

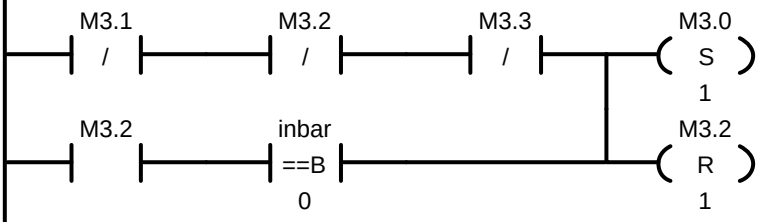


Network 19



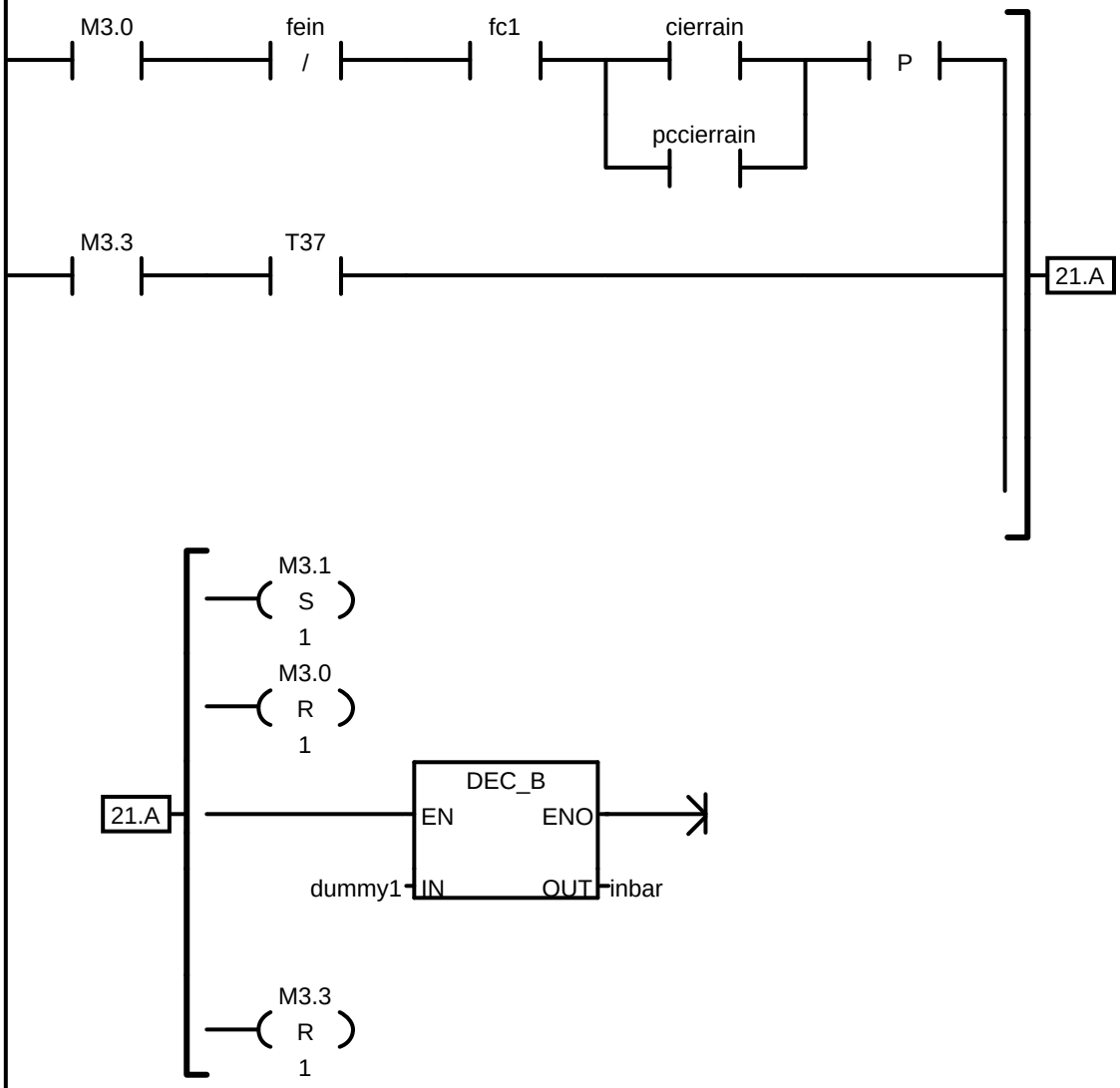
Network 20 ==========

Secuencia que controla la imagen de cierre de la baranda de ingreso. Es decir, la imagen del scada empieza a moverse en cerrado.



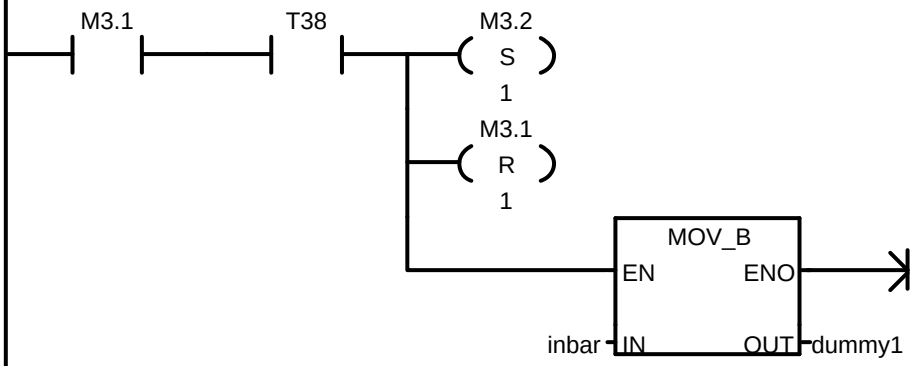
Símbolo	Dirección	Comentario
inbar	VB22	Mover imagen de ingreso

Network 21



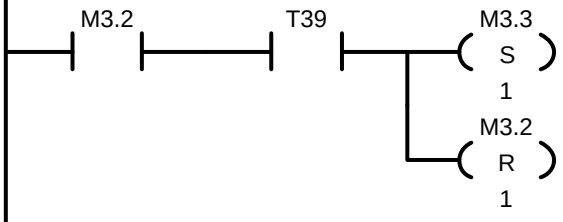
Símbolo	Dirección	Comentario
cierrain	E0.0	plc
dummy1	VB21	Mover imagen de ingreso
fc1	E0.5	plc
fein	E1.1	plc
inbar	VB22	Mover imagen de ingreso
pccierrain	M0.0	pc

Network 22



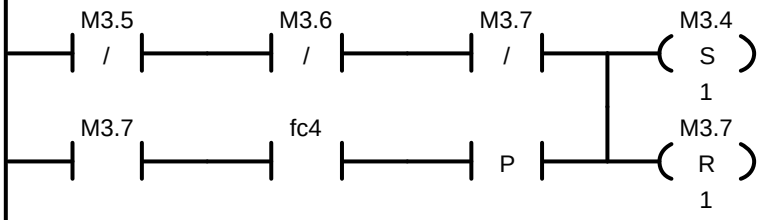
Símbolo	Dirección	Comentario
dummy1	VB21	Mover imagen de ingreso
inbar	VB22	Mover imagen de ingreso

Network 23



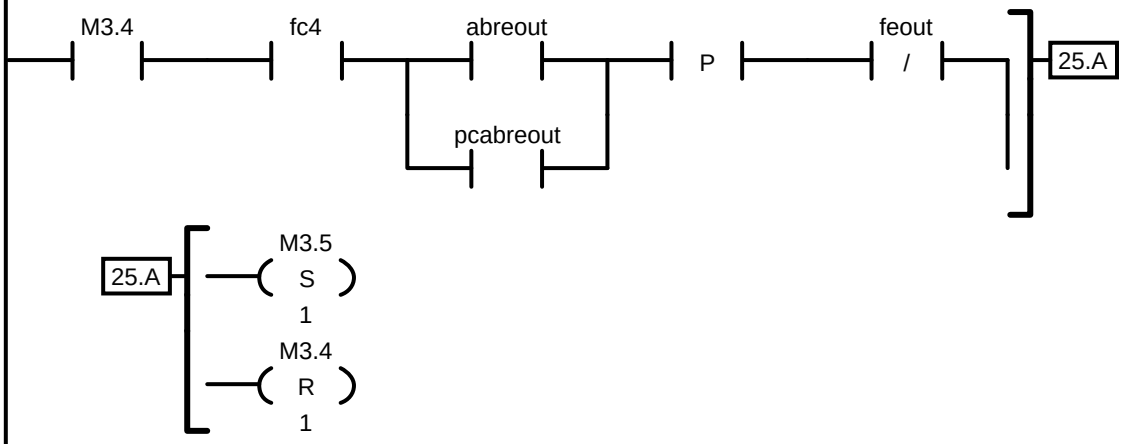
Network 24 ==========

Bloque de control para el motor de apertura/cierre de salida del parqueadero. Abrir y cerrar se controla con un motor en inversion de giro, consta de sistema de frenado por zapata.



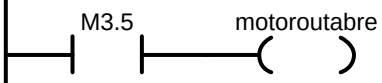
Símbolo	Dirección	Comentario
fc4	E1.0	plc

Network 25



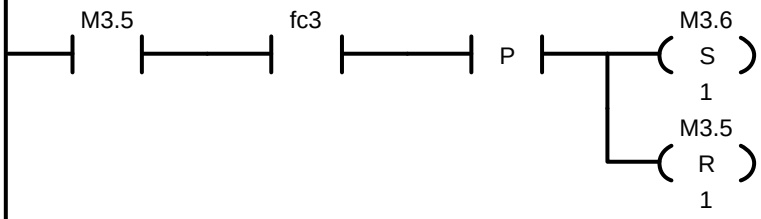
Símbolo	Dirección	Comentario
abreout	E0.3	plc
fc4	E1.0	plc
feout	E1.2	plc
pcabreout	M0.3	pc

Network 26



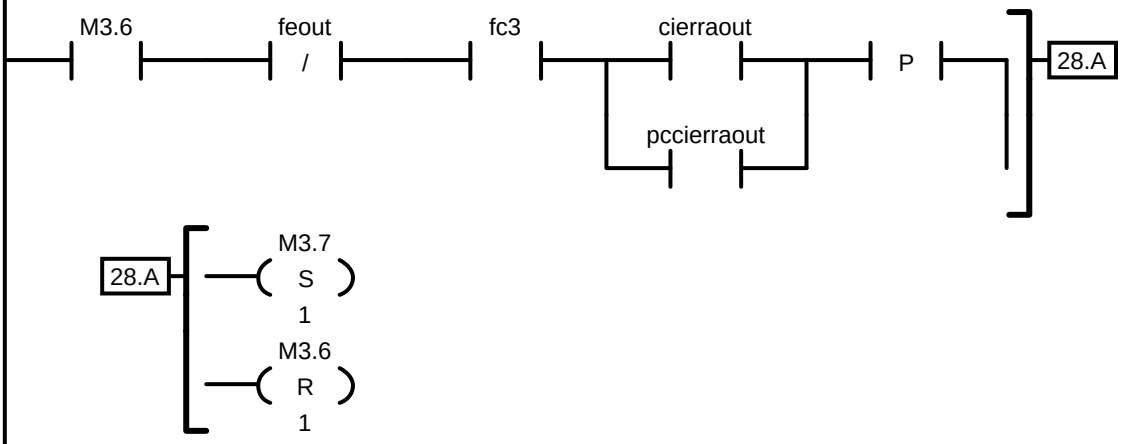
Símbolo	Dirección	Comentario
motoroutabre	A0.4	plc

Network 27



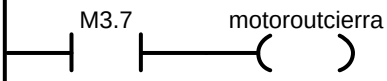
Símbolo	Dirección	Comentario
fc3	E0.7	plc

Network 28



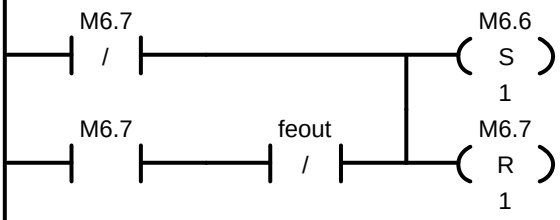
Símbolo	Dirección	Comentario
cierraout	E0.2	plc
fc3	E0.7	plc
feout	E1.2	plc
pccierraout	M0.2	pc

Network 29



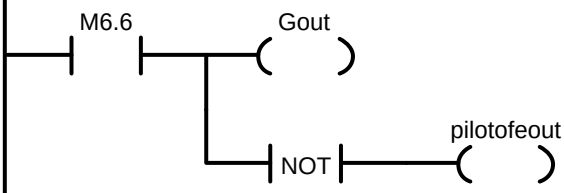
Símbolo	Dirección	Comentario
motoroutcierra	A0.5	plc

Network 30 ********** ********** Bloque de operacion del termico de salida



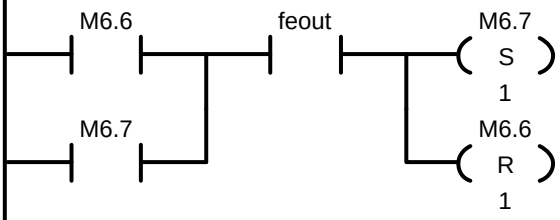
Símbolo	Dirección	Comentario
feout	E1.2	plc

Network 31



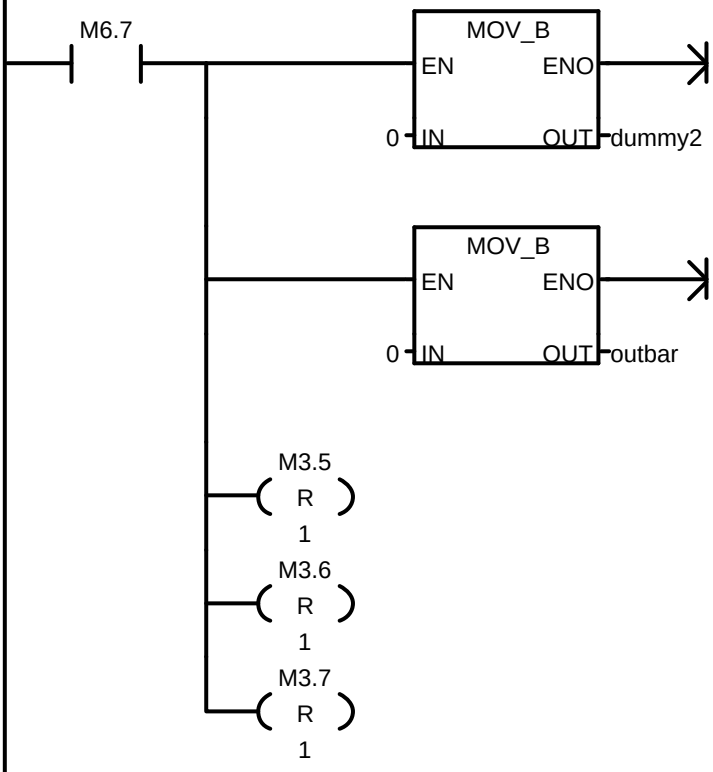
Símbolo	Dirección	Comentario
Gout	A1.2	plc
pilotofeout	A1.0	plc

Network 32



Símbolo	Dirección	Comentario
feout	E1.2	plc

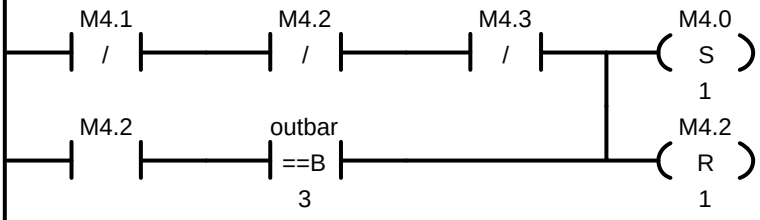
Network 33



Símbolo	Dirección	Comentario
dummy2	VB31	Mover imagen de salida
outbar	VB32	Mover imagen de salida

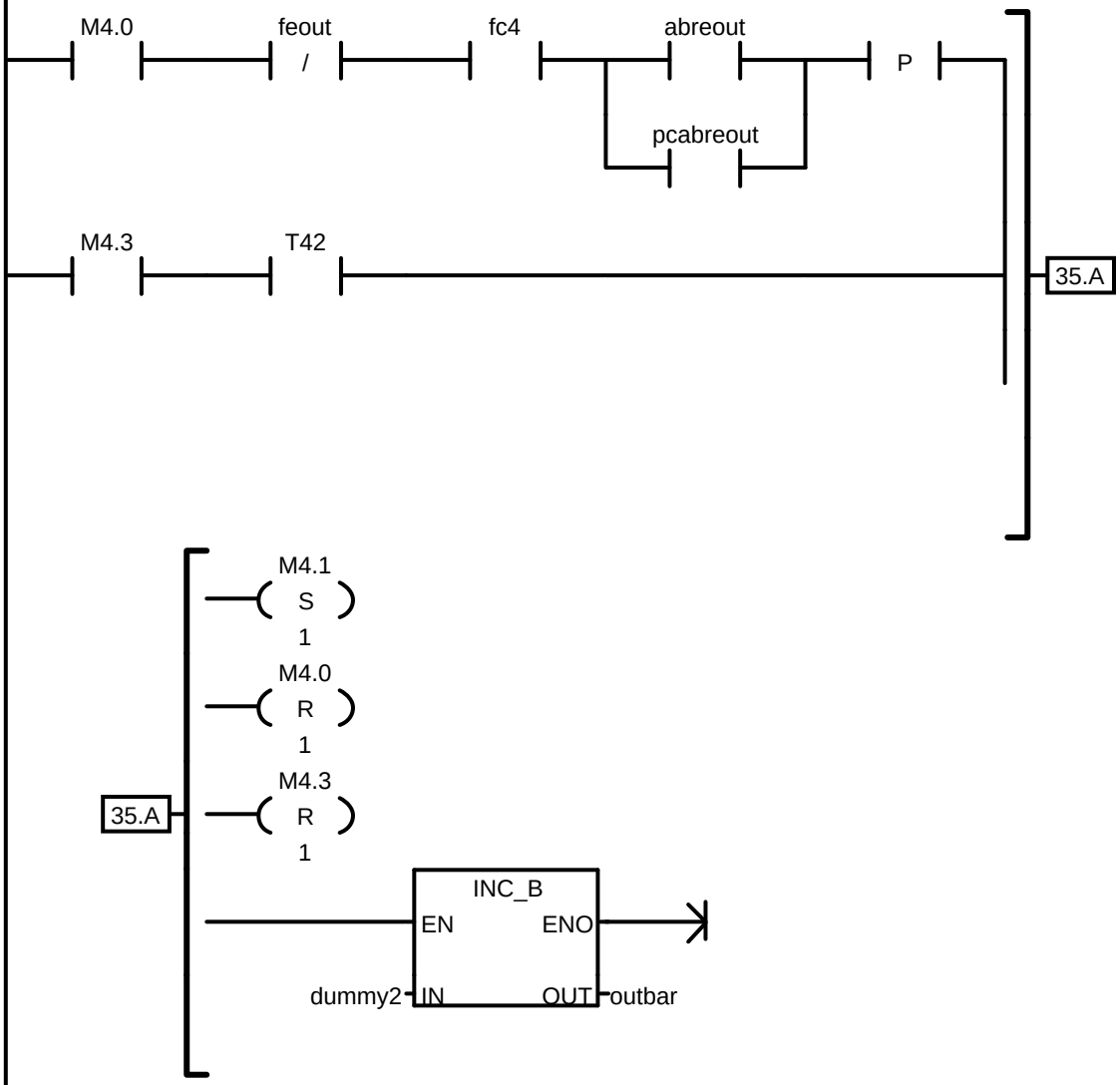
Network 34 ==========

Secuencia que controla la imagen de apertura de la baranda de salida. Es decir, la imagen del scada empieza a moverse en apertura.



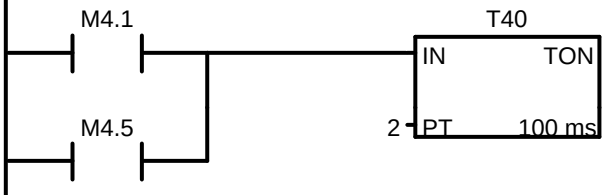
Símbolo	Dirección	Comentario
outbar	VB32	Mover imagen de salida

Network 35

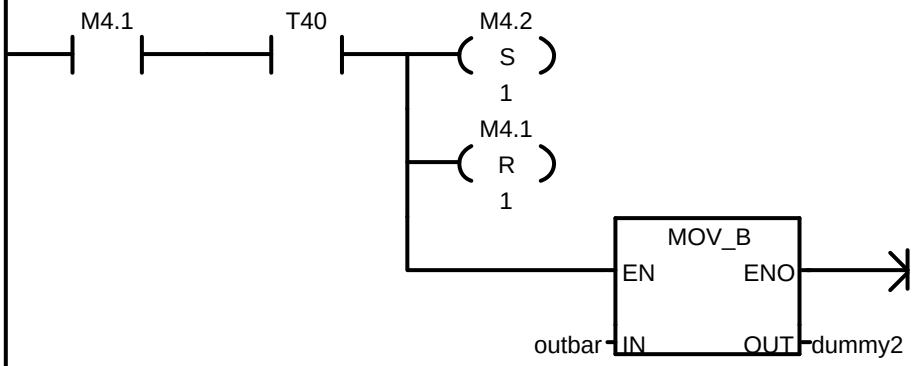


Símbolo	Dirección	Comentario
abreout	E0.3	plc
dummy2	VB31	Mover imagen de salida
fc4	E1.0	plc
feout	E1.2	plc
outbar	VB32	Mover imagen de salida
pcabreout	M0.3	pc

Network 36

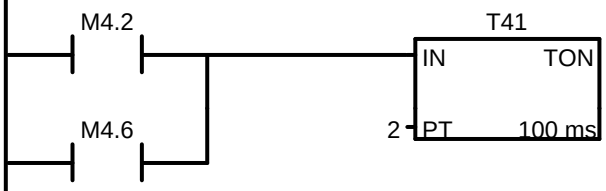


Network 37

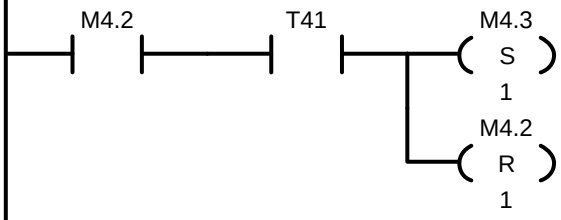


Símbolo	Dirección	Comentario
dummy2	VB31	Mover imagen de salida
outbar	VB32	Mover imagen de salida

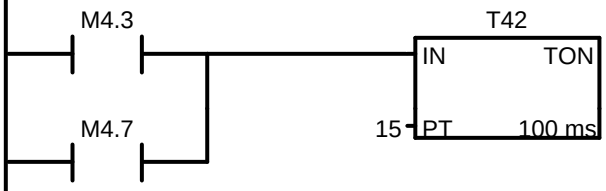
Network 38



Network 39

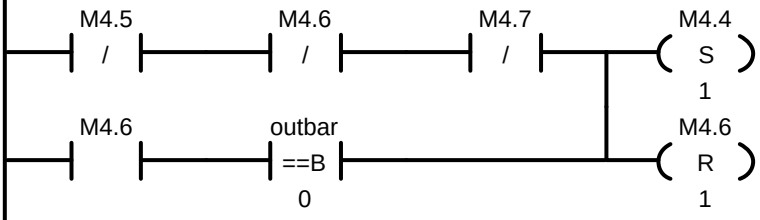


Network 40



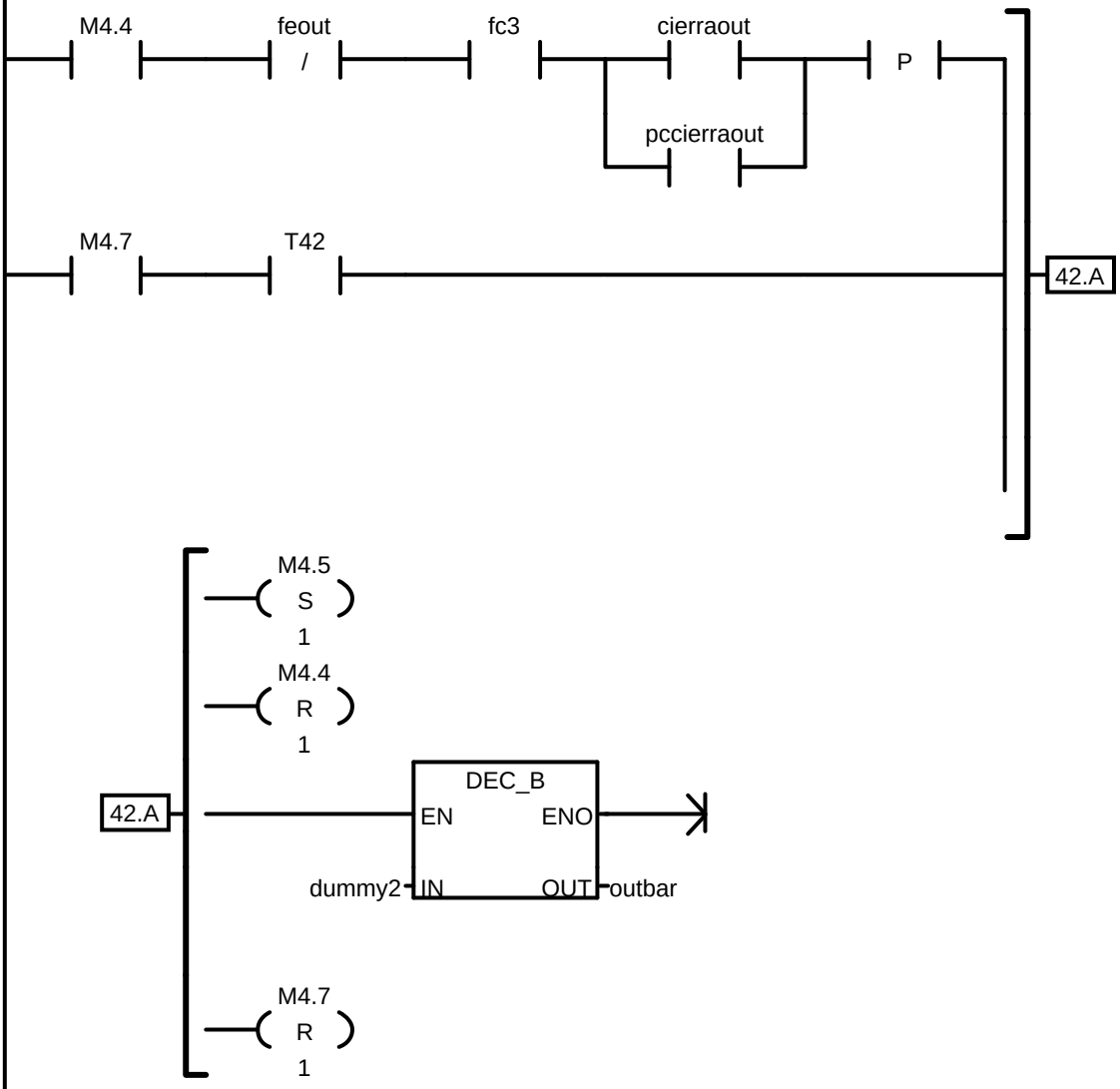
Network 41 ==========

Secuencia que controla la imagen de cierre de la baranda de salida. Es decir, la imagen del scada empieza a moverse en cerrado.



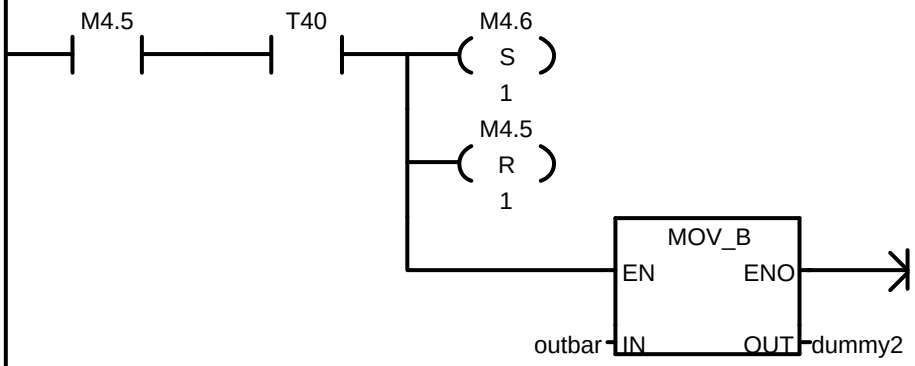
Símbolo	Dirección	Comentario
outbar	VB32	Mover imagen de salida

Network 42



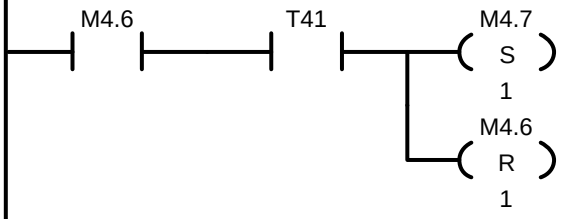
Símbolo	Dirección	Comentario
cierraout	E0.2	plc
dummy2	VB31	Mover imagen de salida
fc3	E0.7	plc
feout	E1.2	plc
outbar	VB32	Mover imagen de salida
pccierraout	M0.2	pc

Network 43



Símbolo	Dirección	Comentario
dummy2	VB31	Mover imagen de salida
outbar	VB32	Mover imagen de salida

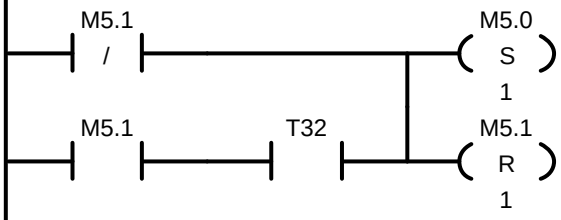
Network 44



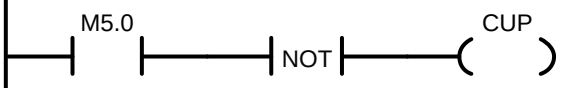
Network 45 ==========

Bloque de control para el contador de vehiculos que ingresan y salen del parqueadero.

Bloque de conteo ascendente.

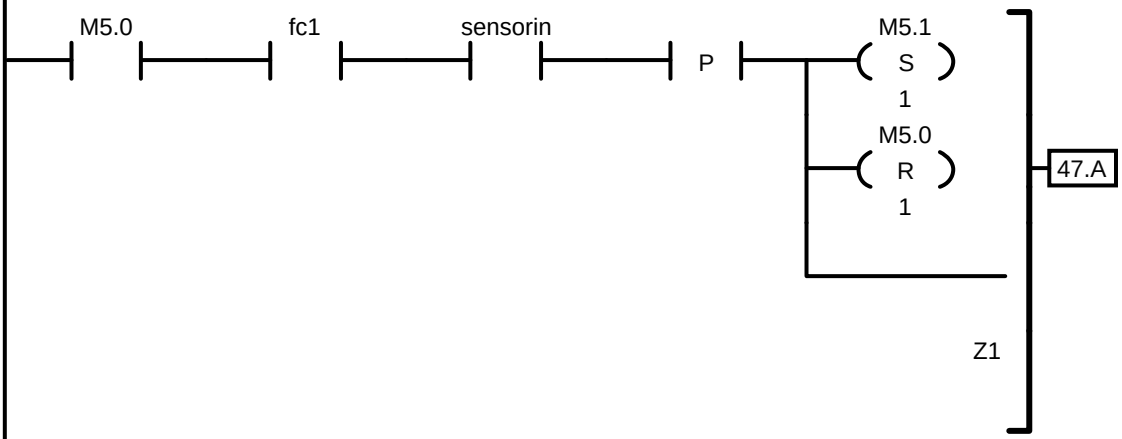


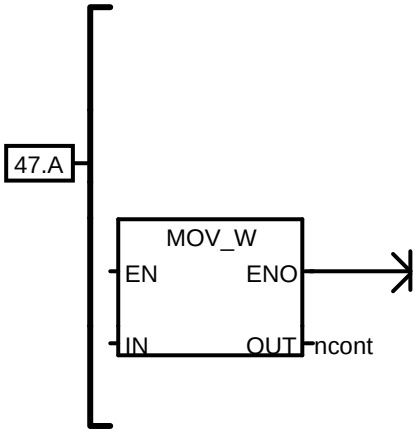
Network 46



Símbolo	Dirección	Comentario
CUP	M1.0	Conteo arriba

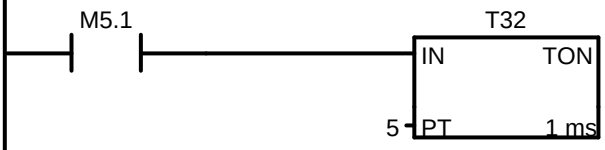
Network 47





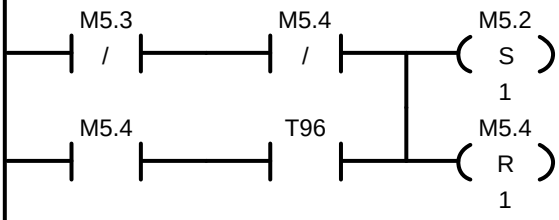
Símbolo	Dirección	Comentario
fc1	E0.5	plc
ncont	VW50	Valor de conteo
sensorin	E1.3	plc

Network 48



Network 49 ----------

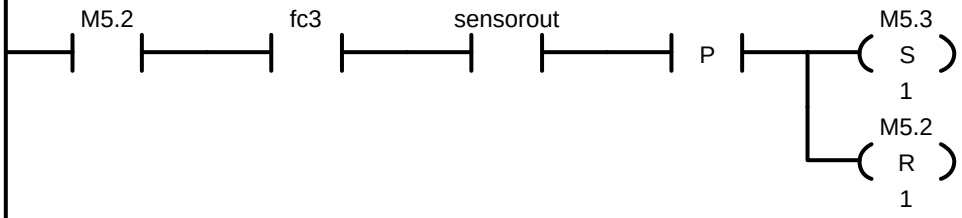
Bloque de conteo descendente.



Network 50

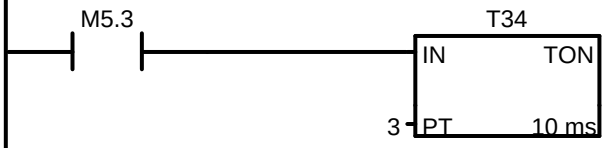
Símbolo	Dirección	Comentario
CDOWN	M1.1	Conteo abajo

Network 51

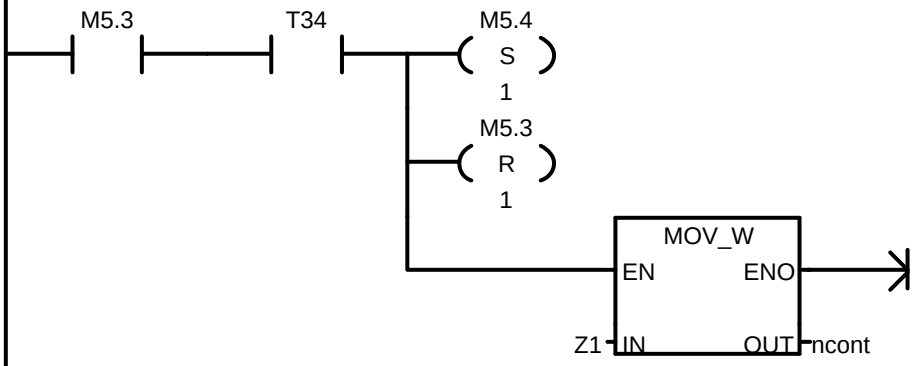


Símbolo	Dirección	Comentario
fc3	E0.7	plc
sensorout	E1.4	plc

Network 52

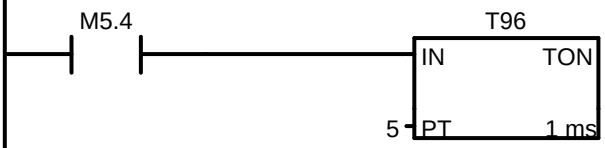


Network 53

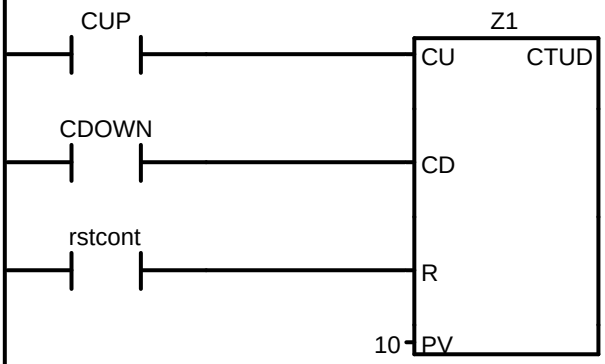


Símbolo	Dirección	Comentario
ncont	VW50	Valor de conteo

Network 54



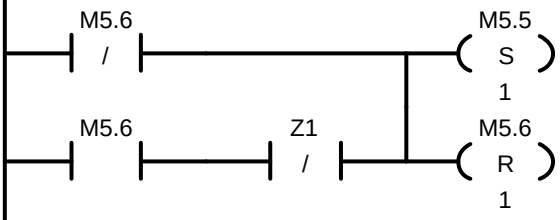
Network 55



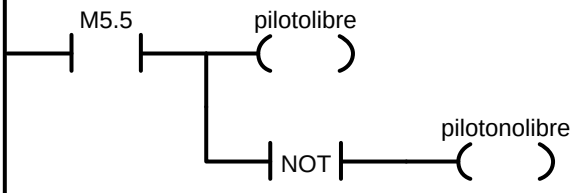
Símbolo	Dirección	Comentario
CDOWN	M1.1	Conteo abajo
CUP	M1.0	Conteo arriba
rstcont	E0.4	plc

Network 56

Indicador luminoso de espacios en el parqueadero.

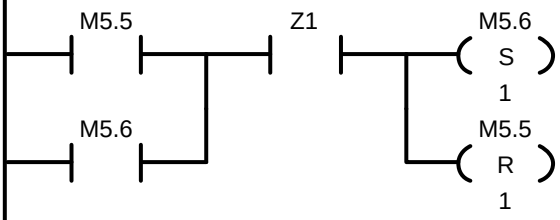


Network 57



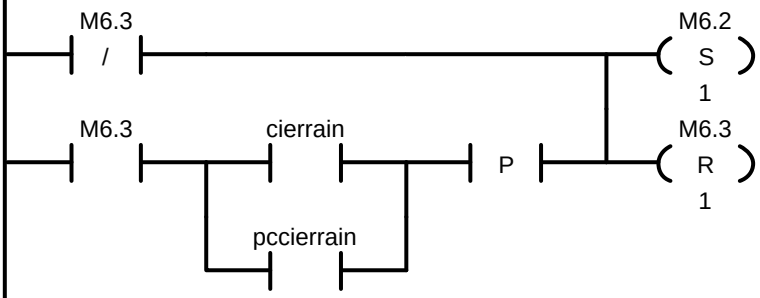
Símbolo	Dirección	Comentario
pilotolibre	A0.6	plc
pilotonolibre	A1.7	plc

Network 58



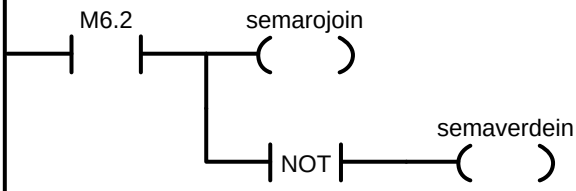
Network 59 ==========
==========
==========

Semaforo de ingreso al parqueadero.



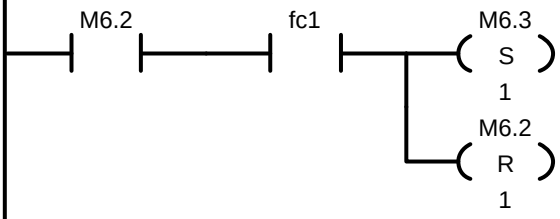
Símbolo	Dirección	Comentario
cierrain	E0.0	plc
pccierrain	M0.0	pc

Network 60



Símbolo	Dirección	Comentario
semarojoin	A1.3	plc
semaverdein	A1.5	plc

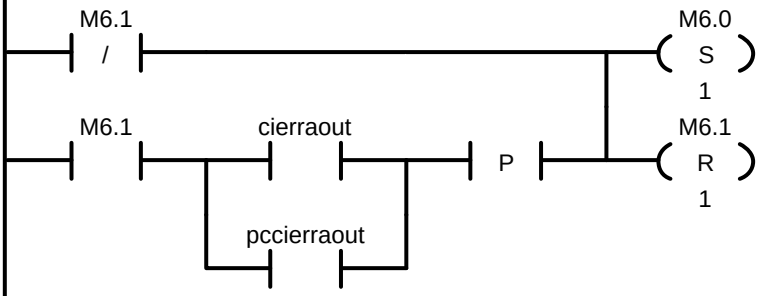
Network 61



Símbolo	Dirección	Comentario
fc1	E0.5	plc

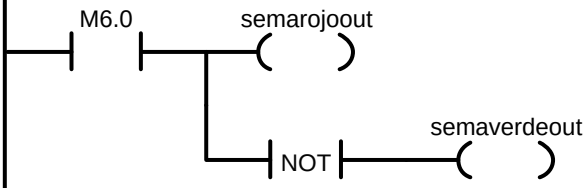
Network 62 ==========

Semaforo de salida del parqueadero.



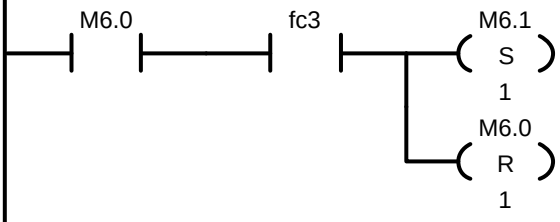
Símbolo	Dirección	Comentario
cierraout	E0.2	plc
pccierraout	M0.2	pc

Network 63



Símbolo	Dirección	Comentario
semarojoout	A1.6	plc
semaverdeout	A1.4	plc

Network 64

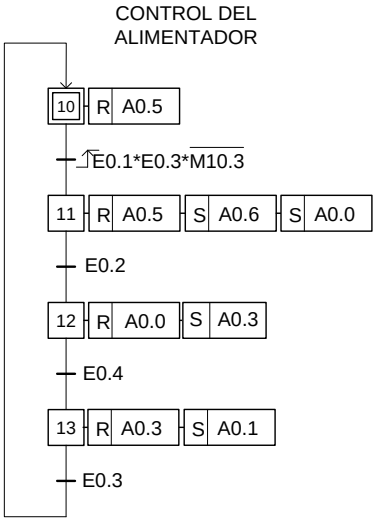


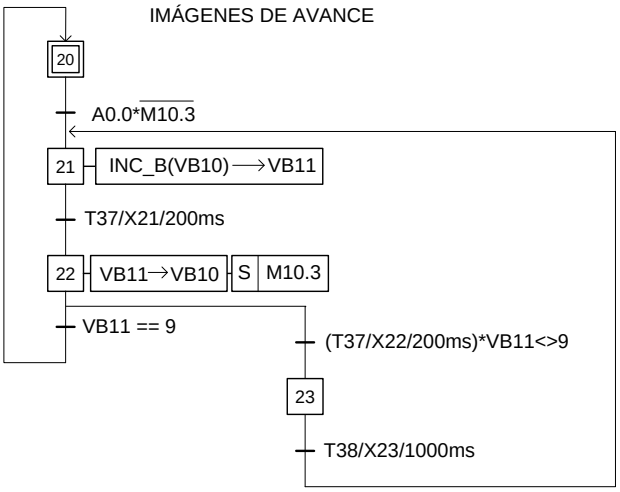
Símbolo	Dirección	Comentario
fc3	E0.7	plc

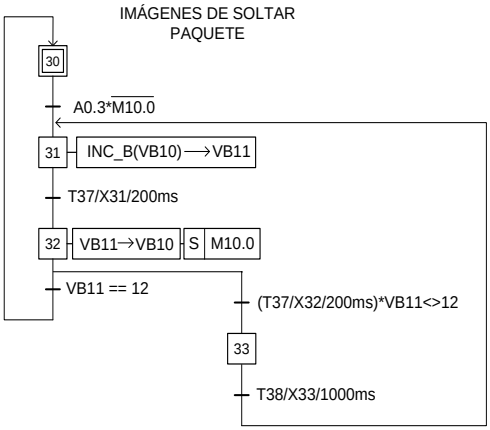
Símbolo	Dirección	Comentario
dummy1	VB21	Mover imagen de ingreso
inbar	VB22	Mover imagen de ingreso
dummy2	VB31	Mover imagen de salida
outbar	VB32	Mover imagen de salida
ncont	VW50	Valor de conteo
CUP	M1.0	Conteo arriba
CDOWN	M1.1	Conteo abajo
cierrain	E0.0	plc
abrein	E0.1	plc
cierraout	E0.2	plc
abreout	E0.3	plc
rstcont	E0.4	plc
fc1	E0.5	plc
fc2	E0.6	plc
fc3	E0.7	plc
fc4	E1.0	plc
fein	E1.1	plc
feout	E1.2	plc
sensorin	E1.3	plc
sensorout	E1.4	plc
COM	M0.5	pc ---- plc
motorinabre	A0.2	plc
motorincierra	A0.3	plc
motoroutabre	A0.4	plc
motoroutcierra	A0.5	plc
pilotolibre	A0.6	plc
pilotofein	A0.7	plc
pilotofeout	A1.0	plc
Goin	A1.1	plc
Gout	A1.2	plc
semarojoin	A1.3	plc
semaverdeout	A1.4	plc
semaverdein	A1.5	plc
semarojoout	A1.6	plc
pilotonolibre	A1.7	plc
pccierrain	M0.0	pc
pcabrein	M0.1	pc
pccierraout	M0.2	pc
pcabreout	M0.3	pc
pcrstcont	M0.4	pc

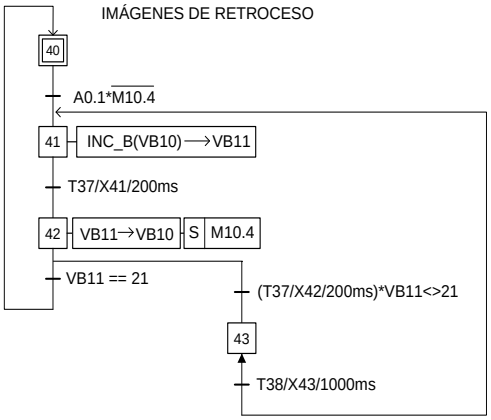
Símbolo	Dirección	Comentario
PRINCIPAL	OB1	CONTROL DE UN PARQUEADERO

Programa principal:

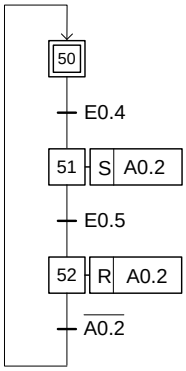




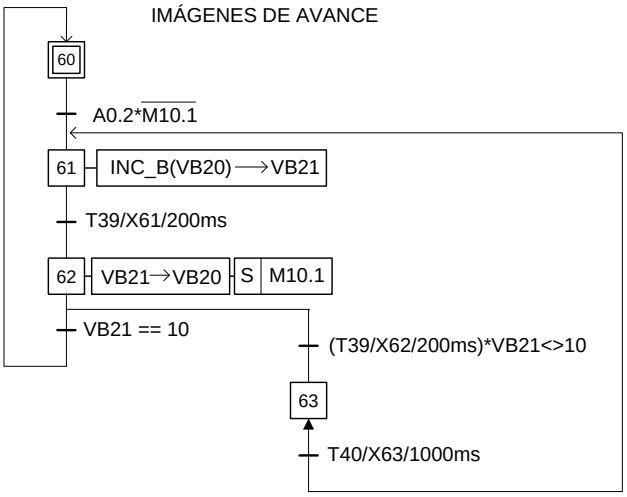




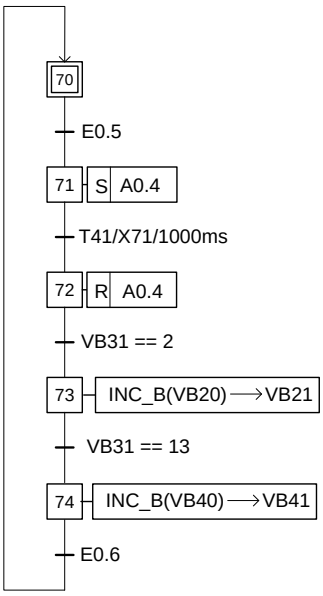
CONTROL DE LA BANDA



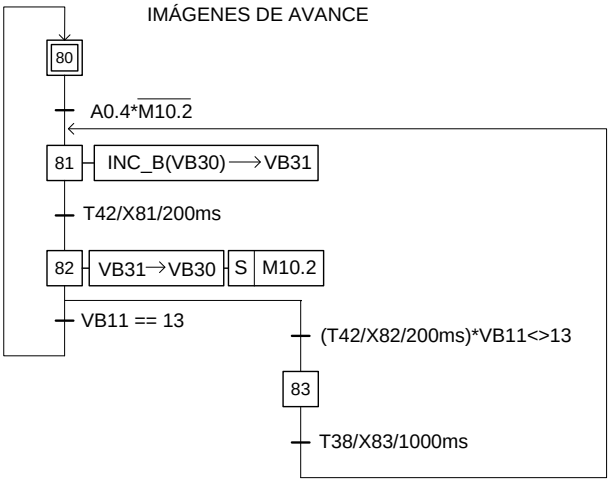
IMÁGENES DE AVANCE

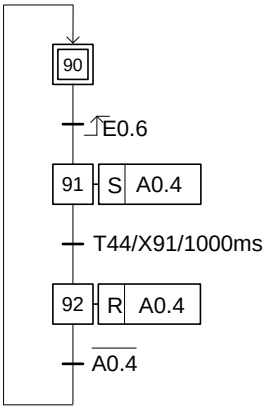


CONTROL DEL COLOCADOR

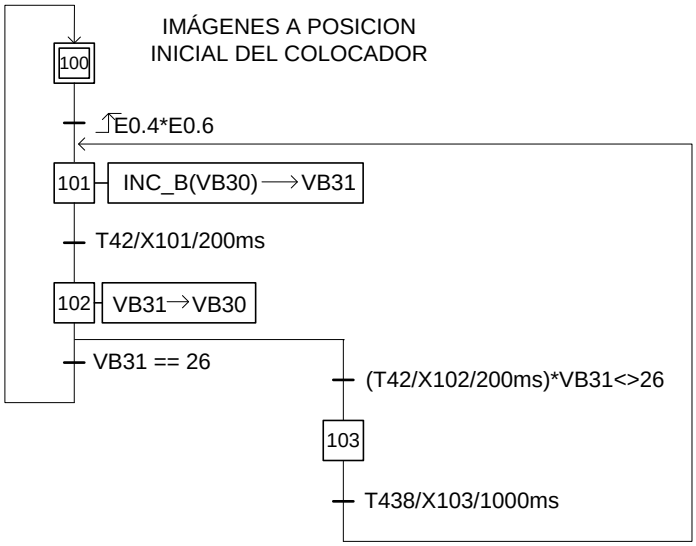


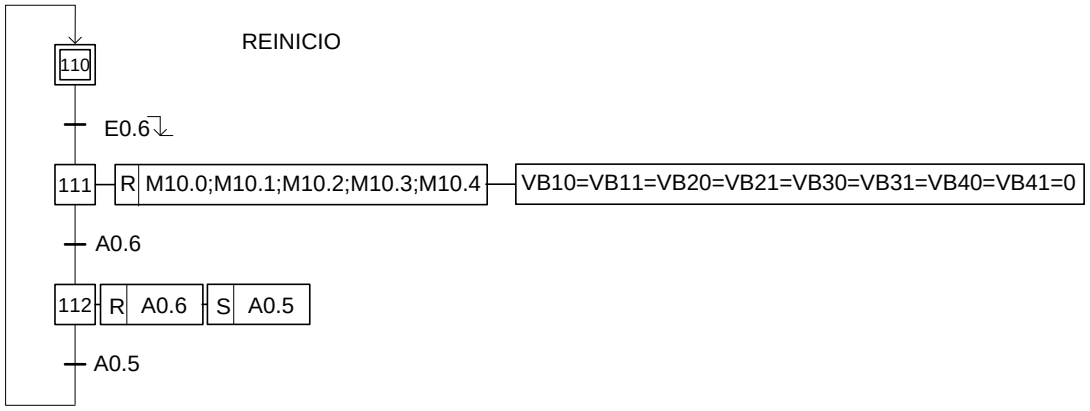
IMÁGENES DE AVANCE

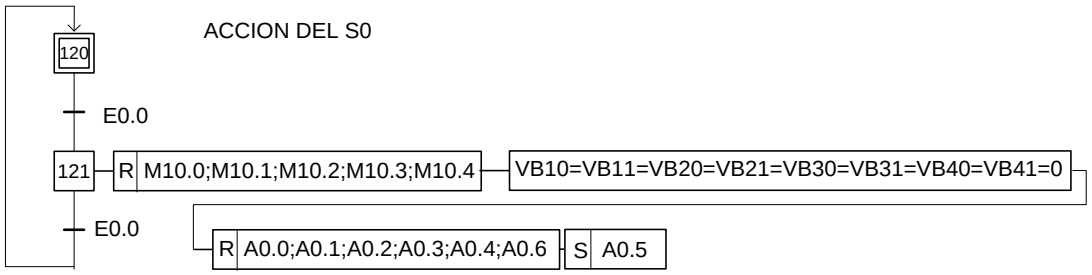


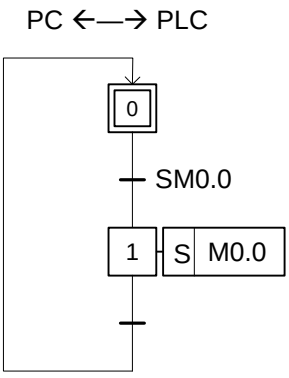


IMÁGENES A POSICION
INICIAL DEL COLOCADOR




REINICIO
110
E0.6
111
R
M10.0;M10.1;M10.2;M10.3;M10.4
VB10=VB11=VB20=VB21=VB30=VB31=VB40=VB41=0
A0.6
112
R
A0.6
S
A0.5
A0.5


ACCION DEL S0
120
E0.0
121
R
M10.0;M10.1;M10.2;M10.3;M10.4
VB10=VB11=VB20=VB21=VB30=VB31=VB40=VB41=0
E0.0
R
A0.0;A0.1;A0.2;A0.3;A0.4;A0.6
S
A0.5


PC ←—→ PLC
0
SM0.0
1
S
M0.0

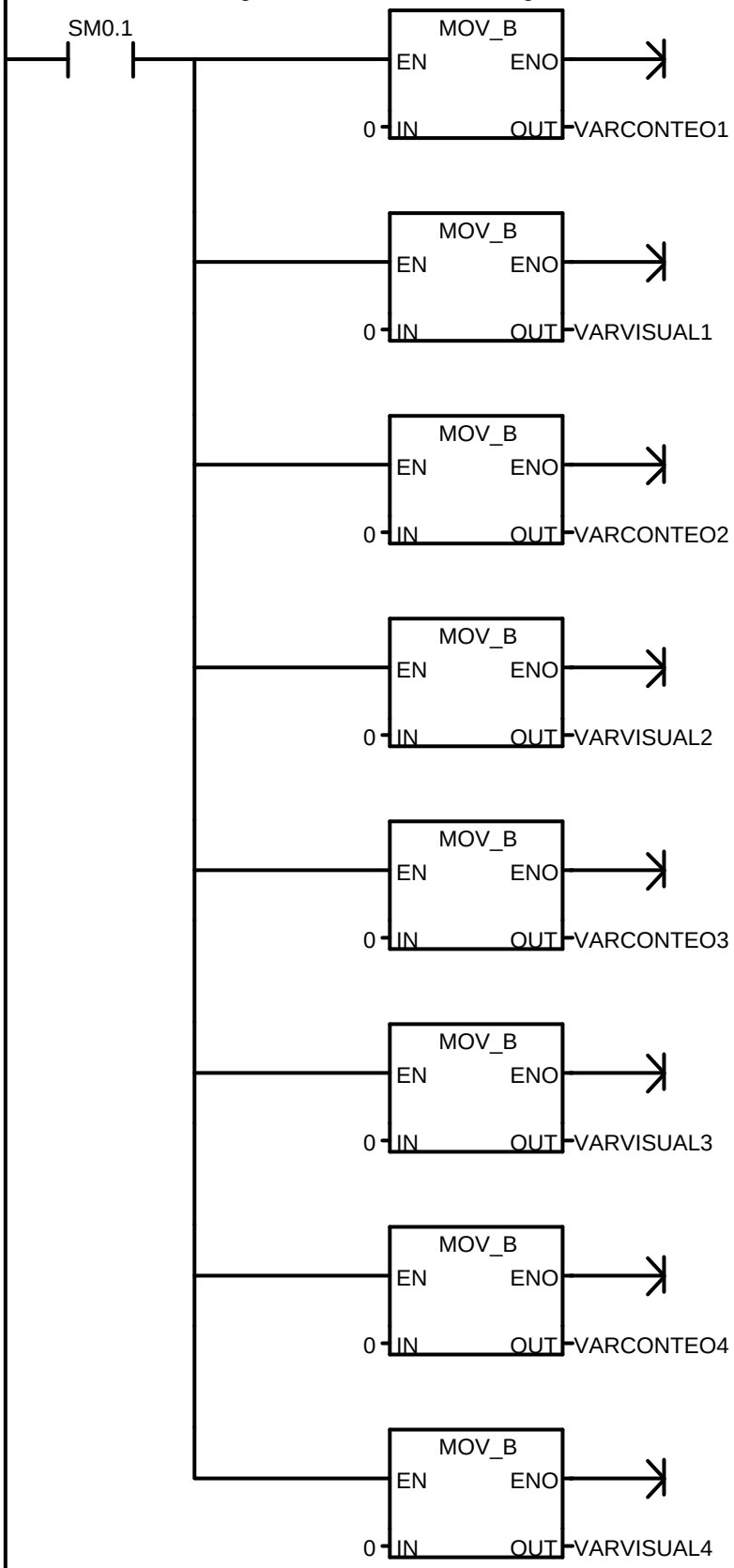
Bloque:	PRINCIPAL	
Autor:		
Fecha de creación:	14.06.2009	17:32:15
Fecha de modificación:	30.05.2010	21:44:04

Símbolo	Tipo var.	Tipo de datos	Comentario
	TEMP		
	TEMP		
	TEMP		
	TEMP		

PROGRAMA PARA CONTROLAR UN SISTEMA SIMPLE DE ACCIONAMIENTO AUTOMATIZADO

Network 1

Inicializacion de los registros: Colocacion de los registros a utilizar a cero (0).

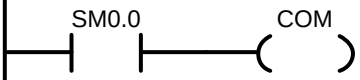


Símbolo	Dirección	Comentario
VARCONTEO1	VB10	CONTEO 1

VARCONTEO2	VB20	CONTEO 2
VARCONTEO3	VB30	CONTEO 3
VARCONTEO4	VB40	CONTEO 4
VARVISUAL1	VB11	VISUAL 1
VARVISUAL2	VB21	VISUAL 2
VARVISUAL3	VB31	VISUAL 3
VARVISUAL4	VB41	VISUAL 4

Network 2

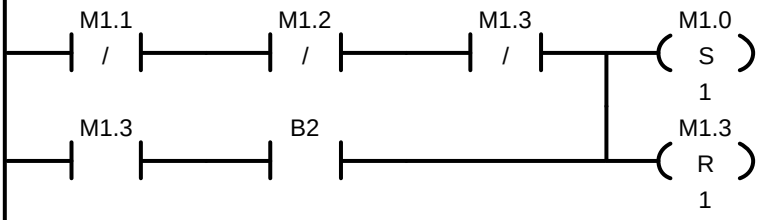
Establecimiento de la comunicacion PC <----> PLC



Símbolo	Dirección	Comentario
COM	M0.0	COMUNICACION

Network 3 ==========

Control del alimentador

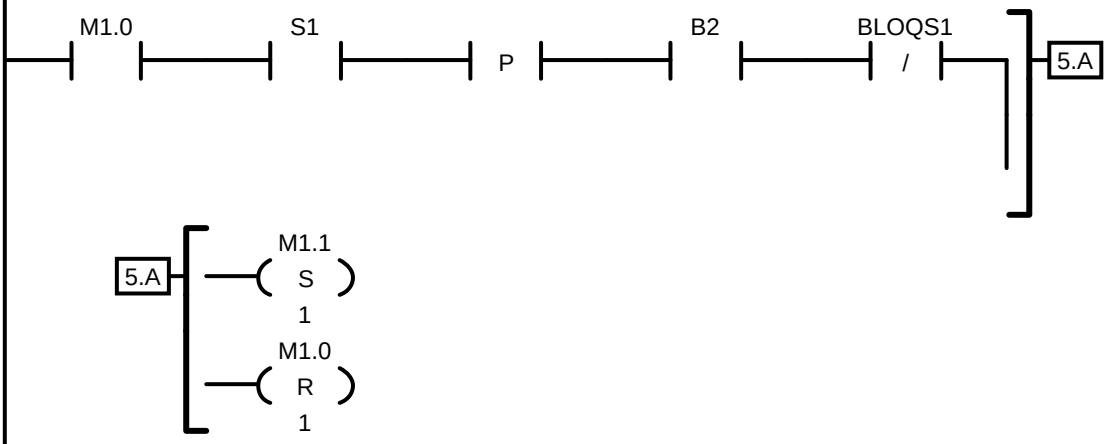


Símbolo	Dirección	Comentario
B2	E0.3	SENSOR DE ALIMENTADOR 2

Network 4

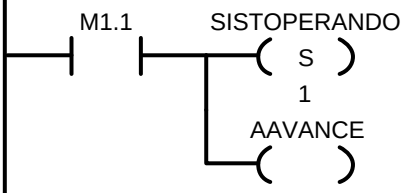
Símbolo	Dirección	Comentario
SISTLISTO	A0.5	LISTO PARA USO
SISTOPERANDO	A0.6	SISTEMA EN OPERACION

Network 5



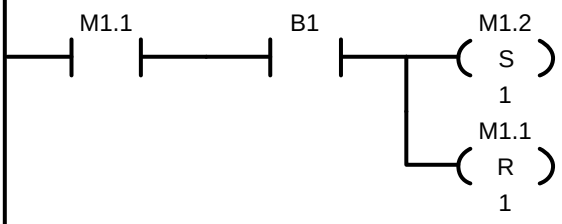
Símbolo	Dirección	Comentario
B2	E0.3	SENSOR DE ALIMENTADOR 2
BLOQS1	M10.3	BLOQUEO DE START
S1	E0.1	INICIO DE ACTIVIDAD

Network 6



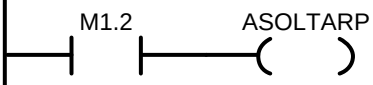
Símbolo	Dirección	Comentario
AAVANCE	A0.0	ALIMENTADOR EN AVANCE
SISTOPERANDO	A0.6	SISTEMA EN OPERACION

Network 7



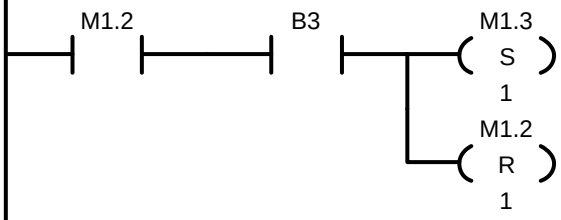
Símbolo	Dirección	Comentario
B1	E0.2	SENSOR DE ALIMENTADOR 1

Network 8



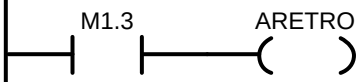
Símbolo	Dirección	Comentario
ASOLTARP	A0.3	SOLTAR CARGA

Network 9



Símbolo	Dirección	Comentario
B3	E0.4	SENSOR DE BANDA 1

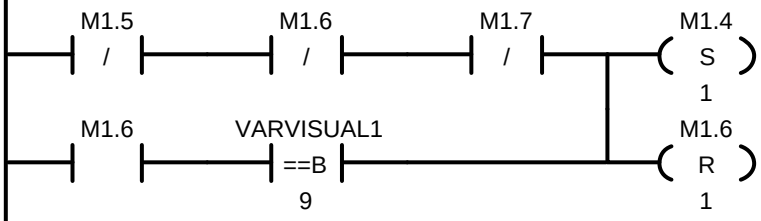
Network 10



Símbolo	Dirección	Comentario
ARETRO	A0.1	ALIMENTADOR EN RETROCESO

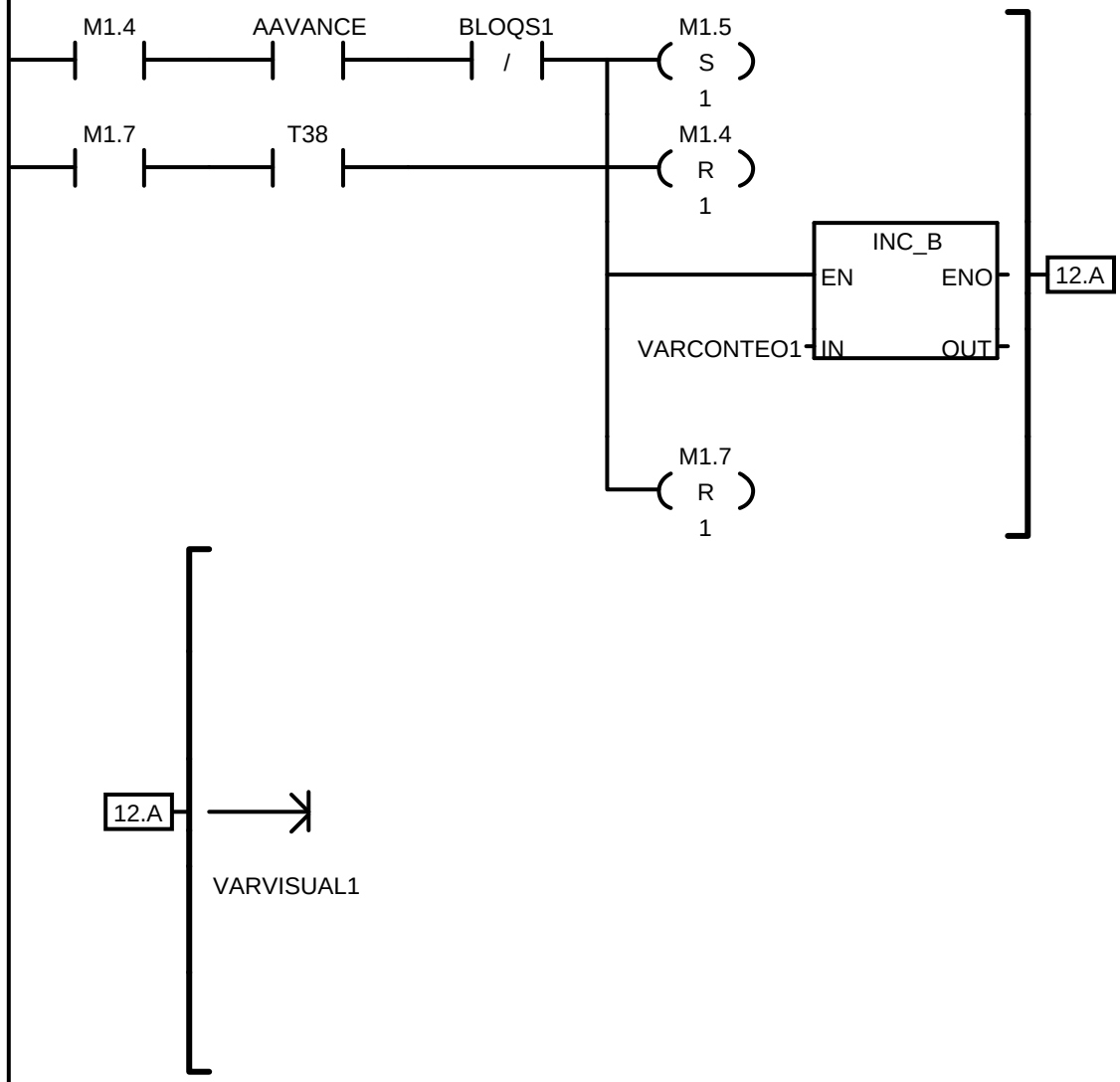
Network 11

Imagenes de movimiento en avance.



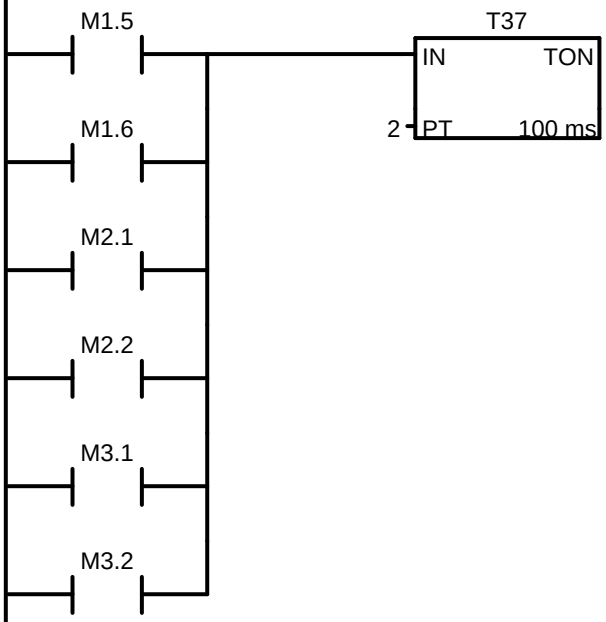
Símbolo	Dirección	Comentario
VARVISUAL1	VB11	VISUAL 1

Network 12

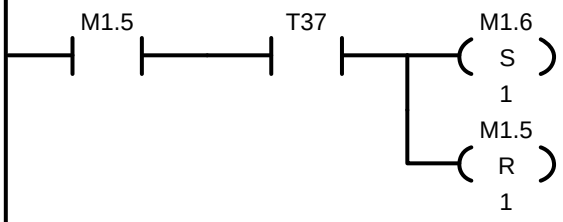


Símbolo	Dirección	Comentario
AAVANCE	A0.0	ALIMENTADOR EN AVANCE
BLOQS1	M10.3	BLOQUEO DE START
VARCONTEO1	VB10	CONTEO 1
VARVISUAL1	VB11	VISUAL 1

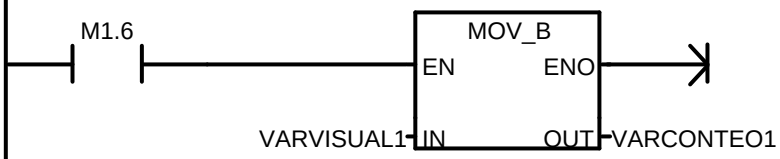
Network 13



Network 14

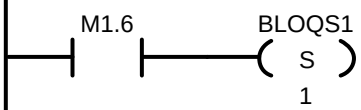


Network 15



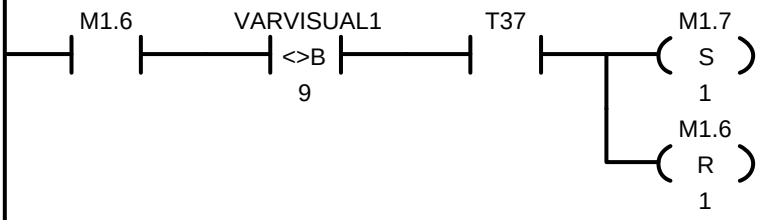
Símbolo	Dirección	Comentario
VARCONTEO1	VB10	CONTEO 1
VARVISUAL1	VB11	VISUAL 1

Network 16



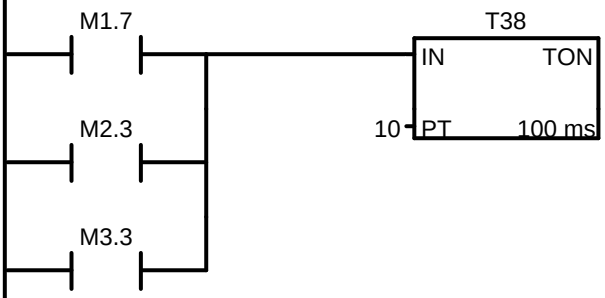
Símbolo	Dirección	Comentario
BLOQS1	M10.3	BLOQUEO DE START

Network 17



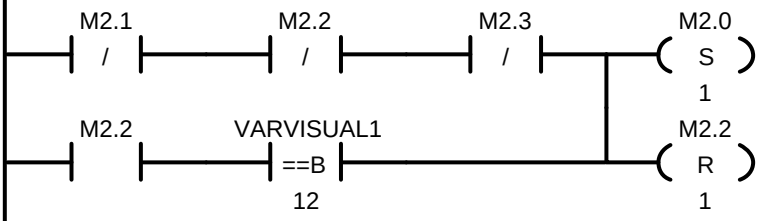
Símbolo	Dirección	Comentario
VARVISUAL1	VB11	VISUAL 1

Network 18



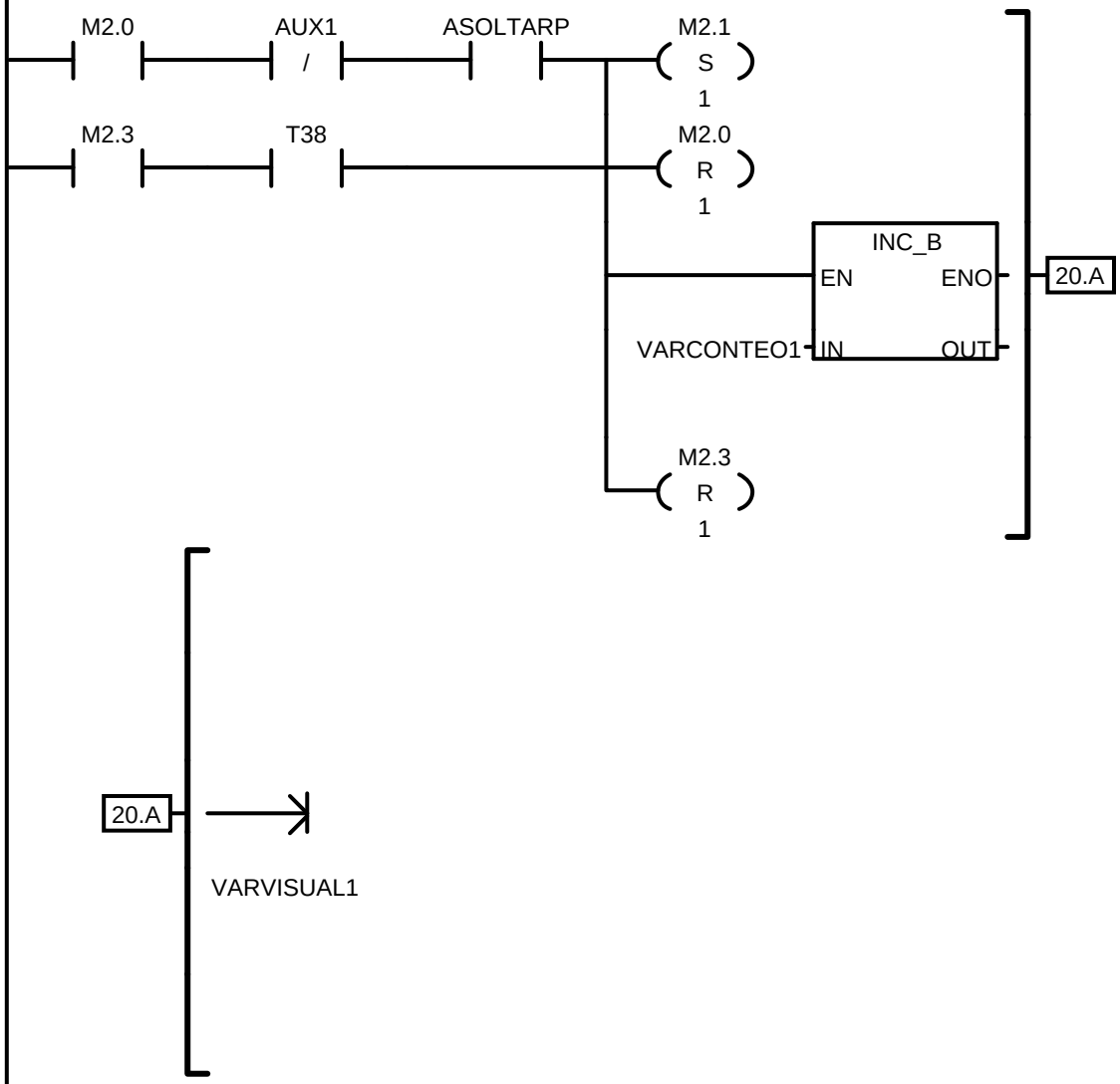
Network 19

Imagenes de soltar el paquete en la banda.



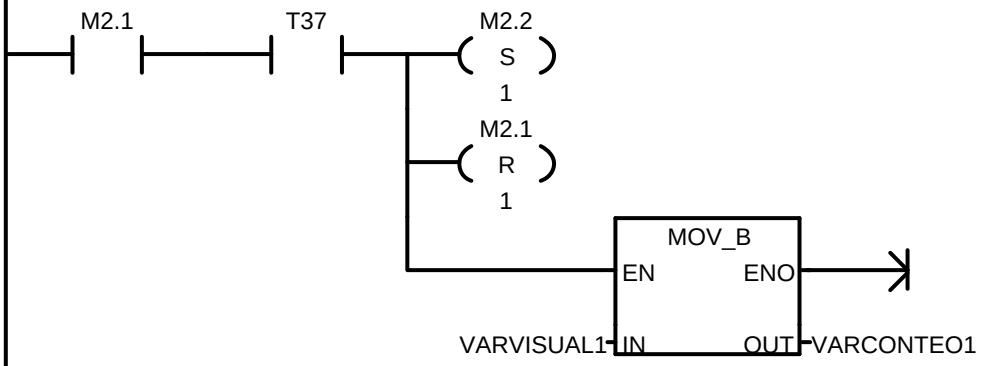
Símbolo	Dirección	Comentario
VARVISUAL1	VB11	VISUAL 1

Network 20



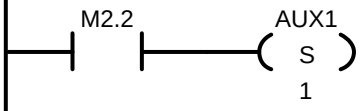
Símbolo	Dirección	Comentario
ASOLTARP	A0.3	SOLTAR CARGA
AUX1	M10.0	MARCA AUXILIAR 1
VARCONTEO1	VB10	CONTEO 1
VARVISUAL1	VB11	VISUAL 1

Network 21



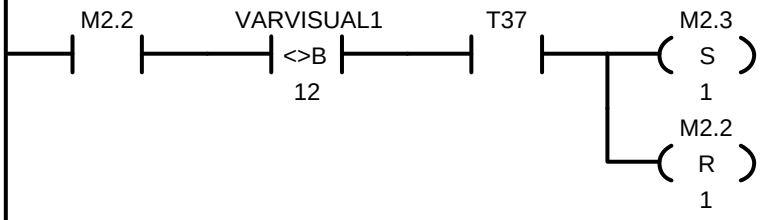
Símbolo	Dirección	Comentario
VARCONTEO1	VB10	CONTEO 1
VARVISUAL1	VB11	VISUAL 1

Network 22



Símbolo	Dirección	Comentario
AUX1	M10.0	MARCA AUXILIAR 1

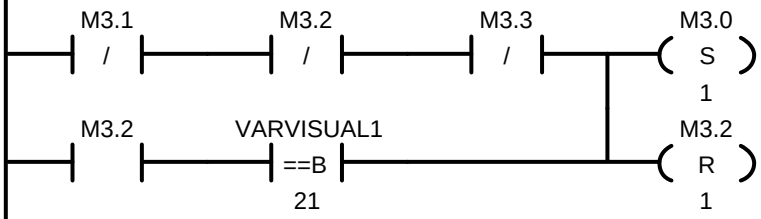
Network 23



Símbolo	Dirección	Comentario
VARVISUAL1	VB11	VISUAL 1

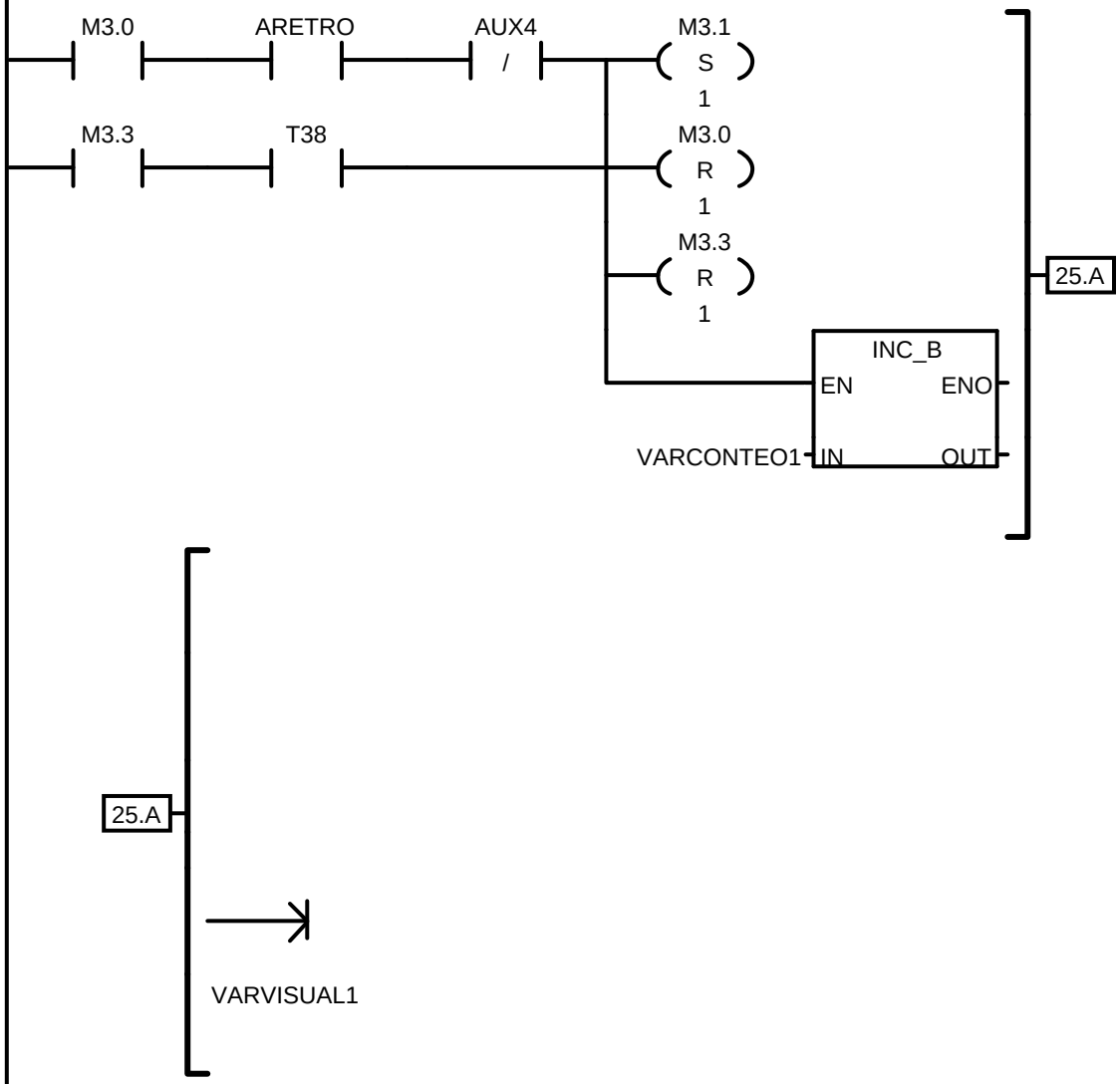
Network 24

Imagenes de retroceso del alimentador.



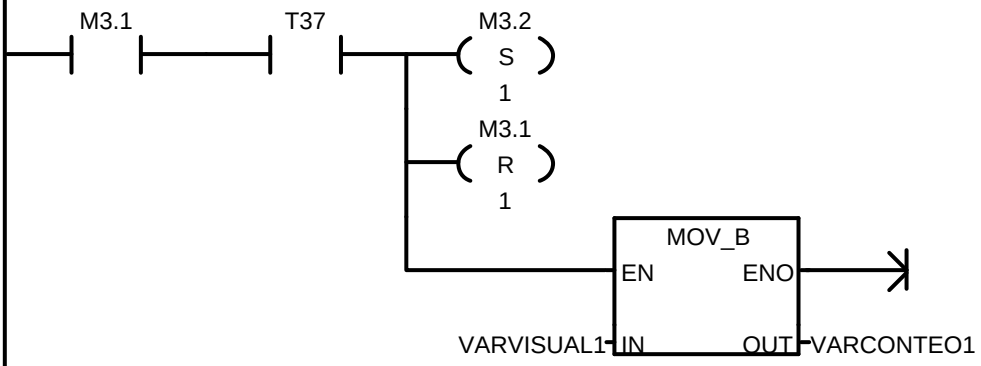
Símbolo	Dirección	Comentario
VARVISUAL1	VB11	VISUAL 1

Network 25



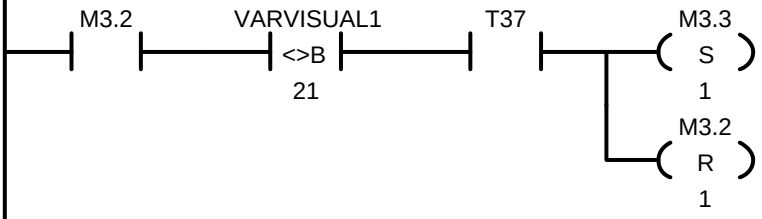
Símbolo	Dirección	Comentario
ARETRO	A0.1	ALIMENTADOR EN RETROCESO
AUX4	M10.4	MARCA AUXILIAR 4
VARCONTEO1	VB10	CONTEO 1
VARVISUAL1	VB11	VISUAL 1

Network 26



Símbolo	Dirección	Comentario
VARCONTEO1	VB10	CONTEO 1
VARVISUAL1	VB11	VISUAL 1

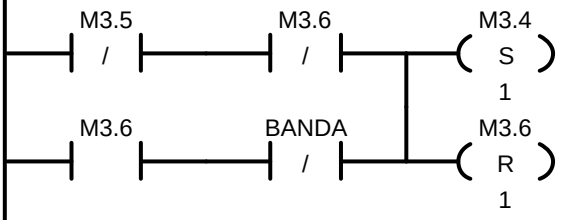
Network 27



Símbolo	Dirección	Comentario
VARVISUAL1	VB11	VISUAL 1

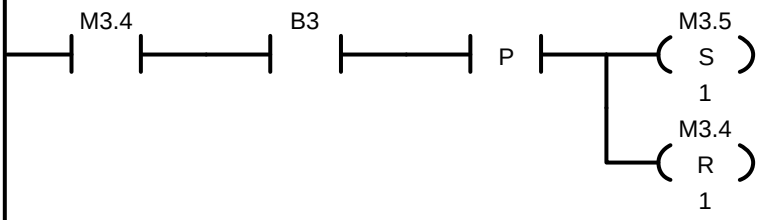
Network 28 ==========

Control de la banda transportadora.



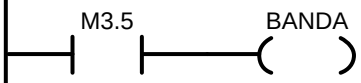
Símbolo	Dirección	Comentario
BANDA	A0.2	BANDA DE TRANSPORTE

Network 29



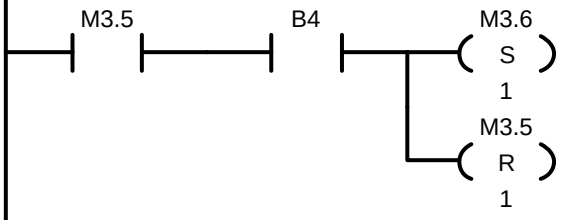
Símbolo	Dirección	Comentario
B3	E0.4	SENSOR DE BANDA 1

Network 30



Símbolo	Dirección	Comentario
BANDA	A0.2	BANDA DE TRANSPORTE

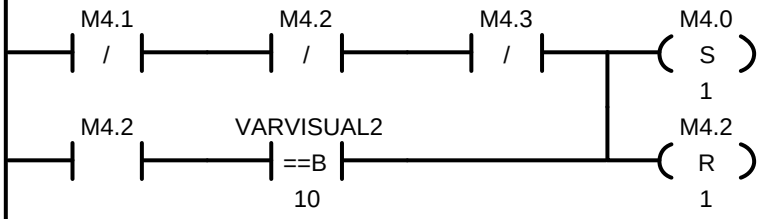
Network 31



Símbolo	Dirección	Comentario
B4	E0.5	SENSOR DE BANDA 2

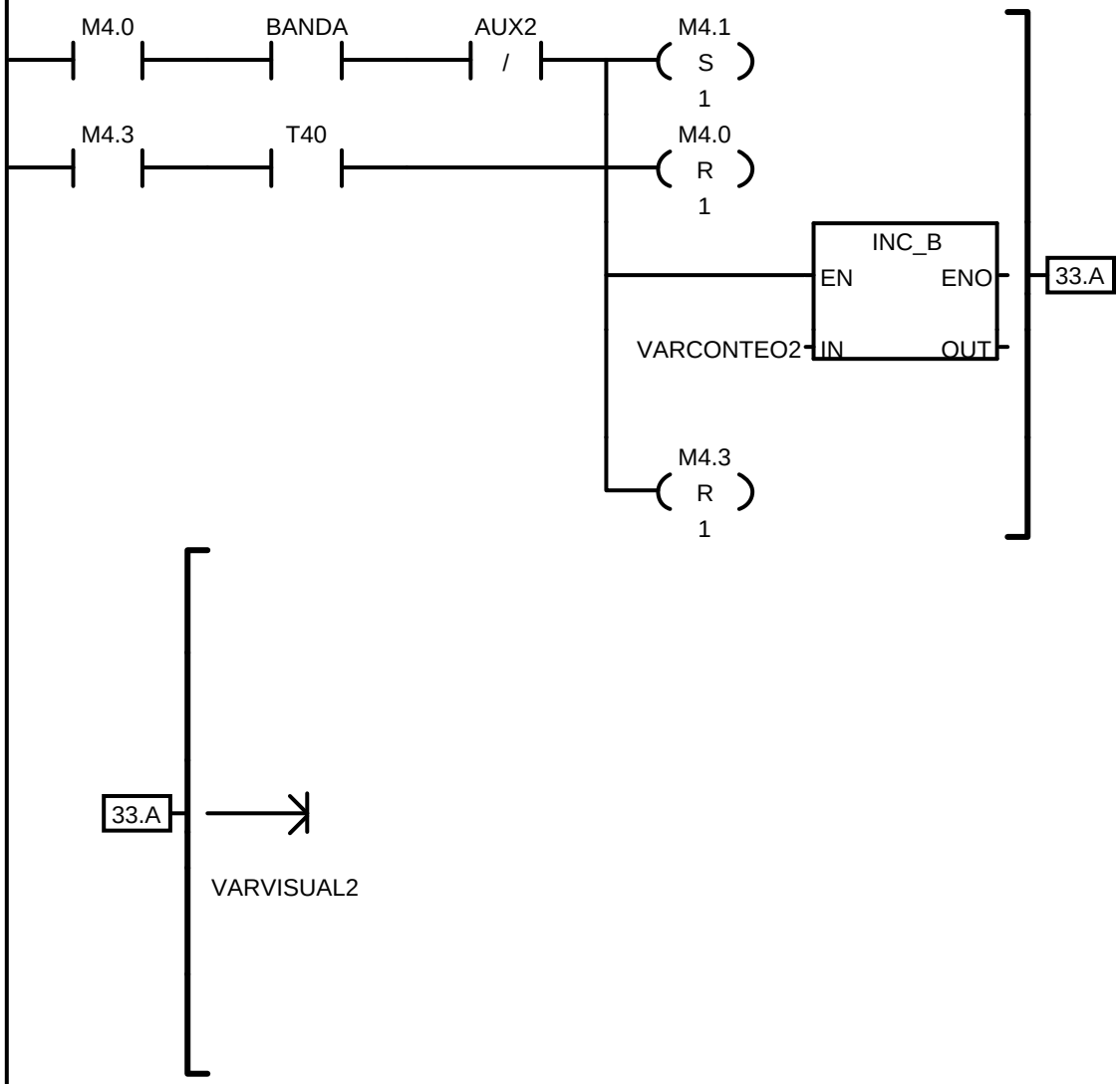
Network 32

Imagenes de la banda activa.



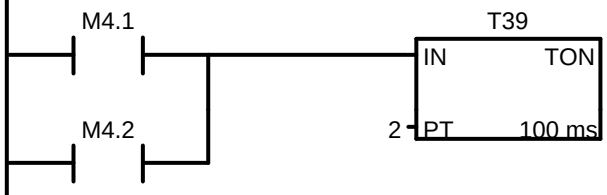
Símbolo	Dirección	Comentario
VARVISUAL2	VB21	VISUAL 2

Network 33

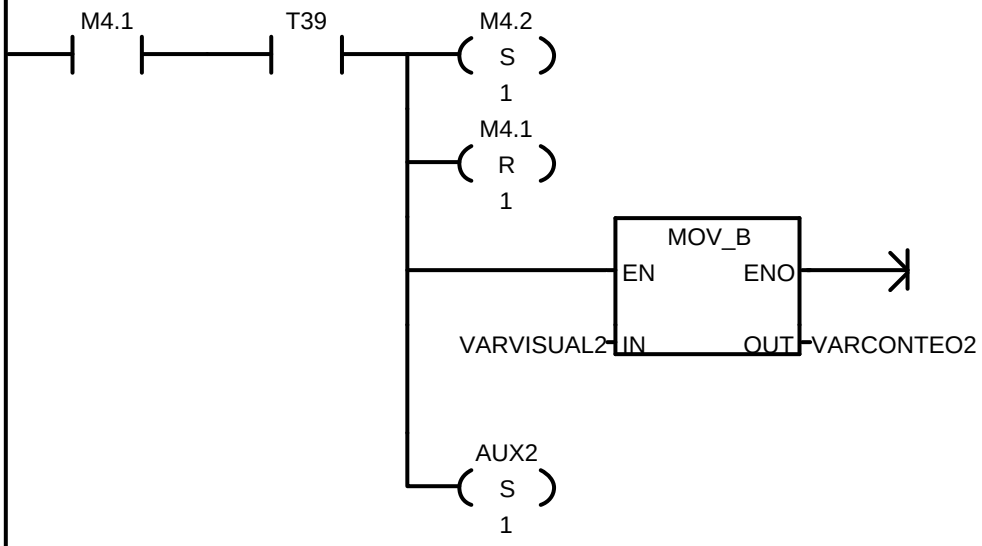


Símbolo	Dirección	Comentario
AUX2	M10.1	MARCA AUXILIAR 2
BANDA	A0.2	BANDA DE TRANSPORTE
VARCONTEO2	VB20	CONTEO 2
VARVISUAL2	VB21	VISUAL 2

Network 34

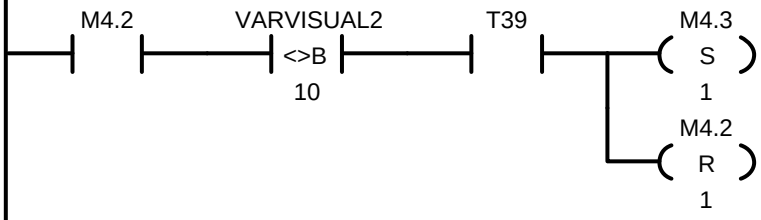


Network 35



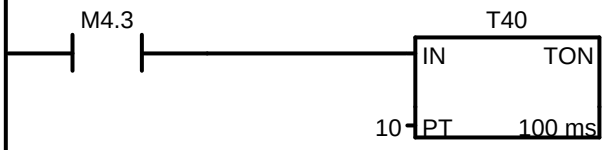
Símbolo	Dirección	Comentario
AUX2	M10.1	MARCA AUXILIAR 2
VARCONTEO2	VB20	CONTEO 2
VARVISUAL2	VB21	VISUAL 2

Network 36



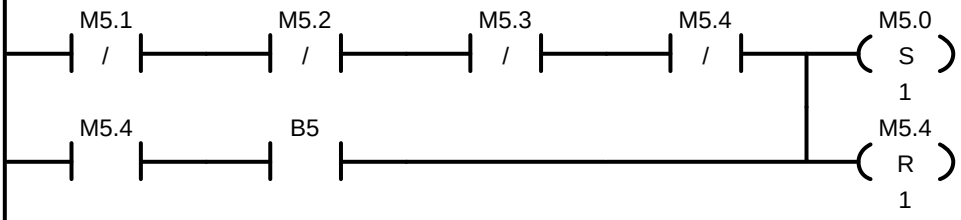
Símbolo	Dirección	Comentario
VARVISUAL2	VB21	VISUAL 2

Network 37



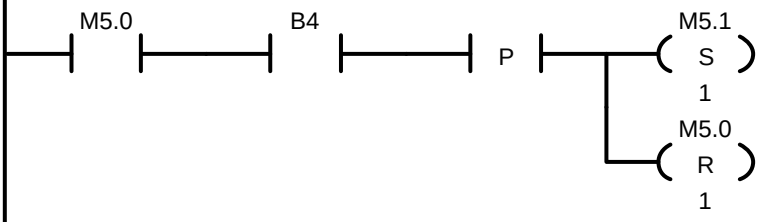
Network 38 ==========
==========

Control del movimiento del colocador.



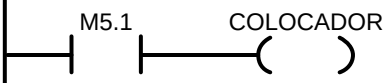
Símbolo	Dirección	Comentario
B5	E0.6	SENSOR DE RECEPCION

Network 39



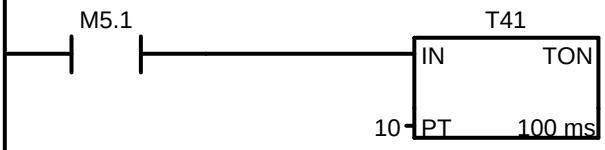
Símbolo	Dirección	Comentario
B4	E0.5	SENSOR DE BANDA 2

Network 40

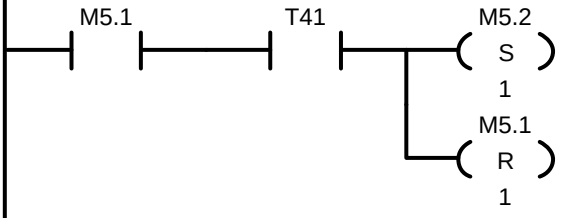


Símbolo	Dirección	Comentario
COLOCADOR	A0.4	BRAZO DE ROBOT

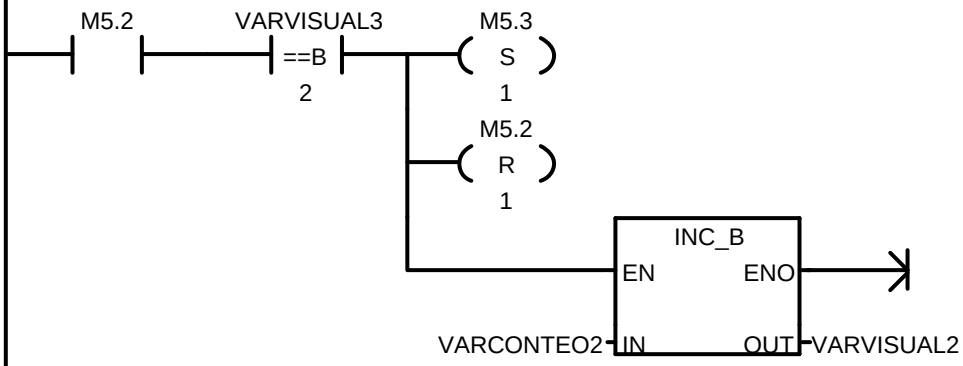
Network 41



Network 42

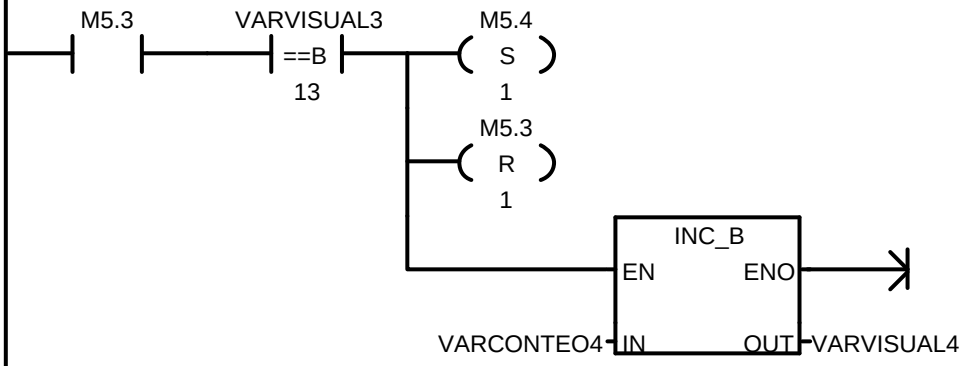


Network 43



Símbolo	Dirección	Comentario
VARCONTEO2	VB20	CONTEO 2
VARVISUAL2	VB21	VISUAL 2
VARVISUAL3	VB31	VISUAL 3

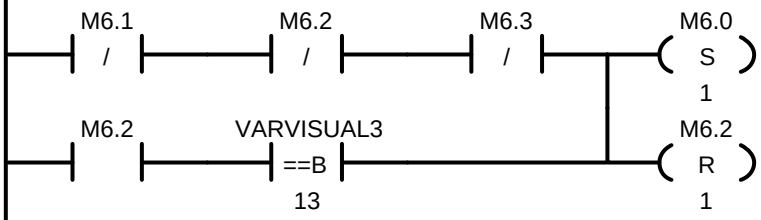
Network 44



Símbolo	Dirección	Comentario
VARCONTEO4	VB40	CONTEO 4
VARVISUAL3	VB31	VISUAL 3
VARVISUAL4	VB41	VISUAL 4

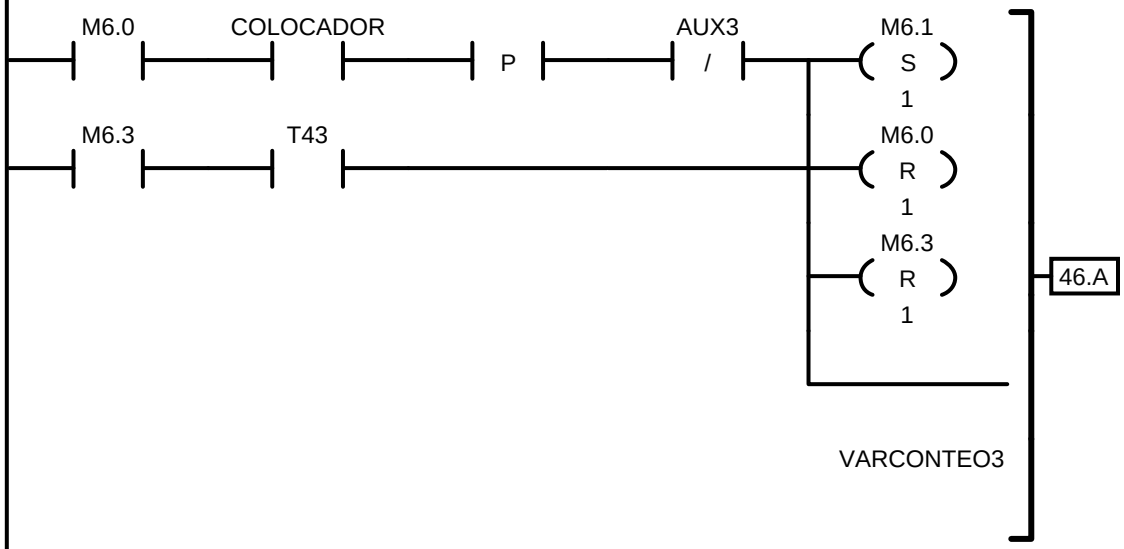
Network 45

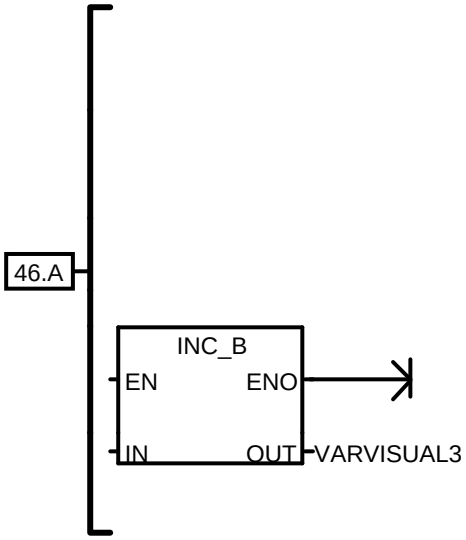
Visualizacion del avance del colocador.



Símbolo	Dirección	Comentario
VARVISUAL3	VB31	VISUAL 3

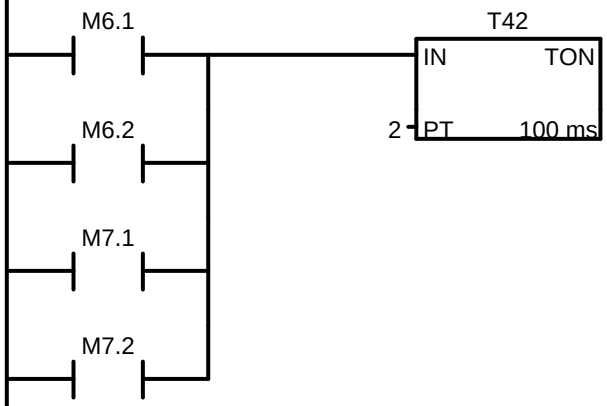
Network 46



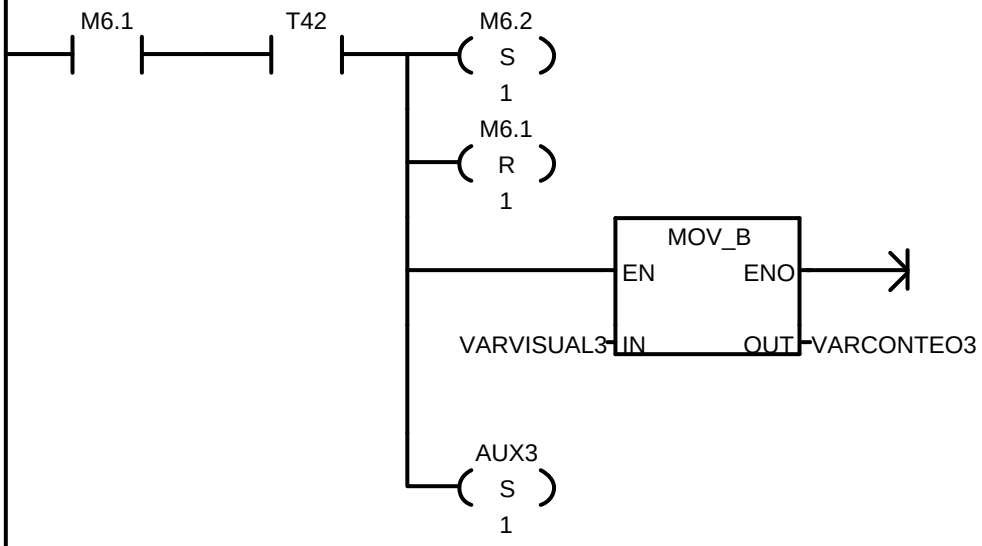


Símbolo	Dirección	Comentario
AUX3	M10.2	MARCA AUXILIAR 3
COLOCADOR	A0.4	BRAZO DE ROBOT
VARCONTEO3	VB30	CONTEO 3
VARVISUAL3	VB31	VISUAL 3

Network 47

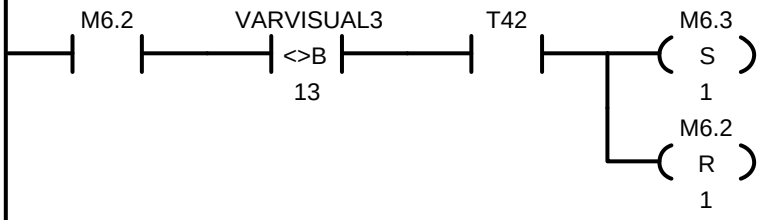


Network 48



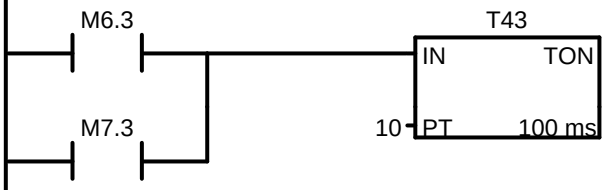
Símbolo	Dirección	Comentario
AUX3	M10.2	MARCA AUXILIAR 3
VARCONTEO3	VB30	CONTEO 3
VARVISUAL3	VB31	VISUAL 3

Network 49



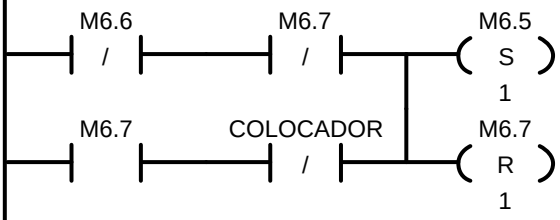
Símbolo	Dirección	Comentario
VARVISUAL3	VB31	VISUAL 3

Network 50



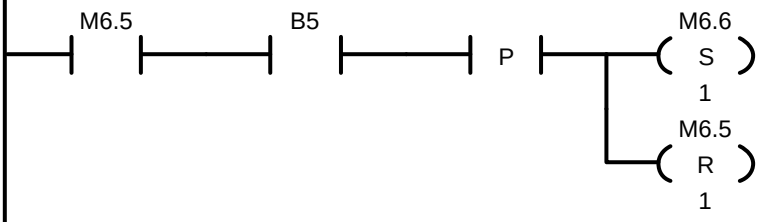
Network 51 ==========

Activacion del colocador en retorno a posicion inicial.



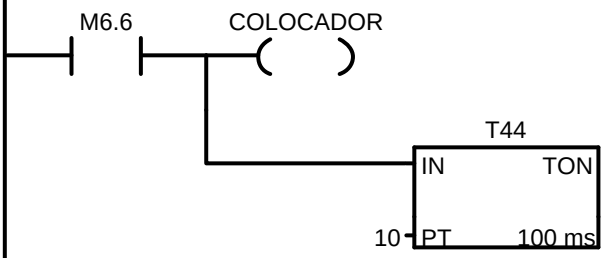
Símbolo	Dirección	Comentario
COLOCADOR	A0.4	BRAZO DE ROBOT

Network 52



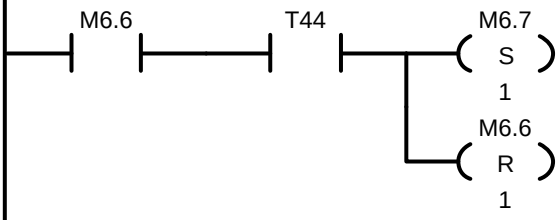
Símbolo	Dirección	Comentario
B5	E0.6	SENSOR DE RECEPCION

Network 53



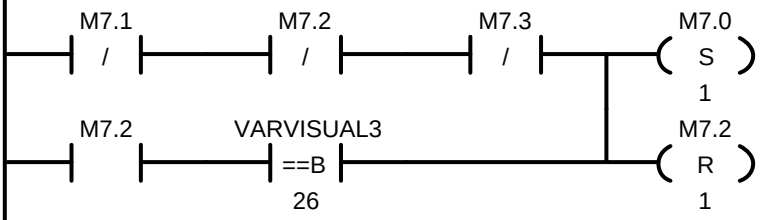
Símbolo	Dirección	Comentario
COLOCADOR	A0.4	BRAZO DE ROBOT

Network 54



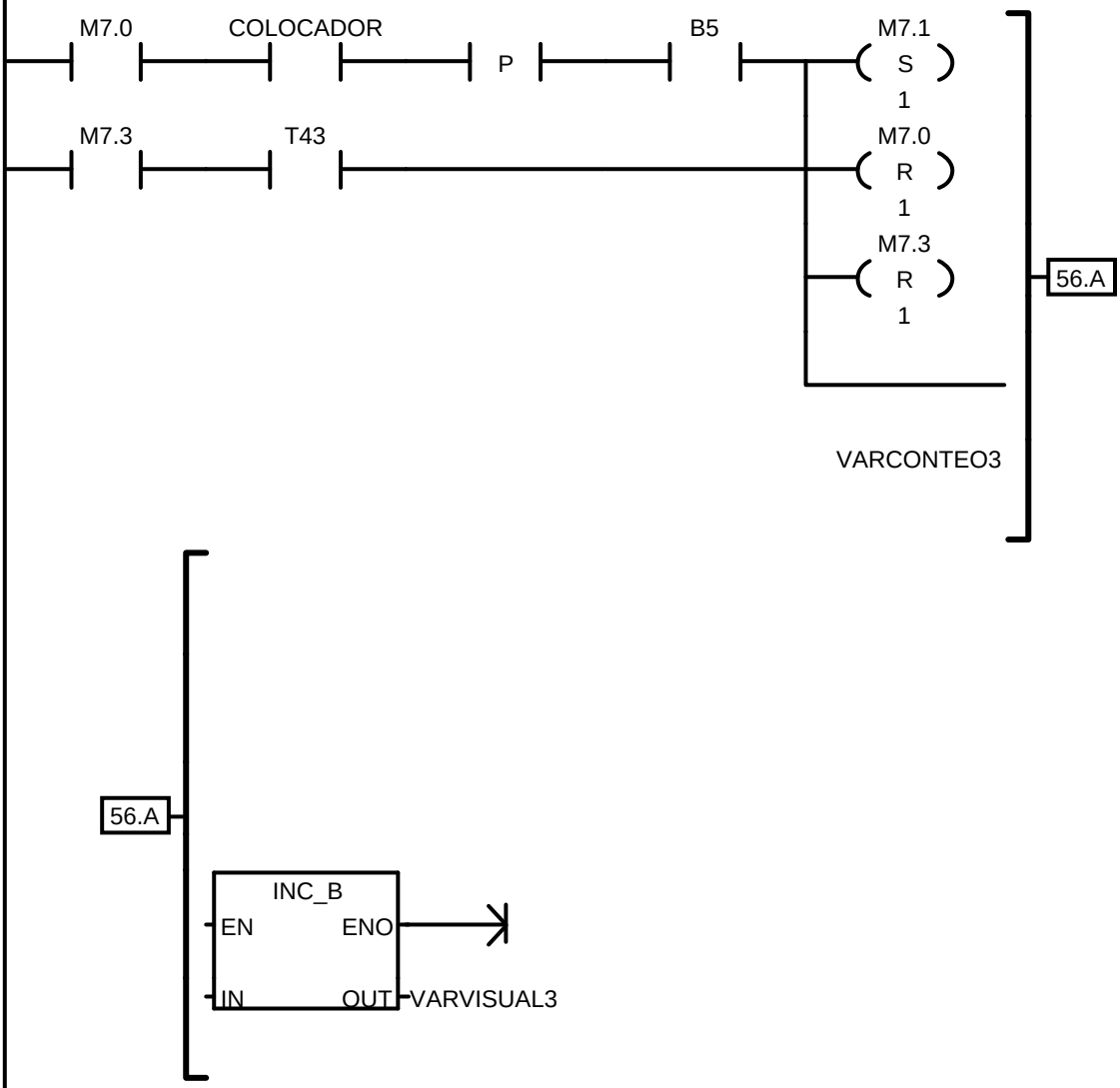
Network 55 ==========

Visualizacion del colocador en retorno.



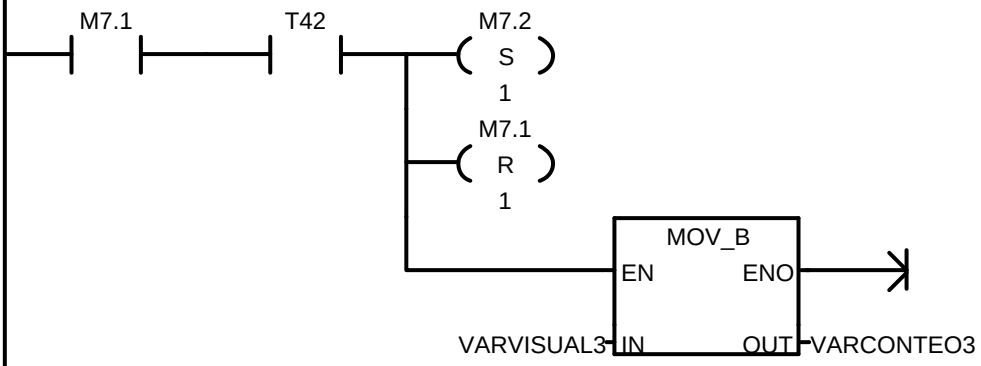
Símbolo	Dirección	Comentario
VARVISUAL3	VB31	VISUAL 3

Network 56



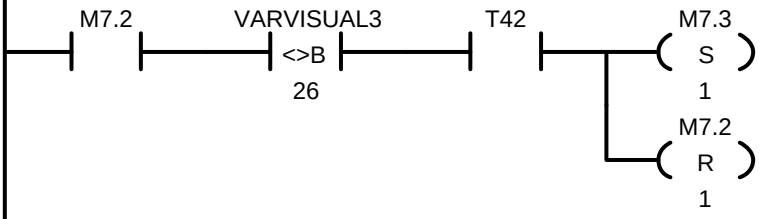
Símbolo	Dirección	Comentario
B5	E0.6	SENSOR DE RECEPCION
COLOCADOR	A0.4	BRAZO DE ROBOT
VARCONTEO3	VB30	CONTEO 3
VARVISUAL3	VB31	VISUAL 3

Network 57



Símbolo	Dirección	Comentario
VARCONTEO3	VB30	CONTEO 3
VARVISUAL3	VB31	VISUAL 3

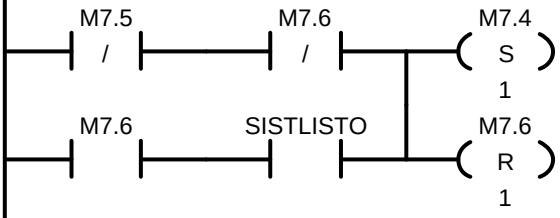
Network 58



Símbolo	Dirección	Comentario
VARVISUAL3	VB31	VISUAL 3

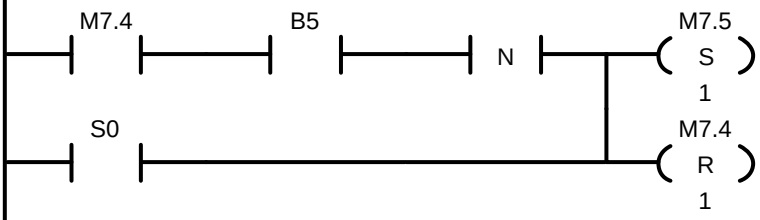
Network 59 ==========

Reinicio ==> dado al retirar el paquete de la recepcion. El sensor deja de ver al paquete.



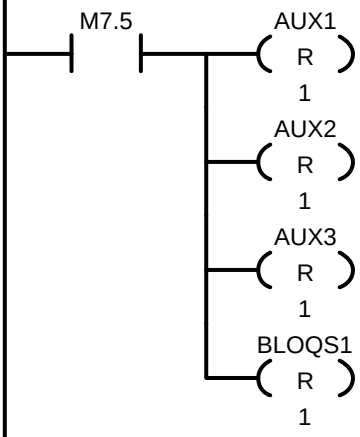
Símbolo	Dirección	Comentario
SISTLISTO	A0.5	LISTO PARA USO

Network 60



Símbolo	Dirección	Comentario
B5	E0.6	SENSOR DE RECEPCION
S0	E0.0	PARO TOTAL

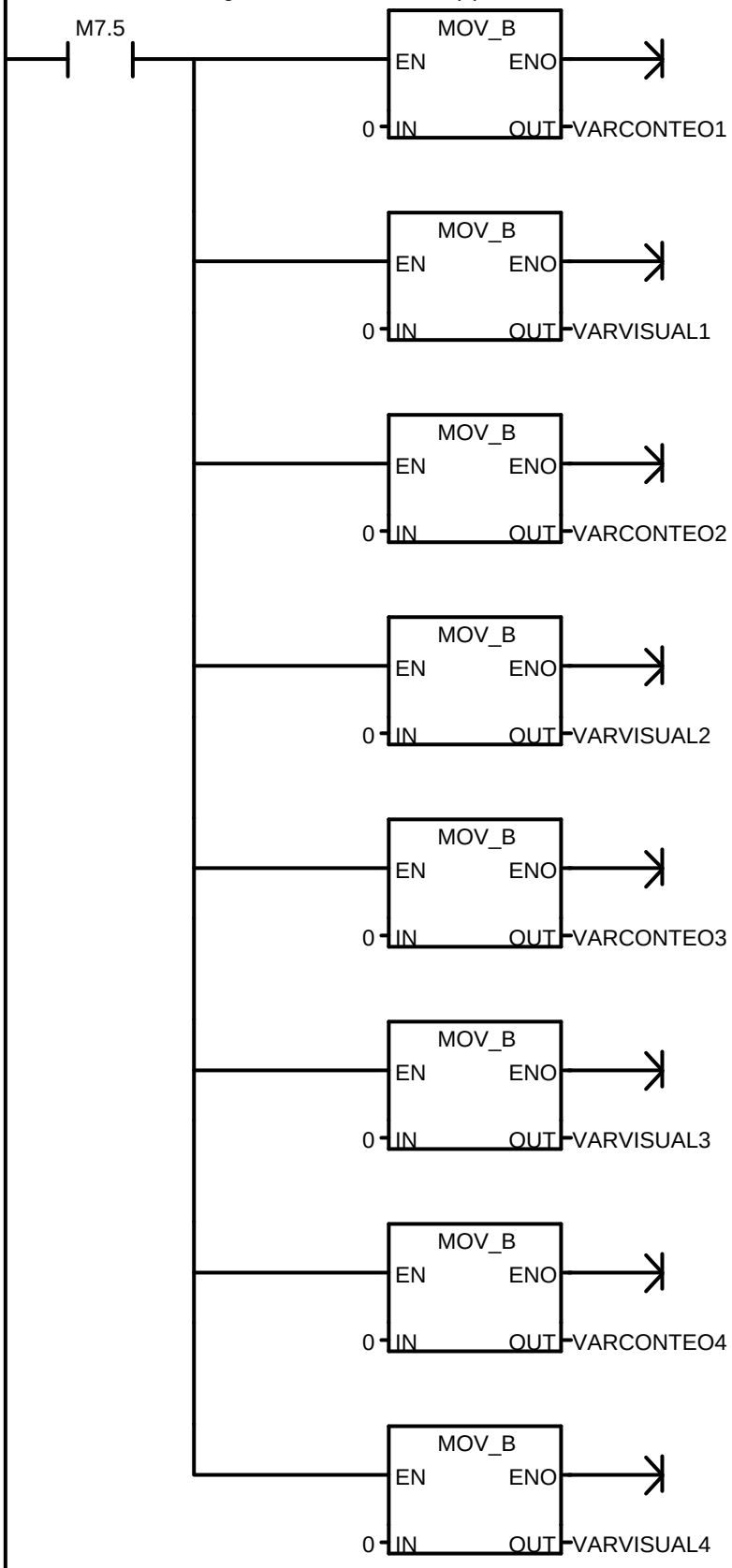
Network 61



Símbolo	Dirección	Comentario
AUX1	M10.0	MARCA AUXILIAR 1
AUX2	M10.1	MARCA AUXILIAR 2
AUX3	M10.2	MARCA AUXILIAR 3
BLOQS1	M10.3	BLOQUEO DE START

Network 62

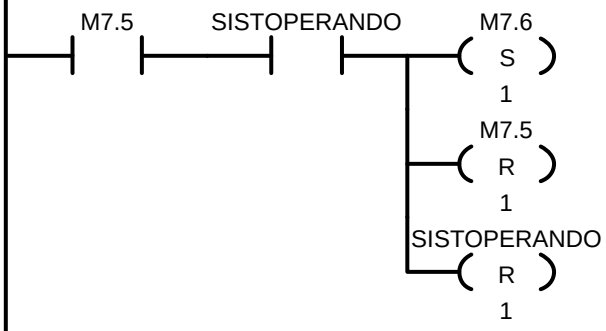
Colocacion de los registros a utilizar a cero (0).



Símbolo	Dirección	Comentario
VARCONTEO1	VB10	CONTEO 1
VARCONTEO2	VB20	CONTEO 2

VARCONTEO3	VB30	CONTEO 3
VARCONTEO4	VB40	CONTEO 4
VARVISUAL1	VB11	VISUAL 1
VARVISUAL2	VB21	VISUAL 2
VARVISUAL3	VB31	VISUAL 3
VARVISUAL4	VB41	VISUAL 4

Network 63

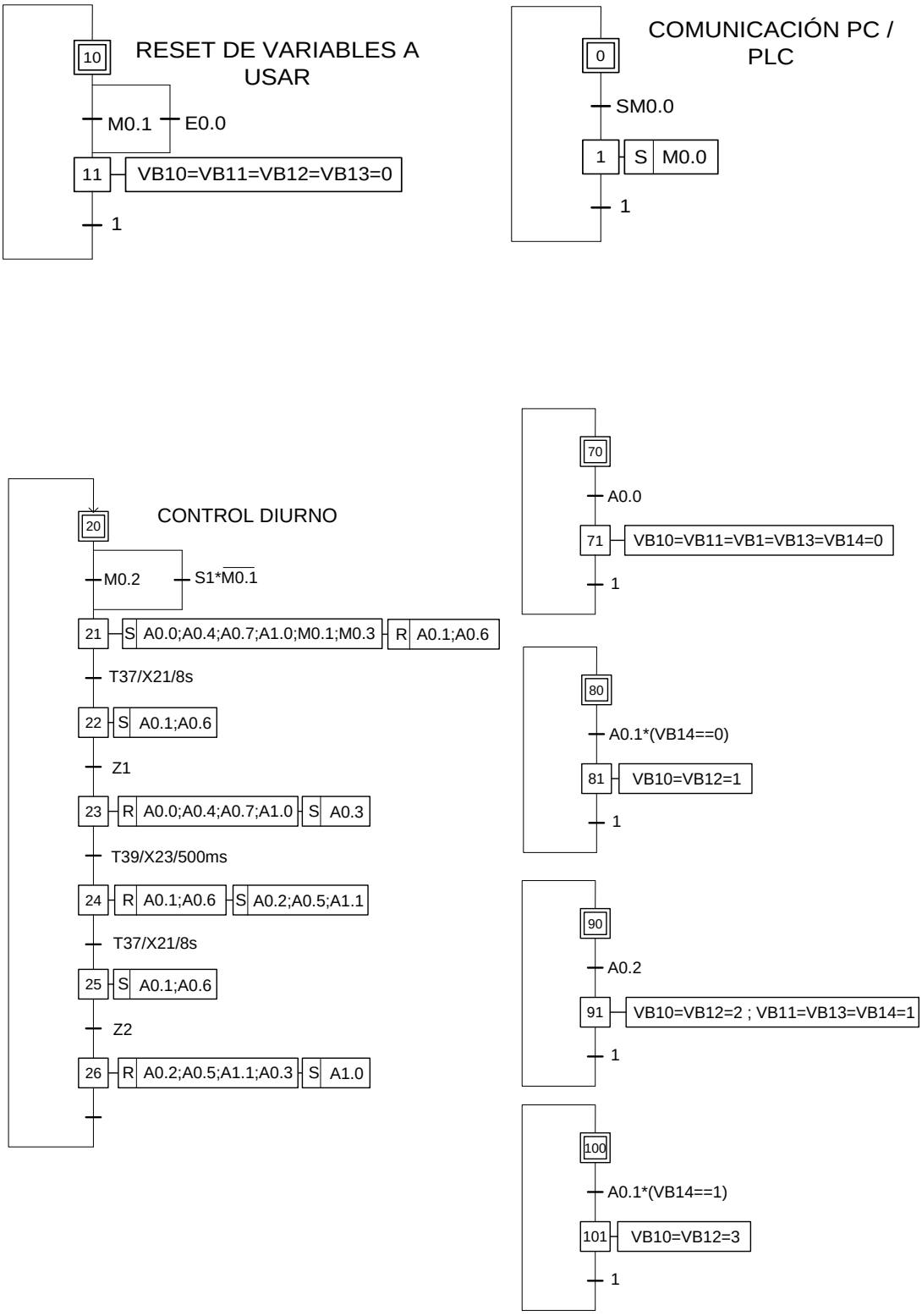


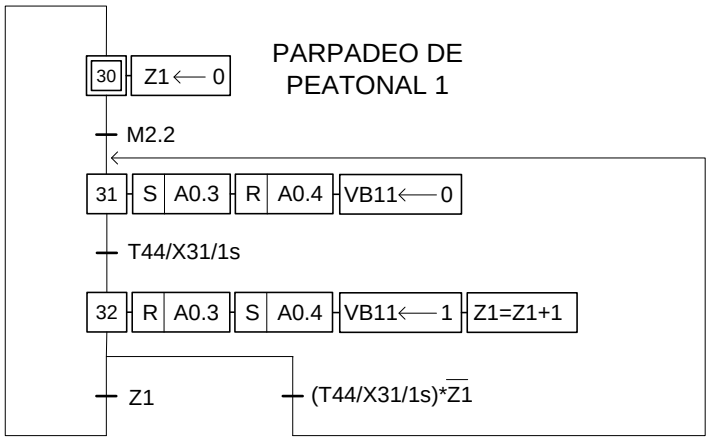
Símbolo	Dirección	Comentario
SISTOPERANDO	A0.6	SISTEMA EN OPERACION

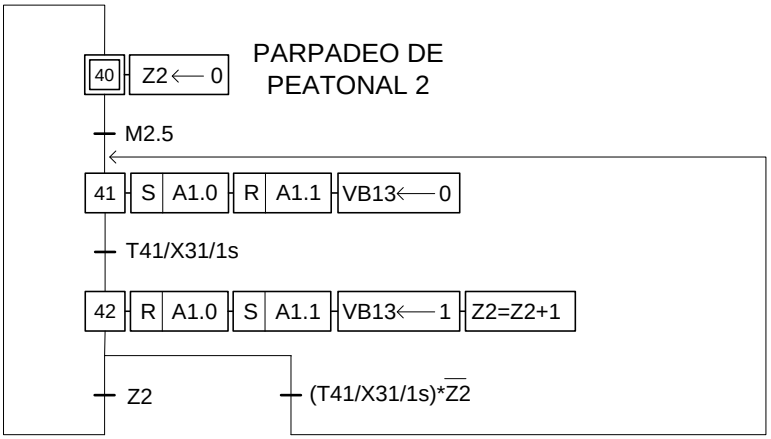
Símbolo	Dirección	Comentario
S0	E0.0	PARO TOTAL
S1	E0.1	INICIO DE ACTIVIDAD
COM	M0.0	COMUNICACION
B1	E0.2	SENSOR DE ALIMENTADOR 1
B2	E0.3	SENSOR DE ALIMENTADOR 2
B3	E0.4	SENSOR DE BANDA 1
B4	E0.5	SENSOR DE BANDA 2
B5	E0.6	SENSOR DE RECEPCION
SISTLISTO	A0.5	LISTO PARA USO
SISTOPERANDO	A0.6	SISTEMA EN OPERACION
AAVANCE	A0.0	ALIMENTADOR EN AVANCE
ARETRO	A0.1	ALIMENTADOR EN RETROCESO
BANDA	A0.2	BANDA DE TRANSPORTE
ASOLTARP	A0.3	SOLTAR CARGA
COLOCADOR	A0.4	BRAZO DE ROBOT
VARCONTEO1	VB10	CONTEO 1
VARVISUAL1	VB11	VISUAL 1
VARCONTEO2	VB20	CONTEO 2
VARVISUAL2	VB21	VISUAL 2
VARCONTEO3	VB30	CONTEO 3
VARVISUAL3	VB31	VISUAL 3
VARCONTEO4	VB40	CONTEO 4
VARVISUAL4	VB41	VISUAL 4
AUX1	M10.0	MARCA AUXILIAR 1
AUX2	M10.1	MARCA AUXILIAR 2
AUX3	M10.2	MARCA AUXILIAR 3
BLOQS1	M10.3	BLOQUEO DE START
AUX4	M10.4	MARCA AUXILIAR 4

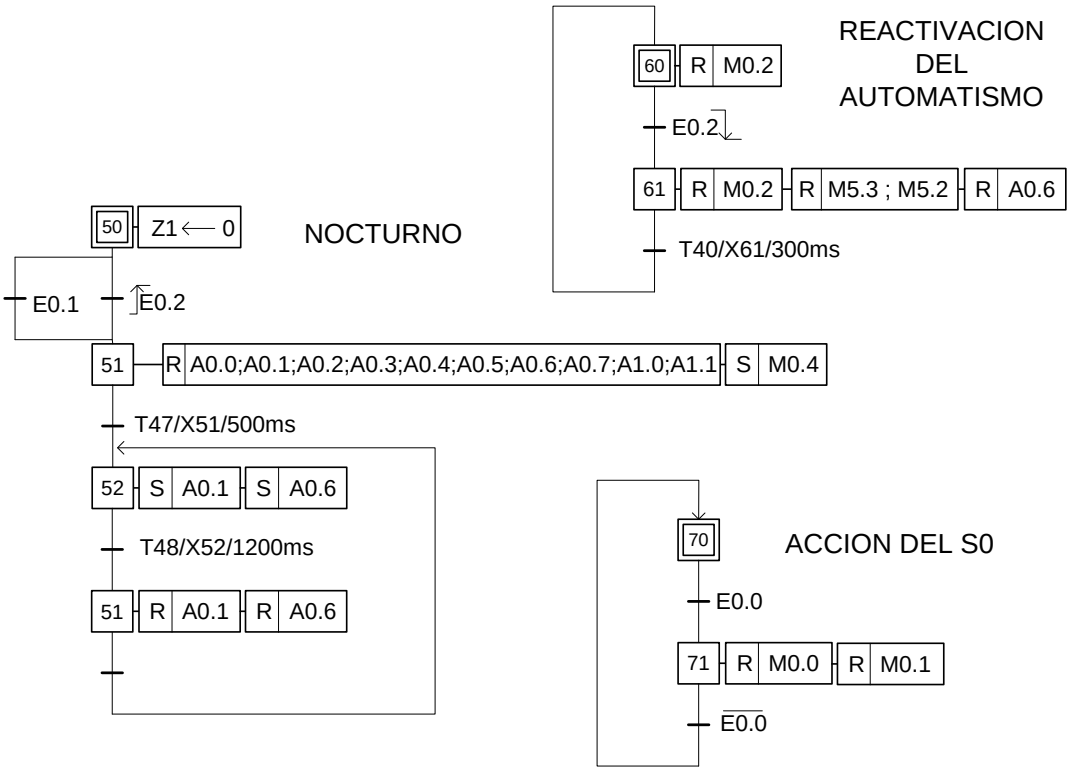
Símbolo	Dirección	Comentario
PRINCIPAL	OB1	PROGRAMA PARA CONTROLAR UN SISTEMA SIMPLE DE ACCIONAMIENTO AUTOMATIZADO

Programa principal:




PARPADEO DE
PEATONAL 1
30
Z1⟵ 0
M2.2
31
S
A0.3
R
A0.4
VB11⟵ 0
T44/X31/1s
32
R
A0.3
S
A0.4
VB11⟵ 1
Z1=Z1+1
Z1
(T44/X31/1s)*Z1


PARPADEO DE
PEATONAL 2
40
Z2⟵ 0
M2.5
41
S
A1.0
R
A1.1
VB13⟵ 0
T41/X31/1s
42
R
A1.0
S
A1.1
VB13⟵ 1
Z2=Z2+1
Z2
(T41/X31/1s)*Z2


REACTIVACION
DEL
AUTOMATISMO
60
R
M0.2
E0.2
61
R
M0.2
R
M5.3 ; M5.2
R
A0.6
T40/X61/300ms
NOCTURNO
50
Z1⟵ 0
E0.1
E0.2
51
R
A0.0;A0.1;A0.2;A0.3;A0.4;A0.5;A0.6;A0.7;A1.0;A1.1
S
M0.4
T47/X51/500ms
52
S
A0.1
S
A0.6
T48/X52/1200ms
51
R
A0.1
R
A0.6
ACCION DEL S0
70
E0.0
71
R
M0.0
R
M0.1
E0.0

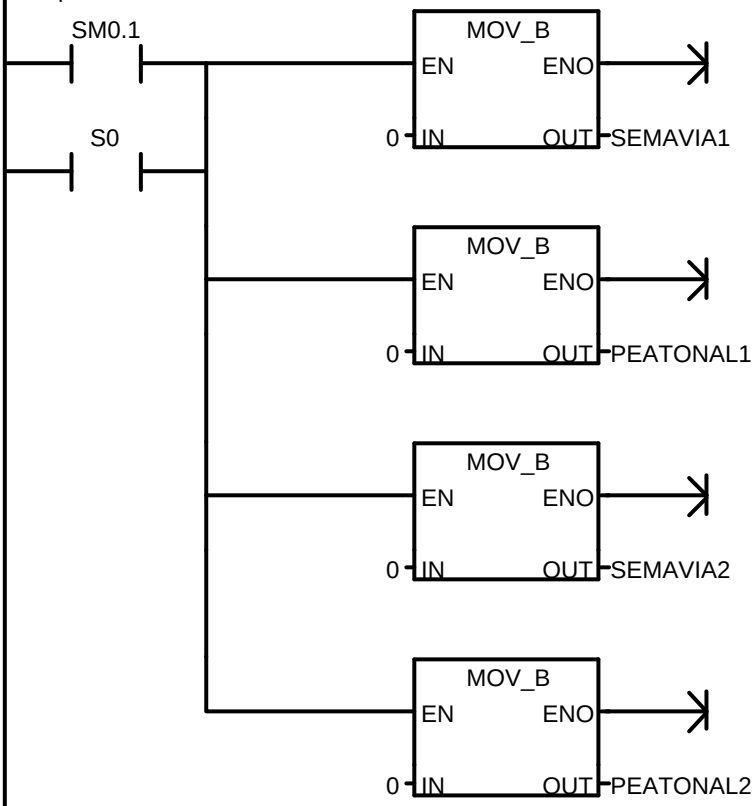
Bloque: PRINCIPAL
Autor:
Fecha de creación: 20.07.2009 20:37:11
Fecha de modificación: 30.05.2010 21:45:22

Símbolo	Tipo var.	Tipo de datos	Comentario
	TEMP		
	TEMP		
	TEMP		
	TEMP		

PROGRAMA PARA CONTROLAR UN SEMAFORO DE DOS VIAS

Network 1

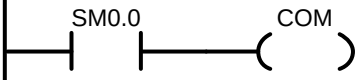
Limpieza de las variables a usar



Símbolo	Dirección	Comentario
PEATONAL1	VB11	VARIABLE DE ACCION 2
PEATONAL2	VB13	VARIABLE DE ACCION 4
S0	E0.0	PARO
SEMAVIA1	VB10	VARIABLE DE ACCION 1
SEMAVIA2	VB12	VARIABLE DE ACCION 3

Network 2 ==========

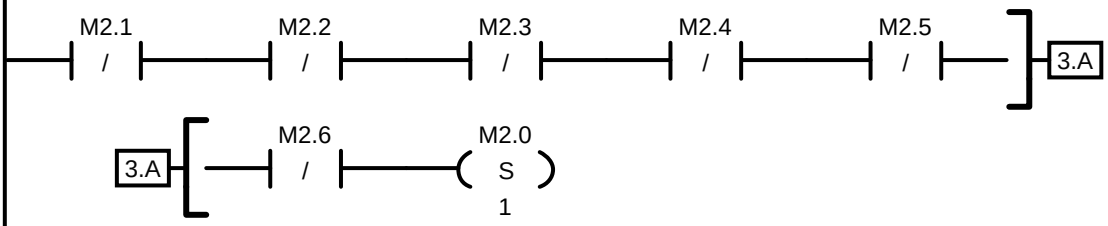
Comunicacion PC <-----> PLC



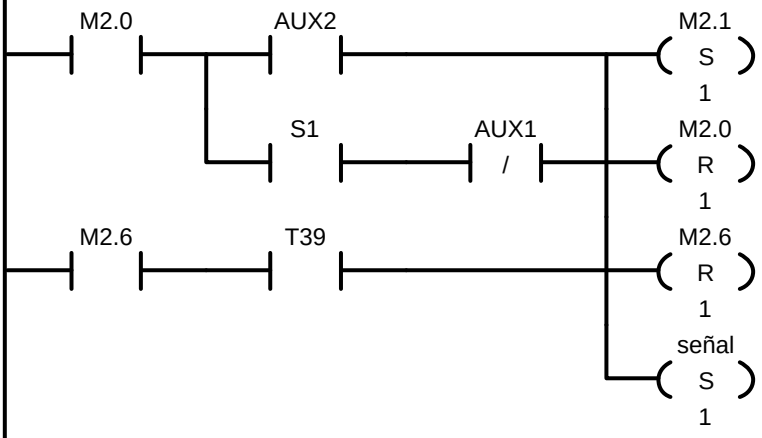
Símbolo	Dirección	Comentario
COM	M0.0	PC <----> PLC

Network 3 ==========

Control Diurno

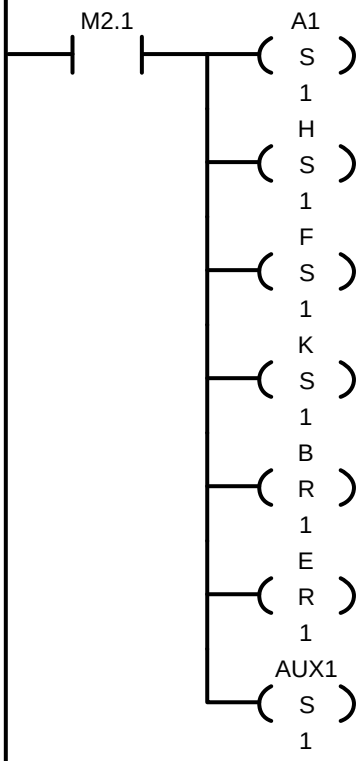


Network 4



Símbolo	Dirección	Comentario
AUX1	M0.1	AUXILIAR DE CONTROL 1
AUX2	M0.2	AUXILIAR DE CONTROL 2
S1	E0.1	MARCHA
señal	M0.3	PILOTO DE ON/OFF

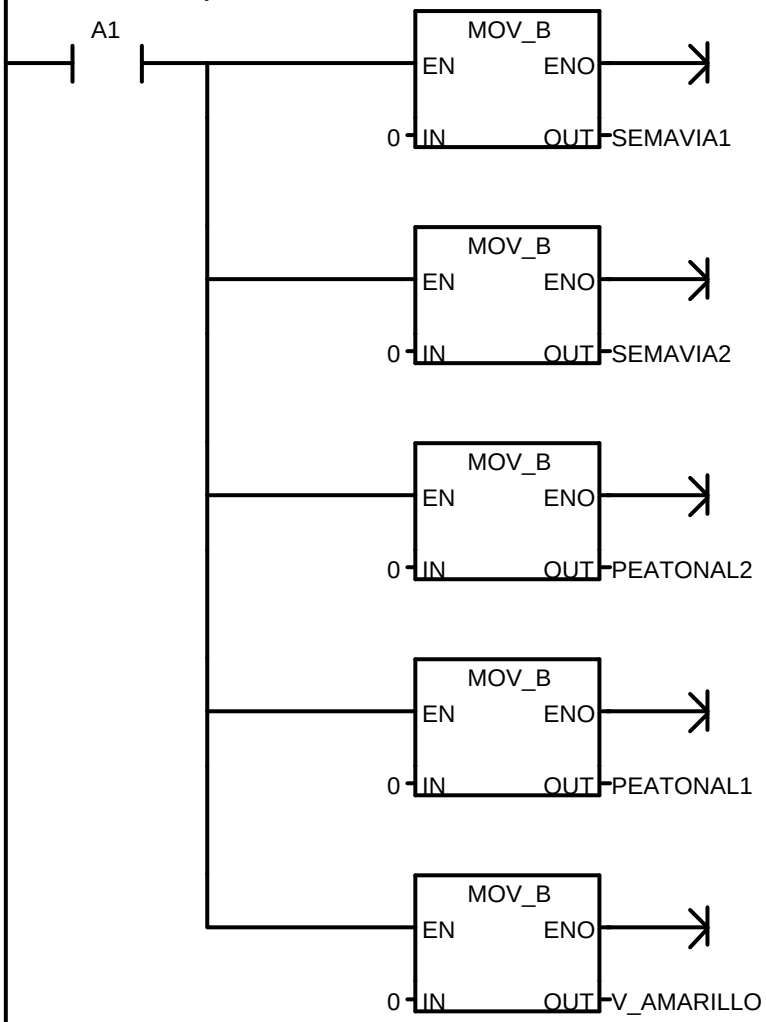
Network 5



Símbolo	Dirección	Comentario
A1	A0.0	ROJO1
AUX1	M0.1	AUXILIAR DE CONTROL 1
B	A0.1	AMARILLO1
E	A0.6	AMARILLO2
F	A0.7	VERDE2
H	A0.4	VERDEPEATONAL1
K	A1.0	ROJOPEATONAL2

Network 6

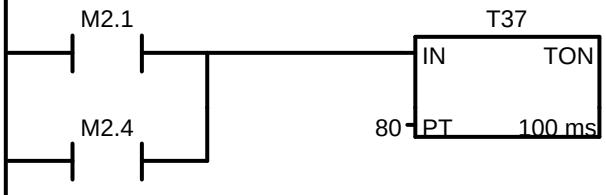
Activacion del Rojo1



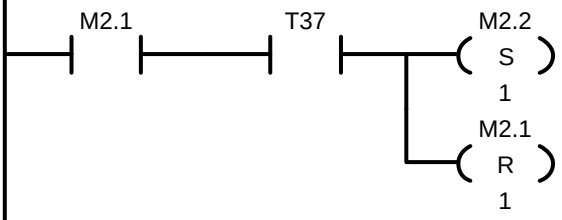
Símbolo	Dirección	Comentario
A1	A0.0	ROJO1
PEATONAL1	VB11	VARIABLE DE ACCION 2
PEATONAL2	VB13	VARIABLE DE ACCION 4
SEMAVIA1	VB10	VARIABLE DE ACCION 1
SEMAVIA2	VB12	VARIABLE DE ACCION 3
V_AMARILLO	VB14	VALOR DE POSTCARGA

Network 7

Timer de paso para la activacion de los amarillos

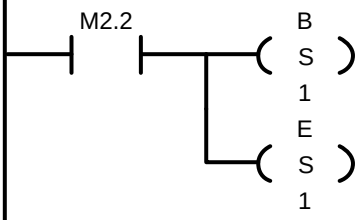


Network 8



Network 9

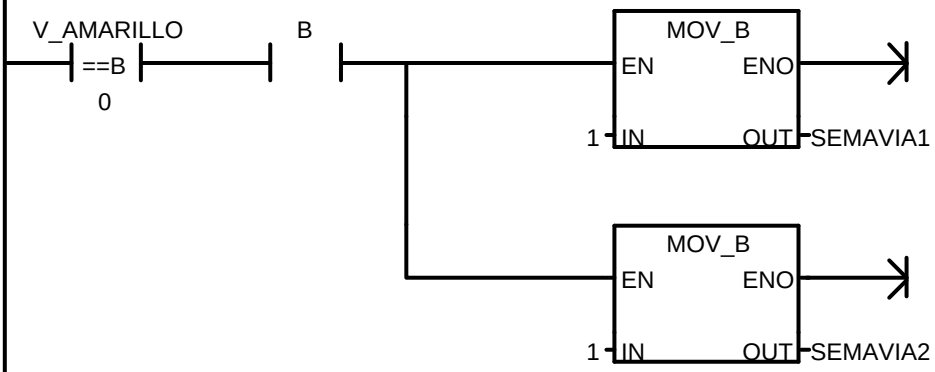
activacion de los amarillos



Símbolo	Dirección	Comentario
B	A0.1	AMARILLO1
E	A0.6	AMARILLO2

Network 10

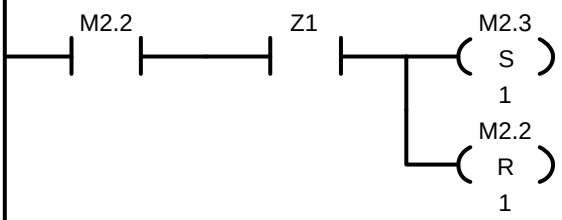
Activacion de los Amarillos



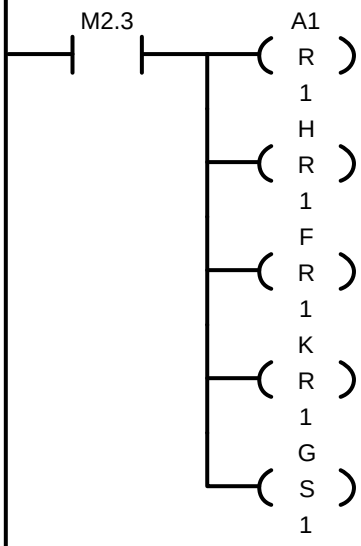
Símbolo	Dirección	Comentario
B	A0.1	AMARILLO1
SEMAVIA1	VB10	VARIABLE DE ACCION 1
SEMAVIA2	VB12	VARIABLE DE ACCION 3
V_AMARILLO	VB14	VALOR DE POSTCARGA

Network 11

cambio de luces

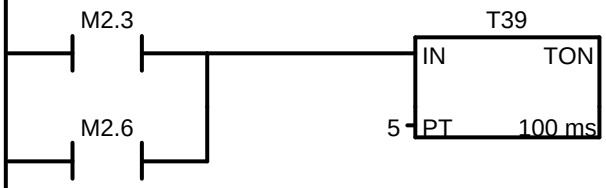


Network 12

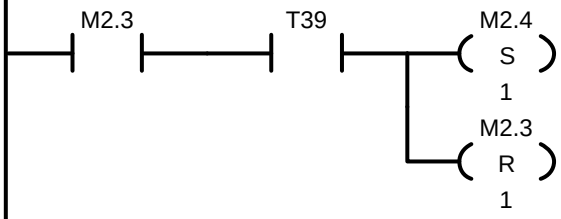


Símbolo	Dirección	Comentario
A1	A0.0	ROJO1
F	A0.7	VERDE2
G	A0.3	ROJOPEATONAL1
H	A0.4	VERDEPEATONAL1
K	A1.0	ROJOPEATONAL2

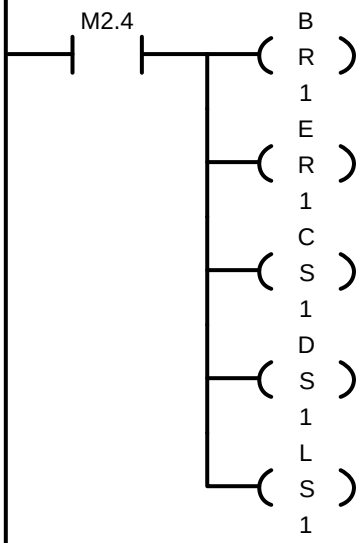
Network 13



Network 14

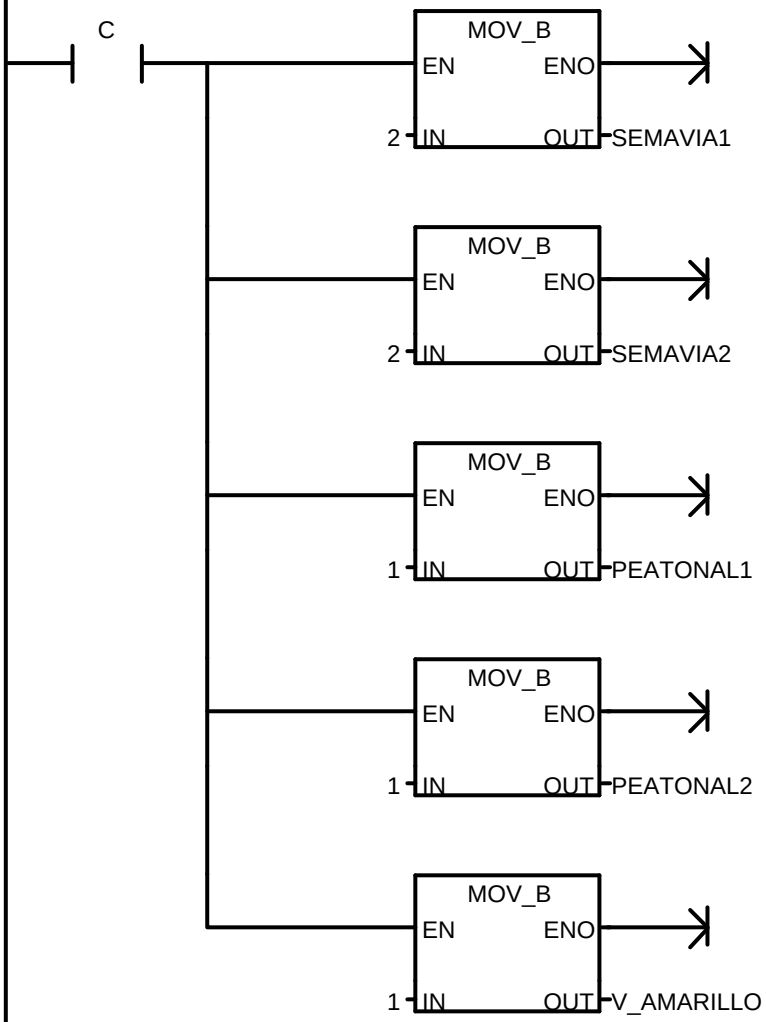


Network 15



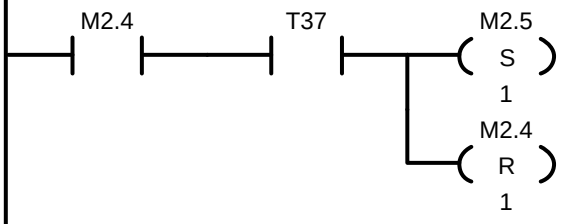
Símbolo	Dirección	Comentario
B	A0.1	AMARILLO1
C	A0.2	VERDE1
D	A0.5	ROJO2
E	A0.6	AMARILLO2
L	A1.1	VERDEPEATONAL2

Network 16

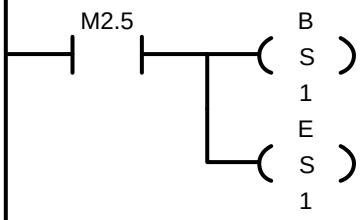


Símbolo	Dirección	Comentario
C	A0.2	VERDE1
PEATONAL1	VB11	VARIABLE DE ACCION 2
PEATONAL2	VB13	VARIABLE DE ACCION 4
SEMAVIA1	VB10	VARIABLE DE ACCION 1
SEMAVIA2	VB12	VARIABLE DE ACCION 3
V_AMARILLO	VB14	VALOR DE POSTCARGA

Network 17

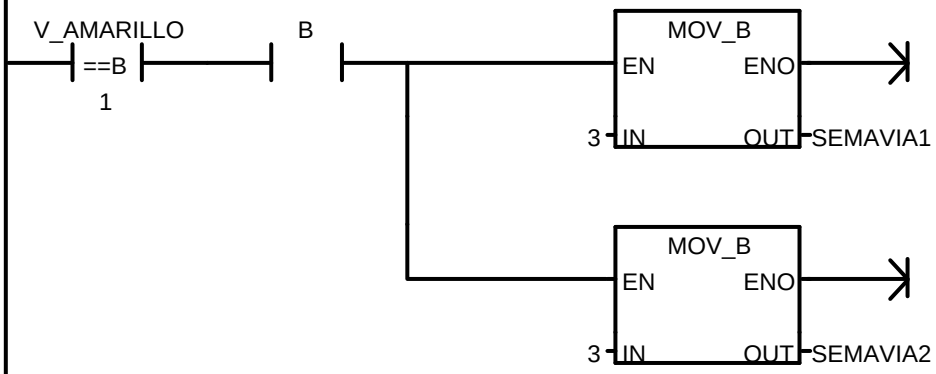


Network 18



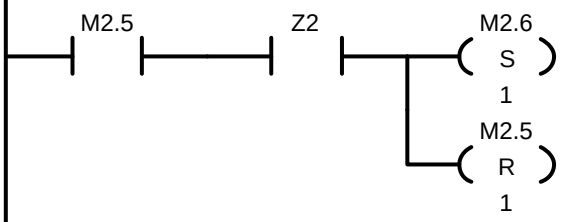
Símbolo	Dirección	Comentario
B	A0.1	AMARILLO1
E	A0.6	AMARILLO2

Network 19

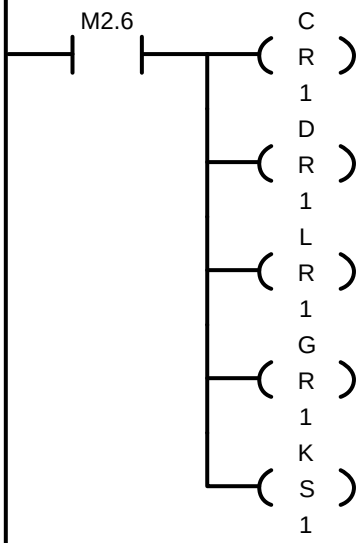


Símbolo	Dirección	Comentario
B	A0.1	AMARILLO1
SEMAVIA1	VB10	VARIABLE DE ACCION 1
SEMAVIA2	VB12	VARIABLE DE ACCION 3
V_AMARILLO	VB14	VALOR DE POSTCARGA

Network 20



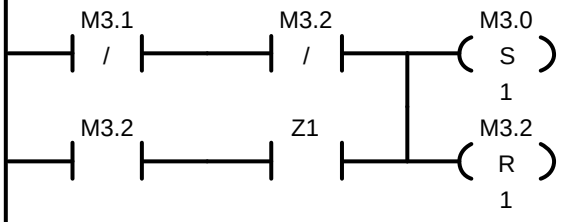
Network 21



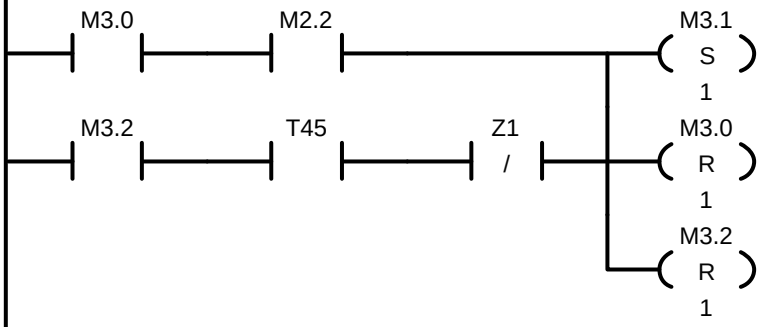
Símbolo	Dirección	Comentario
C	A0.2	VERDE1
D	A0.5	ROJO2
G	A0.3	ROJOPEATONAL1
K	A1.0	ROJOPEATONAL2
L	A1.1	VERDEPEATONAL2

Network 22 ==========

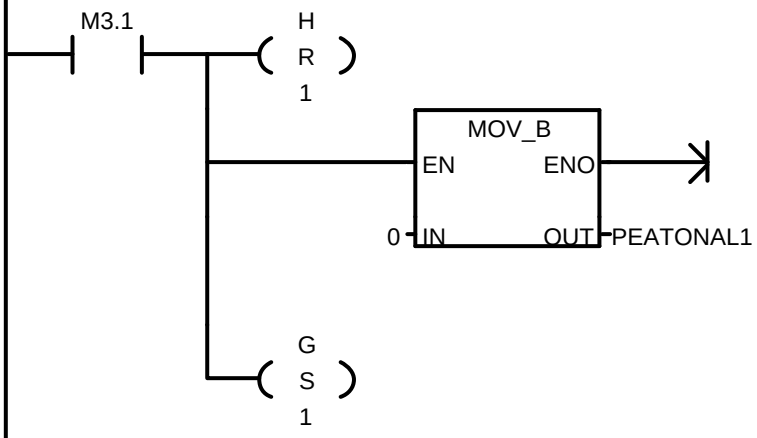
Parpadeo del Peatonal 1



Network 23

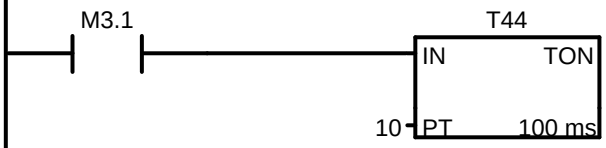


Network 24

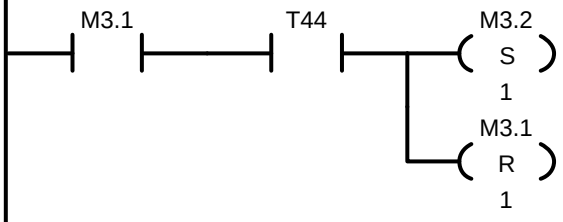


Símbolo	Dirección	Comentario
G	A0.3	ROJOPEATONAL1
H	A0.4	VERDEPEATONAL1
PEATONAL1	VB11	VARIABLE DE ACCION 2

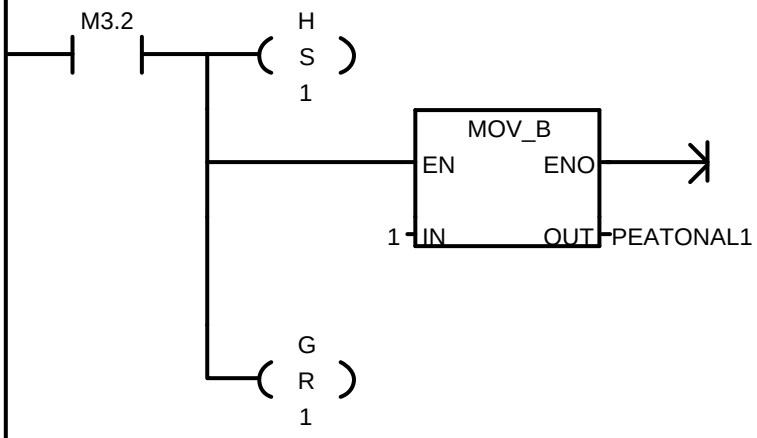
Network 25



Network 26

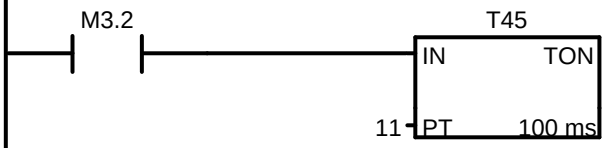


Network 27

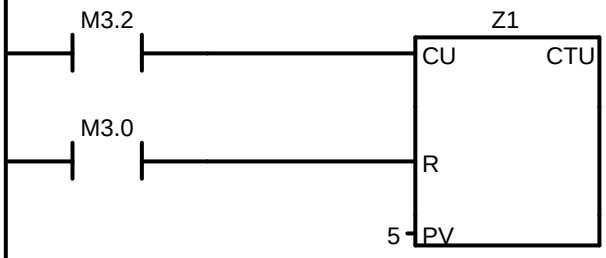


Símbolo	Dirección	Comentario
G	A0.3	ROJOPEATONAL1
H	A0.4	VERDEPEATONAL1
PEATONAL1	VB11	VARIABLE DE ACCION 2

Network 28

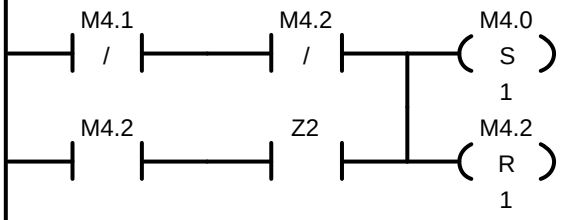


Network 29

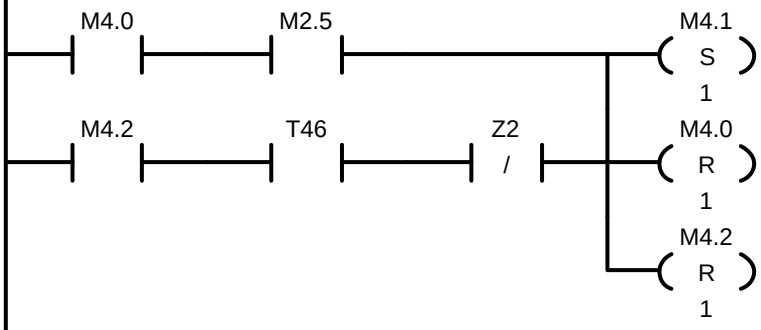


Network 30 ==========

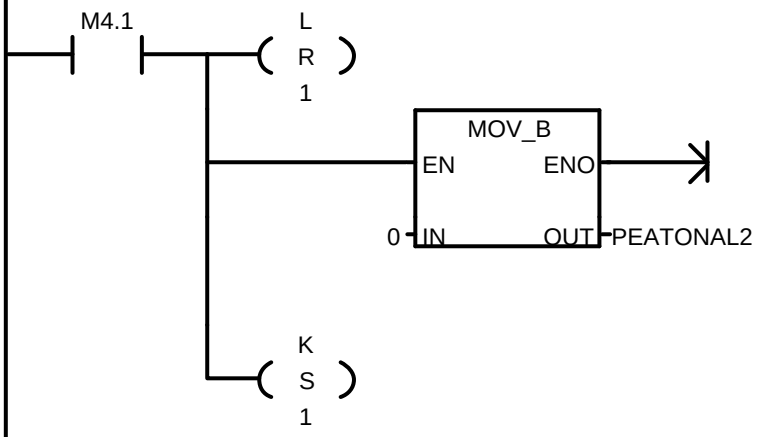
Parpadeo del Peatonal 2



Network 31

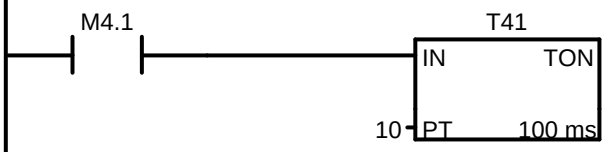


Network 32

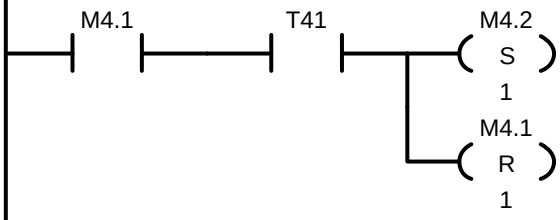


Símbolo	Dirección	Comentario
K	A1.0	ROJOPEATONAL2
L	A1.1	VERDEPEATONAL2
PEATONAL2	VB13	VARIABLE DE ACCION 4

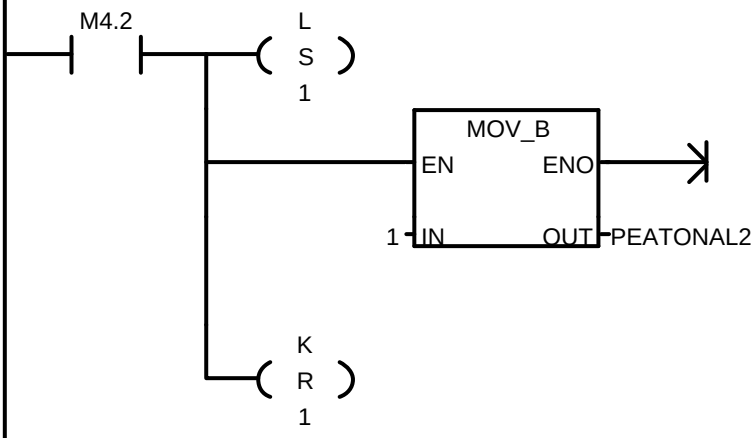
Network 33



Network 34

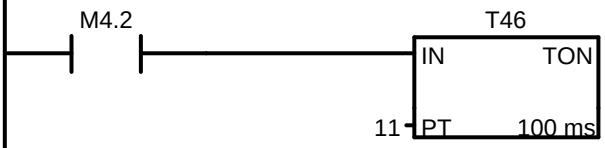


Network 35

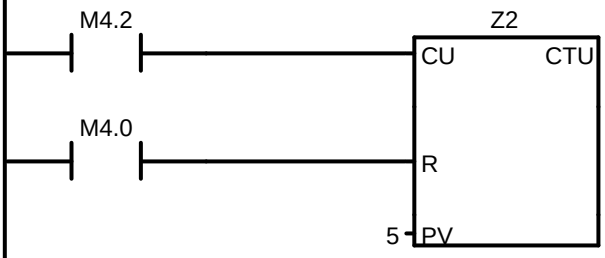


Símbolo	Dirección	Comentario
K	A1.0	ROJOPEATONAL2
L	A1.1	VERDEPEATONAL2
PEATONAL2	VB13	VARIABLE DE ACCION 4

Network 36

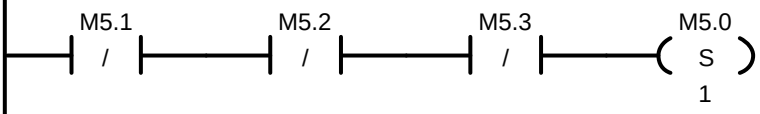


Network 37

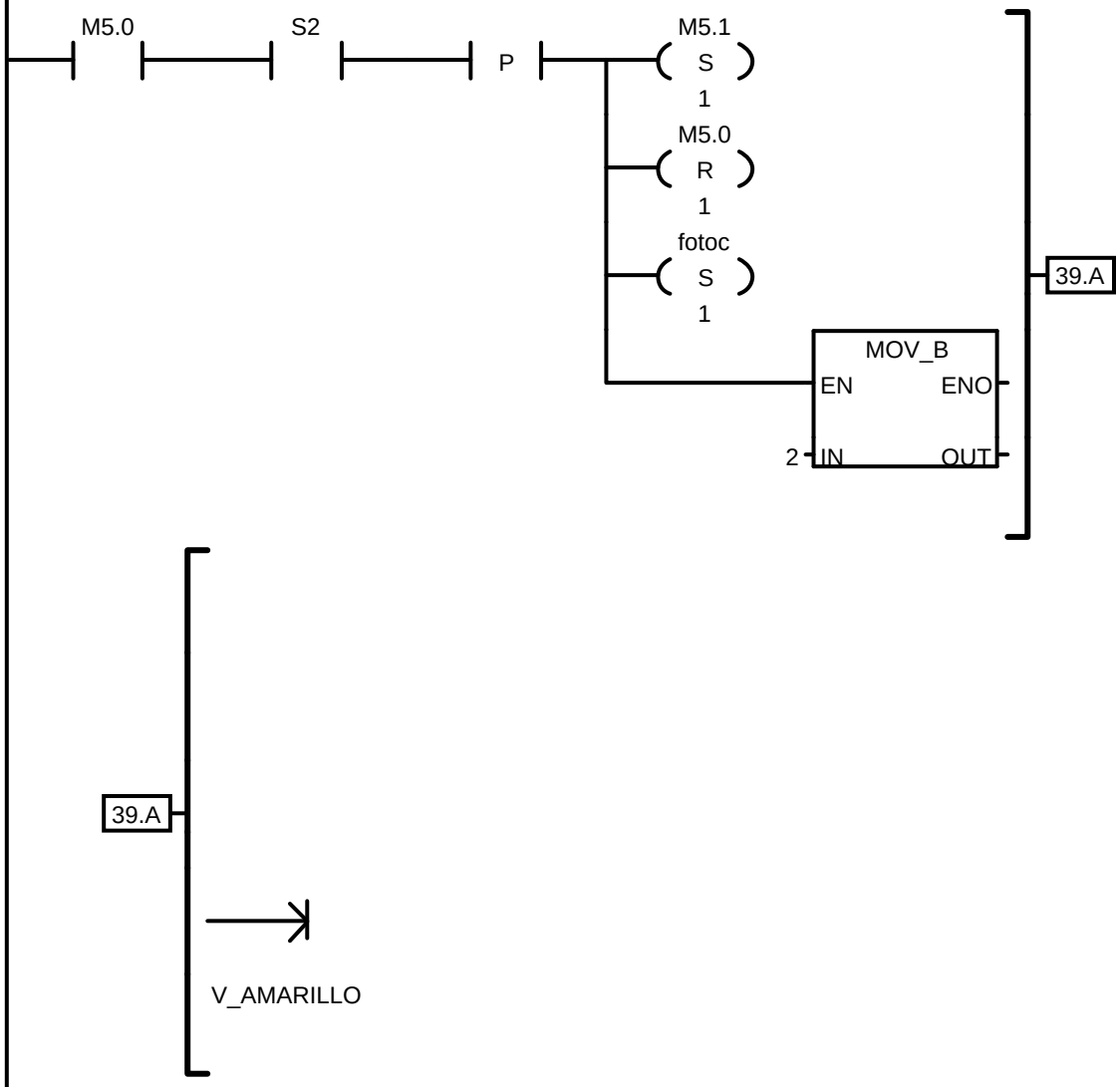


Network 38 ==========

Nocturno

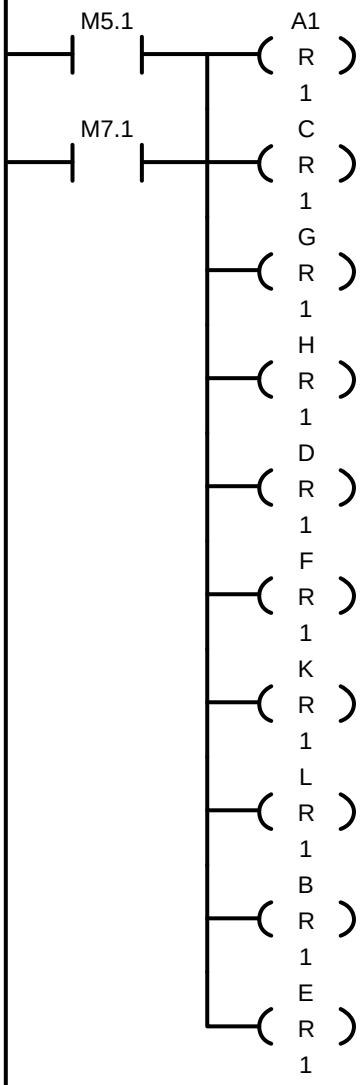


Network 39



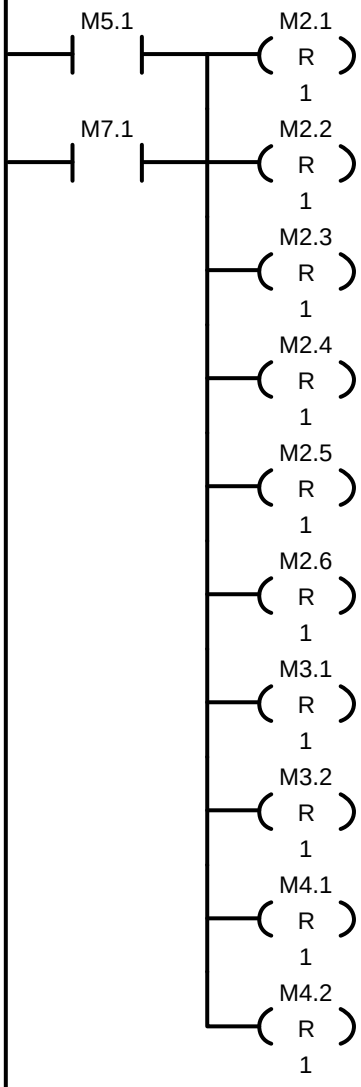
Símbolo	Dirección	Comentario
fotoc	M0.4	ON/OFF FOTOCELDA
S2	E0.2	FOTOCELDA
V_AMARILLO	VB14	VALOR DE POSTCARGA

Network 40

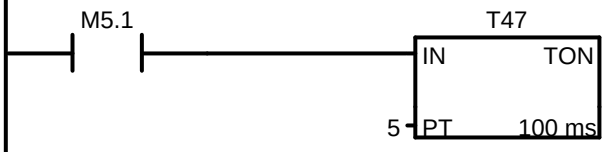


Símbolo	Dirección	Comentario
A1	A0.0	ROJO1
B	A0.1	AMARILLO1
C	A0.2	VERDE1
D	A0.5	ROJO2
E	A0.6	AMARILLO2
F	A0.7	VERDE2
G	A0.3	ROJOPEATONAL1
H	A0.4	VERDEPEATONAL1
K	A1.0	ROJOPEATONAL2
L	A1.1	VERDEPEATONAL2

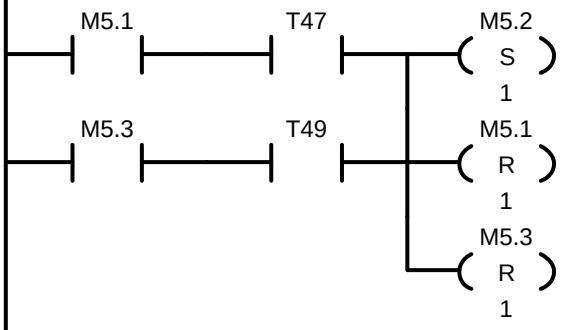
Network 41



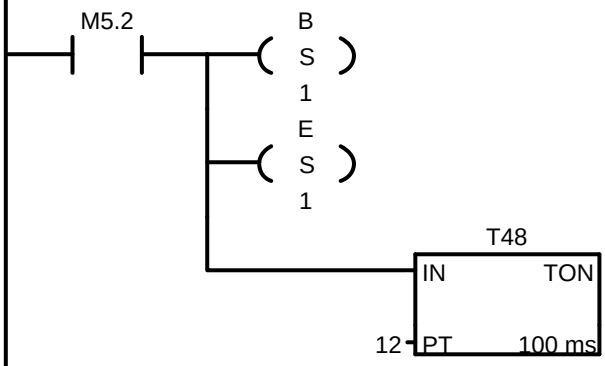
Network 42



Network 43

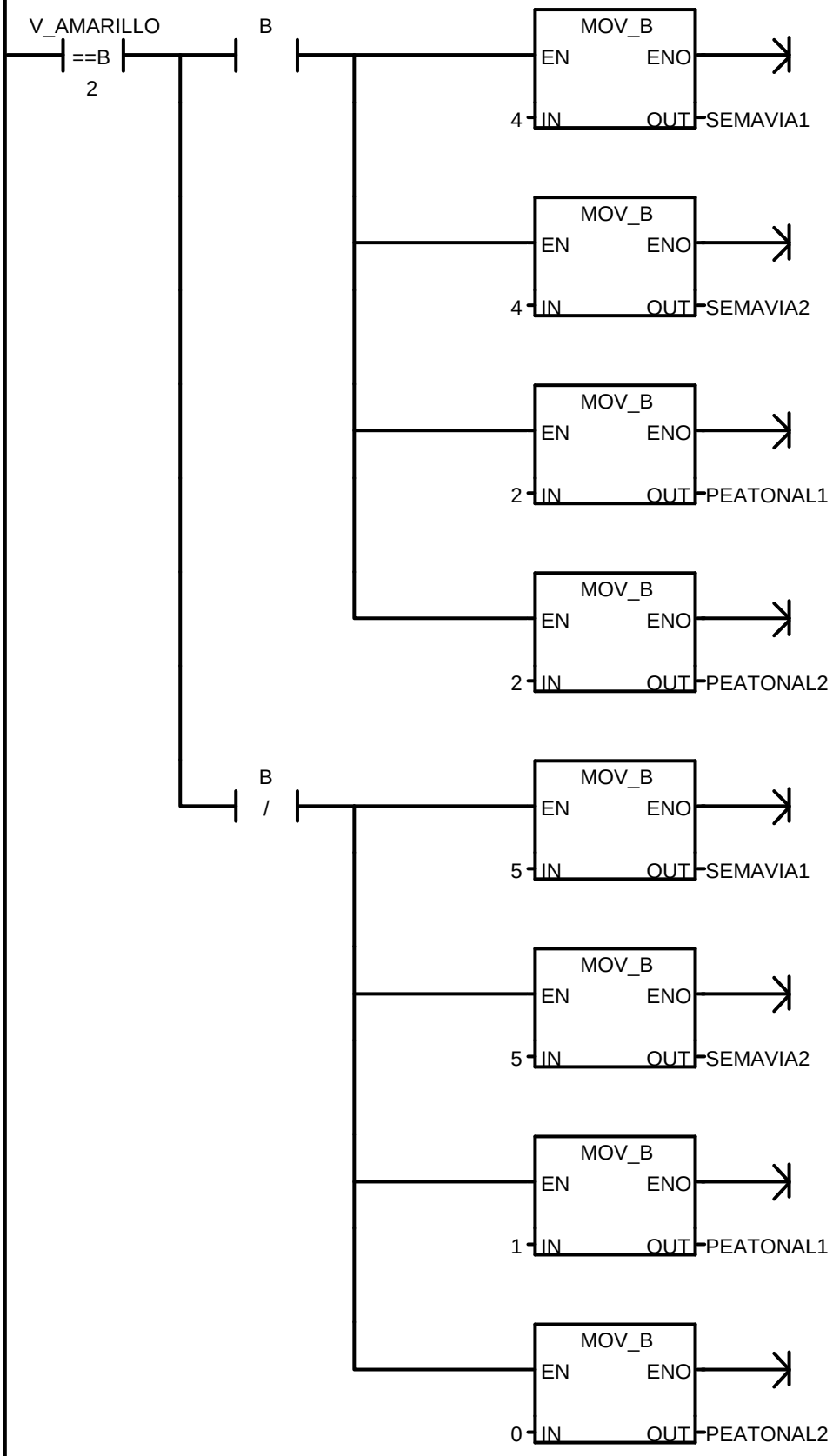


Network 44



Símbolo	Dirección	Comentario
B	A0.1	AMARILLO1
E	A0.6	AMARILLO2

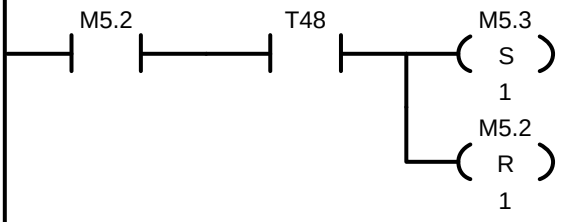
Network 45



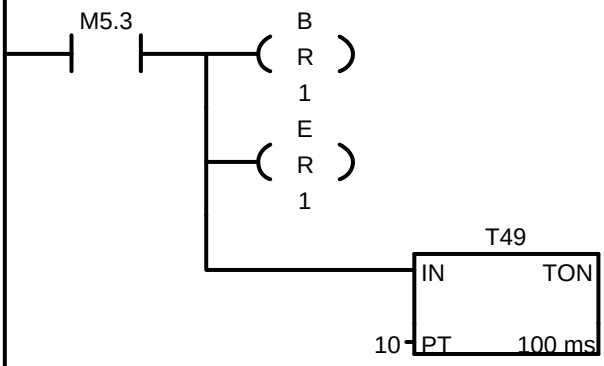
Símbolo	Dirección	Comentario
B	A0.1	AMARILLO1
PEATONAL1	VB11	VARIABLE DE ACCION 2

PEATONAL2	VB13	VARIABLE DE ACCION 4
SEMAVIA1	VB10	VARIABLE DE ACCION 1
SEMAVIA2	VB12	VARIABLE DE ACCION 3
V_AMARILLO	VB14	VALOR DE POSTCARGA

Network 46



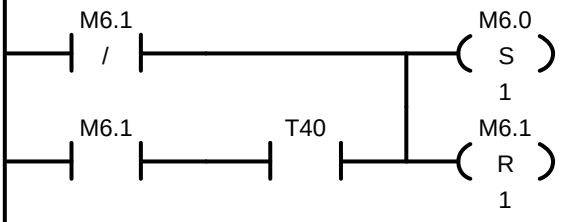
Network 47



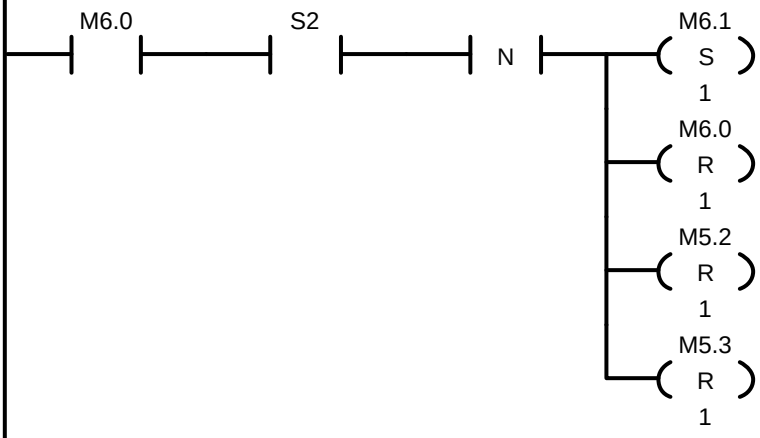
Símbolo	Dirección	Comentario
B	A0.1	AMARILLO1
E	A0.6	AMARILLO2

Network 48 ==========

Regreso al Diurno

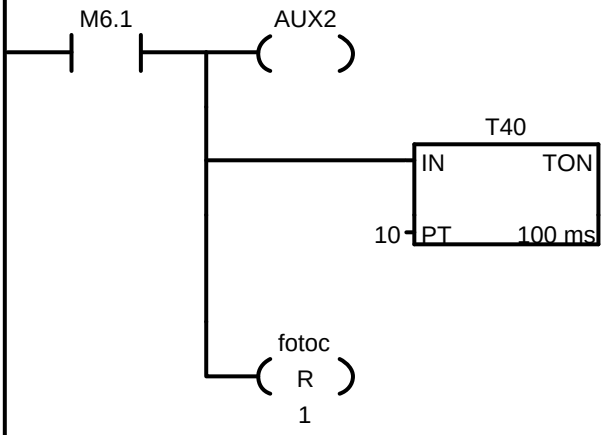


Network 49



Símbolo	Dirección	Comentario
S2	E0.2	FOTOCELDA

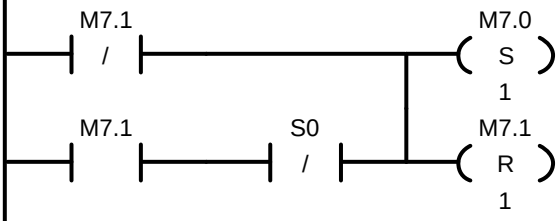
Network 50



Símbolo	Dirección	Comentario
AUX2	M0.2	AUXILIAR DE CONTROL 2
fotoc	M0.4	ON/OFF FOTOCELDA

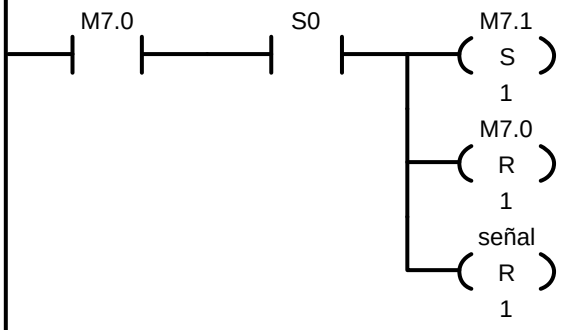
Network 51 ==========

Accion del S0



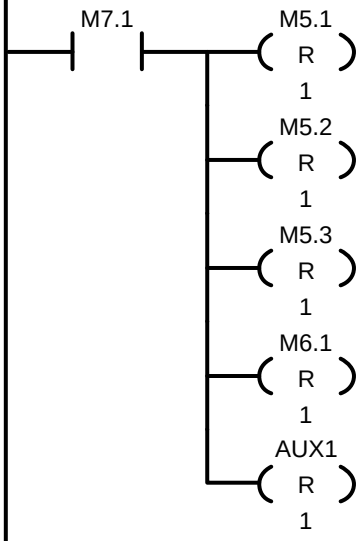
Símbolo	Dirección	Comentario
S0	E0.0	PARO

Network 52



Símbolo	Dirección	Comentario
S0	E0.0	PARO
señal	M0.3	PILOTO DE ON/OFF

Network 53



Símbolo	Dirección	Comentario
AUX1	M0.1	AUXILIAR DE CONTROL 1

Símbolo	Dirección	Comentario
S1	E0.1	MARCHA
S0	E0.0	PARO
S2	E0.2	FOTOCELDA
A1	A0.0	ROJO1
B	A0.1	AMARILLO1
C	A0.2	VERDE1
COM	M0.0	PC <----> PLC
G	A0.3	ROJOPEATONAL1
H	A0.4	VERDEPEATONAL1
D	A0.5	ROJO2
E	A0.6	AMARILLO2
F	A0.7	VERDE2
K	A1.0	ROJOPEATONAL2
L	A1.1	VERDEPEATONAL2
AUX1	M0.1	AUXILIAR DE CONTROL 1
AUX2	M0.2	AUXILIAR DE CONTROL 2
señal	M0.3	PILOTO DE ON/OFF
fotoc	M0.4	ON/OFF FOTOCELDA
SEMAVIA1	VB10	VARIABLE DE ACCION 1
PEATONAL1	VB11	VARIABLE DE ACCION 2
SEMAVIA2	VB12	VARIABLE DE ACCION 3
PEATONAL2	VB13	VARIABLE DE ACCION 4
V_AMARILLO	VB14	VALOR DE POSTCARGA

Símbolo	Dirección	Comentario
PRINCIPAL	OB1	PROGRAMA PARA CONTROLAR UN SEMAFORO DE DOS VIAS

Para resolver ésta práctica se empleó dos CPU-224, un CP243-1 como *cliente*, y un CP243-1 IT como *servidor*. Considerar que aunque el funcionamiento de las unidades CPs son similares, difieren en la configuración que se la realiza por medio de los asistentes del MicroWin. Como también que los seteos que se muestran en las imágenes son para una dinámica solo entre los autómatas.

Al término de los box de configuración se presenta el KOP del programa, se recomienda elaborar un algoritmo gráfico para entender como está hecho.

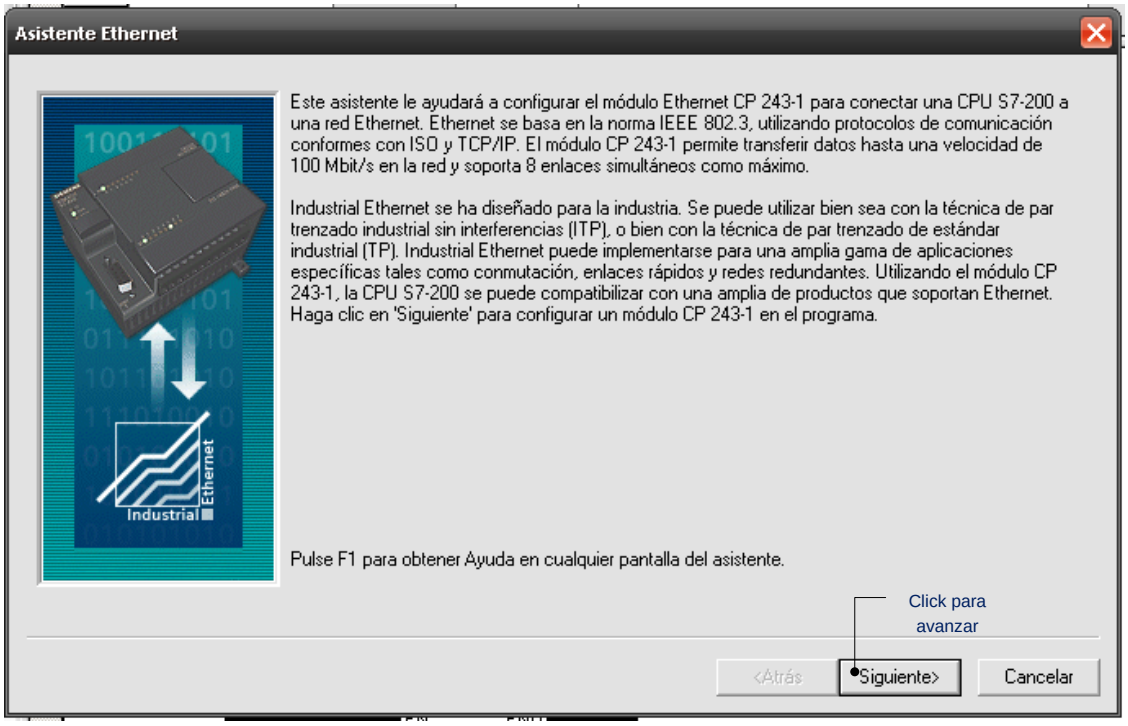
Los asistentes empleados son:

- ✓ El CP243-1 se configura con el Asistente Ethernet.
- ✓ El CP243-1 IT se configura con el Asistente Internet.

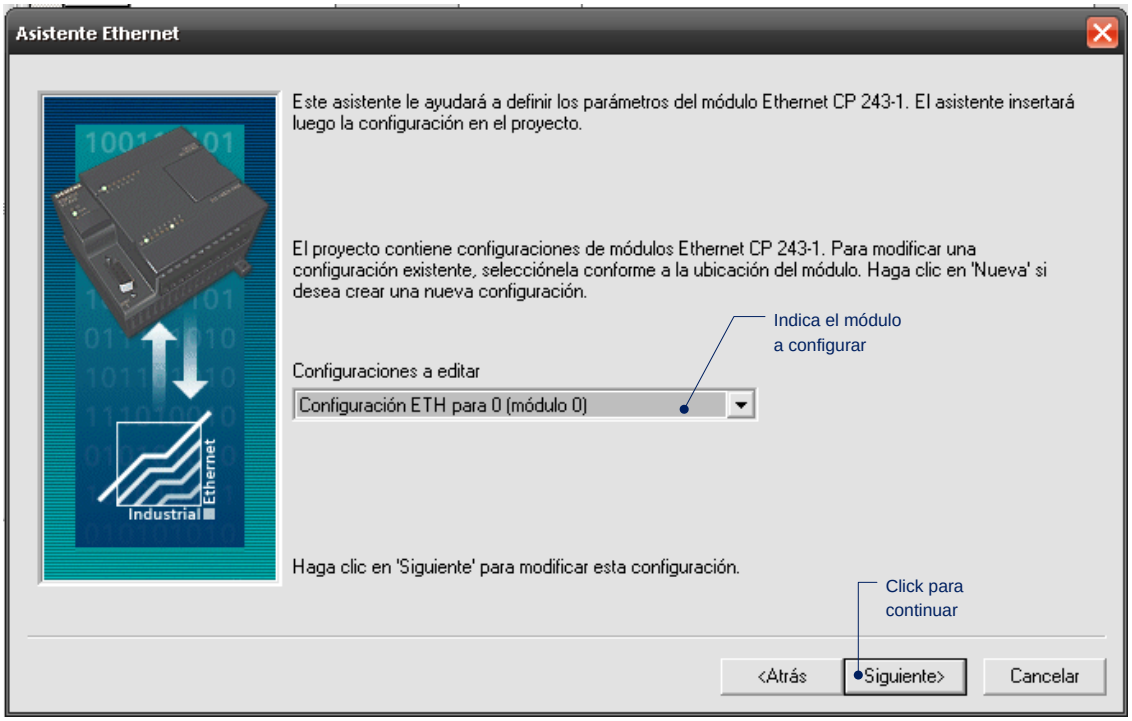
Para el CP243-1 es:

- Iniciar el asistente:

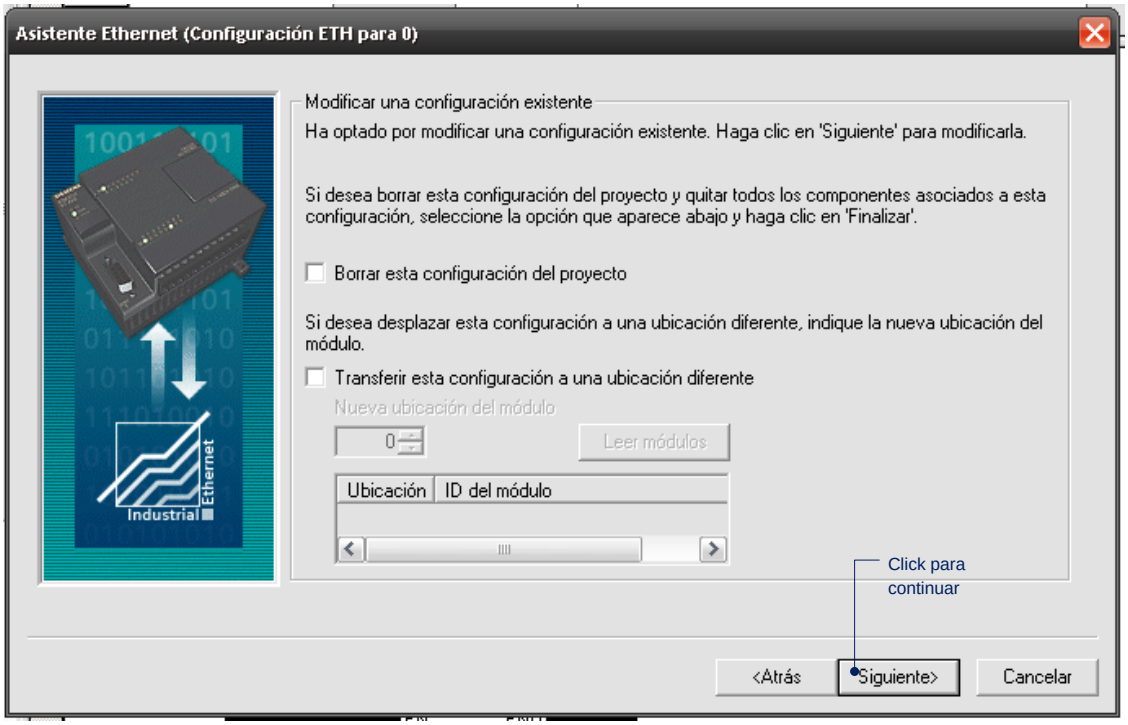
  Aquí se da una breve explicación sobre algunos tópicos de la configuración.



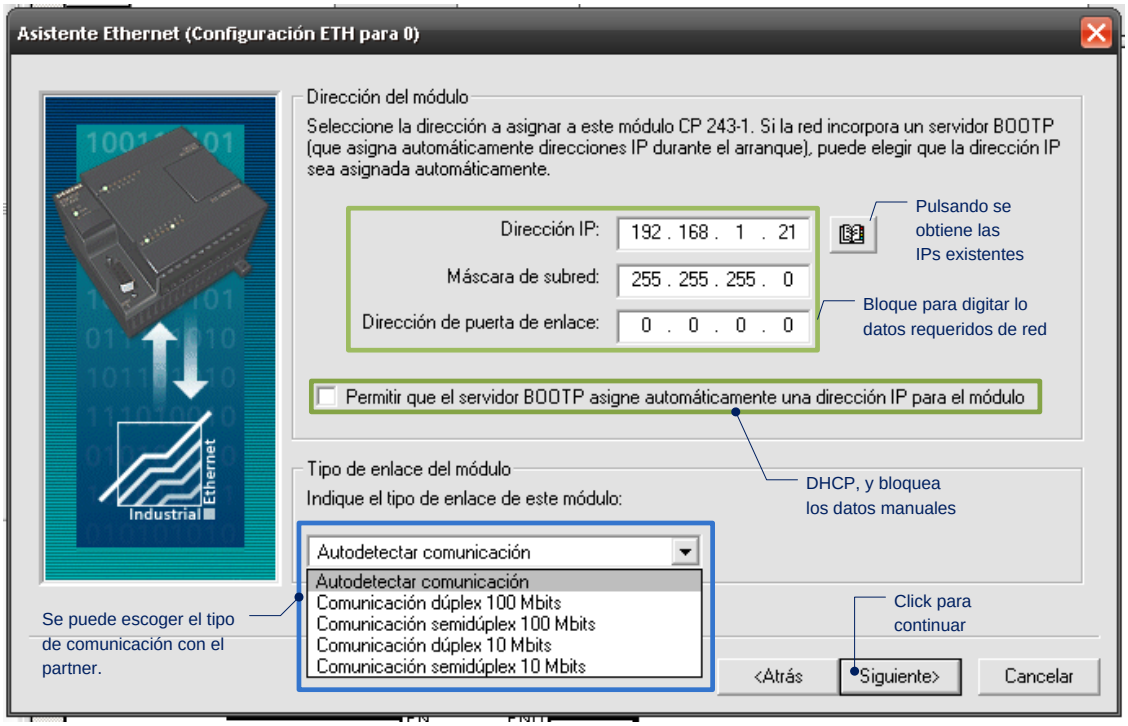
- Se indica para que modulo es la configuración a realizar.



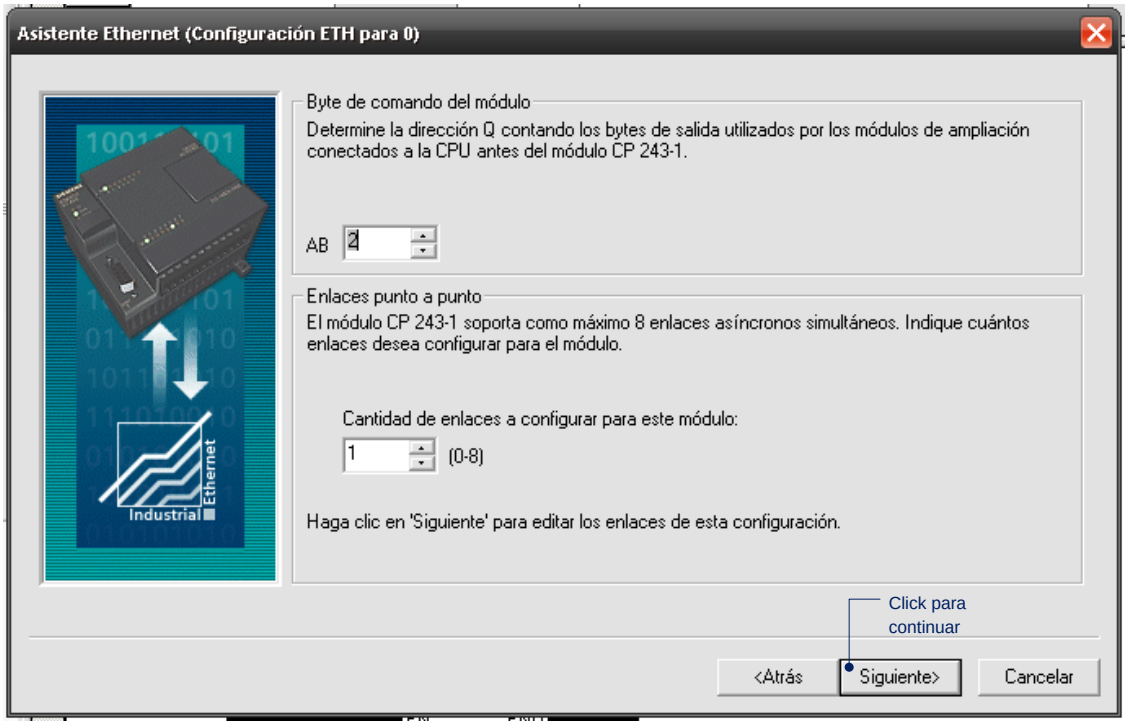
- Se obtiene la posición del CP por "Leer módulos", o se puede dejar para luego.



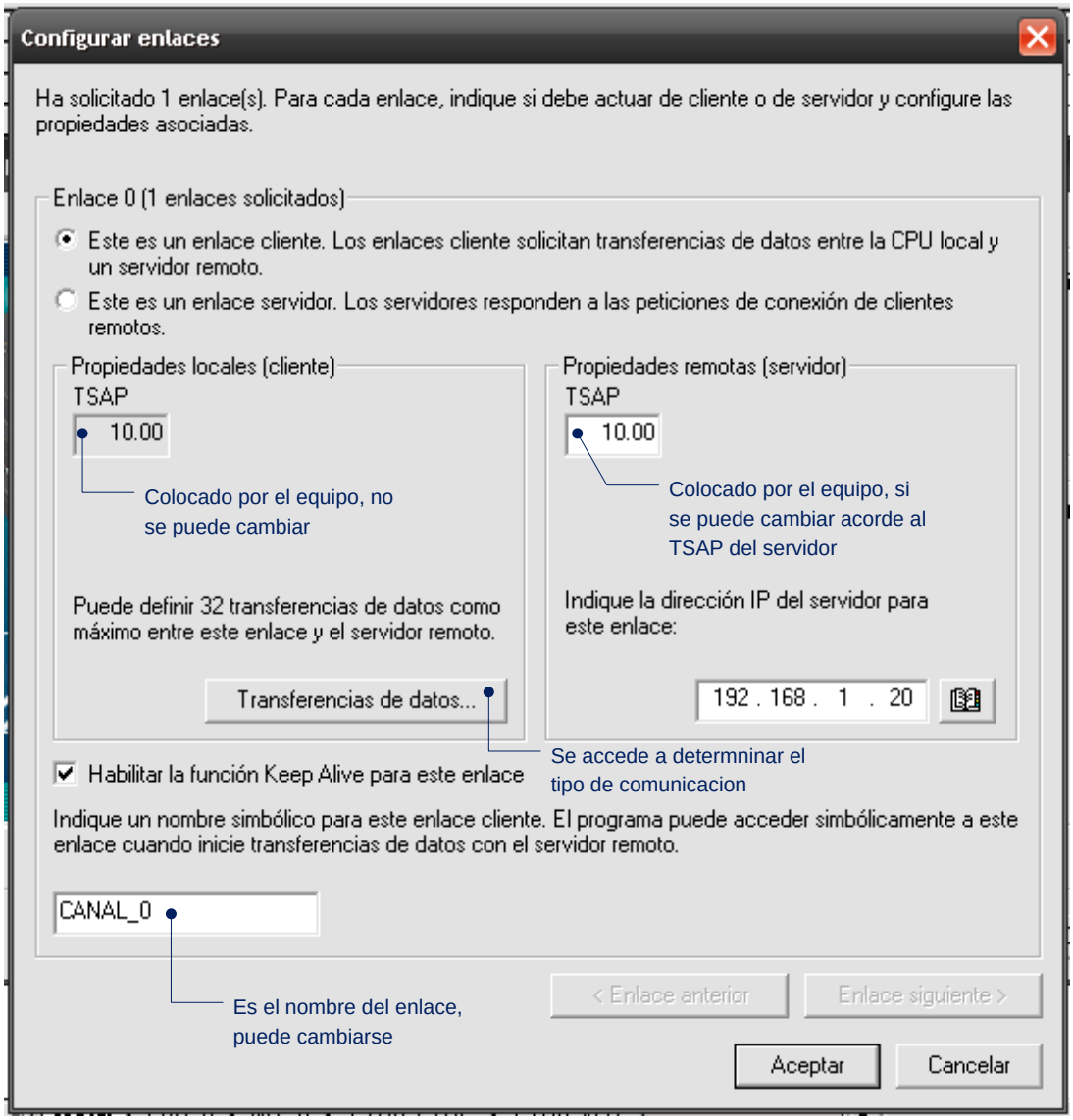
- Bloque para configurar los datos de red.



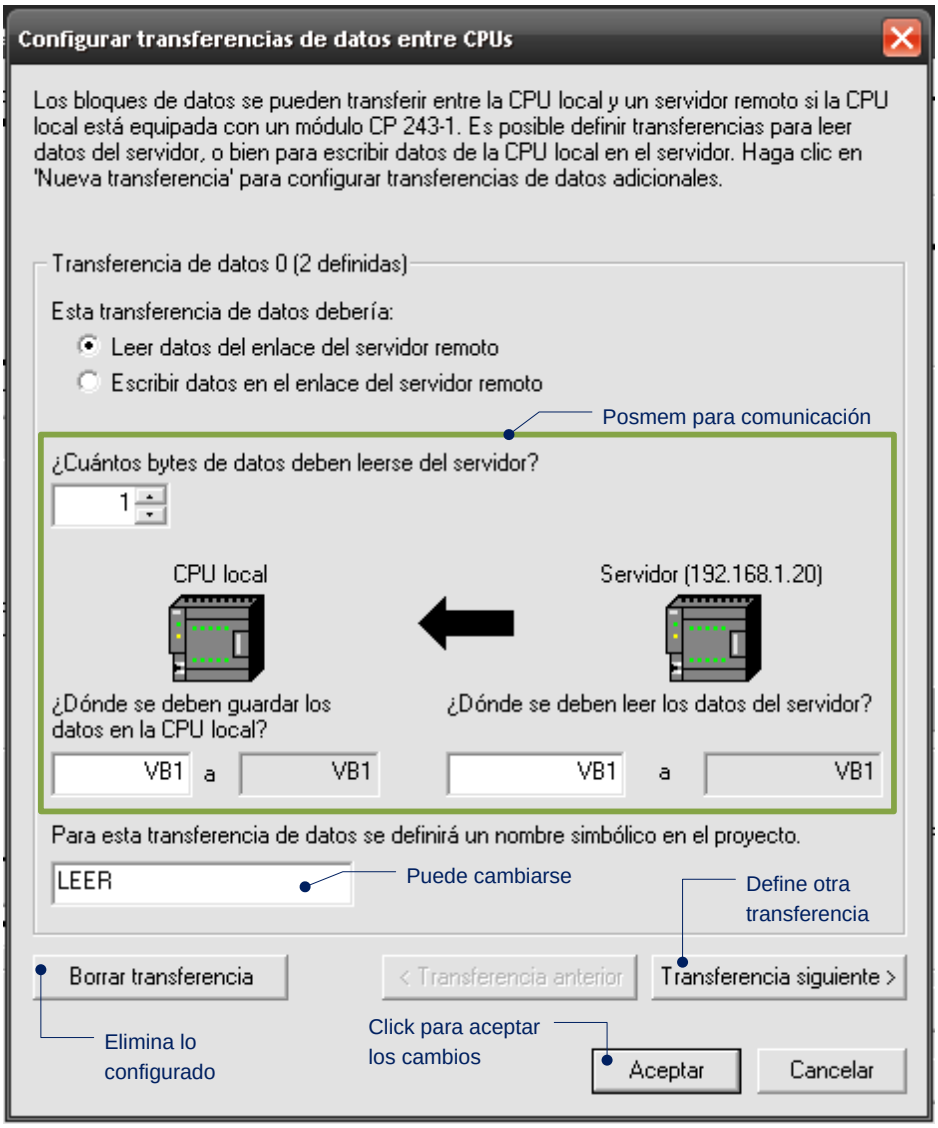
- Configurar el número de enlaces y byte de comando.



- Se escoge el tipo del enlace y parámetros para la comunicación.

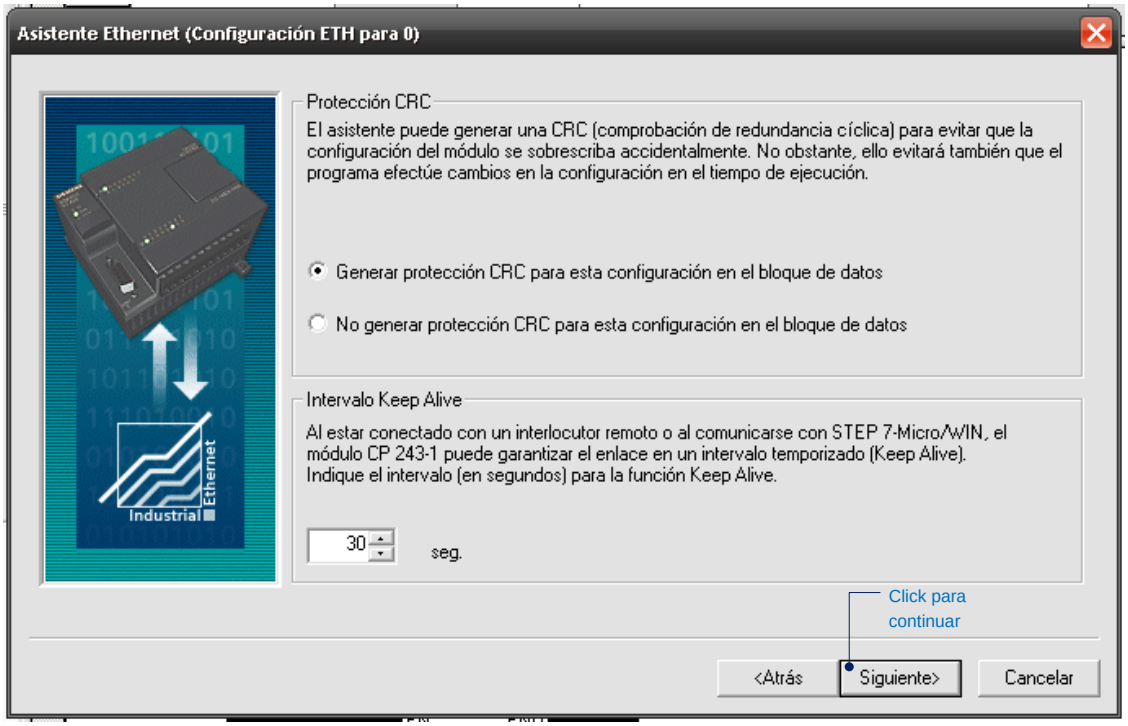


- Para configurar la(s) transferencia(s), se clickea *Transferencia de datos..*, teniéndose.

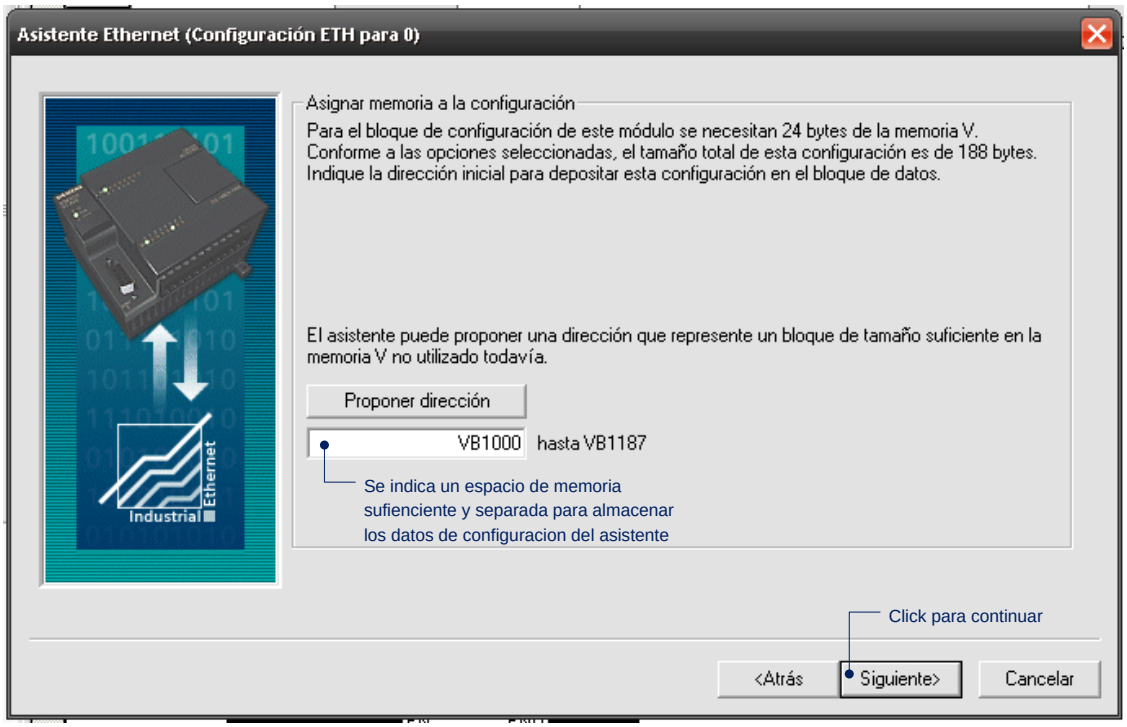


Para efectuar otra configuración de transferencia de datos, se clickea *Transferencia siguiente>*

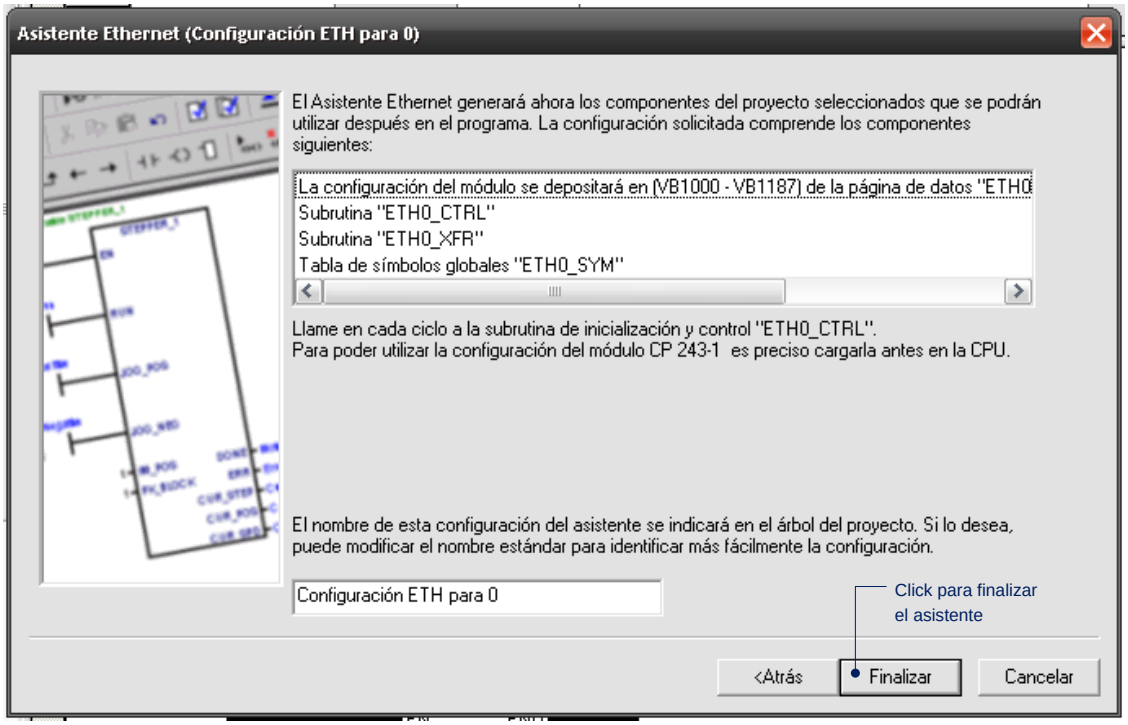
- Configurar ésta protección de acuerdo a la programación que haya planteado.



- Es recomendable colocar un espacio alto de posmem, para eliminar la posibilidad de traslape de datos involuntarios. Se puede aceptar la propuesta o digitar, según la conveniencia.



- Se muestra cuales son las subrutinas que ayudarán a establecer la dinámica de comunicación.

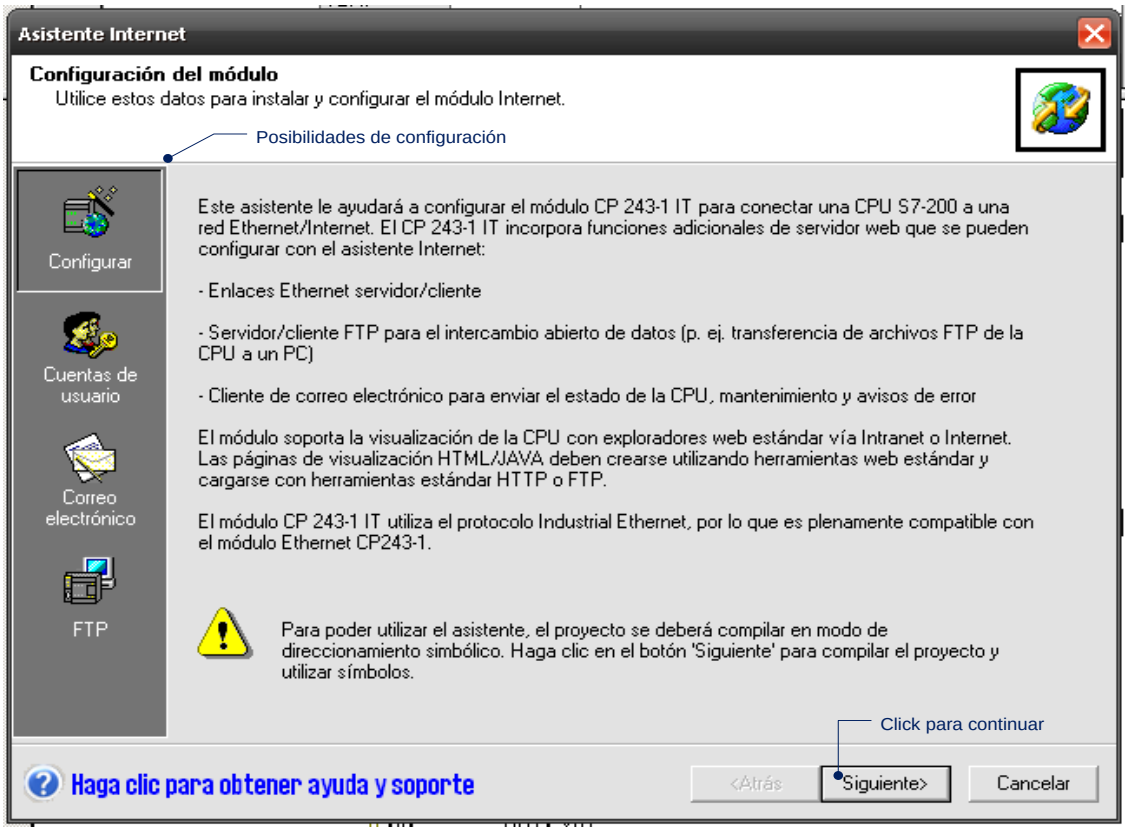


Para el CP243-1 IT se lo hace de forma similar, pero como es un módulo internet y por poseer las cualidades para la web, surgen unos box adicionales. Siendo:

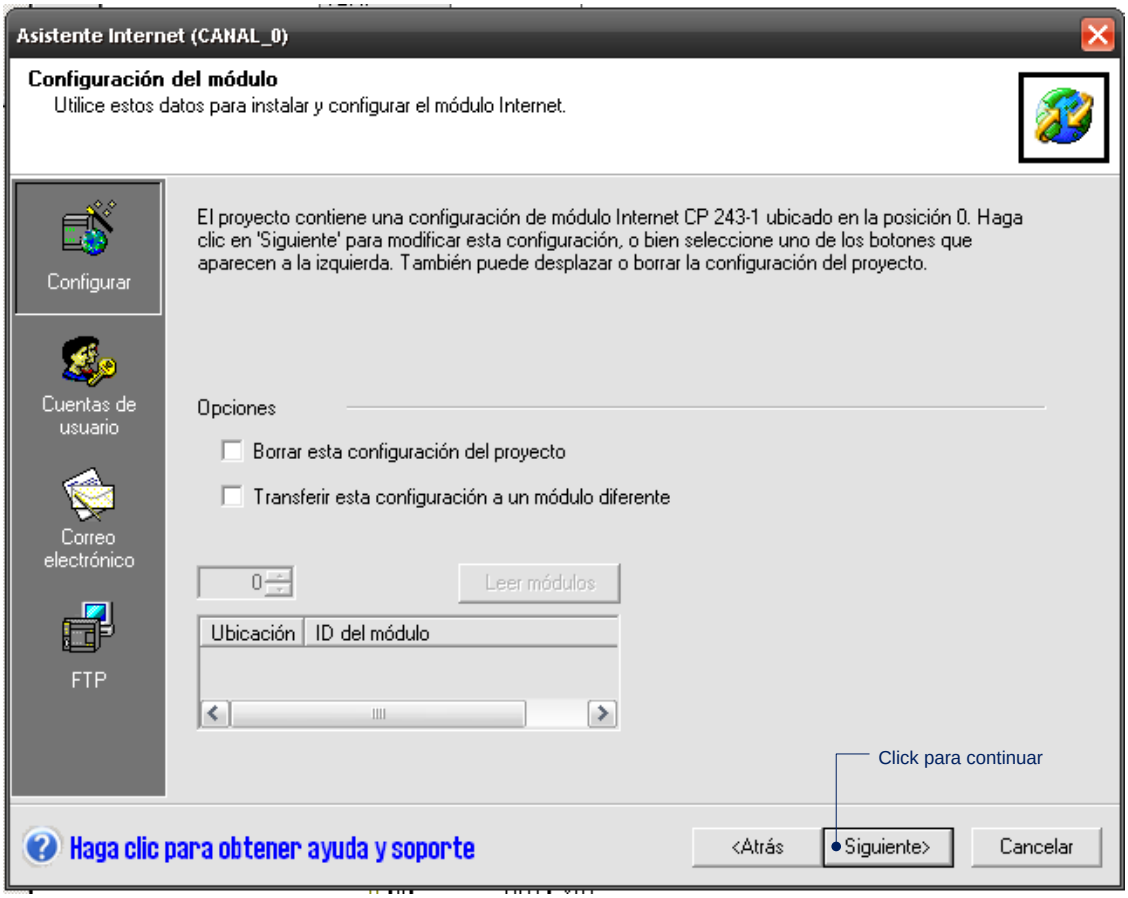
- Iniciar el asistente.

  Al lado izquierdo se presentan cuatro posibilidades de configuración: *Configurar, Cuentas de Usuario, Correo Electrónico, FTP*. De las cuales, en ésta configuración, se escogerá *Configurar*; por el tipo de algoritmo que se tiene planteado.

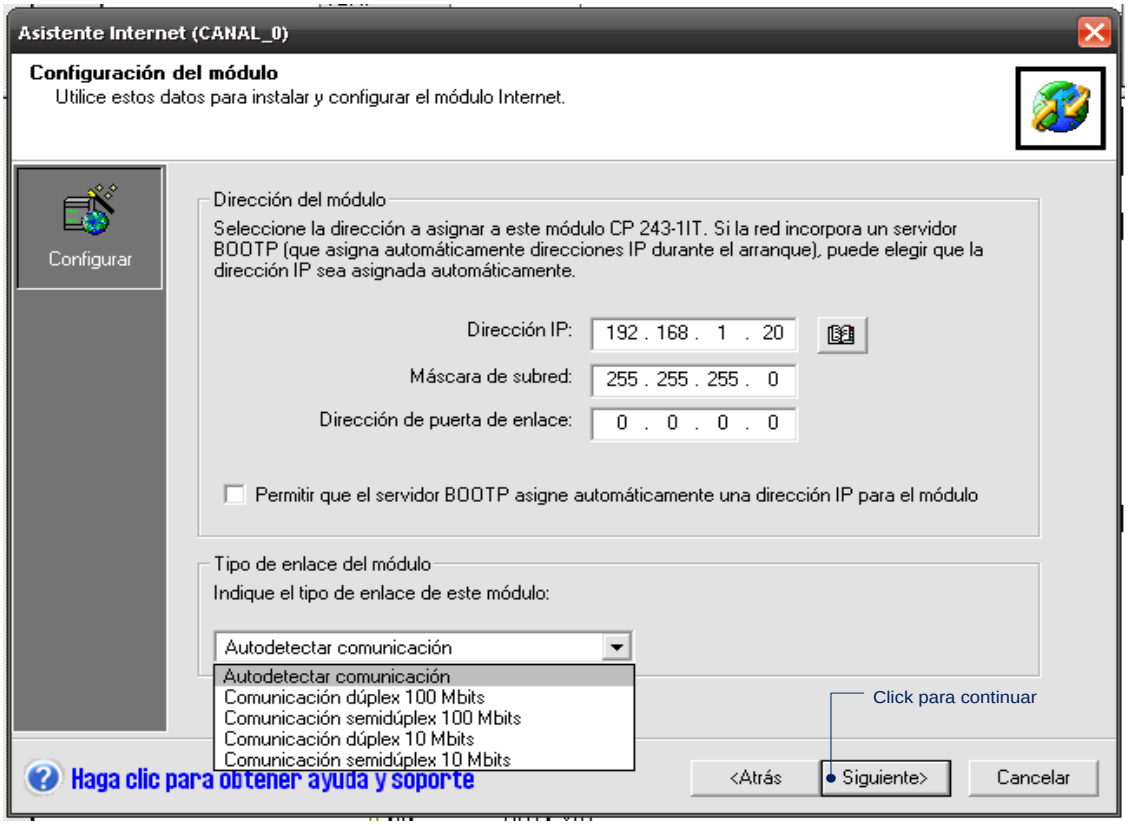
  Para obtener información adicional de las demás posibilidades, consultar el manual del módulo.



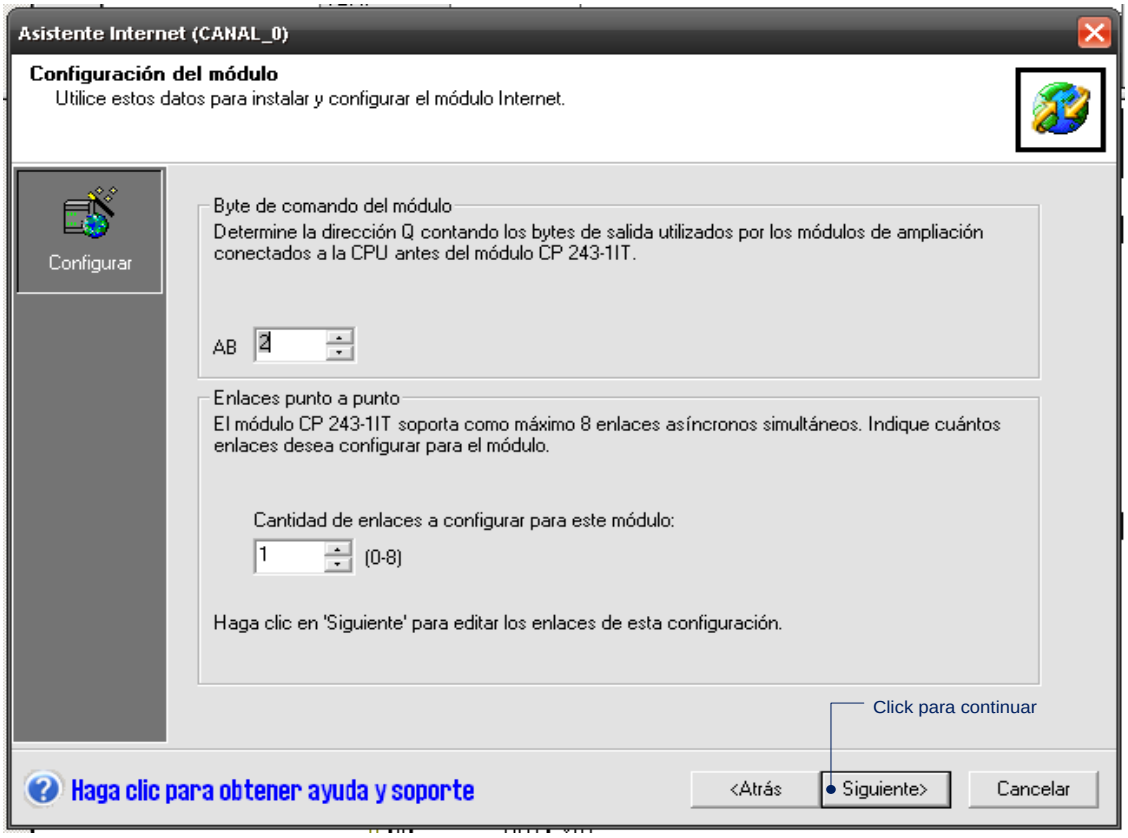
- Se accede al módulo en cuestión.



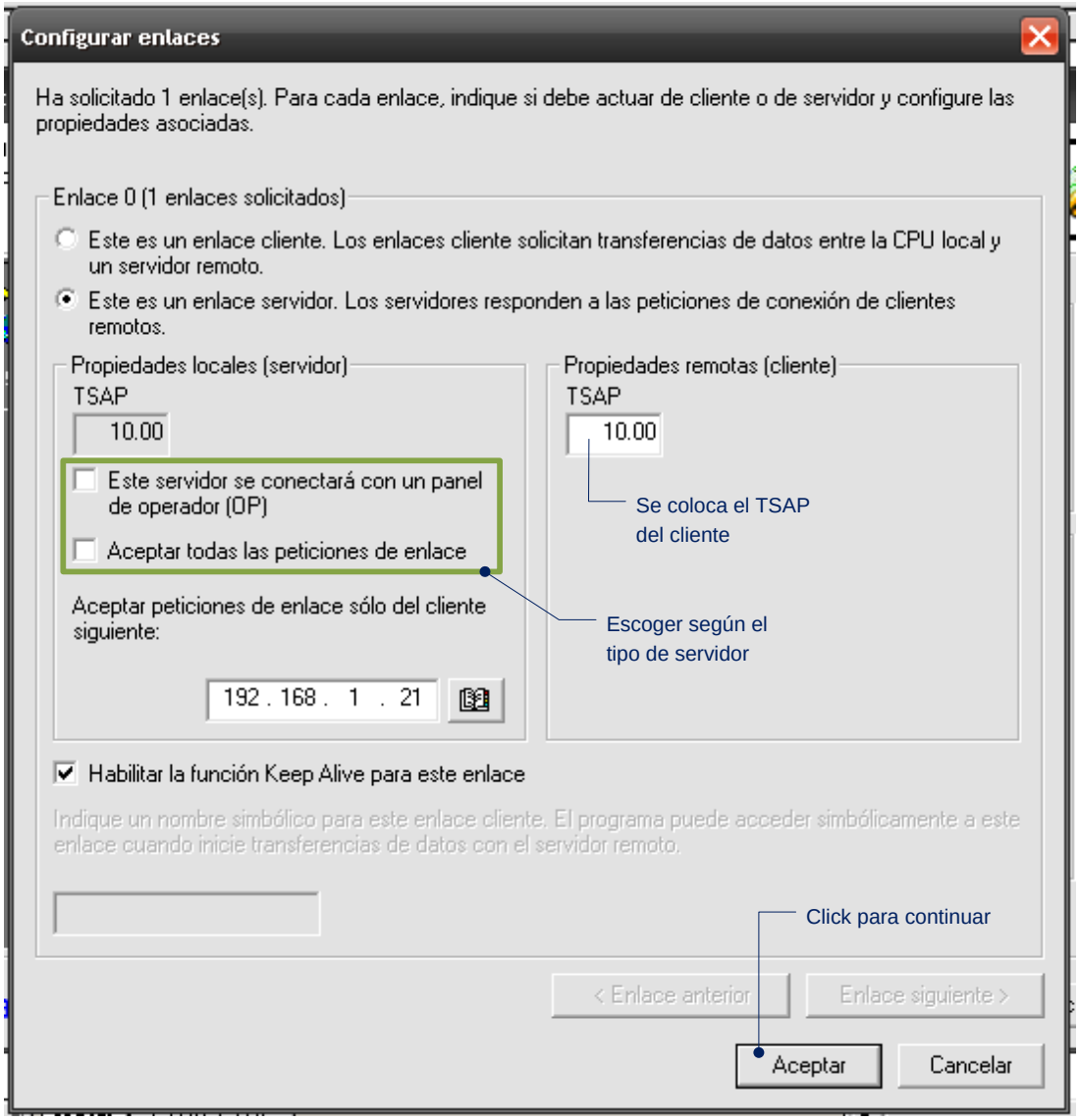
- Bloque para configuración IP y del tipo del enlace.



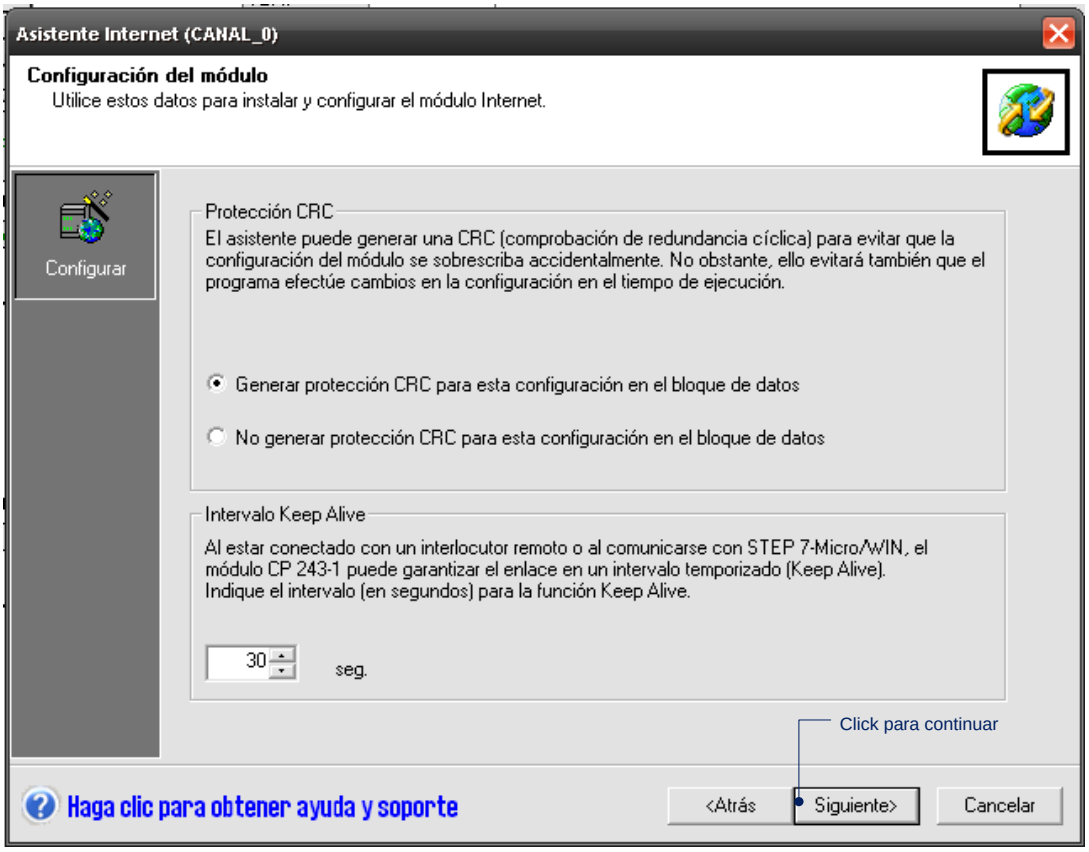
- Configurar el byte de comando y la cantidad de enlaces.



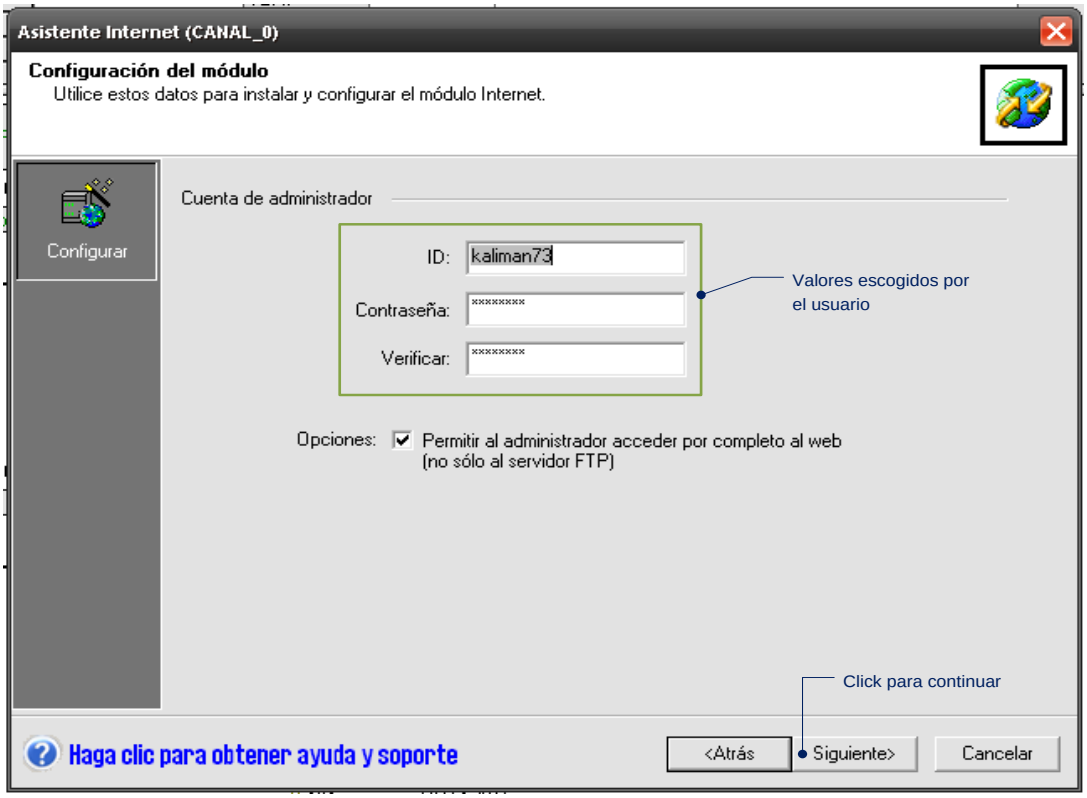
- Se configura como una estación servidor. Por lo que actuará en modo pasivo con los clientes.



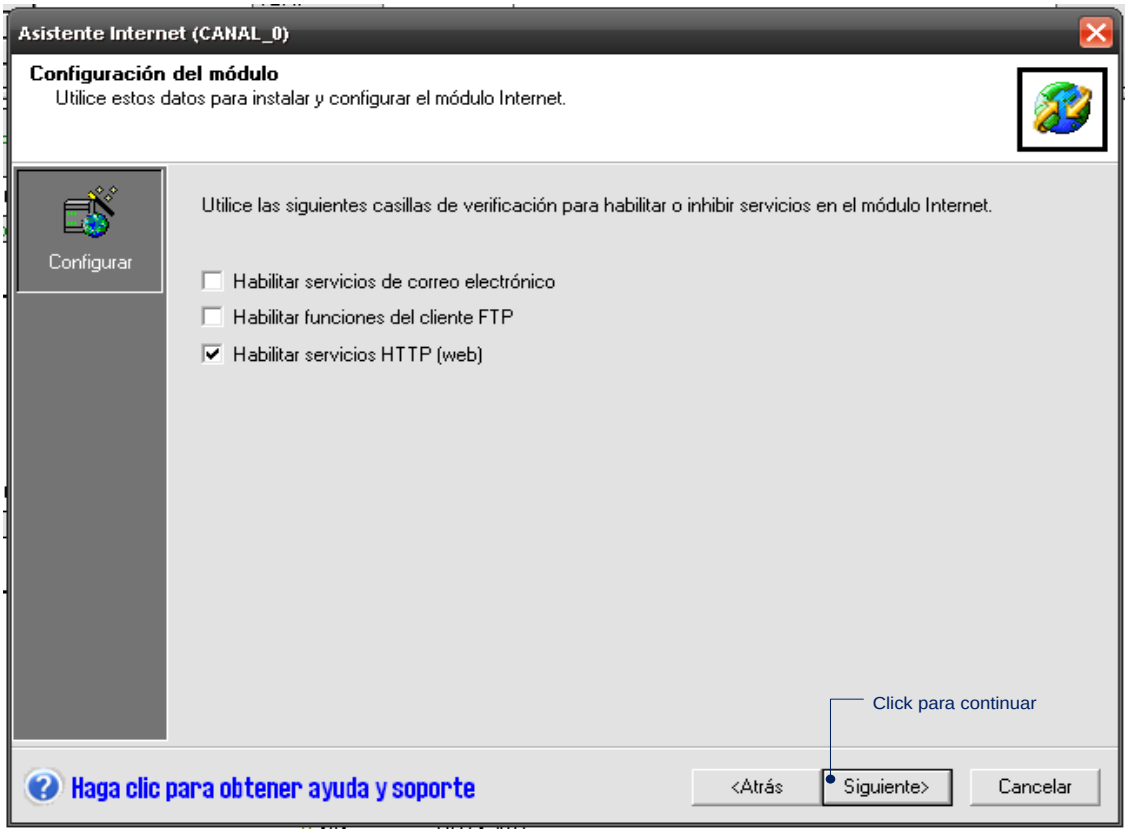
- Configuración del CRC.



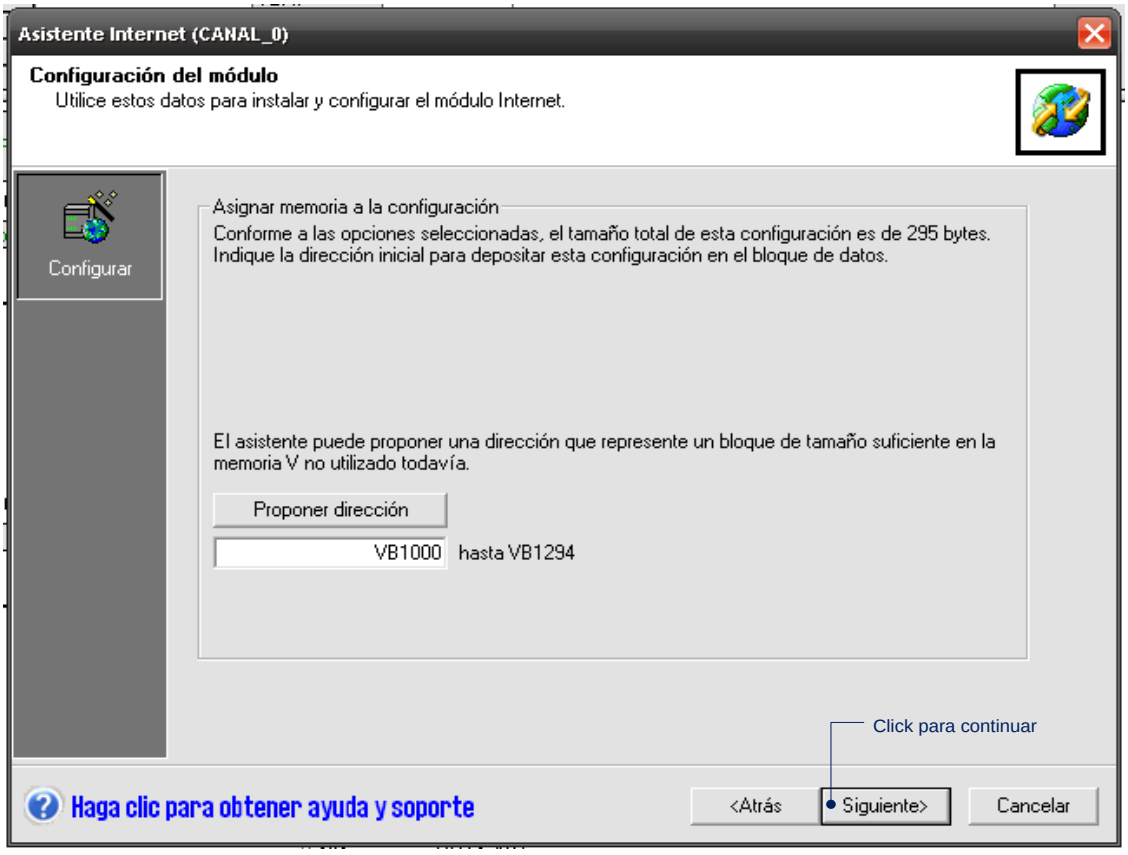
- Estos valores permiten al administrador configurar sus datos.



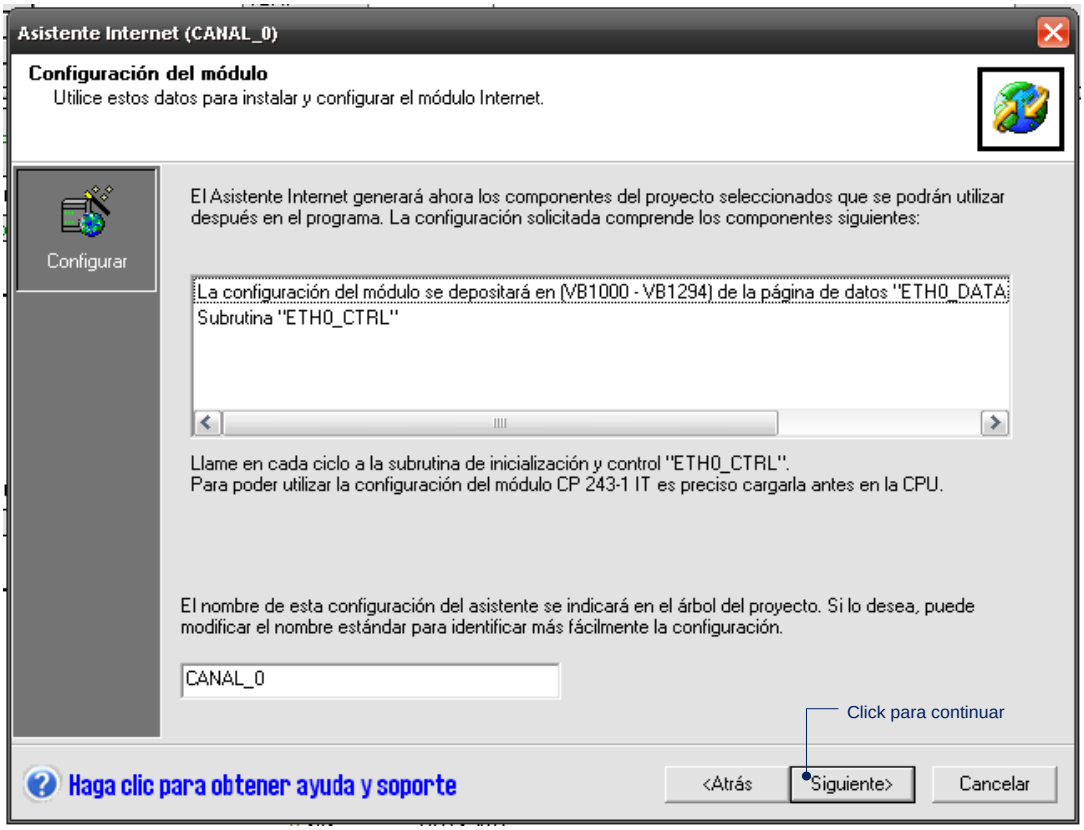
- Servicios del módulo.



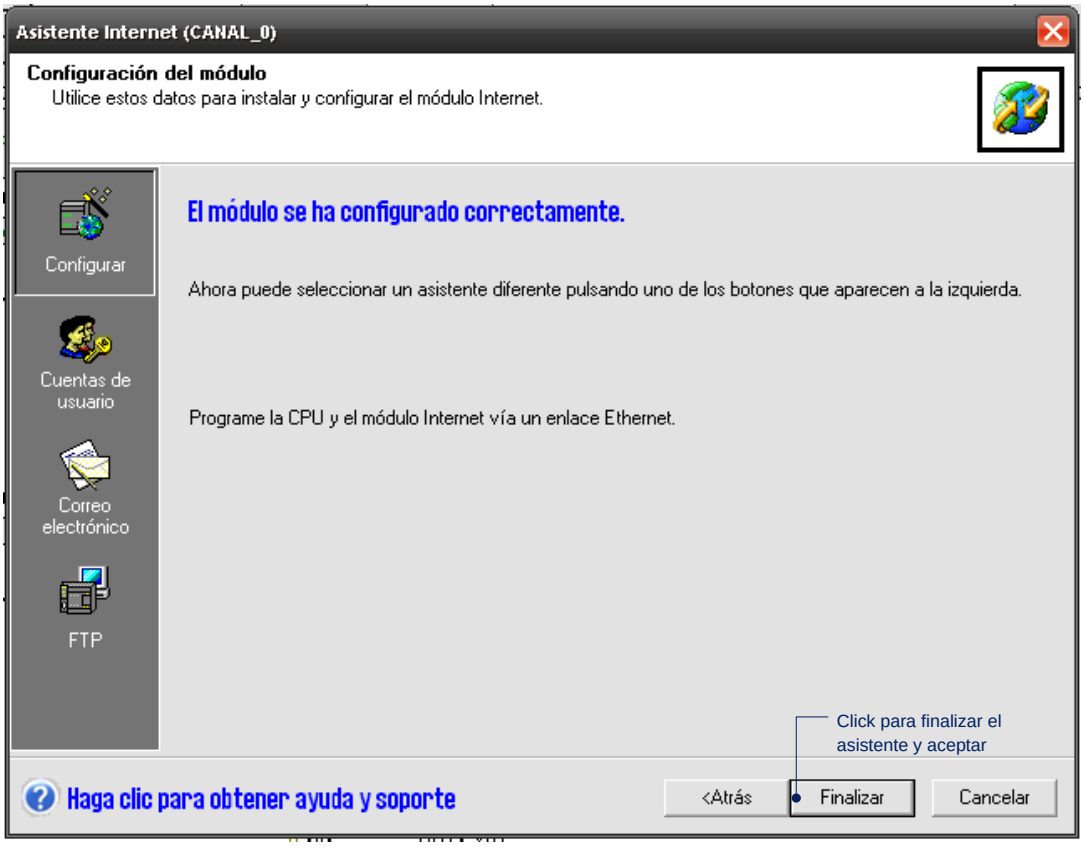
- Escoger el espacio de posmem para colocar los datos de la configuración total.



- Indica que subrutina se ha creado para la comunicación pasiva.



- Finalizar la configuración.



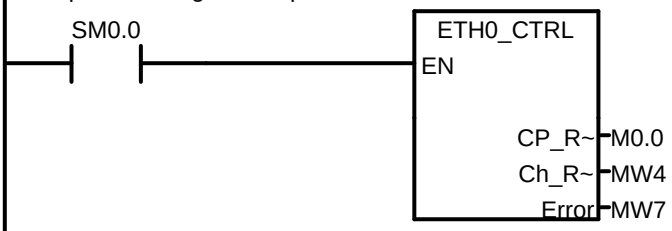
Bloque: MAIN
Autor:
Fecha de creación: 27.07.2001 8:07:16
Fecha de modificación: 30.05.2010 22:12:47

Símbolo	Tipo var.	Tipo de datos	Comentario
	TEMP		
	TEMP		
	TEMP		
	TEMP		

PROGRAMA DEL CLIENTE

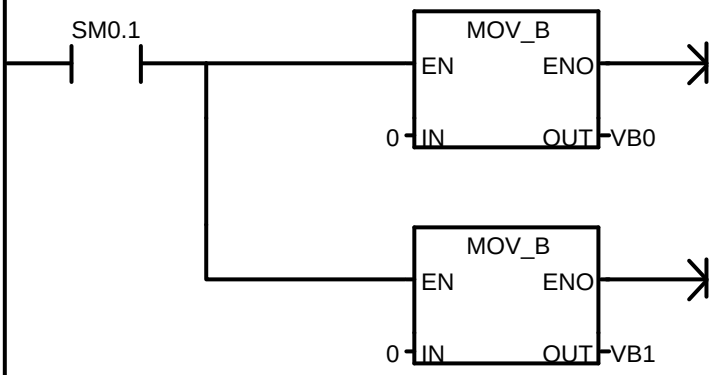
Network 1

Bloque de configuracion para el CP243-1



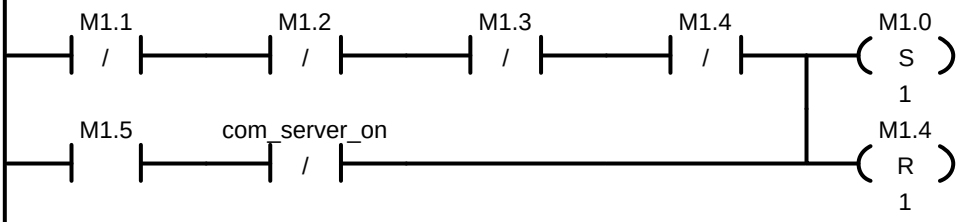
Network 2

Inicializacion de las variables de uso



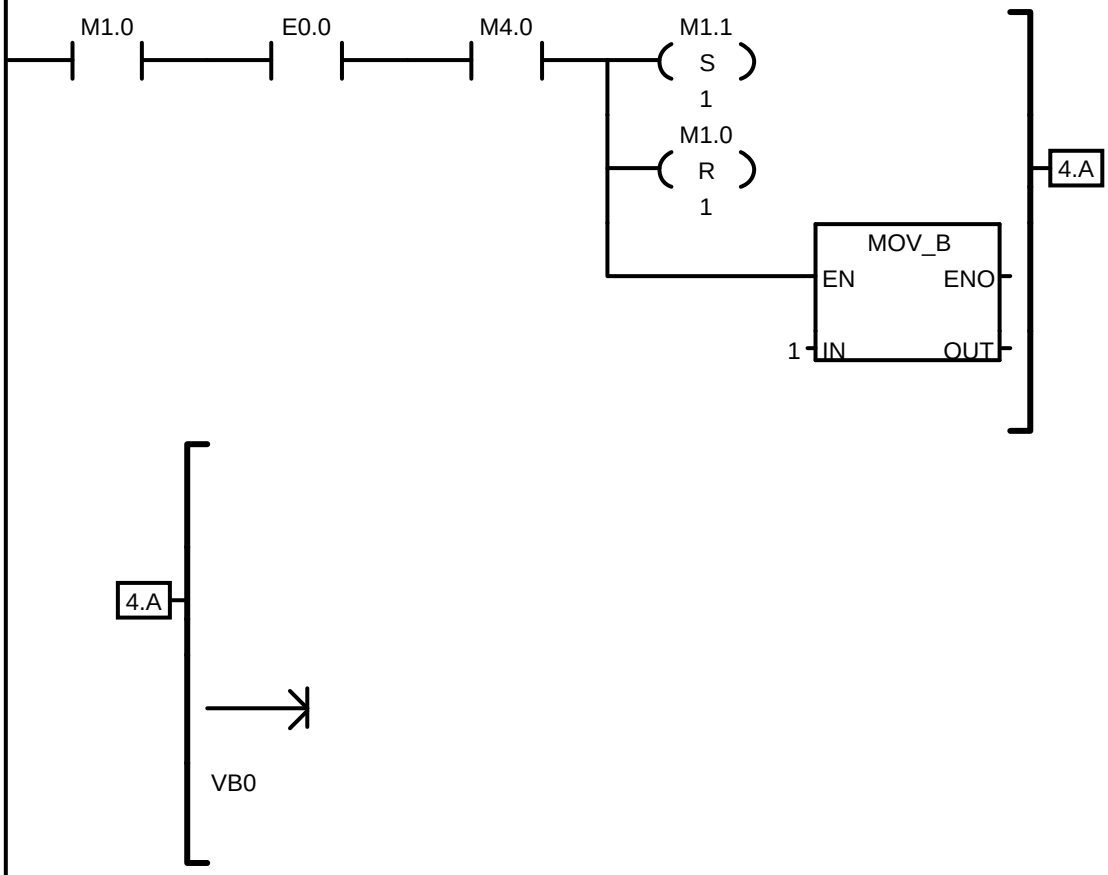
Network 3

Programa para establecer la comunicacion con el servidor

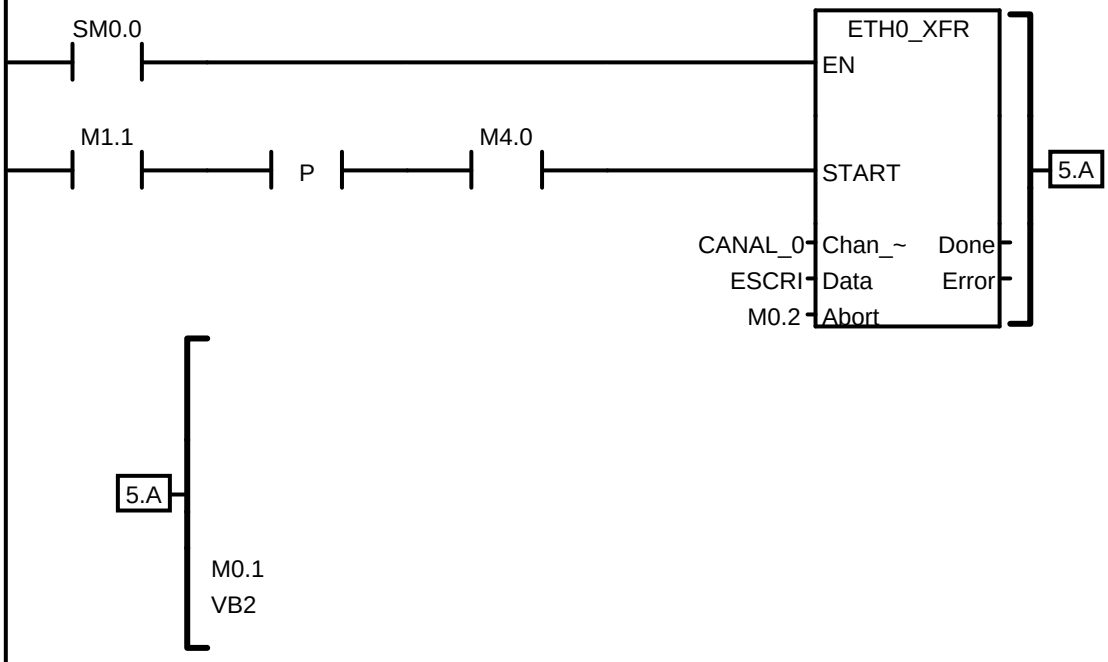


Símbolo	Dirección	Comentario
com_server_on	A0.0	

Network 4

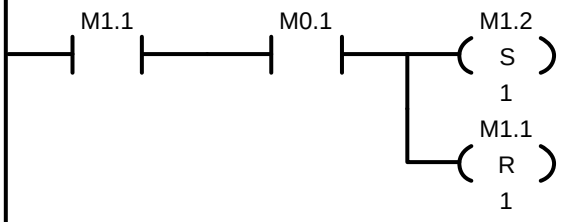


Network 5

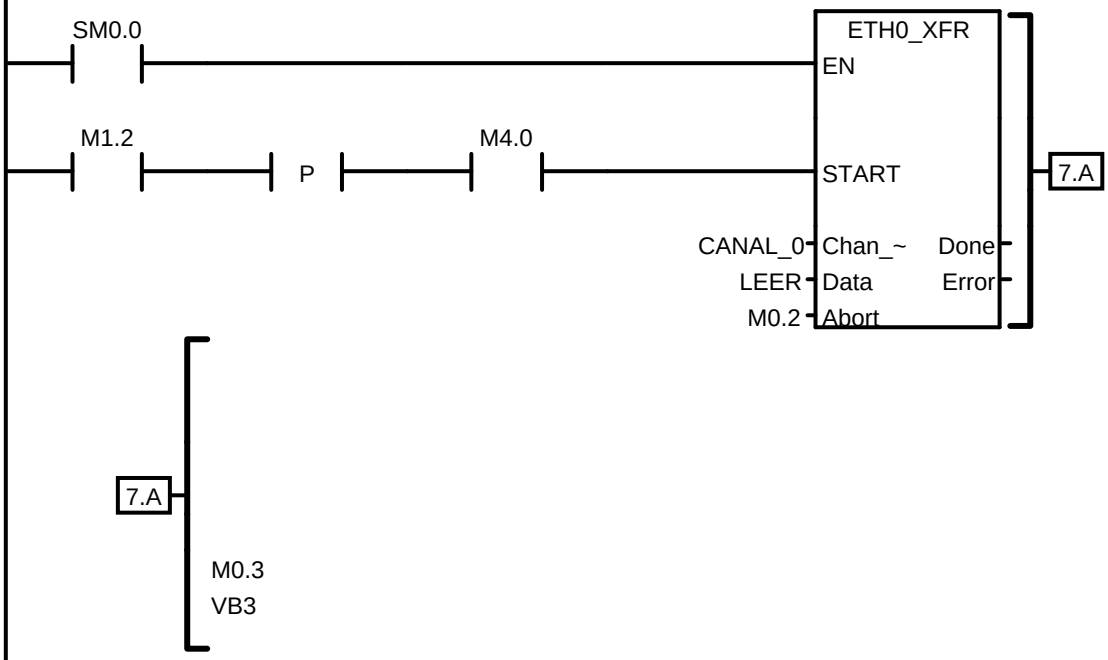


Símbolo	Dirección	Comentario
CANAL_0	VB1158	
ESCRI	VB1160	

Network 6

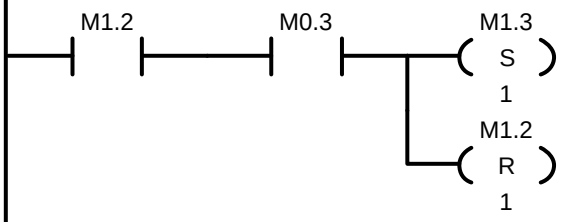


Network 7

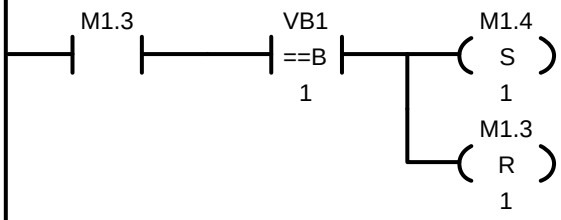


Símbolo	Dirección	Comentario
CANAL_0	VB1158	
LEER	VB1159	

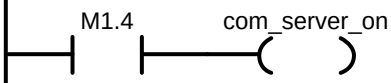
Network 8



Network 9



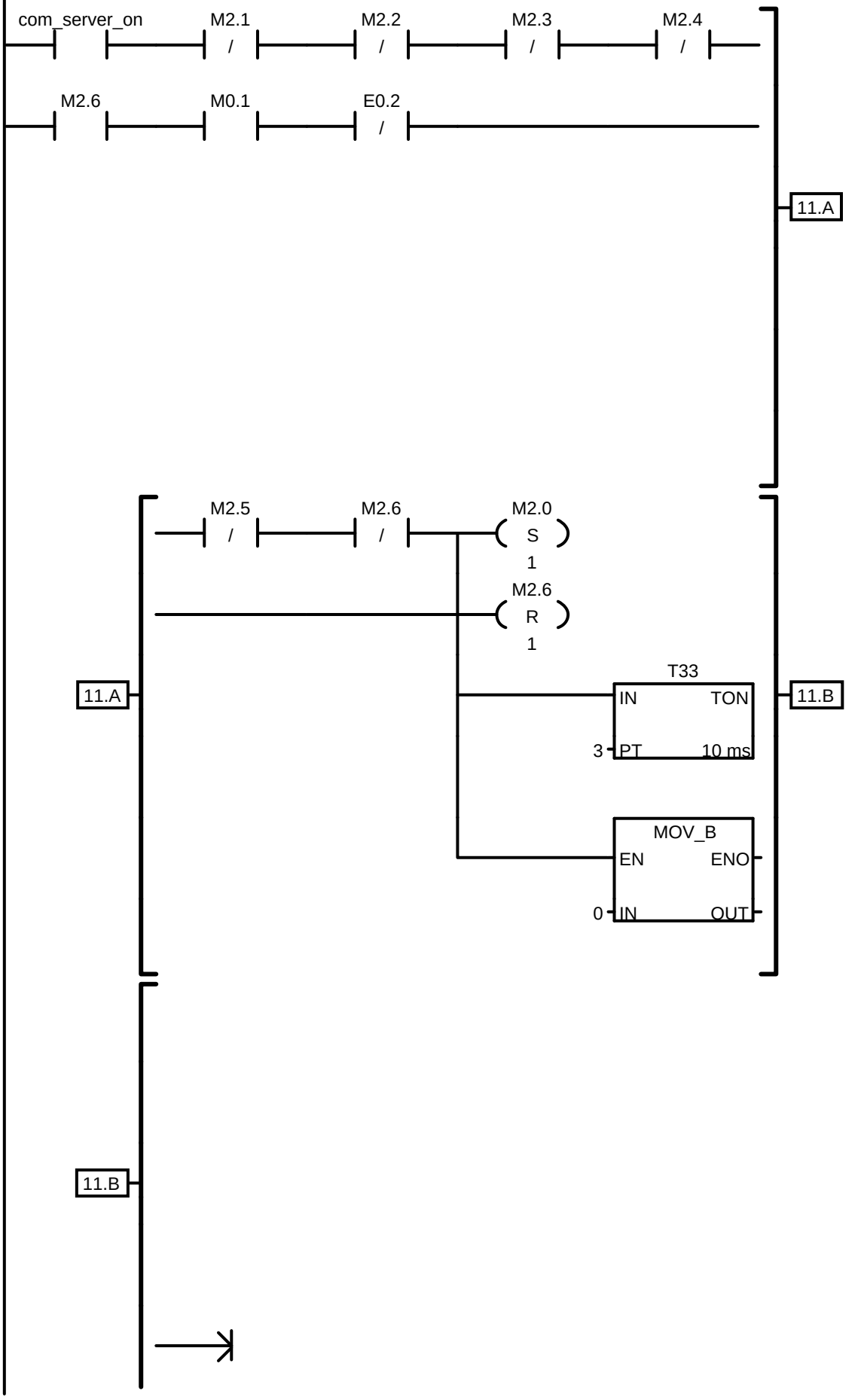
Network 10



Símbolo	Dirección	Comentario
com_server_on	A0.0	

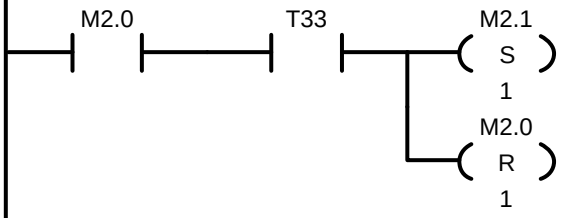
Network 11 **********
****

Programa para controlar la carga y transferir / lee datos hasta / desde el servidor

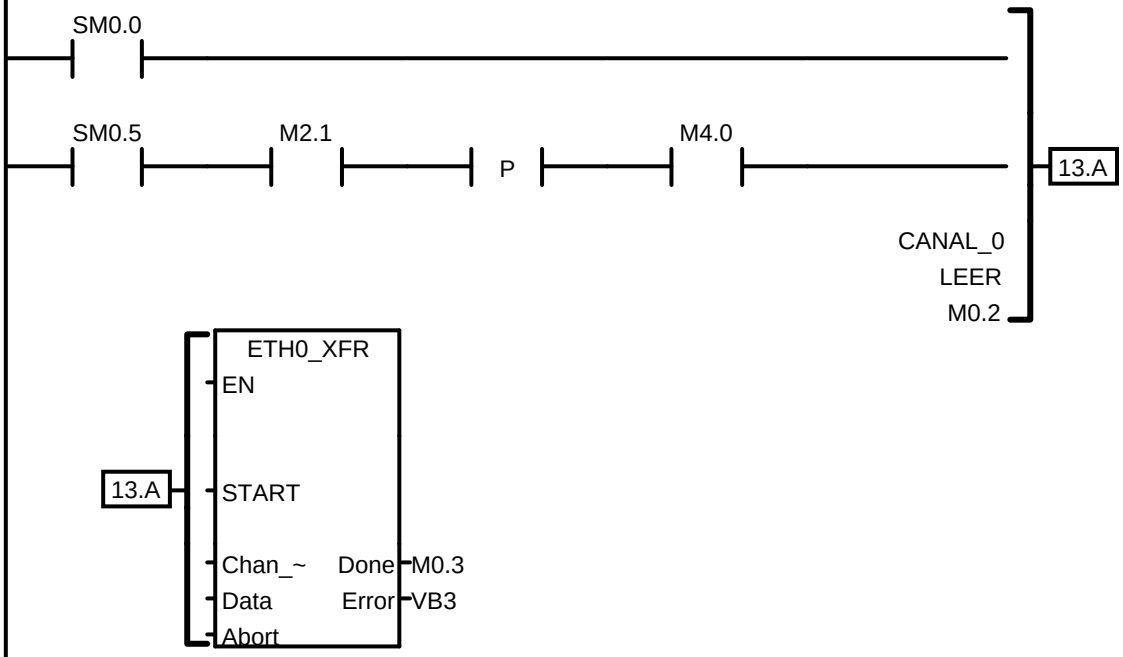


Símbolo	Dirección	Comentario
com_server_on	A0.0	

Network 12

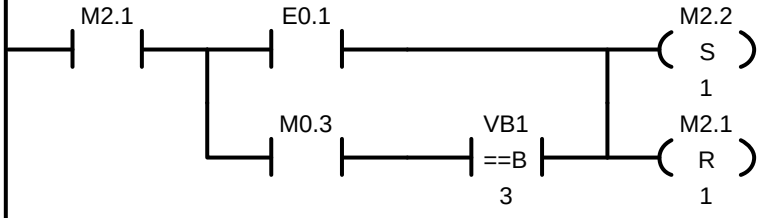


Network 13

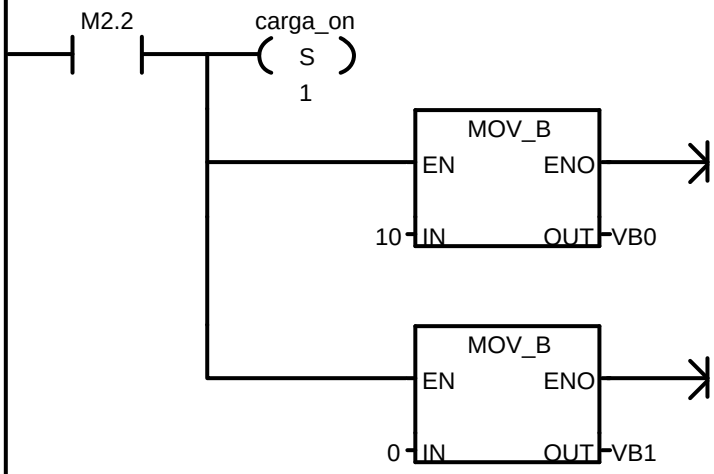


Símbolo	Dirección	Comentario
CANAL_0	VB1158	
LEER	VB1159	

Network 14 ----------

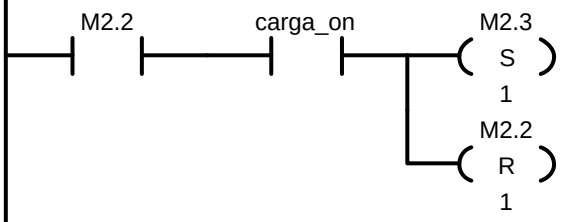


Network 15



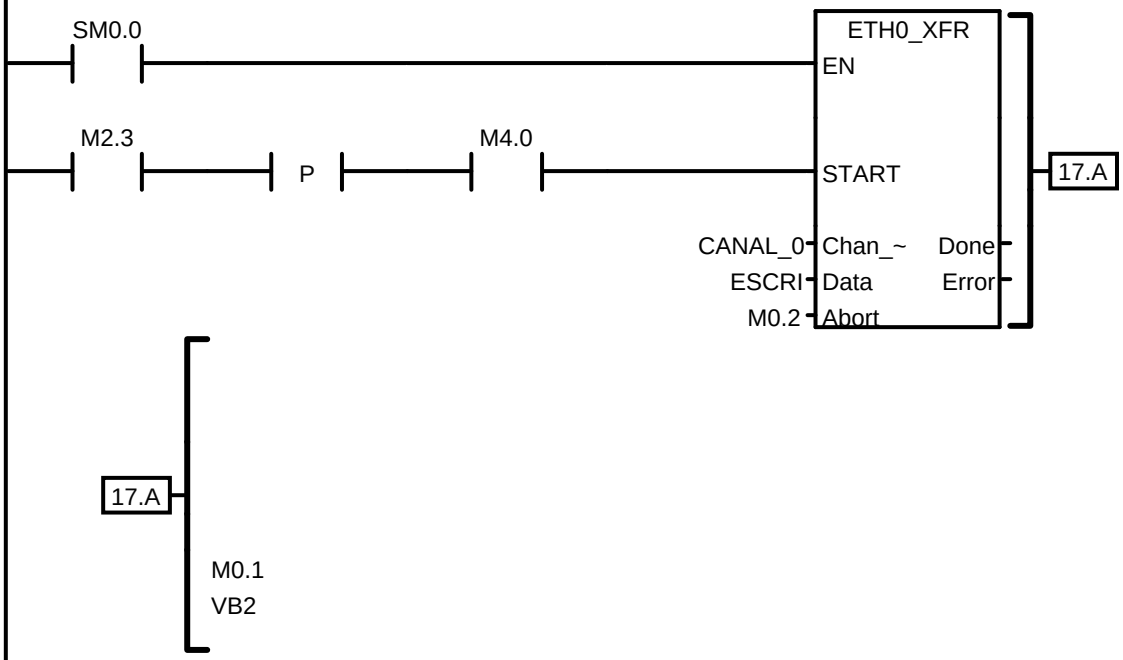
Símbolo	Dirección	Comentario
carga_on	A0.5	

Network 16



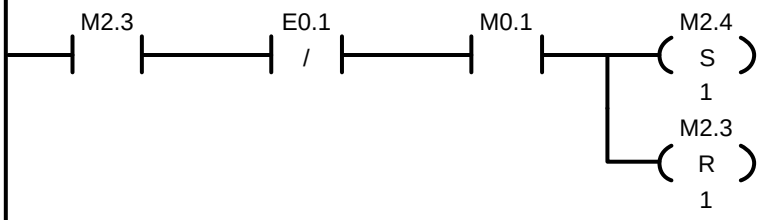
Símbolo	Dirección	Comentario
carga_on	A0.5	

Network 17

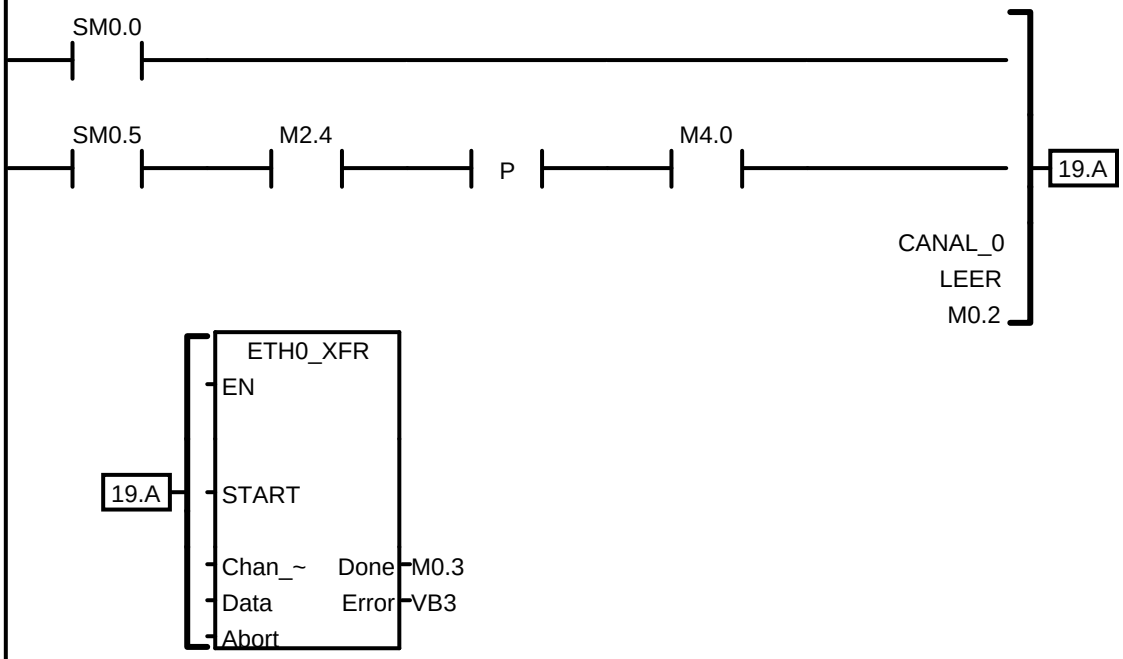


Símbolo	Dirección	Comentario
CANAL_0	VB1158	
ESCRI	VB1160	

Network 18

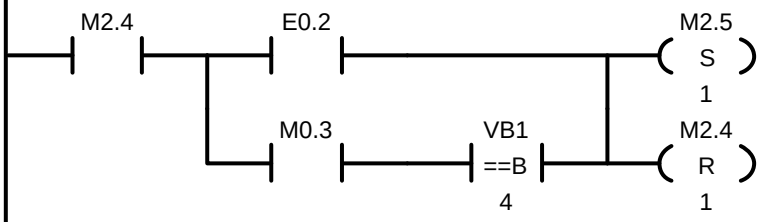


Network 19

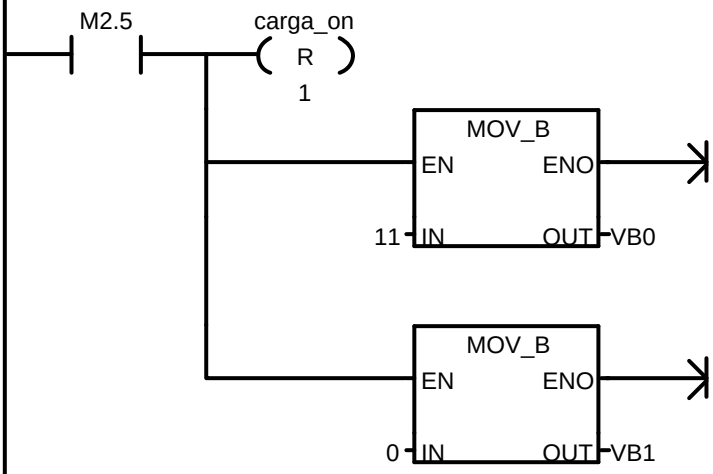


Símbolo	Dirección	Comentario
CANAL_0	VB1158	
LEER	VB1159	

Network 20

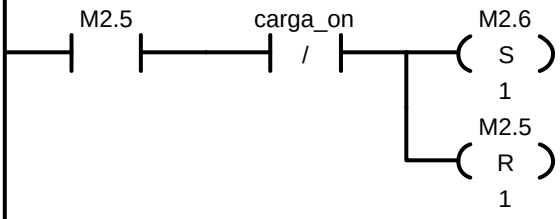


Network 21



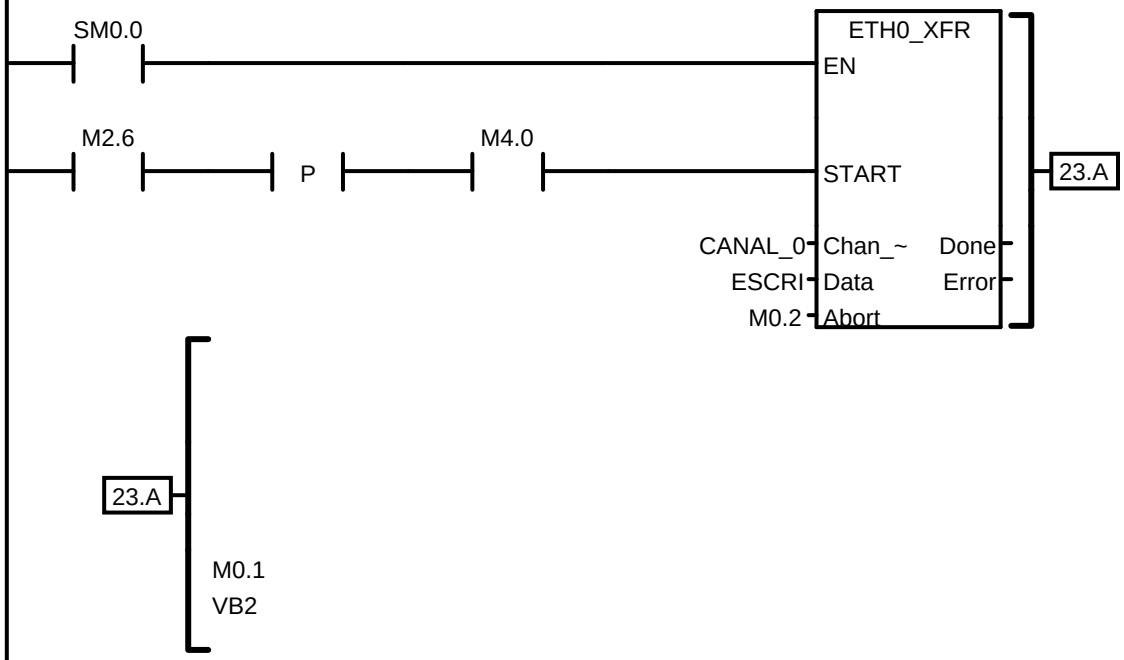
Símbolo	Dirección	Comentario
carga_on	A0.5	

Network 22



Símbolo	Dirección	Comentario
carga_on	A0.5	

Network 23



Símbolo	Dirección	Comentario
CANAL_0	VB1158	
ESCRI	VB1160	

Bloque: ETH0_CTRL
Autor: Asistente Ethernet
Fecha de creación: 11.05.2010 18:15:48
Fecha de modificación: 11.05.2010 18:15:48

	Símbolo	Tipo var.	Tipo de datos	Comentario
	EN	IN	BOOL	
		IN		
		IN_OUT		
L0.0	CP_Ready	OUT	BOOL	El módulo CP 243-1 está listo.
LW1	Ch_Ready	OUT	WORD	Bits de canal listo
LW3	Error	OUT	WORD	Palabra de error
		OUT		
		TEMP		

El asistente Ethernet ha creado esta UOP para utilizarla con un módulo CP 243-1 ubicado en la posición 0. La operación ETHx_CTRL (Control) sirve para habilitar e inicializar el módulo CP 243-1. Utilice esta operación sólo una vez en el proyecto y vigile que el programa de usuario la invoque en cada ciclo. El byte de comando de este módulo se ha definido como QB2.

Los enlaces siguientes ya se han configurado para este módulo:
El enlace 0 es un cliente: TSAP local: 10.00 TSAP remoto: 10.00 Dirección del servidor: 192.168.1.20

Bloque: ETH0_XFR
Autor: Asistente Ethernet
Fecha de creación: 11.05.2010 18:15:48
Fecha de modificación: 11.05.2010 18:15:48

	Símbolo	Tipo var.	Tipo de datos	Comentario
	EN	IN	BOOL	
L0.0	START	IN	BOOL	Enviar comando al CP 243-1 si no está activo
LB1	Chan_ID	IN	BYTE	ID del canal de cliente (0-7)
LB2	Data	IN	BYTE	ID del mensaje a enviar (0-31)
L3.0	Abort	IN	BOOL	1 = Cancelar
		IN		
		IN_OUT		
L3.1	Done	OUT	BOOL	Alto cuando el CP 243-1 finalice el comando
LB4	Error	OUT	BYTE	Estado de error del módulo CP 243-1
		OUT		
		TEMP		

El asistente Ethernet ha creado esta UOP para utilizarla con un módulo CP 243-1 ubicado en la posición 0. La operación ETHx_XFR (Transferir datos) sirve para transferir datos entre el S7-200 y un interlocutor remoto. Ejecute la operación EHTx_CTRL antes de llamar a ETHx_XFR.

Símbolo	Dirección	Comentario
S1_Client	E0.3	
S0_Client	E0.4	
carga_on	A0.5	
com_server_on	A0.0	

Símbolo	Dirección	Comentario
ETH0_CTRL	SBR1	El asistente Ethernet ha creado esta UOP para utilizarla con un módulo CP 243-1 ubicado en la posición 0. La operación ETHx_CTRL (Control) sirve para habilitar e inicializar el módulo CP 243-1. Utilice esta operación sólo una vez en el proyecto y vigile que el programa de usuario la invoque en cada ciclo. El byte de comando de este módulo se ha definido como QB2.
ETH0_XFR	SBR2	El asistente Ethernet ha creado esta UOP para utilizarla con un módulo CP 243-1 ubicado en la posición 0. La operación ETHx_XFR (Transferir datos) sirve para transferir datos entre el S7-200 y un interlocutor remoto. Ejecute la operación EHTx_CTRL antes de llamar a ETHx_XFR.
MAIN	OB1	

Símbolo	Dirección	Comentario
MOD0_CmdByte	AB2	
MOD0_CmdBit	A2.7	
ESCRI	VB1160	
LEER	VB1159	
CANAL_0	VB1158	

Tipo de configuración del asistente
Asistente Ethernet

Nombre de la configuración del asistente
Configuración ETH para 0

Fecha de creación:	11.05.10 18:13:02
Fecha de modificación:	11.05.10 18:13:02

Componentes del proyecto creados con esta configuración del asistente
- La configuración del módulo se depositará en (VB1000 - VB1187) de la página de datos "ETH0_DATA".
- Subrutina "ETH0_CTRL"
- Subrutina "ETH0_XFR"
- Tabla de símbolos globales "ETH0_SYM"

Instrucciones de uso
Llame en cada ciclo a la subrutina de inicialización y control "ETH0_CTRL".
Para poder utilizar la configuración del módulo CP 243-1 es preciso cargarla antes en la CPU.

Información del módulo inteligente

Ubicación del módulo:	0
Byte de comando:	2

Opciones del módulo

Dirección:	192.168.1.21
Máscara de subred:	255.255.255.0
Dirección de puerta de enlace:	0.0.0.0
Tipo de enlace del módulo:	Autodetectar comunicación
Intervalo Keep Alive:	30
Protección CRC:	Habilitada

Enlaces configurados
Nº de enlaces definidos para esta configuración: 1

El enlace 0 es un enlace cliente

Nombre simbólico:	CANAL_0
TSAP local:	10.00
TSAP remoto:	10.00
Dirección del servidor:	192.168.1.20
Keep Alive:	Habilitada

Nº de transferencias de datos definidas para este enlace 0: 2

LEER (Transferencia entre CPUs 0)

Lectura de datos 1 BYTES	Dirección local (VB1 - VB1)	Dirección remota (VB1 - VB1)

ESCRI (Transferencia entre CPUs 1)

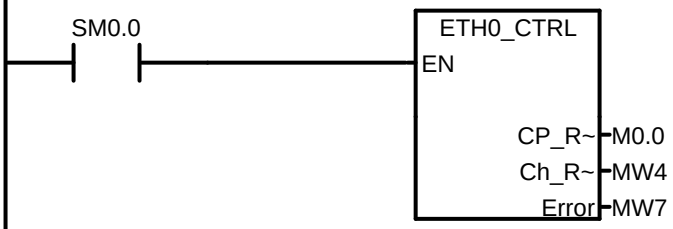
Escritura de datos 1 BYTES	Dirección local (VB0 - VB0)	Dirección remota (VB0 - VB0)

Bloque: MAIN
Autor:
Fecha de creación: 27.07.2001 8:07:16
Fecha de modificación: 30.05.2010 22:08:04

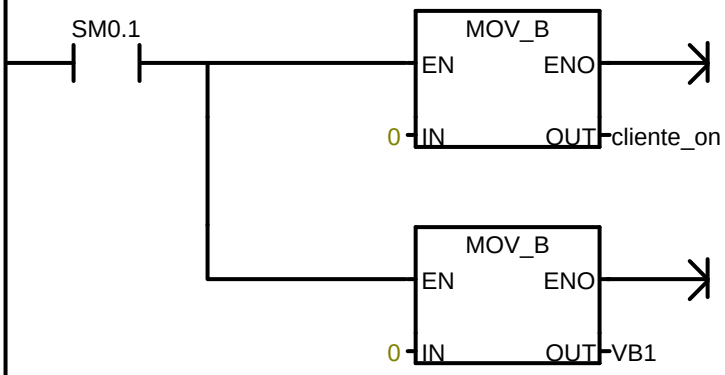
Símbolo	Tipo var.	Tipo de datos	Comentario
	TEMP		
	TEMP		
	TEMP		
	TEMP		

PROGRAMA PARA EL SERVIDOR

Network 1 Programa para establecer la comunicacion con el cliente
Configuracion del CP243-1

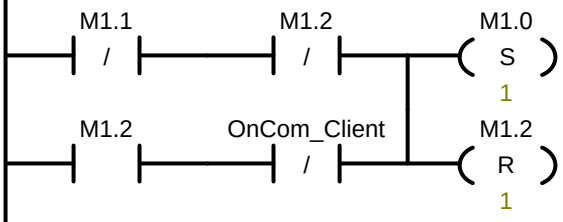


Network 2



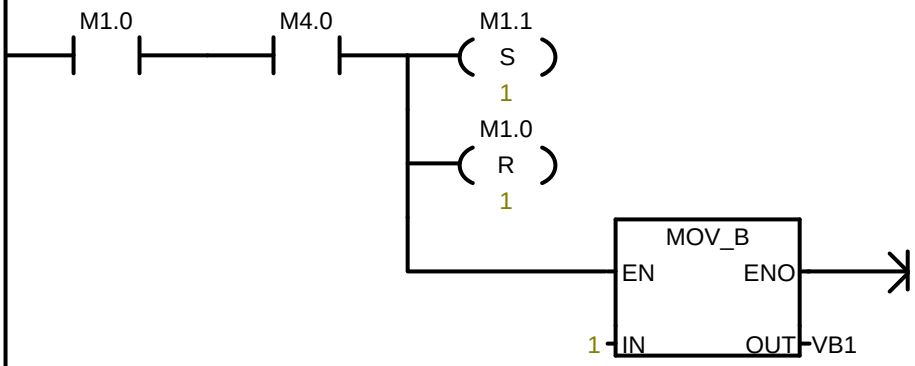
Símbolo	Dirección	Comentario
cliente_on	VB0	

Network 3

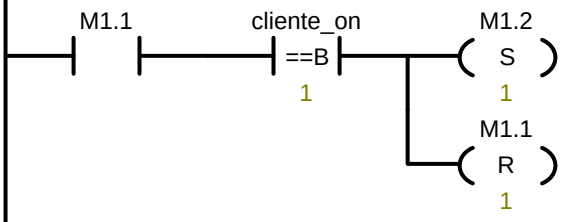


Símbolo	Dirección	Comentario
OnCom_Client	A0.0	

Network 4

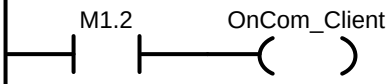


Network 5



Símbolo	Dirección	Comentario
cliente_on	VB0	

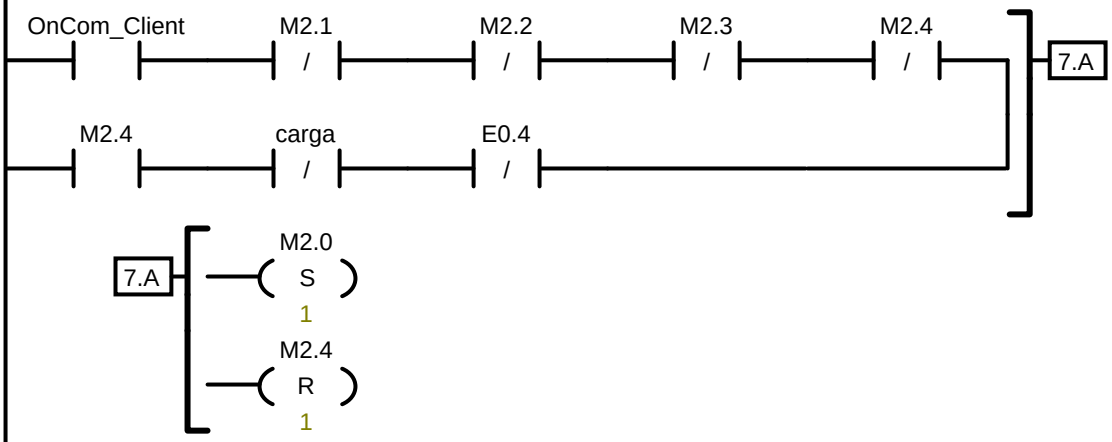
Network 6



Símbolo	Dirección	Comentario
OnCom_Client	A0.0	

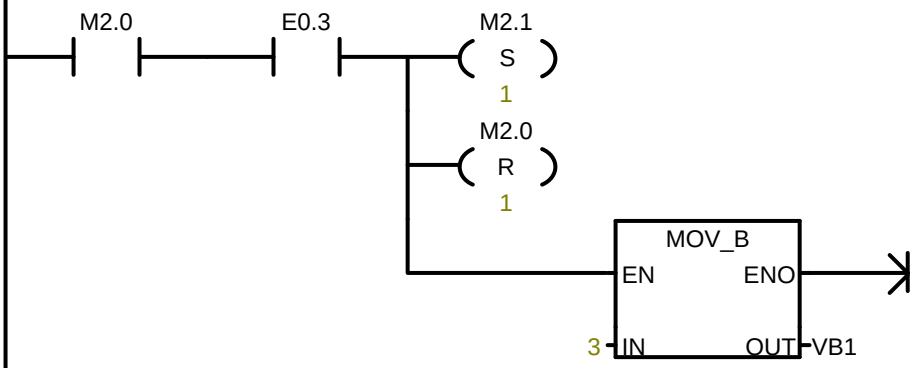
Network 7 ********** **********

Programa para controlar la carga

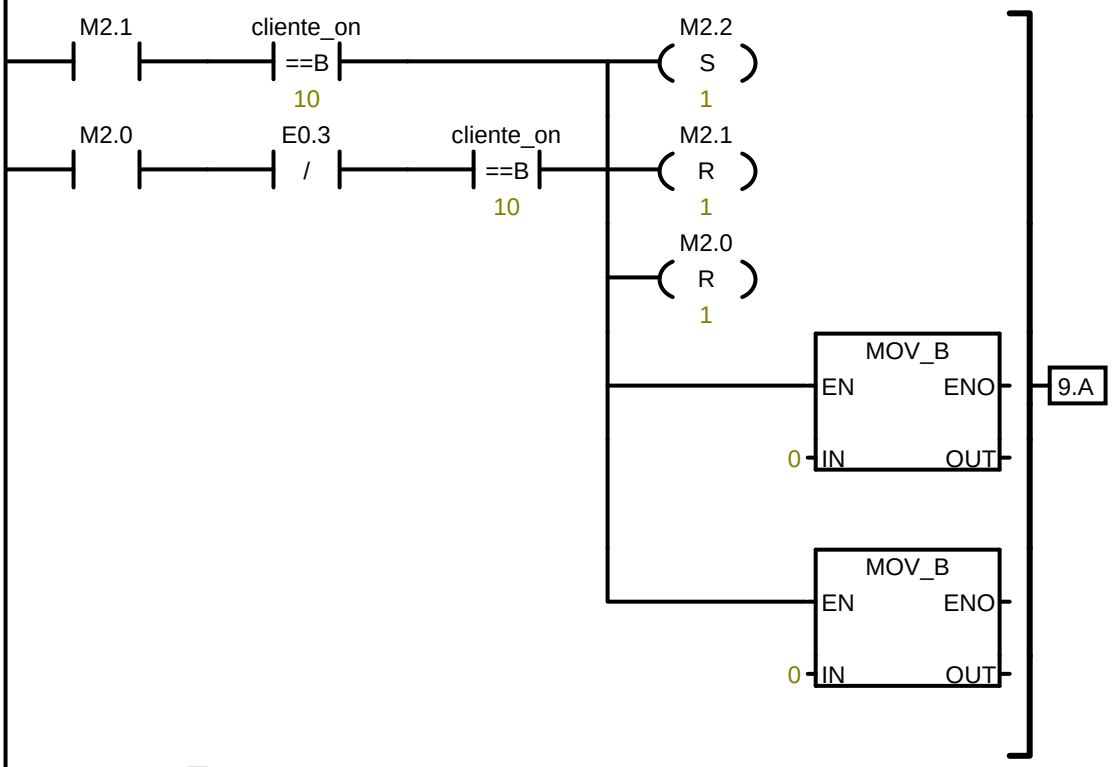


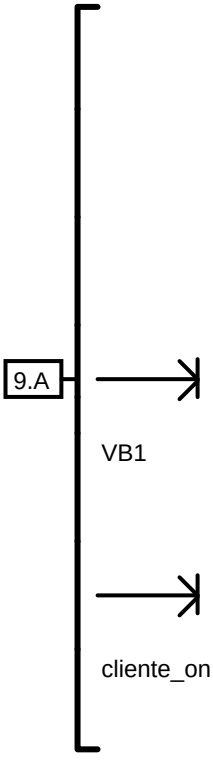
Símbolo	Dirección	Comentario
carga	A0.5	
OnCom_Client	A0.0	

Network 8



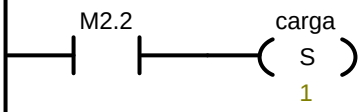
Network 9





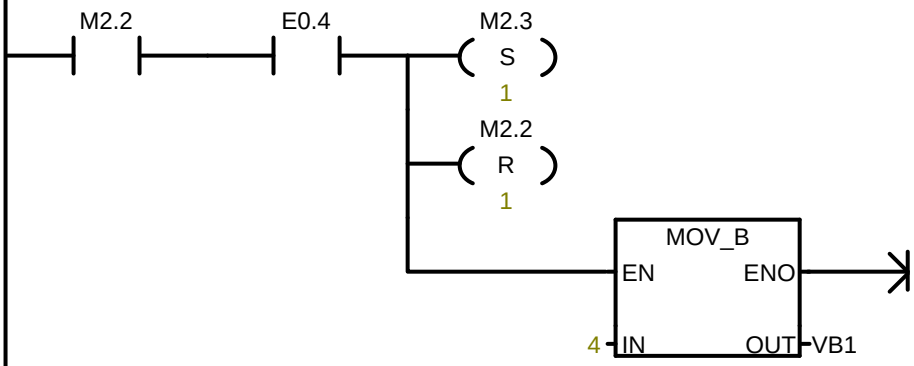
Símbolo	Dirección	Comentario
cliente_on	VB0	

Network 10

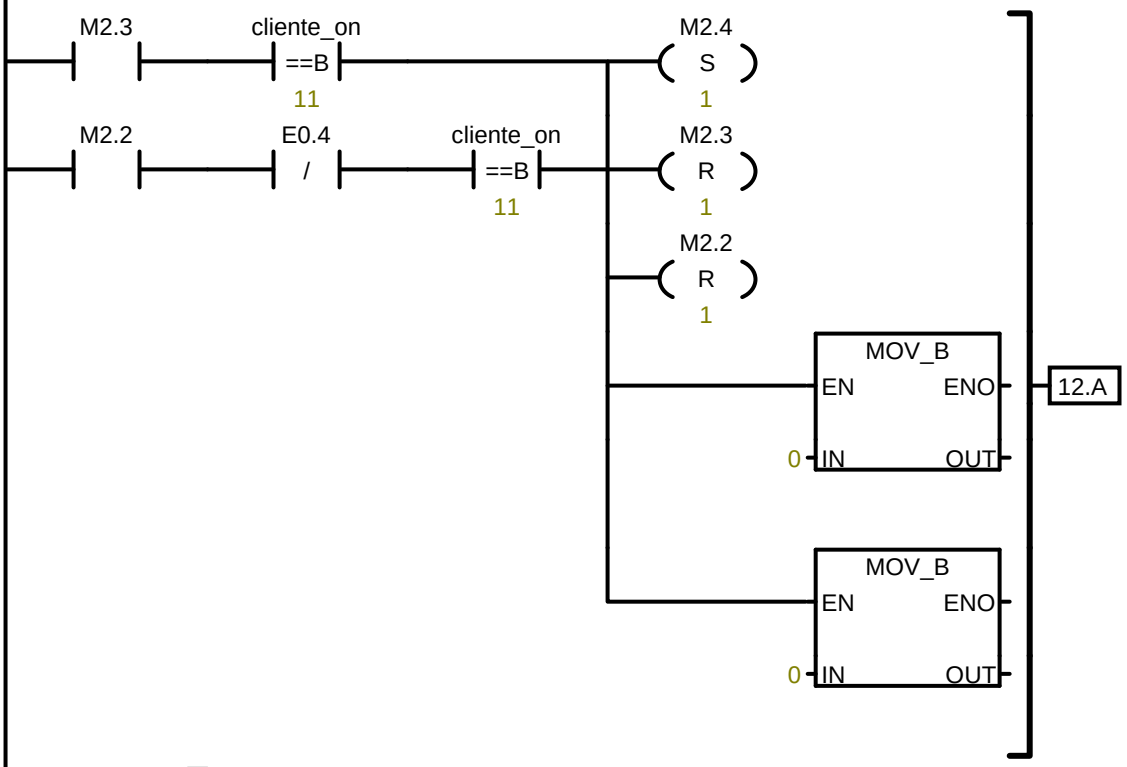


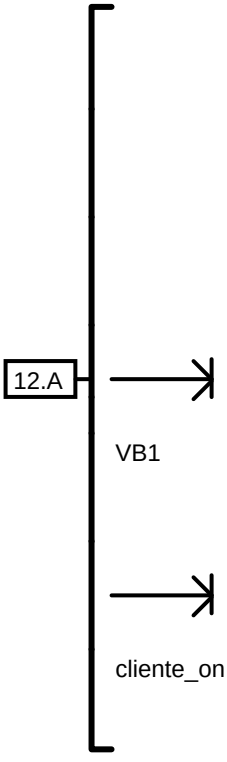
Símbolo	Dirección	Comentario
carga	A0.5	

Network 11



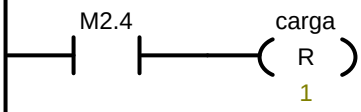
Network 12





Símbolo	Dirección	Comentario
cliente_on	VB0	

Network 13



Símbolo	Dirección	Comentario
carga	A0.5	

Bloque: ETH0_CTRL
Autor: Asistente Ethernet
Fecha de creación: 11.05.2010 18:02:55
Fecha de modificación: 11.05.2010 18:02:55

	Símbolo	Tipo var.	Tipo de datos	Comentario
	EN	IN	BOOL	
		IN		
		IN_OUT		
L0.0	CP_Ready	OUT	BOOL	El módulo CP 243-1 está listo.
LW1	Ch_Ready	OUT	WORD	Bits de canal listo
LW3	Error	OUT	WORD	Palabra de error
		OUT		
		TEMP		

El asistente Internet ha creado esta UOP para utilizarla con un módulo CP 243-IT ubicado en la posición 0. La operación ETHx_CTRL (Control) sirve para habilitar e inicializar el módulo CP 243-1. Utilice esta operación sólo una vez en el proyecto y vigile que el programa de usuario la invoque en cada ciclo. El byte de comando de este módulo se ha definido como QB2.

Los enlaces siguientes ya se han configurado para este módulo:
El enlace 0 es un servidor: TSAP local: 10.00 TSAP remoto: 10.00 Direcciones de clientes: 192.168.1.21

Símbolo	Dirección	Comentario
OnCom_Client	A0.0	
carga	A0.5	
cliente_on	VB0	

Símbolo	Dirección	Comentario
ETH0_CTRL	SBR1	El asistente Internet ha creado esta UOP para utilizarla con un módulo CP 243-IT ubicado en la posición 0. La operación ETHx_CTRL (Control) sirve para habilitar e inicializar el módulo CP 243-1. Utilice esta operación sólo una vez en el proyecto y vigile que el programa de usuario la invoque en cada ciclo. El byte de comando de este módulo se ha definido como QB2.
MAIN	OB1	

Tipo de configuración del asistente
Asistente Internet

Nombre de la configuración del asistente
CANAL_0

Fecha de creación:	11.05.10 18:00:41
Fecha de modificación:	11.05.10 18:00:41

Componentes del proyecto creados con esta configuración del asistente
- La configuración del módulo se depositará en (VB1000 - VB1294) de la página de datos "ETH0_DATA".
- Subrutina "ETH0_CTRL"

Instrucciones de uso
Llame en cada ciclo a la subrutina de inicialización y control "ETH0_CTRL".
Para poder utilizar la configuración del módulo CP 243-1 IT es preciso cargarla antes en la CPU.

Información del módulo inteligente

Ubicación del módulo:	0
Byte de comando:	2

Opciones del módulo

Dirección:	192.168.1.20
Máscara de subred:	255.255.255.0
Dirección de puerta de enlace:	0.0.0.0
Tipo de enlace del módulo:	Autodetectar comunicación
Intervalo Keep Alive:	30
Protección CRC:	Habilitada

Enlaces configurados
Nº de enlaces definidos para esta configuración: 1

El enlace 0 es un enlace servidor

TSAP local:	10.00
TSAP remoto:	10.00
Keep Alive:	Habilitada
El servidor aceptará las peticiones de conexión de:	192.168.1.21

Servicios de Internet:

Servicios de correo electrónico:	Inhibida
Servicios del cliente FTP:	Inhibida
Servicios HTTP:	Habilitada
Inicio de sesión de administrador:	kaliman73

Configuración del correo electrónico

Dirección de correo electrónico:	
Servidor SMTP 1:	0.0.0.0
Puerto:	25
Servidor SMTP 2:	0.0.0.0
Puerto:	25

Correo electrónico 0 No configurada

Correo electrónico 1 No configurada

Correo electrónico 2 No configurada

Correo electrónico 3 No configurada

Correo electrónico 4 No configurada

Correo electrónico 5 No configurada

Correo electrónico 6 No configurada

Correo electrónico 7 No configurada

Correo electrónico 8 No configurada

Correo electrónico 9 No configurada

Correo electrónico 10 No configurada

Correo electrónico 11 No configurada

Correo electrónico 12 No configurada

Correo electrónico 13 No configurada

Correo electrónico 14 No configurada

Correo electrónico 15 No configurada

Correo electrónico 16 No configurada

Correo electrónico 17 No configurada

Correo electrónico 18 No configurada

Correo electrónico 19 No configurada

Correo electrónico 20 No configurada

Correo electrónico 21 No configurada

Correo electrónico 22 No configurada

Correo electrónico 23 No configurada

Correo electrónico 24 No configurada

Correo electrónico 25 No configurada

Correo electrónico 26 No configurada

Correo electrónico 27 No configurada

Correo electrónico 28 No configurada

Correo electrónico 29 No configurada

Correo electrónico 30 No configurada

Correo electrónico 31 No configurada

Funciones del cliente FTP

Función FTP 0 No configurada

Función FTP 1 No configurada

Función FTP 2 No configurada

Función FTP 3 No configurada

Función FTP 4 No configurada

Función FTP 5 No configurada

Función FTP 6 No configurada

Función FTP 7 No configurada

Función FTP 8 No configurada

Función FTP 9 No configurada

Función FTP 10 No configurada

Función FTP 11 No configurada

Función FTP 12 No configurada

Función FTP 13 No configurada

Función FTP 14 No configurada

Función FTP 15 No configurada

Función FTP 16 No configurada

Función FTP 17 No configurada

Función FTP 18 No configurada

Función FTP 19 No configurada

Función FTP 20 No configurada

Función FTP 21 No configurada

Función FTP 22 No configurada

Función FTP 23 No configurada

Función FTP 24 No configurada

Función FTP 25 No configurada

Función FTP 26 No configurada

Función FTP 27 No configurada

Función FTP 28 No configurada

Función FTP 29 No configurada

Función FTP 30 No configurada

Función FTP 31 No configurada

Cuentas de usuario

Cuenta de usuario 0 No configurada

Cuenta de usuario 1 No configurada

Cuenta de usuario 2 No configurada

Cuenta de usuario 3 No configurada

Cuenta de usuario 4 No configurada

Cuenta de usuario 5 No configurada

Cuenta de usuario 6 No configurada

Cuenta de usuario 7 No configurada

Elemento	Bloque	Ubicación	Contexto
E0.3	MAIN (OB1)	Network 8	-| |-
E0.3	MAIN (OB1)	Network 9	-|/|-
E0.4	MAIN (OB1)	Network 7	-|/|-
E0.4	MAIN (OB1)	Network 11	-| |-
E0.4	MAIN (OB1)	Network 12	-|/|-
AB2	ETH0_CTRL (SBR1)	***	BIW
OnCom_Client	MAIN (OB1)	Network 3	-|/|-
OnCom_Client	MAIN (OB1)	Network 6	-( )
OnCom_Client	MAIN (OB1)	Network 7	-| |-
carga	MAIN (OB1)	Network 7	-|/|-
carga	MAIN (OB1)	Network 10	-(S)
carga	MAIN (OB1)	Network 13	-(R)
A2.7	ETH0_CTRL (SBR1)	***	RI
A2.7	ETH0_CTRL (SBR1)	***	SI
&VB1000	ETH0_CTRL (SBR1)	***	MOVD
VW1271	ETH0_CTRL (SBR1)	***	FILL
VW1273	ETH0_CTRL (SBR1)	***	FILL (rango)
VW1275	ETH0_CTRL (SBR1)	***	FILL (rango)
VW1277	ETH0_CTRL (SBR1)	***	FILL (rango)
VW1279	ETH0_CTRL (SBR1)	***	FILL (rango)
VW1281	ETH0_CTRL (SBR1)	***	FILL (rango)
VW1283	ETH0_CTRL (SBR1)	***	FILL (rango)
VW1285	ETH0_CTRL (SBR1)	***	FILL (rango)
VW1287	ETH0_CTRL (SBR1)	***	FILL (rango)
VW1291	ETH0_CTRL (SBR1)	***	FILL
VW1293	ETH0_CTRL (SBR1)	***	FILL (rango)
cliente_on	MAIN (OB1)	Network 2	MOV_B
cliente_on	MAIN (OB1)	Network 5	-|==B|-
cliente_on	MAIN (OB1)	Network 9	-|==B|-
cliente_on	MAIN (OB1)	Network 9	-|==B|-
cliente_on	MAIN (OB1)	Network 9	MOV_B
cliente_on	MAIN (OB1)	Network 12	-|==B|-
cliente_on	MAIN (OB1)	Network 12	-|==B|-
cliente_on	MAIN (OB1)	Network 12	MOV_B
VB1	MAIN (OB1)	Network 2	MOV_B
VB1	MAIN (OB1)	Network 4	MOV_B
VB1	MAIN (OB1)	Network 8	MOV_B
VB1	MAIN (OB1)	Network 9	MOV_B
VB1	MAIN (OB1)	Network 11	MOV_B
VB1	MAIN (OB1)	Network 12	MOV_B
VB1000	ETH0_CTRL (SBR1)	***	BMB
VB1000	ETH0_CTRL (SBR1)	***	BMB
VB1001	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1001	ETH0_CTRL (SBR1)	***	BMB (rango)

Elemento	Bloque	Ubicación	Contexto
VB1002	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1002	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1003	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1003	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1004	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1004	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1005	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1005	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1006	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1006	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1007	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1007	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1008	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1008	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1009	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1009	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1010	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1010	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1011	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1011	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1012	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1012	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1013	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1013	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1014	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1014	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1015	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1015	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1016	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1016	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1017	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1017	ETH0_CTRL (SBR1)	***	BMB (rango)

Elemento	Bloque	Ubicación	Contexto
VB1018	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1018	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1019	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1019	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1020	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1020	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1021	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1021	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1022	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1022	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1023	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1023	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1024	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1024	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1025	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1025	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1026	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1026	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1027	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1027	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1028	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1028	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1029	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1029	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1030	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1030	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1031	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1031	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1032	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1032	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1033	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1033	ETH0_CTRL (SBR1)	***	BMB (rango)

Elemento	Bloque	Ubicación	Contexto
VB1034	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1034	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1035	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1035	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1036	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1036	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1037	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1037	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1038	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1038	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1039	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1039	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1040	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1040	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1041	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1041	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1042	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1042	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1043	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1043	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1044	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1044	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1045	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1045	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1046	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1046	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1047	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1047	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1048	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1048	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1049	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1049	ETH0_CTRL (SBR1)	***	BMB (rango)

Elemento	Bloque	Ubicación	Contexto
VB1050	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1050	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1051	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1051	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1052	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1052	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1053	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1053	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1054	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1054	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1055	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1055	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1056	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1056	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1057	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1057	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1058	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1058	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1059	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1059	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1060	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1060	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1061	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1061	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1062	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1062	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1063	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1063	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1064	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1064	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1065	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1065	ETH0_CTRL (SBR1)	***	BMB (rango)

Elemento	Bloque	Ubicación	Contexto
VB1066	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1066	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1067	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1067	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1068	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1068	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1069	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1069	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1070	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1070	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1071	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1071	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1072	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1072	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1073	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1073	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1074	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1074	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1075	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1075	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1076	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1076	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1077	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1077	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1078	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1078	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1079	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1079	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1080	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1080	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1081	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1081	ETH0_CTRL (SBR1)	***	BMB (rango)

Elemento	Bloque	Ubicación	Contexto
VB1082	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1082	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1083	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1083	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1084	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1084	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1085	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1085	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1086	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1086	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1087	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1087	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1088	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1088	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1089	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1089	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1090	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1090	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1091	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1091	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1092	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1092	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1093	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1093	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1094	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1094	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1095	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1095	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1096	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1096	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1097	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1097	ETH0_CTRL (SBR1)	***	BMB (rango)

Elemento	Bloque	Ubicación	Contexto
VB1098	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1098	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1099	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1099	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1100	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1100	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1101	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1101	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1102	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1102	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1103	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1103	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1104	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1104	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1105	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1105	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1106	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1106	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1107	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1107	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1108	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1108	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1109	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1109	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1110	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1110	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1111	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1111	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1112	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1112	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1113	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1113	ETH0_CTRL (SBR1)	***	BMB (rango)

Elemento	Bloque	Ubicación	Contexto
VB1114	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1114	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1115	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1115	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1116	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1116	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1117	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1117	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1118	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1118	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1119	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1119	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1120	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1120	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1121	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1121	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1122	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1122	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1123	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1123	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1124	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1124	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1125	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1125	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1126	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1126	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1127	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1127	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1128	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1128	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1129	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1129	ETH0_CTRL (SBR1)	***	BMB (rango)

Elemento	Bloque	Ubicación	Contexto
VB1130	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1130	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1131	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1131	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1132	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1132	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1133	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1133	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1134	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1134	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1135	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1135	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1136	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1136	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1137	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1137	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1138	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1138	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1139	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1139	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1140	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1140	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1141	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1141	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1142	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1142	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1143	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1143	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1144	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1144	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1145	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1145	ETH0_CTRL (SBR1)	***	BMB (rango)

Elemento	Bloque	Ubicación	Contexto
VB1146	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1146	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1147	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1147	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1148	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1148	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1149	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1149	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1150	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1150	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1151	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1151	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1152	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1152	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1153	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1153	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1154	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1154	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1155	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1155	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1156	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1156	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1157	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1157	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1158	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1158	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1159	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1159	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1160	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1160	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1161	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1161	ETH0_CTRL (SBR1)	***	BMB (rango)

Elemento	Bloque	Ubicación	Contexto
VB1162	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1162	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1163	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1163	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1164	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1164	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1165	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1165	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1166	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1166	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1167	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1167	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1168	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1168	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1169	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1169	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1170	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1170	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1171	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1171	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1172	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1172	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1173	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1173	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1174	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1174	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1175	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1175	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1176	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1176	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1177	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1177	ETH0_CTRL (SBR1)	***	BMB (rango)

Elemento	Bloque	Ubicación	Contexto
VB1178	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1178	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1179	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1179	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1180	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1180	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1181	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1181	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1182	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1182	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1183	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1183	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1184	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1184	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1185	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1185	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1186	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1186	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1187	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1187	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1188	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1188	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1189	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1189	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1190	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1190	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1191	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1191	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1192	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1192	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1193	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1193	ETH0_CTRL (SBR1)	***	BMB (rango)

Elemento	Bloque	Ubicación	Contexto
VB1194	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1194	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1195	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1195	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1196	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1196	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1197	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1197	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1198	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1198	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1199	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1199	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1200	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1200	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1201	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1201	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1202	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1202	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1203	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1203	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1204	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1204	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1205	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1205	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1206	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1206	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1207	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1207	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1208	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1208	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1209	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1209	ETH0_CTRL (SBR1)	***	BMB (rango)

Elemento	Bloque	Ubicación	Contexto
VB1210	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1210	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1211	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1211	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1212	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1212	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1213	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1213	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1214	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1214	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1215	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1215	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1216	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1216	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1217	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1217	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1218	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1218	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1219	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1219	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1220	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1220	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1221	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1221	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1222	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1222	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1223	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1223	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1224	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1224	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1225	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1225	ETH0_CTRL (SBR1)	***	BMB (rango)

Elemento	Bloque	Ubicación	Contexto
VB1226	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1226	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1227	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1227	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1228	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1228	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1229	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1229	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1230	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1230	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1231	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1231	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1232	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1232	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1233	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1233	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1234	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1234	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1235	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1235	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1236	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1236	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1237	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1237	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1238	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1238	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1239	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1239	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1240	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1240	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1241	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1241	ETH0_CTRL (SBR1)	***	BMB (rango)

Elemento	Bloque	Ubicación	Contexto
VB1242	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1242	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1243	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1243	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1244	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1244	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1245	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1245	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1246	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1246	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1247	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1247	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1248	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1248	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1249	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1249	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1250	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1250	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1251	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1251	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1252	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1252	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1253	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1253	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1254	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1254	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1255	ETH0_CTRL (SBR1)	***	BMB
VB1255	ETH0_CTRL (SBR1)	***	BMB
VB1256	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1256	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1257	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1257	ETH0_CTRL (SBR1)	***	BMB (rango)

Elemento	Bloque	Ubicación	Contexto
VB1258	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1258	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1259	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1259	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1260	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1260	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1261	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1261	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1262	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1262	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1263	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1263	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1264	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1264	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1265	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1265	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1266	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1266	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1267	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1267	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1268	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1268	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1269	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1269	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1270	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1270	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1271	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1271	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1272	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1272	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1273	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1273	ETH0_CTRL (SBR1)	***	BMB (rango)

Elemento	Bloque	Ubicación	Contexto
VB1274	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1274	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1275	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1275	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1276	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1276	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1277	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1277	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1278	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1278	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1279	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1279	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1280	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1280	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1281	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1281	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1282	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1282	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1283	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1283	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1284	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1284	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1285	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1285	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1286	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1286	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1287	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1287	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1288	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1288	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1289	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1289	ETH0_CTRL (SBR1)	***	BMB (rango)

Elemento	Bloque	Ubicación	Contexto
VB1290	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1290	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1291	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1291	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1292	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1292	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1293	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1293	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1294	ETH0_CTRL (SBR1)	***	BMB (rango)
VB1294	ETH0_CTRL (SBR1)	***	BMB (rango)
V1272.5	ETH0_CTRL (SBR1)	***	U
V1272.5	ETH0_CTRL (SBR1)	***	=
V1272.6	ETH0_CTRL (SBR1)	***	R
V1272.7	ETH0_CTRL (SBR1)	***	U
V1272.7	ETH0_CTRL (SBR1)	***	R (rango)
V1274.5	ETH0_CTRL (SBR1)	***	U
V1274.5	ETH0_CTRL (SBR1)	***	=
V1274.6	ETH0_CTRL (SBR1)	***	R
V1274.7	ETH0_CTRL (SBR1)	***	U
V1274.7	ETH0_CTRL (SBR1)	***	R (rango)
V1276.5	ETH0_CTRL (SBR1)	***	U
V1276.5	ETH0_CTRL (SBR1)	***	=
V1276.6	ETH0_CTRL (SBR1)	***	R
V1276.7	ETH0_CTRL (SBR1)	***	U
V1276.7	ETH0_CTRL (SBR1)	***	R (rango)
V1278.5	ETH0_CTRL (SBR1)	***	U
V1278.5	ETH0_CTRL (SBR1)	***	=
V1278.6	ETH0_CTRL (SBR1)	***	R
V1278.7	ETH0_CTRL (SBR1)	***	U
V1278.7	ETH0_CTRL (SBR1)	***	R (rango)
V1280.5	ETH0_CTRL (SBR1)	***	U
V1280.5	ETH0_CTRL (SBR1)	***	=

Elemento	Bloque	Ubicación	Contexto
V1280.6	ETH0_CTRL (SBR1)	***	R
V1280.7	ETH0_CTRL (SBR1)	***	U
V1280.7	ETH0_CTRL (SBR1)	***	R (rango)
V1282.5	ETH0_CTRL (SBR1)	***	U
V1282.5	ETH0_CTRL (SBR1)	***	=
V1282.6	ETH0_CTRL (SBR1)	***	R
V1282.7	ETH0_CTRL (SBR1)	***	U
V1282.7	ETH0_CTRL (SBR1)	***	R (rango)
V1284.5	ETH0_CTRL (SBR1)	***	U
V1284.5	ETH0_CTRL (SBR1)	***	=
V1284.6	ETH0_CTRL (SBR1)	***	R
V1284.7	ETH0_CTRL (SBR1)	***	U
V1284.7	ETH0_CTRL (SBR1)	***	R (rango)
V1286.5	ETH0_CTRL (SBR1)	***	U
V1286.5	ETH0_CTRL (SBR1)	***	=
V1286.6	ETH0_CTRL (SBR1)	***	R
V1286.7	ETH0_CTRL (SBR1)	***	U
V1286.7	ETH0_CTRL (SBR1)	***	R (rango)
V1289.0	ETH0_CTRL (SBR1)	***	R
V1289.1	ETH0_CTRL (SBR1)	***	R
V1289.1	ETH0_CTRL (SBR1)	***	LDN
V1289.1	ETH0_CTRL (SBR1)	***	S
V1289.2	ETH0_CTRL (SBR1)	***	=
V1289.2	ETH0_CTRL (SBR1)	***	LDN
V1289.3	ETH0_CTRL (SBR1)	***	=
V1289.3	ETH0_CTRL (SBR1)	***	O
V1289.3	ETH0_CTRL (SBR1)	***	UN
V1292.5	ETH0_CTRL (SBR1)	***	U
V1292.5	ETH0_CTRL (SBR1)	***	=
V1292.6	ETH0_CTRL (SBR1)	***	R
V1292.7	ETH0_CTRL (SBR1)	***	U
V1292.7	ETH0_CTRL (SBR1)	***	R (rango)

Elemento	Bloque	Ubicación	Contexto
V1294.5	ETH0_CTRL (SBR1)	***	U
V1294.5	ETH0_CTRL (SBR1)	***	=
V1294.6	ETH0_CTRL (SBR1)	***	R
V1294.7	ETH0_CTRL (SBR1)	***	U
V1294.7	ETH0_CTRL (SBR1)	***	R (rango)
MW4	MAIN (OB1)	Network 1	ETH0_CTRL
MW7	MAIN (OB1)	Network 1	ETH0_CTRL
M0.0	MAIN (OB1)	Network 1	ETH0_CTRL
M1.0	MAIN (OB1)	Network 3	-(S)
M1.0	MAIN (OB1)	Network 4	-| |-
M1.0	MAIN (OB1)	Network 4	-(R)
M1.1	MAIN (OB1)	Network 3	-|/|-
M1.1	MAIN (OB1)	Network 4	-(S)
M1.1	MAIN (OB1)	Network 5	-| |-
M1.1	MAIN (OB1)	Network 5	-(R)
M1.2	MAIN (OB1)	Network 3	-|/|-
M1.2	MAIN (OB1)	Network 3	-| |-
M1.2	MAIN (OB1)	Network 3	-(R)
M1.2	MAIN (OB1)	Network 5	-(S)
M1.2	MAIN (OB1)	Network 6	-| |-
M2.0	MAIN (OB1)	Network 7	-(S)
M2.0	MAIN (OB1)	Network 8	-| |-
M2.0	MAIN (OB1)	Network 8	-(R)
M2.0	MAIN (OB1)	Network 9	-| |-
M2.0	MAIN (OB1)	Network 9	-(R)
M2.1	MAIN (OB1)	Network 7	-|/|-
M2.1	MAIN (OB1)	Network 8	-(S)
M2.1	MAIN (OB1)	Network 9	-| |-
M2.1	MAIN (OB1)	Network 9	-(R)
M2.2	MAIN (OB1)	Network 7	-|/|-
M2.2	MAIN (OB1)	Network 9	-(S)
M2.2	MAIN (OB1)	Network 10	-| |-
M2.2	MAIN (OB1)	Network 11	-| |-
M2.2	MAIN (OB1)	Network 11	-(R)
M2.2	MAIN (OB1)	Network 12	-| |-
M2.2	MAIN (OB1)	Network 12	-(R)
M2.3	MAIN (OB1)	Network 7	-|/|-
M2.3	MAIN (OB1)	Network 11	-(S)
M2.3	MAIN (OB1)	Network 12	-| |-
M2.3	MAIN (OB1)	Network 12	-(R)
M2.4	MAIN (OB1)	Network 7	-|/|-
M2.4	MAIN (OB1)	Network 7	-| |-
M2.4	MAIN (OB1)	Network 7	-(R)
M2.4	MAIN (OB1)	Network 12	-(S)
M2.4	MAIN (OB1)	Network 13	-| |-
M4.0	MAIN (OB1)	Network 4	-| |-
SMD246	ETH0_CTRL (SBR1)	***	MOVD
SMW220	ETH0_CTRL (SBR1)	***	MOVW
SMB8	ETH0_CTRL (SBR1)	***	OB<>
SMB233	ETH0_CTRL (SBR1)	***	MOVB
SMB245	ETH0_CTRL (SBR1)	***	LDB>

Elemento	Bloque	Ubicación	Contexto
SMB245	ETH0_CTRL (SBR1)	***	UB<>
SMB245	ETH0_CTRL (SBR1)	***	UB<>
SM0.0	MAIN (OB1)	Network 1	-| |-
SM0.0	ETH0_CTRL (SBR1)	***	LD
SM0.0	ETH0_CTRL (SBR1)	***	LD
SM0.0	ETH0_CTRL (SBR1)	***	LDN
SM0.1	MAIN (OB1)	Network 2	-| |-
SM0.1	ETH0_CTRL (SBR1)	***	LD
SM0.1	ETH0_CTRL (SBR1)	***	LD
SM0.3	ETH0_CTRL (SBR1)	***	UN
SM9.7	ETH0_CTRL (SBR1)	***	LD
SM222.2	ETH0_CTRL (SBR1)	***	ON
SM222.2	ETH0_CTRL (SBR1)	***	LD
SM222.7	ETH0_CTRL (SBR1)	***	LD
SM225.5	ETH0_CTRL (SBR1)	***	LD
SM225.6	ETH0_CTRL (SBR1)	***	UN
SM225.7	ETH0_CTRL (SBR1)	***	LD
SM226.5	ETH0_CTRL (SBR1)	***	LD
SM226.6	ETH0_CTRL (SBR1)	***	UN
SM226.7	ETH0_CTRL (SBR1)	***	LD
SM227.5	ETH0_CTRL (SBR1)	***	LD
SM227.6	ETH0_CTRL (SBR1)	***	UN
SM227.7	ETH0_CTRL (SBR1)	***	LD
SM228.5	ETH0_CTRL (SBR1)	***	LD
SM228.6	ETH0_CTRL (SBR1)	***	UN
SM228.7	ETH0_CTRL (SBR1)	***	LD
SM229.5	ETH0_CTRL (SBR1)	***	LD
SM229.6	ETH0_CTRL (SBR1)	***	UN
SM229.7	ETH0_CTRL (SBR1)	***	LD
SM230.5	ETH0_CTRL (SBR1)	***	LD
SM230.6	ETH0_CTRL (SBR1)	***	UN
SM230.7	ETH0_CTRL (SBR1)	***	LD
SM231.5	ETH0_CTRL (SBR1)	***	LD

Elemento	Bloque	Ubicación	Contexto
SM231.6	ETH0_CTRL (SBR1)	***	UN
SM231.7	ETH0_CTRL (SBR1)	***	LD
SM232.5	ETH0_CTRL (SBR1)	***	LD
SM232.6	ETH0_CTRL (SBR1)	***	UN
SM232.7	ETH0_CTRL (SBR1)	***	LD
SM233.6	ETH0_CTRL (SBR1)	***	UN
SM233.7	ETH0_CTRL (SBR1)	***	U
SM234.5	ETH0_CTRL (SBR1)	***	LD
SM234.6	ETH0_CTRL (SBR1)	***	UN
SM234.7	ETH0_CTRL (SBR1)	***	LD
SM235.5	ETH0_CTRL (SBR1)	***	LD
SM235.6	ETH0_CTRL (SBR1)	***	UN
SM235.7	ETH0_CTRL (SBR1)	***	LD
SM236.5	ETH0_CTRL (SBR1)	***	LD
SM237.5	ETH0_CTRL (SBR1)	***	LD
#Ch_Ready	ETH0_CTRL (SBR1)	***	MOVW
#Error	ETH0_CTRL (SBR1)	***	MOVW
#Error	ETH0_CTRL (SBR1)	***	MOVW
#Error	ETH0_CTRL (SBR1)	***	MOVW
#CP_Ready	ETH0_CTRL (SBR1)	***	=
L1.0	ETH0_CTRL (SBR1)	***	=
L1.1	ETH0_CTRL (SBR1)	***	=
L1.2	ETH0_CTRL (SBR1)	***	=
L1.3	ETH0_CTRL (SBR1)	***	=
L1.4	ETH0_CTRL (SBR1)	***	=
L1.5	ETH0_CTRL (SBR1)	***	=
L1.6	ETH0_CTRL (SBR1)	***	=
L1.7	ETH0_CTRL (SBR1)	***	=
L2.0	ETH0_CTRL (SBR1)	***	=
L2.1	ETH0_CTRL (SBR1)	***	=
L2.2	ETH0_CTRL (SBR1)	***	=
L2.3	ETH0_CTRL (SBR1)	***	=

Byte	9	8	7	6	5	4	3	2	1	0
VB0	.	.	.	.	.	.	.	.	B	B
VB10	.	.	.	.	.	.	.	.	.	.
VB20	.	.	.	.	.	.	.	.	.	.
VB30	.	.	.	.	.	.	.	.	.	.
VB40	.	.	.	.	.	.	.	.	.	.
VB50	.	.	.	.	.	.	.	.	.	.
VB60	.	.	.	.	.	.	.	.	.	.
VB70	.	.	.	.	.	.	.	.	.	.
VB80	.	.	.	.	.	.	.	.	.	.
VB90	.	.	.	.	.	.	.	.	.	.
VB100	.	.	.	.	.	.	.	.	.	.
VB110	.	.	.	.	.	.	.	.	.	.
VB120	.	.	.	.	.	.	.	.	.	.
VB130	.	.	.	.	.	.	.	.	.	.
VB140	.	.	.	.	.	.	.	.	.	.
VB150	.	.	.	.	.	.	.	.	.	.
VB160	.	.	.	.	.	.	.	.	.	.
VB170	.	.	.	.	.	.	.	.	.	.
VB180	.	.	.	.	.	.	.	.	.	.
VB190	.	.	.	.	.	.	.	.	.	.
VB200	.	.	.	.	.	.	.	.	.	.
VB210	.	.	.	.	.	.	.	.	.	.
VB220	.	.	.	.	.	.	.	.	.	.
VB230	.	.	.	.	.	.	.	.	.	.
VB240	.	.	.	.	.	.	.	.	.	.
VB250	.	.	.	.	.	.	.	.	.	.
VB260	.	.	.	.	.	.	.	.	.	.
VB270	.	.	.	.	.	.	.	.	.	.
VB280	.	.	.	.	.	.	.	.	.	.
VB290	.	.	.	.	.	.	.	.	.	.
VB300	.	.	.	.	.	.	.	.	.	.
VB310	.	.	.	.	.	.	.	.	.	.
VB320	.	.	.	.	.	.	.	.	.	.
VB330	.	.	.	.	.	.	.	.	.	.
VB340	.	.	.	.	.	.	.	.	.	.
VB350	.	.	.	.	.	.	.	.	.	.
VB360	.	.	.	.	.	.	.	.	.	.
VB370	.	.	.	.	.	.	.	.	.	.
VB380	.	.	.	.	.	.	.	.	.	.
VB390	.	.	.	.	.	.	.	.	.	.
VB400	.	.	.	.	.	.	.	.	.	.
VB410	.	.	.	.	.	.	.	.	.	.
VB420	.	.	.	.	.	.	.	.	.	.
VB430	.	.	.	.	.	.	.	.	.	.
VB440	.	.	.	.	.	.	.	.	.	.
VB450	.	.	.	.	.	.	.	.	.	.
VB460	.	.	.	.	.	.	.	.	.	.
VB470	.	.	.	.	.	.	.	.	.	.
VB480	.	.	.	.	.	.	.	.	.	.
VB490	.	.	.	.	.	.	.	.	.	.
VB500	.	.	.	.	.	.	.	.	.	.
VB510	.	.	.	.	.	.	.	.	.	.
VB520	.	.	.	.	.	.	.	.	.	.
VB530	.	.	.	.	.	.	.	.	.	.
VB540	.	.	.	.	.	.	.	.	.	.
VB550	.	.	.	.	.	.	.	.	.	.
VB560	.	.	.	.	.	.	.	.	.	.
VB570	.	.	.	.	.	.	.	.	.	.
VB580	.	.	.	.	.	.	.	.	.	.

Byte	9	8	7	6	5	4	3	2	1	0
VB590	.	.	.	.	.	.	.	.	.	.
VB600	.	.	.	.	.	.	.	.	.	.
VB610	.	.	.	.	.	.	.	.	.	.
VB620	.	.	.	.	.	.	.	.	.	.
VB630	.	.	.	.	.	.	.	.	.	.
VB640	.	.	.	.	.	.	.	.	.	.
VB650	.	.	.	.	.	.	.	.	.	.
VB660	.	.	.	.	.	.	.	.	.	.
VB670	.	.	.	.	.	.	.	.	.	.
VB680	.	.	.	.	.	.	.	.	.	.
VB690	.	.	.	.	.	.	.	.	.	.
VB700	.	.	.	.	.	.	.	.	.	.
VB710	.	.	.	.	.	.	.	.	.	.
VB720	.	.	.	.	.	.	.	.	.	.
VB730	.	.	.	.	.	.	.	.	.	.
VB740	.	.	.	.	.	.	.	.	.	.
VB750	.	.	.	.	.	.	.	.	.	.
VB760	.	.	.	.	.	.	.	.	.	.
VB770	.	.	.	.	.	.	.	.	.	.
VB780	.	.	.	.	.	.	.	.	.	.
VB790	.	.	.	.	.	.	.	.	.	.
VB800	.	.	.	.	.	.	.	.	.	.
VB810	.	.	.	.	.	.	.	.	.	.
VB820	.	.	.	.	.	.	.	.	.	.
VB830	.	.	.	.	.	.	.	.	.	.
VB840	.	.	.	.	.	.	.	.	.	.
VB850	.	.	.	.	.	.	.	.	.	.
VB860	.	.	.	.	.	.	.	.	.	.
VB870	.	.	.	.	.	.	.	.	.	.
VB880	.	.	.	.	.	.	.	.	.	.
VB890	.	.	.	.	.	.	.	.	.	.
VB900	.	.	.	.	.	.	.	.	.	.
VB910	.	.	.	.	.	.	.	.	.	.
VB920	.	.	.	.	.	.	.	.	.	.
VB930	.	.	.	.	.	.	.	.	.	.
VB940	.	.	.	.	.	.	.	.	.	.
VB950	.	.	.	.	.	.	.	.	.	.
VB960	.	.	.	.	.	.	.	.	.	.
VB970	.	.	.	.	.	.	.	.	.	.
VB980	.	.	.	.	.	.	.	.	.	.
VB990	.	.	.	.	.	.	.	.	.	.
VB1000	B	B	B	B	B	B	B	B	B	B
VB1010	B	B	B	B	B	B	B	B	B	B
VB1020	B	B	B	B	B	B	B	B	B	B
VB1030	B	B	B	B	B	B	B	B	B	B
VB1040	B	B	B	B	B	B	B	B	B	B
VB1050	B	B	B	B	B	B	B	B	B	B
VB1060	B	B	B	B	B	B	B	B	B	B
VB1070	B	B	B	B	B	B	B	B	B	B
VB1080	B	B	B	B	B	B	B	B	B	B
VB1090	B	B	B	B	B	B	B	B	B	B
VB1100	B	B	B	B	B	B	B	B	B	B
VB1110	B	B	B	B	B	B	B	B	B	B
VB1120	B	B	B	B	B	B	B	B	B	B
VB1130	B	B	B	B	B	B	B	B	B	B
VB1140	B	B	B	B	B	B	B	B	B	B
VB1150	B	B	B	B	B	B	B	B	B	B
VB1160	B	B	B	B	B	B	B	B	B	B
VB1170	B	B	B	B	B	B	B	B	B	B

Byte	9	8	7	6	5	4	3	2	1	0
VB1180	B	B	B	B	B	B	B	B	B	B
VB1190	B	B	B	B	B	B	B	B	B	B
VB1200	B	B	B	B	B	B	B	B	B	B
VB1210	B	B	B	B	B	B	B	B	B	B
VB1220	B	B	B	B	B	B	B	B	B	B
VB1230	B	B	B	B	B	B	B	B	B	B
VB1240	B	B	B	B	B	B	B	B	B	B
VB1250	B	B	B	B	B	B	B	B	B	B
VB1260	B	B	B	B	B	B	B	B	B	B
VB1270	B	b	B	b	B	b	B	b	B	B
VB1280	b	B	B	b	B	b	B	b	B	b
VB1290	.	.	.	.	.	b	B	b	B	B
MB0	.	W	W	.	W	b	.	b	b	b
SMB0	b	B	.	.	.	.	.	.	.	b
SMB10	.	.	.	.	.	.	.	.	.	.
SMB20	.	.	.	.	.	.	.	.	.	.
SMB30	.	.	.	.	.	.	.	.	.	.
SMB40	.	.	.	.	.	.	.	.	.	.
SMB50	.	.	.	.	.	.	.	.	.	.
SMB60	.	.	.	.	.	.	.	.	.	.
SMB70	.	.	.	.	.	.	.	.	.	.
SMB80	.	.	.	.	.	.	.	.	.	.
SMB90	.	.	.	.	.	.	.	.	.	.
SMB100	.	.	.	.	.	.	.	.	.	.
SMB110	.	.	.	.	.	.	.	.	.	.
SMB120	.	.	.	.	.	.	.	.	.	.
SMB130	.	.	.	.	.	.	.	.	.	.
SMB140	.	.	.	.	.	.	.	.	.	.
SMB150	.	.	.	.	.	.	.	.	.	.
SMB160	.	.	.	.	.	.	.	.	.	.
SMB170	.	.	.	.	.	.	.	.	.	.
SMB180	.	.	.	.	.	.	.	.	.	.
SMB190	.	.	.	.	.	.	.	.	.	.
SMB200	.	.	.	.	.	.	.	.	.	.
SMB210	.	.	.	.	.	.	.	.	.	.
SMB220	b	b	b	b	b	.	.	b	W	W
SMB230	.	.	b	b	b	b	b	b	b	b
SMB240	D	D	D	D	B	.	.	.	.	.

Bit	7	6	5	4	3	2	1	0
E0.0	.	.	.	b	b	.	.	.
A0.0	.	.	b	.	.	.	.	b
A1.0	.	.	.	.	.	.	.	.
A2.0	b	B	B	B	B	B	B	B
M0.0	.	.	.	.	.	.	.	b
M1.0	.	.	.	.	.	b	b	b
M2.0	.	.	.	b	b	b	b	b
M3.0	.	.	.	.	.	.	.	.
M4.0	W	W	W	W	W	W	W	b
M5.0	W	W	W	W	W	W	W	W
M6.0	.	.	.	.	.	.	.	.
M7.0	W	W	W	W	W	W	W	W
M8.0	W	W	W	W	W	W	W	W

En ésta parte se va a dar un enfoque inductivo – analítico – sintético de los conocimientos que se debe tener para poder efectuar lo planteado. La información es recopilada de los fabricantes y/o dueños de los equipos y/o softwares que se emplearán, mostrándose "COMO ES". Lo que se ha hecho es ponerlos en una secuencia que sea viable para un fácil entendimiento (según el Autor de este Trabajo de Graduación).

Conecte LabVIEW a Cualquier Red Industrial y PLC[1]

Modbus TCP y Modbus Serial

Modbus TCP y Modbus Serial son dos de los protocolos de trabajo comúnmente utilizados. LabVIEW 8.0 de NI, introdujo el Modbus TCP nativo y soporta el Modbus Serial en cualquier Puerto Ethernet o serial con dos módulos LabVIEW adicionados: LabVIEW Real-Time y LabVIEW DSC. Cualquiera de estos dos módulos le permite crear un servidor de E/S Modbus TCP o Modbus Serial a través de un asistente de configuración gráfica.

En una versión previa de LabVIEW, o no cuenta con LabVIEW Real-Time o los módulos DSC, puede utilizar la biblioteca Modbus de LabVIEW, el cual proporciona un conjunto de VIs de menor nivel para crear aplicaciones Modbus de Maestro o Esclavo con cualquier ethernet de puertos seriales.

Modbus TCP también es una herramienta útil para utilizar compuertas para una amplia gama de opciones de conectividad.

Tarjetas de Comunicación Insertables

Al utilizar una computadora de escritorio estándar o un chasis PXI, se toma ventaja de la disponibilidad de ranuras PCI o PXI para tarjetas de comunicación de conexión directa. Algunas de las ventajas que tiene el utilizar las tarjetas de conexión directa incluyen:

---

1 http://zone.ni.com/devzone/cda/tut/p/id/6296

- Comunicación directa con las redes de trabajo industrial existentes, proporcionando conectividad con todos los componentes conectados.
- Comunicación determinística con el procesador.
- Funciones de Alto Nivel (API) para un rápido desarrollo de aplicaciones.

Los tipo tarjetas de comunicación de conexión directa para PCI, PXI, y PCMCIA trabajan en las siguientes redes de trabajo industrial: *CAN, DeviceNet, CANopen,* Serial *(RS232, RS422, y RS485), y FOUNDATION Fieldbus*. Adicionalmente, otras redes de trabajo industrial, como Profibus, están soportadas por tarjetas de conexión directa de terceros proveedores y en algunas ocasiones disponibles con dispositivos de comunicación LabVIEW y LabVIEW Real-Time.

Interfases DeviceNet

DeviceNet se utiliza comúnmente en aplicaciones industriales y resulta ser una solución abierta y sencilla que permite la comunicación de hasta 64 dispositivos en un solo bus, reduciendo el costo y complejidad del cableado así como la instalación de dispositivos de automatización, proveyendo también la interoperabilidad de componentes similares de varios proveedores.

DeviceNet está basado en un la capa física de CAN (Controller Area Network), y resulta una solución de bajo costo para la conexión de dispositivos industriales, como lo son los sensores fotoeléctricos, lectores de códigos de barras, E/S, PCs industriales, PLCs, desplegados, e interfaces de hombre-máquina (HMIs) a una red de trabajo. La conectividad directa proporciona una mejora en la comunicación entre dispositivos así como diagnósticos a nivel dispositivo de difícil acceso o disponible a través de interfaces cableadas de E/S.

Las interfases DeviceNet de conexión directa de National Instruments pueden funcionar tanto como maestro (escaneo) o como esclavo. Las interfaces DeviceNet de NI se ofrecen como factores de forma PCI, PXI, y PCMCIA y utilizan el conector industrial combicon de 5 pines para tener acceso a los dispositivos y redes de trabajo DeviceNet.

El tablero PXI DeviceNet es compatible con LabVIEW Real-Time para control y comunicación determinística para redes de trabajo y dispositivos DeviceNet. Todos los tableros DeviceNet se envían junto con el software DNET de NI, proporcionando funciones de alto nivel fáciles de usar para un desarrollo rápido de aplicaciones. Adicionalmente, DNET de NI ofrece dos usos para configuraciones de redes de trabajo y configuración; NI Configurador y Analizador.

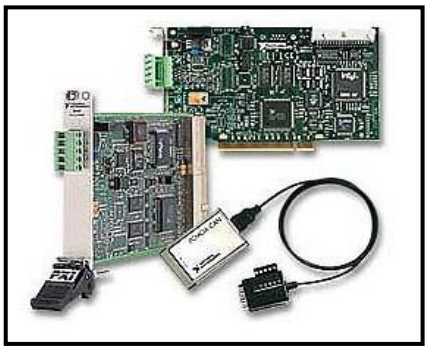


Figura 22.1 : Interfases DeviceNet de Conexión Directa de National Instruments

Configuración DeviceNet de NI: El Configurador es una potente herramienta de configuración con soporte de Hojas de Datos Electrónica (EDS). Cada dispositivo DeviceNet tiene su propio archivo EDS, disponible gracias al proveedor de cada dispositivo. El Configurador puede buscar una red de trabajo DeviceNet para determinar información acerca de dispositivos conectados, cargar automáticamente los archivos EDS relacionados, leer y escribir los parámetros del dispositivo, y cambiar la Identificación del MAC dispositivo.

Analizador DeviceNet de NI: El Analizador realiza un monitoreo de la red de trabajo DeviceNet e interpreta los mensajes CAN capturados de acuerdo al protocolo DeviceNet. Despliega los mensajes junto con sus parámetros. Se puede desplegar cierto tipo de mensajes utilizando filtros potentes y opciones de búsqueda. También se puede obtener estadísticas del mensaje en el Analizador. El Analizador es útil para problemas de disparo y análisis de las redes de trabajo y sistemas DeviceNet.

Interfases CANopen

La interfase CANopen es un protocolo de alto nivel basado en capas físicas CAN y desarrollado como una red de trabajo estandarizada con capacidades de configuración altamente flexibles. Originalmente diseñado para aplicaciones de control de movimiento, el protocolo CANopen es común en muchos segmentos de la industria incluyendo equipos médicos, vehículos no terrestres, transporte público y automatización de construcción.

La biblioteca CANopen de LabVIEW, proporciona funciones LabVIEW de alto nivel y fáciles de usar para crear aplicaciones CANopen. Debido a que las funciones CANopen trabajan encima del software CAN de NI, todos los dispositivos CAN de la Serie 2 de National Instruments para PCI, PXI, y PCMCIA puedan operar con completa funcionalidad como interfaces maestras CANopen.

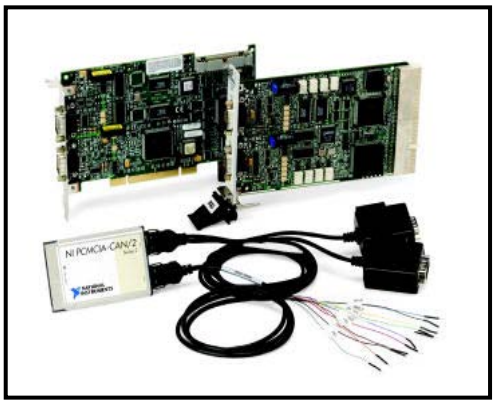


Figura 22.2: Interfases CANopen de National Instruments

La Biblioteca CANopen para LabVIEW de NI proporciona funcionalidad para cubrir el espectro completo de aplicaciones maestras CANopen, incluyendo transmisión y recepción del servicio de objetos de datos (SDOs) y procesamiento de objetos de datos (PDOs), administración de redes de trabajo, monitoreo de nodos y latidos, emergencias y objetos de sincronización. Creándose aplicaciones que se adhieran completamente al CAN en el estándar de Automatización (CiA) DS310 standard.

Trabaja con el Controlador SoftMotion de NI para dispositivos de movimiento CANopen, ayudando a agregar con facilidad cualquier E/S CANopen en

sus redes de trabajo CANopen en movimiento. El Controlador SoftMotion de NI es un motor de movimiento suave que ayuda a crear una interfaz entre el software Motion de NI y dispositivos inteligentes distribuidos. Lográndose programar interfases basadas en dispositivos Accelnet y Xenus a partir de Copley con el Motion API de NI en LabVIEW.

Interfase Serial (RS232, RS422, y RS485)

Este es un protocolo de comunicación estándar para la mayoría de las PCs. La mayoría de las computadoras de escritorio y laptop (no las actuales) incluyen uno o más puertos seriales basados en RS232. El Serial es también un protocolo de comunicación común para instrumentación en muchos dispositivos, y numerosos dispositivos compatibles con GPIB los cuales, vienen ya con un puerto RS232. Además, se puede utilizar la comunicación serial para adquirir datos junto con un dispositivo de muestreo remoto. Mientras que el RS232 es el protocolo serial más común, los protocolos seriales RS422 y RS485 también pueden utilizarse.

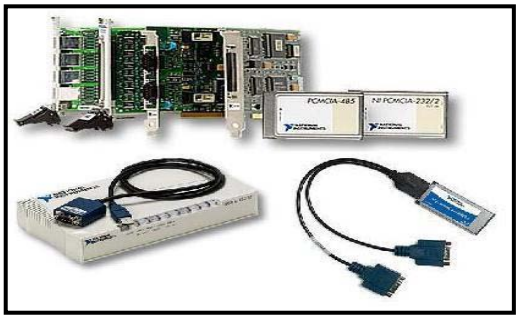


Figura 22.3: *Interfases* ***Seriales de National Instruments***

Interfases Profibus

PROFIBUS fue desarrollada in 1989 y es uno de los buses de campo más populares. Es el estándar para los PLCs de automatización de Siemens para conectarse entre ellos, sensores inteligentes, actuadores y E/S. Existe una variedad de métodos distintos que se pueden utilizar para conectar LabVIEW a redes y dispositivos PROFIBUS. Las compañías en sociedad, como Comsoft, ofrecen interfases PCI y cPCI PROFIBUS, las

cuales se encuentran soportadas en sistemas operativos como Windows y LabVIEW Real-Time para sistemas basados en PXI y PCs.

OPC

OLE for Process Control (OPC) es el nombre original para el estándar desarrollado en 1996 por una fuerza de automatización industrial. El estándar especificó la comunicación de los datos de la planta en tiempo real entre dispositivos de control a partir de diferentes fabricantes. El estándar ahora es mantenido por la Fundación OPC y ha sido renombrada a *Acceso estándar de Datos OPC*. La versión actual de la especificación de Acceso de Datos OPC es OPC Data Access 3.0.

Fue diseñado para cubrir la brecha entre las aplicaciones basadas en Windows y las aplicaciones del hardware de control de procesos. Es un estándar abierto que permite una metodología consistente para tener acceso a datos de campo a partir de dispositivos en la planta. Esta metodología continúa siendo la misma a pesar del tipo y fuente de datos. Tradicionalmente, cuando un paquete de software requería acceso a los datos a partir de un dispositivo, una interface o dispositivo de comunicación, debía escribirse. El propósito de la OPC es definir una interface común escrita una sola vez para usarse posteriormente en paquetes de software SCADA, HMI, o estándar.

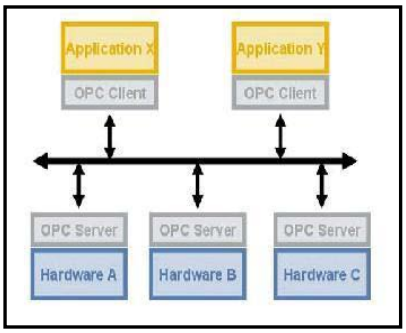


Figura 22.4: OPC está diseñado para mejorar la conectividad del sistema empresarial

Una vez que el servidor OPC está escrito para un dispositivo en particular, puede ser utilizado nuevamente por una aplicación capaz de actuar como un cliente OPC. Los servidores OPC utilizan la tecnología OLE de Microsoft (también conocida como el Componente del Modelo de Objeto, o COM) para comunicarse con clientes.

LabVIEW como un Servidor OPC: Con LabVIEW 8 y versiones más actuales, usted puede publicar cualquier tipo de datos que tenga en un servidor nativo OPC utilizando una variable compartida. Usando este método, se publica cualquier dato que tenga en LabVIEW, en cualquier aplicación que pueda actuar como Cliente OPC.

Agregando la Funcionalidad de Cliente OPC a LabVIEW: El Módulo de LabVIEW DSC extiende el ambiente de desarrollo gráfico LabVIEW con funcionalidad adicional para un rápido desarrollo de aplicaciones de medición distribuida, control y monitoreo de canales. El Módulo DSC proporciona herramientas al ambiente LabVIEW, haciéndole fácil graficar los históricos de las tendencias en tiempo real, mejorar la seguridad en pantallas principales, tener acceso a datos automáticamente, así como agregar un sistema de alarma, escala y seguridad a la variable compartida.

Adicionalmente, una de las mayores características que el modulo LabVIEW DSC proporciona, es la habilidad para LabVIEW de funcionar como un cliente OPC abierto, proporcionando una conectividad fácil con cualquier servidor implementando los fundamentos OPC a partir de la interface del servidor OPC. El Módulo DSC encuentra instalado a todos los servidores OPC y lee cualquier información disponible acerca de las capacidades del servidor, y selecciona directamente desde el servidor.

Publicando Datos a partir de los Dispositivos de Adquisición de Datos de NI (DAQ) con un Servidor OPC: Mientras que el Módulo LabVIEW DSC es requerido para agregar funcionalidad de cliente OPC a LabVIEW, el software NI-DAQmx proporciona la habilidad para publicar datos a partir de cualquier dispositivo DAQ de National Instruments. Gracias a un servidor nativo OPC, simplificando la creación de aplicaciones de adquisición y control de datos distribuidos. Todos los dispositivos NI-DAQmx disponibles para uso con el motor de variable compartida LabVIEW 8 a través de versiones NI-DAQmx 8 y más actuales.

Convertidores de Terceros

Cualquier computadora o Controlador de Automatización Programable (PAC) de NI con Ethernet o puerto serial puede comunicarse con PLCs, sensores inteligentes, y actuadores en una gran variedad de redes de trabajo industrial por medio del uso de convertidores de terceros proveedores. Con las características nativas del Modbus de LabVIEW 8 ya sea con los módulos DSC o Real-Time o con la biblioteca Modbus para

LabVIEW de National Instruments, usted puede utilizar cualquier Ethernet o puerto serial como un Modbus TCP o Modbus serial maestro o esclavo.

Connect LabVIEW to Any PLC Using OPC[2]

The LabVIEW Datalogging and Supervisory Control (DSC) Module is used in this tutorial. This module includes tools for *logging data* to a networked historical database, real-time and historical trending, managing alarms and events, networking LabVIEW Real-Time targets and OPC devices into one complete system, and adding security to user interfaces. With these features, LabVIEW becomes a HMI/SCADA package for industrial control applications.

Requirements

- Windows XP/2000
- NI LabVIEW Full Development System and LabVIEW DSC
- NI OPC Servers

View Existing PLC Tags with NI OPC Servers

1. Launch NI OPC Servers by selecting **Start » Programs » National Instruments»NI OPC Servers»NI OPC Servers**. With NI OPC Servers, you can create, configure, and view tags that are associated with your PLCs.
2. NI OPC Servers should launch with a PLC simulation project already loaded. This project simulates PLCs that have already been created and configured in NI OPC Servers.

Note: If this simulation project is not already loaded, in NI OPC Servers, select **File»Open…** and browse to C:\Program Files\National Instruments\Shared\NI OPC Servers\Projects\simdemo.opf. The project should look like Figure 22.5.

2 http://zone.ni.com/devzone/cda/tut/p/id/7450

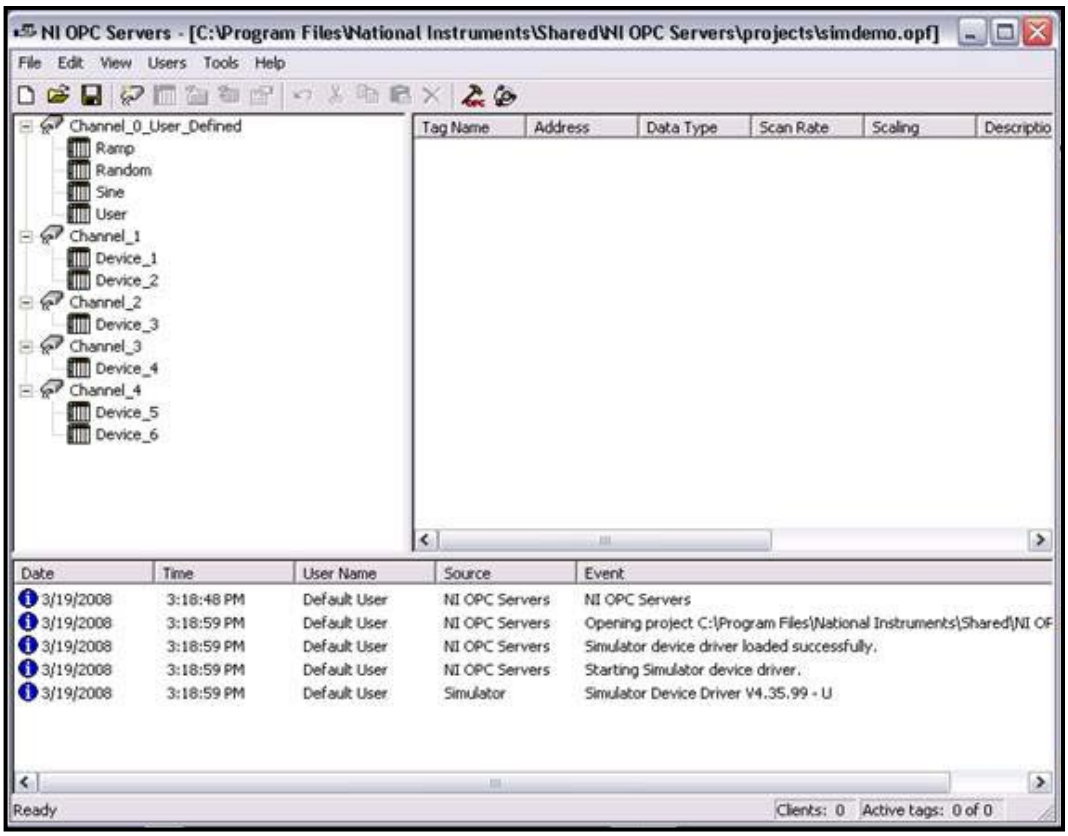


Figure 22.5: NI OPC Servers Displaying Simulated PLCs

3. View the Sine tags by expanding **Channel_0_User_Defined** and selecting **Sine**. The tags populate in the right-most window. These tags, which are bound to registers on the PLCs, are read by LabVIEW.
4. View the data from the PLCs' OPC tags:

    4.1 In NI OPC Servers, select **Tools»Launch OPC Quick Client**. This launches the OPC Quick Client, which you use to view the OPC tag data.

    4.2 Expand the **National Instruments.NIOPCServers** folder and select **Channel_0_User_Defined.Sine**. This selects the device to monitor.

    4.3 Notice that all the Sine tags populate in the right-most window and are updating with simulated sine data as shown in Figure 22.6.

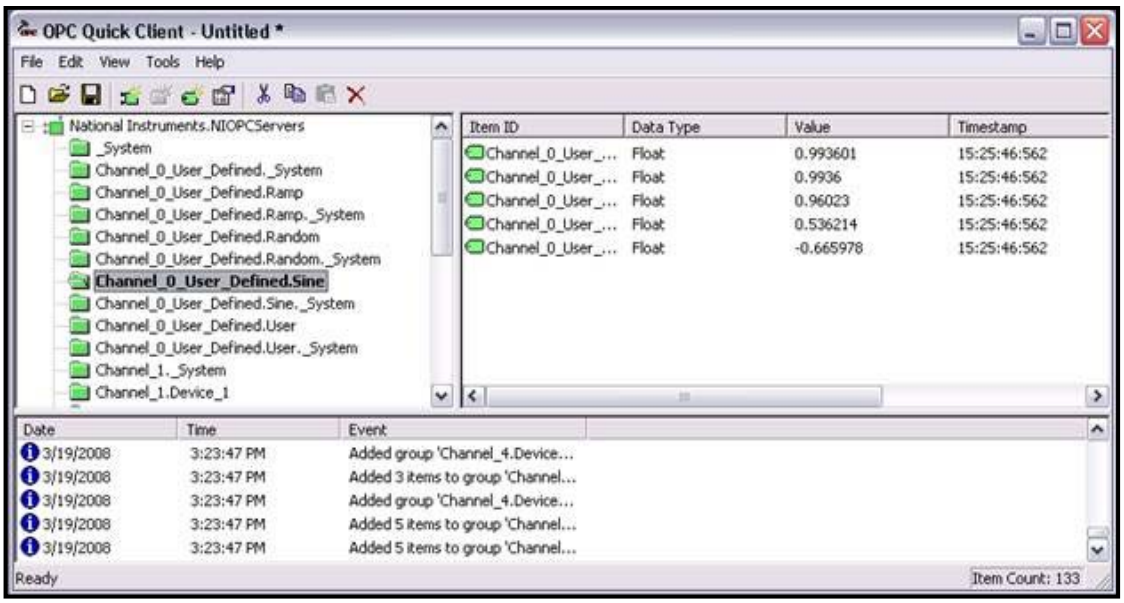


Figure 22.6: NI OPC Quick Client Displaying Simulated Sine OPC Tags

Note: For a list of supported devices/drivers for NI OPC, visit ni.com/opc.

Connect LabVIEW to OPC Tags by Creating an I/O Server

In this section, create a LabVIEW interface to the OPC tags called an I/O Server. The I/O Server automatically updates LabVIEW with the current tag values at a rate you specify.

1. In the **Getting Started** window of LabVIEW, click **File»New Project**. This opens a new LabVIEW Project.
2. If the **Context Help** window is not visible, press **Ctrl+H** to display the window. Keep this window open for helpful information about items under your cursor.
3. In the LabVIEW Project window, right-click **My Computer** and select **New»I/O Server**, as shown in Figure 22.7.

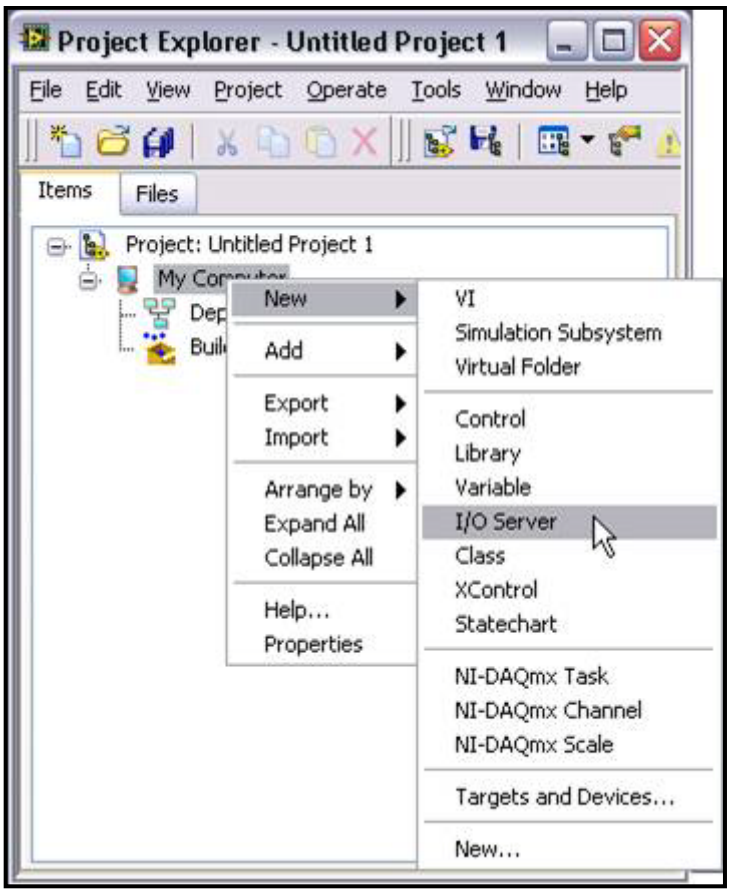


Figure 22.7: Creating a New I/O Server through the LabVIEW Project

4. Select **OPC Client** in the Create New I/O Server Window and click **Continue.**
5. Choose **National Instruments.NIOPCServers** from the Registered OPC servers field and set **Update rate (ms)** to 100. This creates a connection from LabVIEW to the OPC tags, which updates every 100 ms. (Figure 22.8)

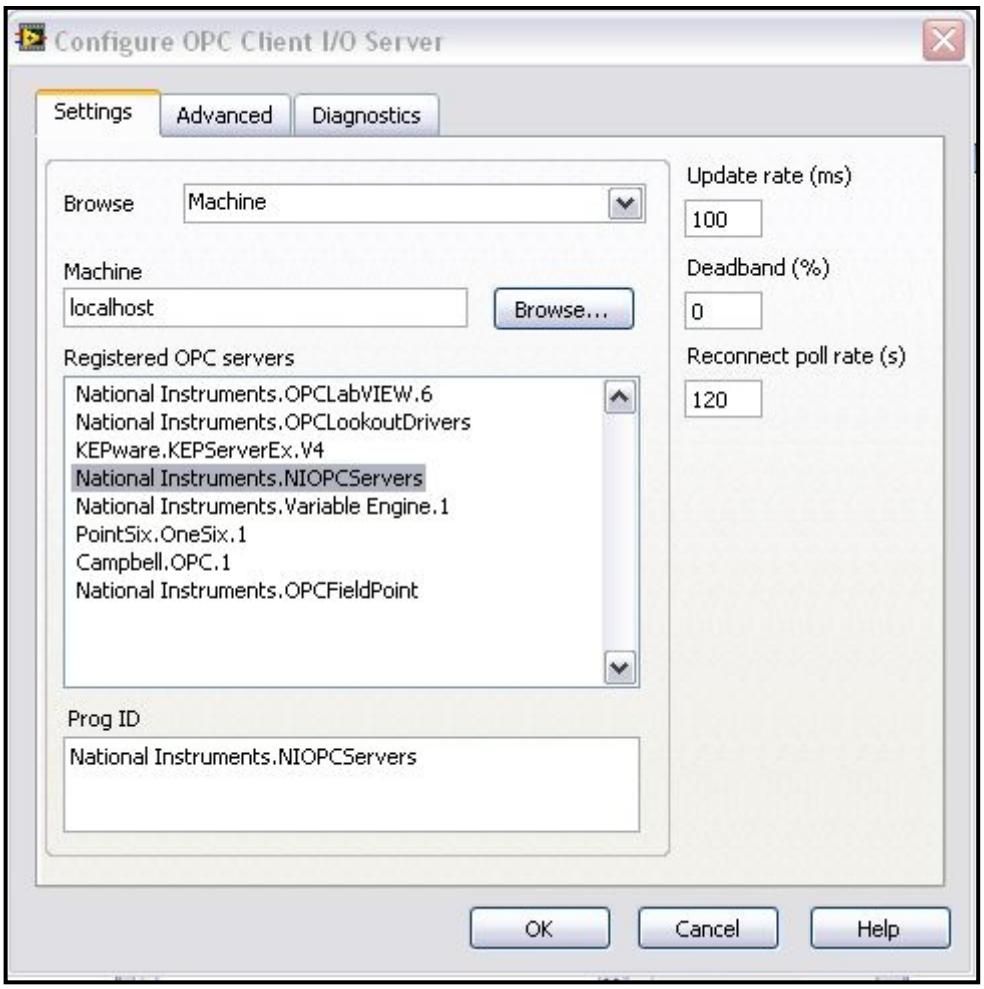


Figure 22.8: Configuring the OPC Client I/O Server

6. Select **OK**. A library is automatically created in your project explorer window to manage the I/O Server.
7. Save the project as OPCDemoProject and the library as OPCDemoLibrary by selecting **File»Save All** from the project explorer window.

Create Shared Variables that Connect to the OPC Tags through the I/O Server

In this section, create shared variables, which are bound to the OPC tags, giving you native access in LabVIEW to PLC data. With the shared variable, you can share data across LabVIEW applications on a single computer or across the network.

1. Create new shared variables that are bound to the PLCs' OPC tags.

1.1 In the LabVIEW Project window, right-click My Computer and select **New»Library**. This creates a new library for the shared variables, which are used to connect to the PLCs' OPC tags.

1.2 Right-click the newly created library and select **Create Bound Variables…**

1.3 In the Create Bound Variables window, select the OPC tags to bind the shared variables to by browsing down to the simulated sine data from the OPC server as shown in Figure 22.9.

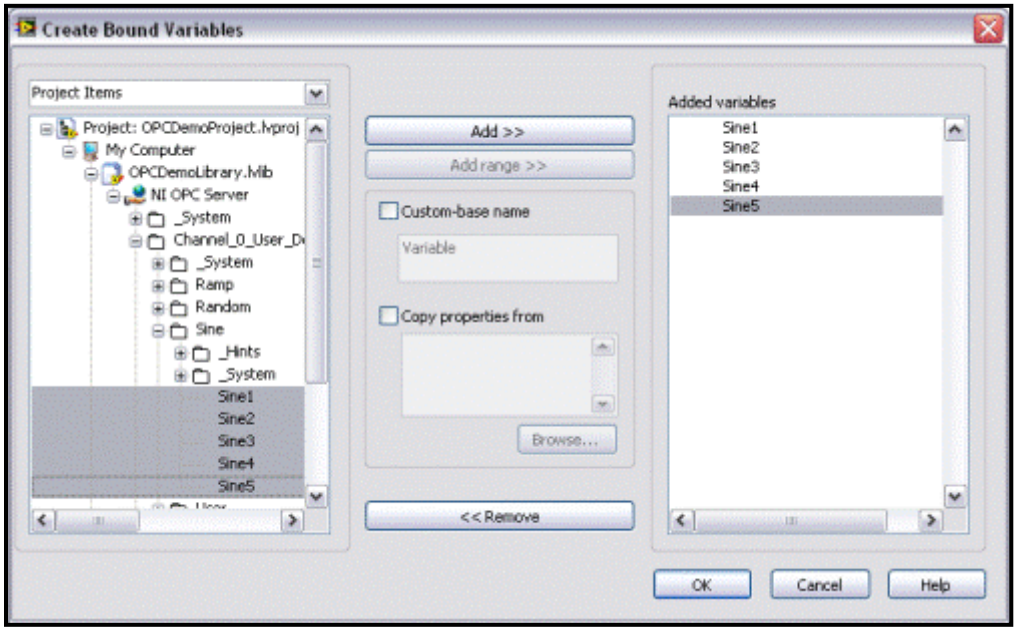


Figure 22.9: Select OPC Tags to Bind to Shared Variables

1.4 Select all the sine items and click **Add** and **OK.** This creates shared variables that are bound to the PLCs' OPC tags and loads them into the Multiple Variable Editor.

1.5 In the Multiple Variable Editor, select **Done**. This adds the new shared variables to the library that was created earlier.

Note: The LabVIEW DSC Module enhances shared variables by adding the ability to log data, alarms, and events directly to a database without ever writing a LabVIEW application.

2. Save the new library as OPCItems.lvlib in the project explorer window by right-clicking the library and selecting **Save As**.

3. Deploy the shared variables by right-clicking the OPCItems library and selecting **Deploy**. This publishes the shared variables, making them available on the network to other computers, OPC clients, and the LabVIEW Real-Time PAC.

You now have access to PLC data natively in LabVIEW through the shared variables.

Viewing Shared Variables with Distributed System Manager

1. From the Project Explorer window, select **Tools»Distributed System Manager**. This opens a window that you can use to manage your shared variables in various ways (view, deploy, undeploy, etc.).
2. In the Tree pane of the Variable Manager, expand the localhost item under the My Systems category. Right-click the **OPCItems** library, and select **Watch List** to display the shared variables, which are bound to the PLCs' OPC tags.
3. The shared variables will be updating with the simulated sine data.

Note: The Distributed System Manager is new in LabVIEW 8.6. Previous versions of LabVIEW did this by going to **Tools»Shared Variable»Variable Manager** and dragging the shared variables into the **Watched Variables:** window.

Using OPC Tag Data in LabVIEW

1. From the project explorer, right-click My Computer and select **New»VI**. This creates a new virtual instrument or VI. A VI is used to create a user interface and executable graphical code.
2. By default, you see the Front Panel, which is the user interface of the VI. LabVIEW has many built-in UI components, such as graphs, charts, dials, and so on, that you can use to build a powerful, intuitive UI. Select **View»Controls Palette** or right-click anywhere on the Front Panel to bring up the Controls palette. Mouse over the various categories to explore the UI components in LabVIEW.
3. Select a waveform chart from the Controls palette by selecting **Express»Graph Indicators»Chart**, and place it on the Front Panel, as shown in Figure 22.10.

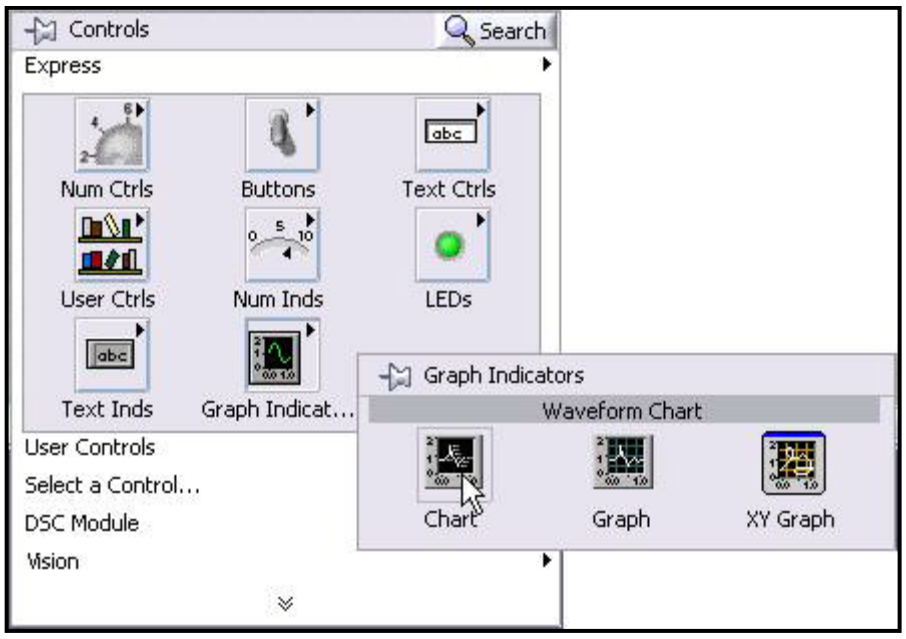


Figure 22.10: Select a Waveform Chart from the Controls Palette

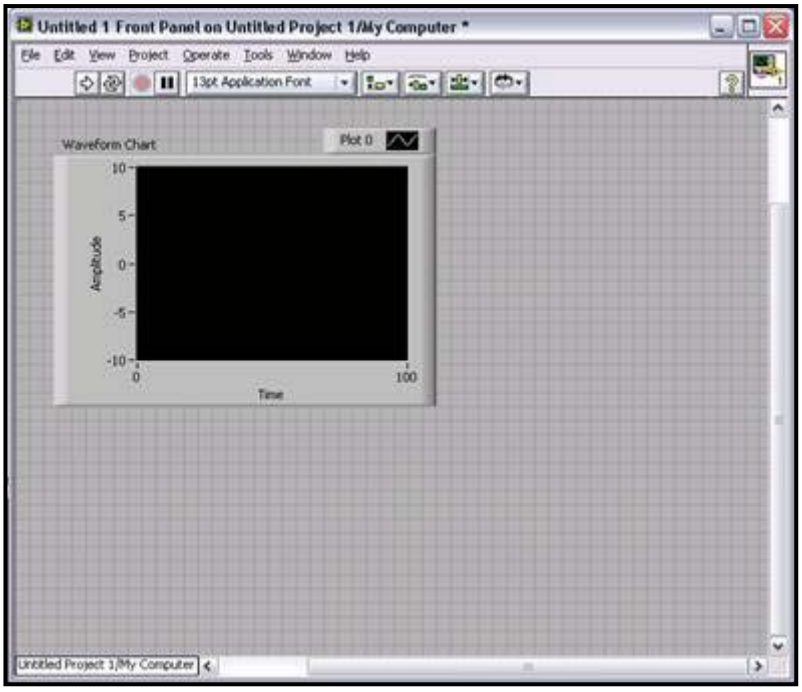


Figure 22.11: Waveform Chart Placed on the Front Panel

4. In the VI, select **Window»Show Block Diagram** or press **Ctrl+E** to show the Block Diagram. The Block Diagram is where you build the behavior of your application. Notice the icon on the Block Diagram, which represents the chart on the Front Panel. By passing data into this terminal, you can display it in the chart on the Front Panel.

5. In the project explorer, expand the **OPCItems** library and select the **Sine1** shared variable.
6. Drag and drop the **Sine1** shared variable from the project explorer to the Block Diagram of the VI. The shared variable acts as a source of data to other terminals on the Block Diagram.
7. Select **View»Tools Palette** or press Shift+right-click to show the Tools palette, which contains various tools for building the Block Diagram. By default you use the **Automatic Tool Selection** tool, which selects the appropriate tool based on the location of the cursor.
8. Select the **Connect Wire** tool as shown in Figure 22.12. This tool is used to wire terminals together on the Block Diagram.

Figure 22.12: Select the Connect Wire Tool

9. Use the Connect Wire tool to wire the **Sine1** shared variable to the **waveform chart** by clicking on the Sine1 shared variable and then on the waveform chart, as shown in Figure 22.13.

Figure 22.13: Connecting Block Diagram Items

Now data flows from the shared variable to the waveform chart when the VI is running.

10. Select the **Automatic Tool Selection** tool from the Tools palette.

Figure 22.14: Automatic Tool Selection from the Tools Palette

11. Open the Functions palette by selecting **View»Functions Palette** or right-clicking anywhere on the Block Diagram. The Functions palette contains hundreds of analysis functions, control functions, and structures for graphical programming.

12. Select a while loop from the Functions palette by navigating to **Express»Execution Control»While Loop**. Once you select the while loop, your cursor appears as shown in Figure 22.15. This allows you to wrap a while loop around a section of code.

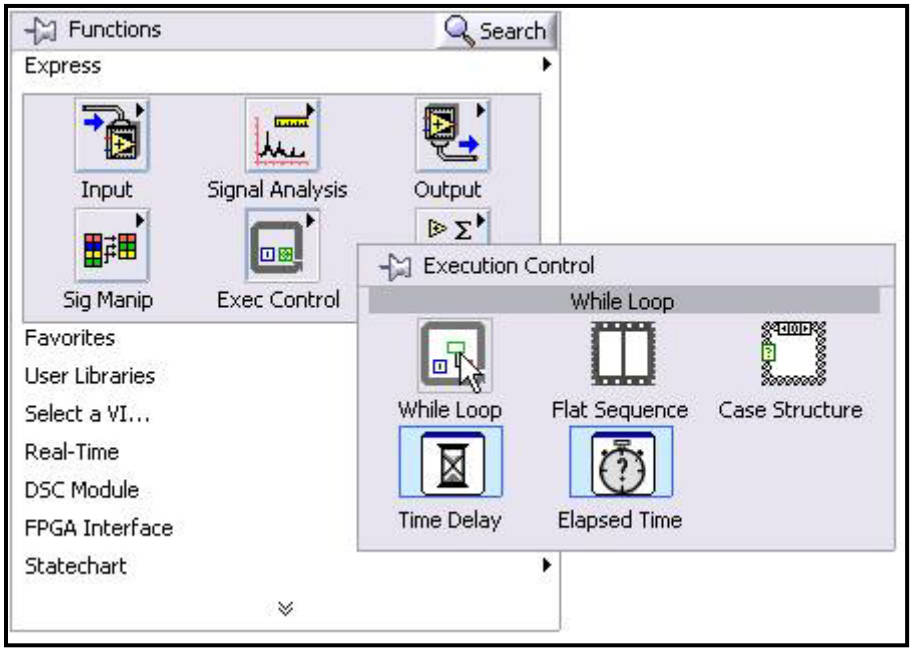
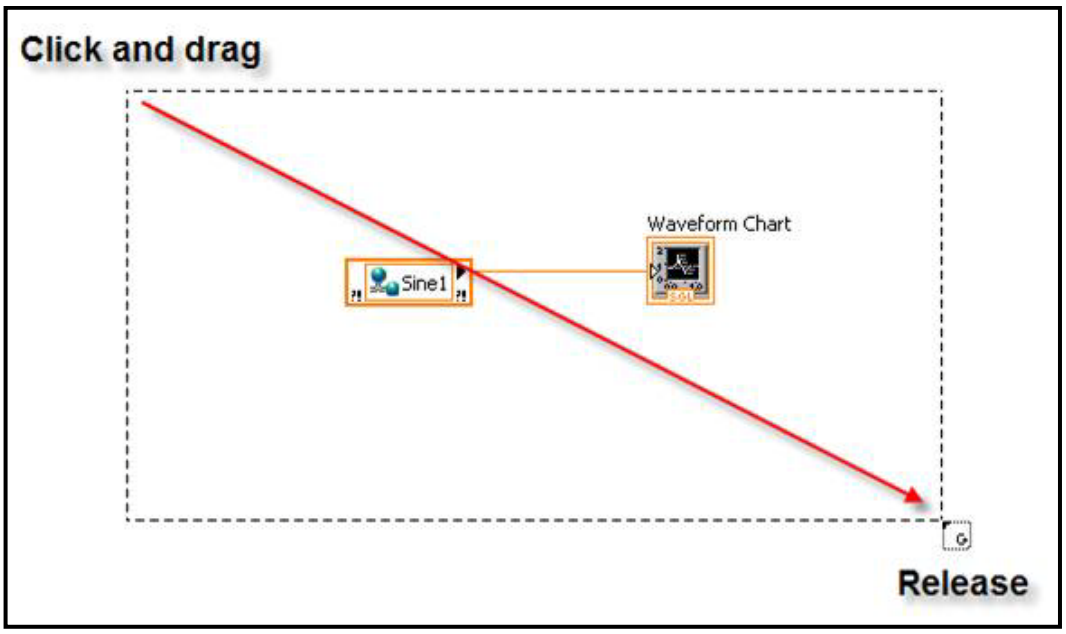


Figure 22.15: Selecting a While Loop

13. Using the while loop cursor, place a while loop around the shared variable and waveform chart by clicking and dragging the cursor.



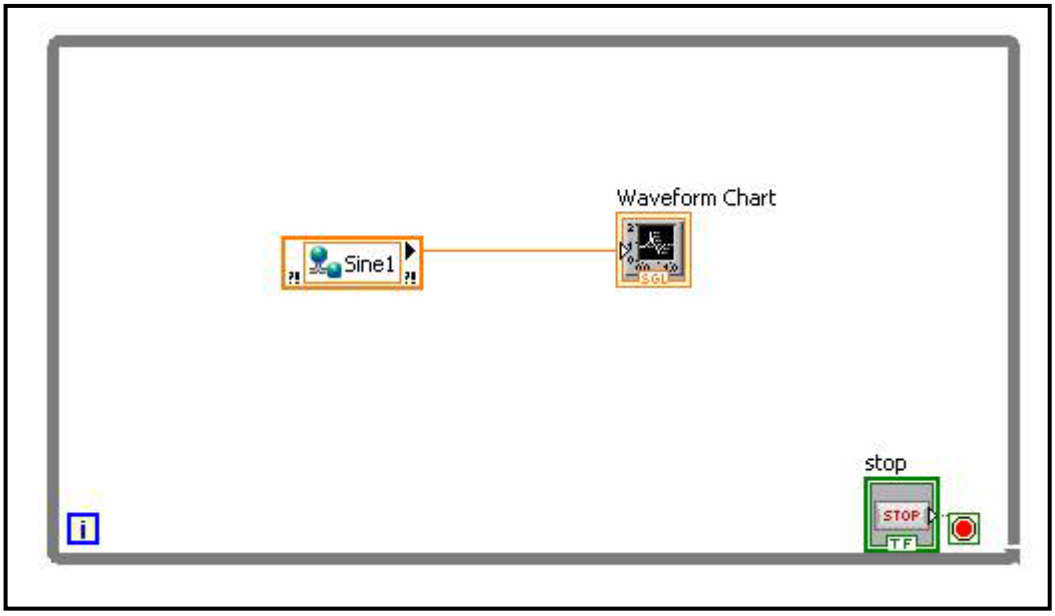


Figure 22.16: Placing a While Loop around the Shared Variable and Waveform Chart

The while loop causes the code within it to execute continuously until stopped by the user or additional logic in the VI.

14. A timed loop, which is an advanced while loop, contains additional configuration options for timing and execution control. Convert the while loop into a timed loop by right-clicking the **while loop** and selecting **Replace with Timed Loop**.

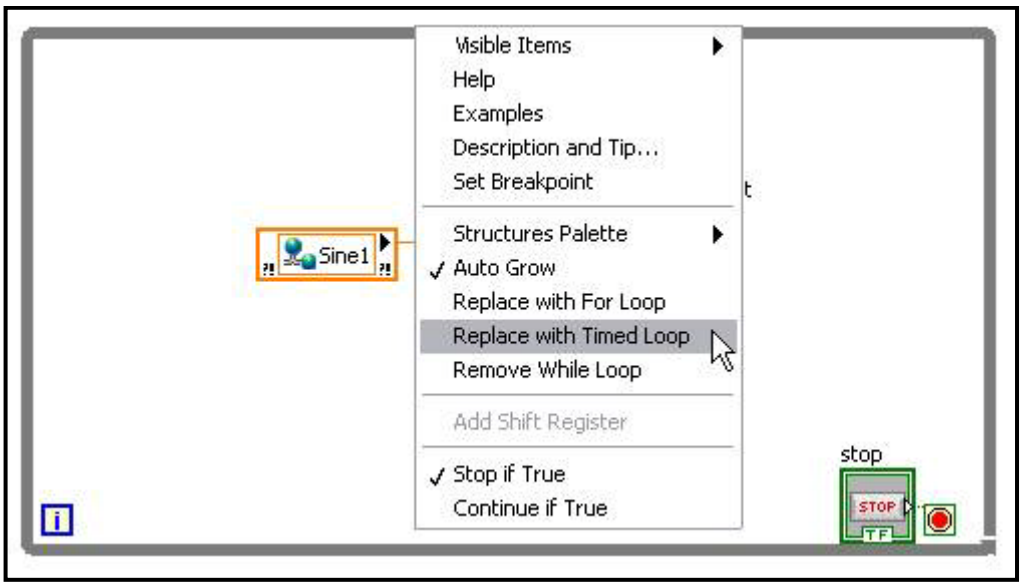


Figure 22.17: Converting the While Loop into a Timed Loop

15. To configure the timed loop, double-click the timed loop input node as seen in Figure 22.18.

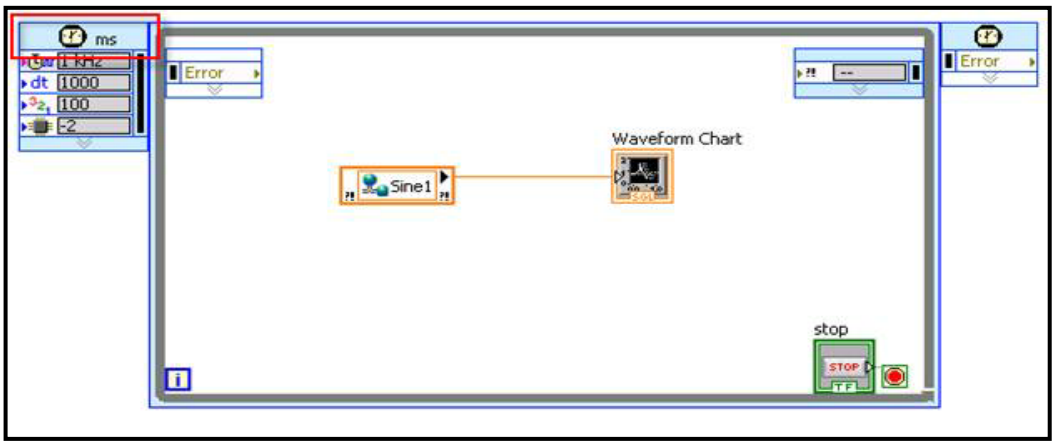


Figure 22.18: Configuring the Timed Loop

16. In the Loop Timing Attributes field, set **Period** to 100 ms and click **OK**. This configures the timed loop to execute the contained code every 100 ms.

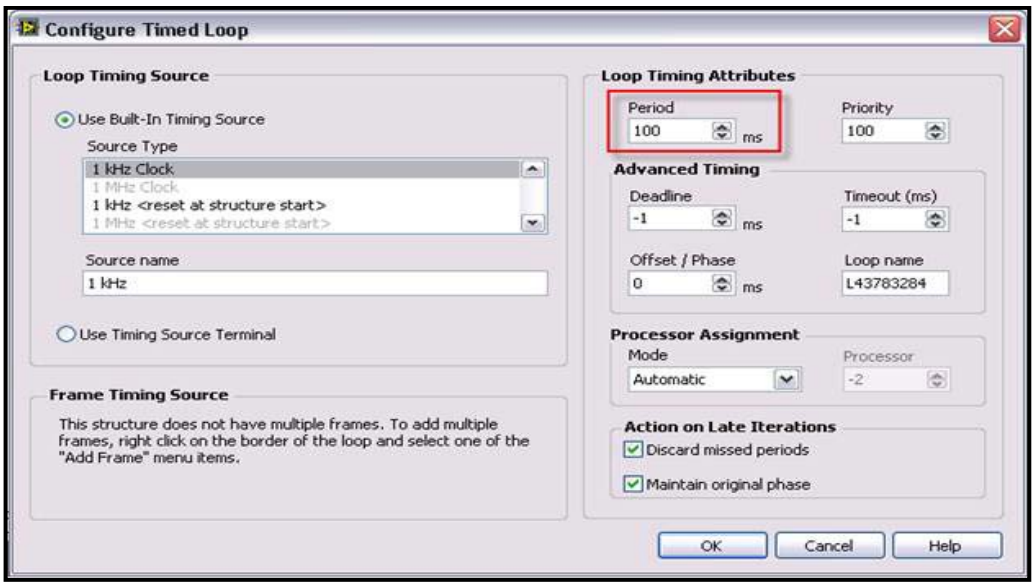


Figure 22.19: Configuring the Timed Loop to Execute Contained Code Every 100 ms

17. Return to the Front Panel by selecting **Window»Show Front Panel** or pressing **Ctrl+E.**

18. Click the **Run** button on the toolbar to execute the VI.

19. Click **Close** on the **Deploy…** window once the deployment completes. When the application begins executing, you see the Sine1 sine wave displayed on the waveform chart.

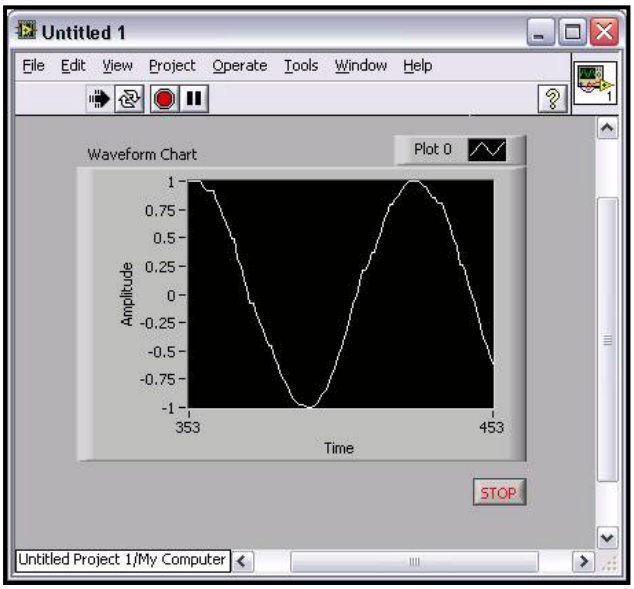


Figure 22.20: Completed Front Panel – Displaying PLC Data on a Waveform Chart.

Congratulations! You successfully accessed PLC data in your LabVIEW application, so you can incorporate powerful analysis and control functions in your solution.

Using the LabVIEW Shared Variable[3]

Creating Shared Variables

Using the shared variable, you can share data between loops on a single diagram or between VIs across the network. In contrast to many existing data sharing methods in LabVIEW, such as UDT/TCP, LabVIEW queues, and Real-Time FIFOs, you typically configure the shared variable at edit time using property dialogs, and you do not need to include configuration code in your application.

You can create three types of shared variables: single-process, network-published, and time-triggered. This paper discusses the single-process and the network-published shared variables in detail. Refer to the document Using Time-Triggered Networks to Communicate Deterministically Over Ethernet with the LabVIEW 8 Real-Time Module for more information about the time-triggered shared variable. To create a shared variable, right-click on a computing device such as "My Computer" or a real-time target in the project tree, and select **New»Variable** to display the shared variable properties dialog. Specify the configuration for the new variable in the dialog presented.

You must have a project open to create a shared variable. To add a shared variable to a project, right-click a target, a project library, or a folder within a project library in the **Project Explorer** window and select **New»Variable** from the shortcut menu to display the **Shared Variable Properties** dialog box. Select among the shared variable configuration options and click the **OK** button.

If you right-click a target or a folder that is not inside a project library and select **New»Variable** from the shortcut menu to create a shared variable, LabVIEW creates a new project library and places the shared variable inside. Refer to the *Shared Variable Lifetime* section for more information about variables and libraries.

3 http://zone.ni.com/devzone/cda/tut/p/id/4679

Figure 22.21 shows the **Shared Variable Properties** dialog box for a single-process shared variable. The LabVIEW Real-Time Module and the LabVIEW Datalogging and Supervisory Control (DSC) Module provide additional features and configurable properties to shared variables. Although in this example both the LabVIEW Real-Time Module and the LabVIEW DSC Module are installed, you can use the features the LabVIEW DSC Module adds only for network-published shared variables.

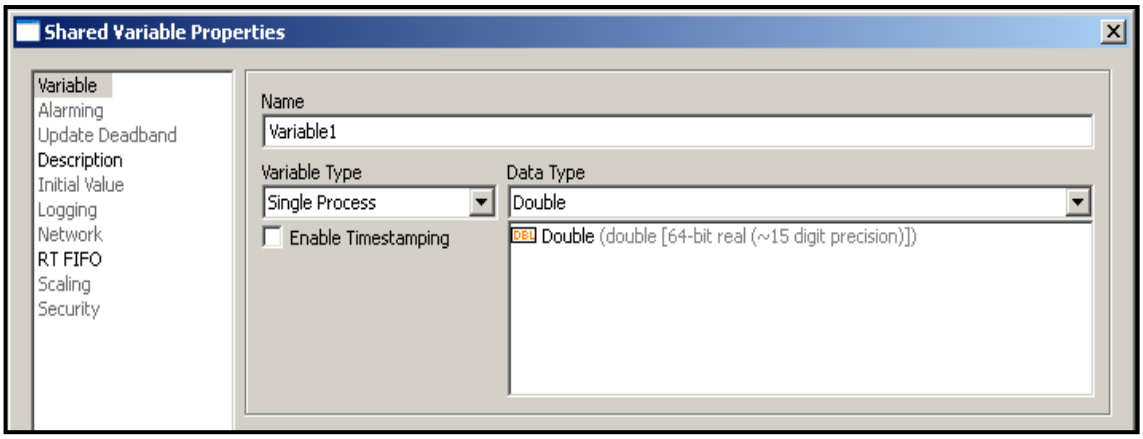


Figure 22.21: Single-Process Shared Variable Properties

Data Type

You can select from a large number of standard data types for a new shared variable. In addition to these standard data types, you can specify a custom data type by selecting **Custom** from the **Data Type** pull-down list and navigating to a custom control. However, some features such as scaling and real-time FIFOs will not work with some custom datatypes. Also, if you have the LabVIEW DSC Module installed, alarming is limited to bad status notifications when using custom datatypes.

After you configure the shared variable properties and click the **OK** button, the shared variable appears in your **Project Explorer** window under the library or target you selected, as shown in Figure 22.22.

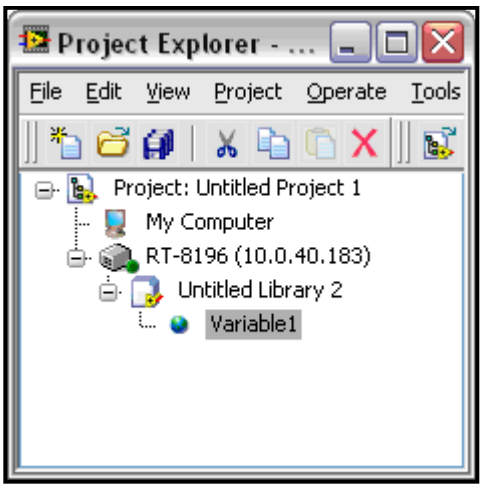


Figure 22.22: Shared Variable in the Project

The target to which the shared variable belongs is the target from which LabVIEW deploys and hosts the shared variable. Refer to the *Deployment and Hosting* section for more information about deploying and hosting shared variables.

Variable References

After you add a shared variable to a LabVIEW project, you can drag the shared variable to the block diagram of a VI to read or write the shared variable, as shown in Figure 22.23. The read and write nodes on the diagram are called Shared Variable nodes.

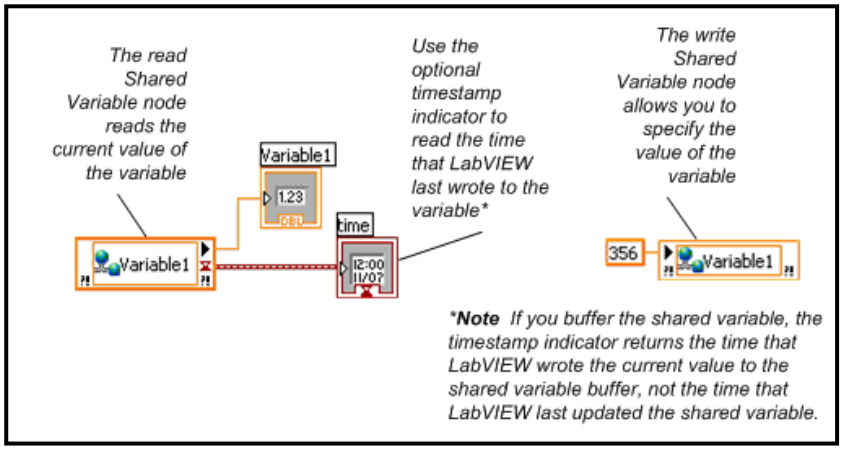


Figure 22.23: Reading and Writing to a Shared Variable Using a Shared Variable Node

You can set a Shared Variable node as absolute or target-relative depending on how you want the node to connect to the variable. An absolute Shared Variable node connects to the shared variable on the target on which you created the variable. A target-relative Shared Variable node connects to the shared variable on the target on which you run the VI that contains the node.

If you move a VI that contains a target-relative Shared Variable node to a new target, you also must move the shared variable to the new target. Use target-relative Shared Variable nodes when you expect to move VIs and variables to other targets.

Shared Variable nodes are absolute by default. Right-click a node and select **Change to Target Relative** or **Change to Absolute** to change how the Shared Variable node connects to the shared variable.

You can right-click a shared variable in the **Project Explorer** window and edit the shared variable properties at any time. The LabVIEW project propagates the new settings to all shared variable references in memory. When you save the variable library, these changes are also applied to the variable definition stored on disk.

Single-Process Shared Variable

Use single-process variables to transfer data between two different locations on the same VI that cannot be connected by wires, such as parallel loops on the same VI, or two different VIs within the same application instance. The underlying implementation of the single-process shared variable is similar to that of the LabVIEW global variable. The main advantage of single-process shared variables over traditional global variables is the ability to convert a single-process shared variable into a network-published shared variable that any node on a network can access.

Single-Process Shared Variables and LabVIEW Real-Time

In order to maintain determinism, a real-time application requires the use of a nonblocking, deterministic mechanism to transfer data from deterministic sections of

the code, such as higher-priority timed loops and time-critical priority VIs, to nondeterministic sections of the code. When you install the LabVIEW Real-Time Module, you can configure a shared variable to use real-time FIFOs by enabling the real-time FIFO feature from the **Shared Variable Properties** dialog box. National Instruments recommends using real-time FIFOs to transfer data between a time-critical and a lower-priority loop. You can avoid using the low-level real-time FIFO VIs by enabling the real-time FIFO on a single-process shared variable.

In LabVIEW versions prior to 8.6, LabVIEW creates a real-time FIFO the first time a Shared Variable node attempts to write to or read from a shared variable. This behavior results in a slightly longer execution time for the first use of each shared variable compared with subsequent uses. If an application requires extremely precise timing, include either initial "warm-up" iterations in the time-critical loop to account for this fluctuation in access times or read the variable at least once outside the time-critical loop. In LabVIEW 8.6 and later, LabVIEW creates a real-time FIFO when the VI is reserved for execution (when the top-level VI in the application starts running in most cases), so no special consideration of the first execution of the Shared Variable node is needed.

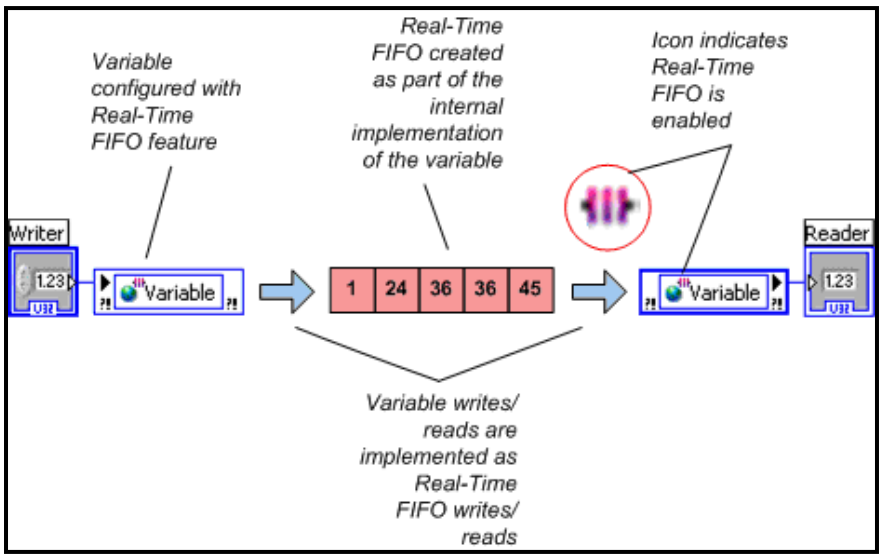


Figure 22.24: Real-Time FIFO-Enabled Shared Variables

LabVIEW creates a single, real-time FIFO for each single-process shared variable even if the shared variable has multiple writers or readers. To ensure data integrity, multiple

writers block each other as do multiple readers. However, a reader does not block a writer, and a writer does not block a reader. National Instruments recommends avoiding multiple writers or multiple readers of single-process shared variables used in time-critical loops.

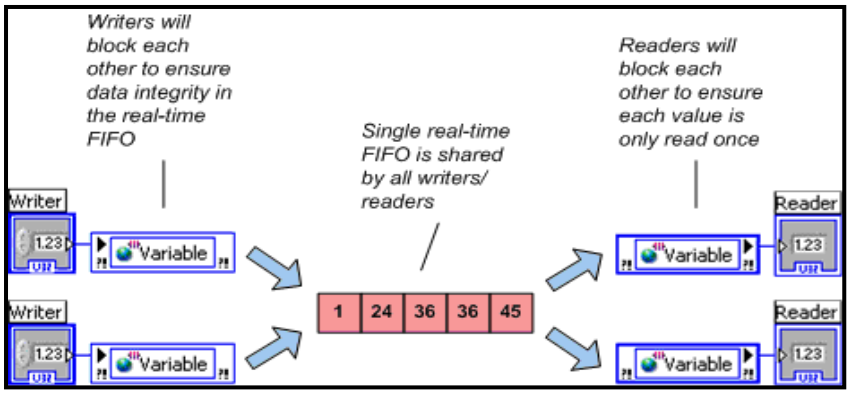


Figure 22.25: Multiple Writers and Readers Sharing a Single FIFO

By enabling the real-time FIFO, you can select between two slightly different types of FIFO-enabled variables: the single-element and the multielement buffer. One distinction between these two types of buffers is that the single-element FIFO does not report warnings on overflow or underflow conditions. A second distinction is the value that LabVIEW returns when multiple readers read an empty buffer. Multiple readers of the single-element FIFO receive the same value, and the single-element FIFO returns the same value until a writer writes to that variable again. Multiple readers of an empty multielement FIFO each get the last value that they read from the buffer or the default value for the data type of the variable if they have not read from the variable before. This behavior is shown below.

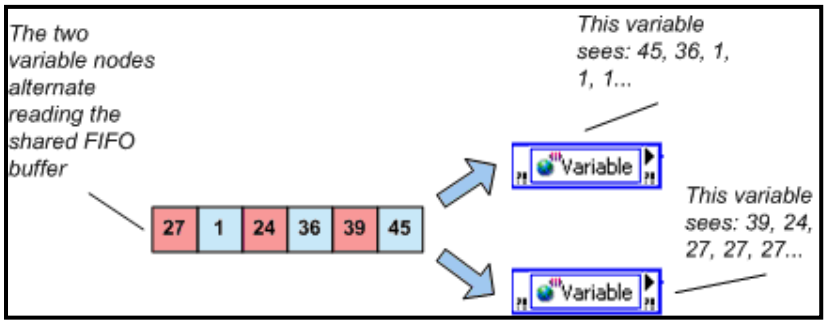


Figure 22.26: Last Read Behavior and the Multielement Real-Time FIFO Shared Variable

If an application requires that each reader get every data point written to a multielement FIFO shared variable, use a separate shared variable for each reader.

Network-Published Shared Variable

Using the network-published shared variable, you can write to and read from shared variables across an Ethernet network. The networking implementation is handled enterly by the network-publiseh variable.

In addition to making your data available over the network, the network-published shared variable adds many features not available with the single-process shared variable. To provide this additional functionality, the internal implementation of the network-published shared variable is considerably more complex than that of the single-process shared variable. The next few sections discuss aspects of this implementation and offer recommendations for obtaining the best performance from the network-published shared variable.

NI-PSP

The NI Publish and Subscribe Protocol (NI-PSP) is a networking protocol optimized to be the transport for Network Shared Variables.

New in LabVIEW 8.5!

The underlying layer of NI-PSP has been rewritten to be substantially more efficient. On modern machines running modern OS kernels, we found that the performance of TCP was significantly better than trying to duplicate much of it's functionality in user mode. Taking advantage of this in LabVIEW 8.5, the lowest level protocol underneath NI-PSP has been rearchitected to use TCP/IP and has been thouroughly retuned with an eye toward performance on both desktop systems and NI's RT targets (see below for comparative benchmarks). The NI-PSP layer has remained unchanged and so you should see performance gains without modifying your application.

Important note:

LabVIEW will only use this new NI-PSP implementation if it is installed on both sides of the communication pipeline. In other words, only if LabVIEW 8.5 or later is installed on all the targets involved in the communication. If one of the targets is running an older version of LabVIEW, then both will step-down to the older protocol.

Theory of Operation of LogosXT

Figure 22.27 illustrates the software stack of the network shared variable. It is important to understand this because the theory of operation in play here is particular to the level of the stack called LogosXT. LogosXT is the layer of the stack responsible for optimizing throughput for the shared variable.

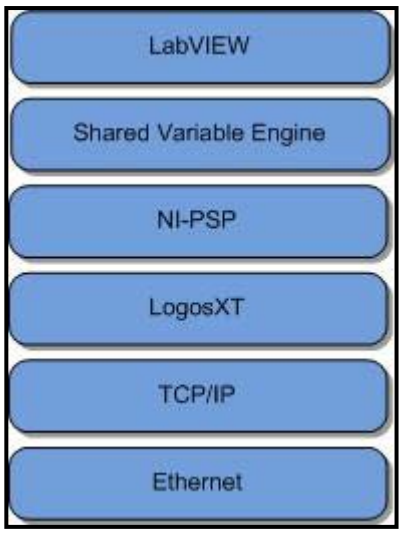


Figure 22.27: The Shared Variable Network Stack

Figure 22.28 shows the principle components of the LogosXT transmission algorithm. In essence, it is very simple. There are two important actors:

1. An 8 Kilobyte (KB) transmit buffer
2. A 10 millisecond (ms) timer thread

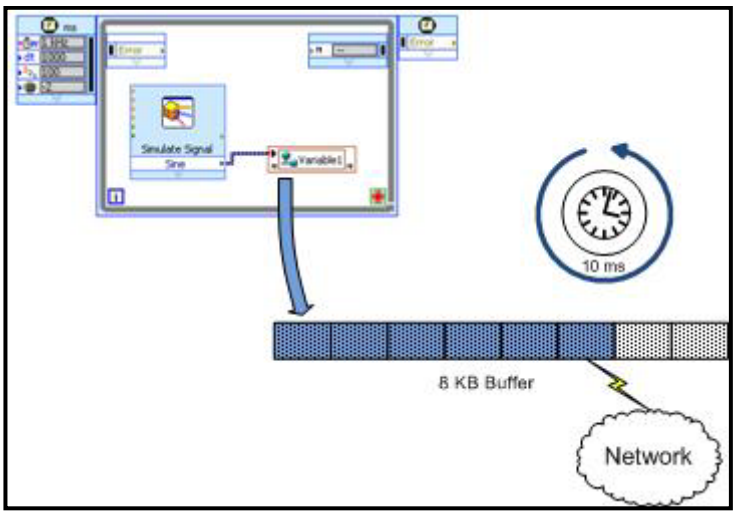


Figure 22.28: LogosXT Actors. The buffer will be transmitted if full or after 10ms has expired

These numbers were arrived at by thoroughly profiling various packet sizes and times to optimize for data **throughput**. The algorithm is as follows:

- IF the transmit buffer is filled to capacity (8KB) before the 10ms timer fires, then the data in that buffer is sent immediately to TCP on the same thread that initiated the write. In the case of the shared variable, the thread will be the shared variable engine thread.
- IF 10ms passes without the buffer ever filling up to capacity, then the data will be sent on the timer's thread.

*Important note: There is **one** transmit buffer for all the connections between two distinct endpoints. That is, all the variables representing connections between two different machines will share one buffer. Do not confuse this transmit buffer with the Buffering property of shared variables. This transmit buffer is a very low level buffer that multiplexes variables into one TCP connection and optimizes network throughput.*

It is important to understand the functionality of this layer of the network stack because it has side effects on the code on your LabVIEW diagram. The algorithm waits 10ms because it is always more efficient for throughput to send as much as possible in a single send operation. Every network operation has a fixed overhead both in time and in packet size. If we send many small packets (call it N packets) containing a total of B bytes, then we pay the network overhead N times. If instead, we send one large packet

containing B bytes, then we only pay the fixed overhead once and overall throughput is much greater.

This algorithm works very well if what you want to do is stream data from or to a target at the highest throughput rate possible. If, on the other hand, what you want to do is send small packets infrequently, for example sending commands to a target to perform some operation such as opening a relay (1 byte of boolean data), but you want it to get there as quickly as possible, then what you need to optimize for is called latency. In LabVIEW 8.5 there did not exist a hook to force LogosXT to flush its buffer. Instead, it was virtually guaranteed that there would be at least 10ms of latency built into the system as the program waited for the transmit buffer to be filled before finally timing every 10ms and sending what data it had.

If it is more important in your application for latency to be optimized, then in LabVIEW 8.5.1, you will find a new function in the Shared Variable palette called Flush.vi. Use this VI to force the transmit buffers in LogosXT to be flushed through the shared variable engine and across the network. This will drastically lower latency.

However, because as was noted above, all shared variables connecting one machine to one other machine share the same transmit buffer, by calling Flush, you are going to affect many of the shared variables on your system. If you have other variables that depend on high throughput, you will adversely affect them by calling Flush.vi (Figure 22.29).

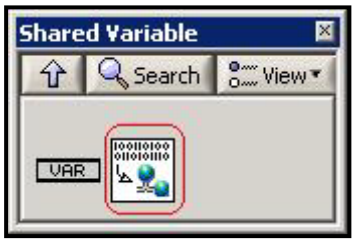


Figure 22.29: Flush.vi

Deployment and Hosting

You must ***deploy*** network-published shared variables to a shared variable engine (SVE) that ***hosts*** the shared variable values on the network. When you write to a shared variable node, LabVIEW sends the new value to the SVE that deployed and hosts the variable. The SVE processing loop then ***publishes*** the value so that ***subscribers*** get the updated value. Figure 22.30 illustrates this process. To use client/server terminology, the SVE is the ***server*** for a shared variable and all references are the ***clients***, regardless of whether they write to or read from the variable. The SVE client is part of the implementation of each shared variable node, and in this paper the term ***client*** and ***subscriber*** are interchangeable.

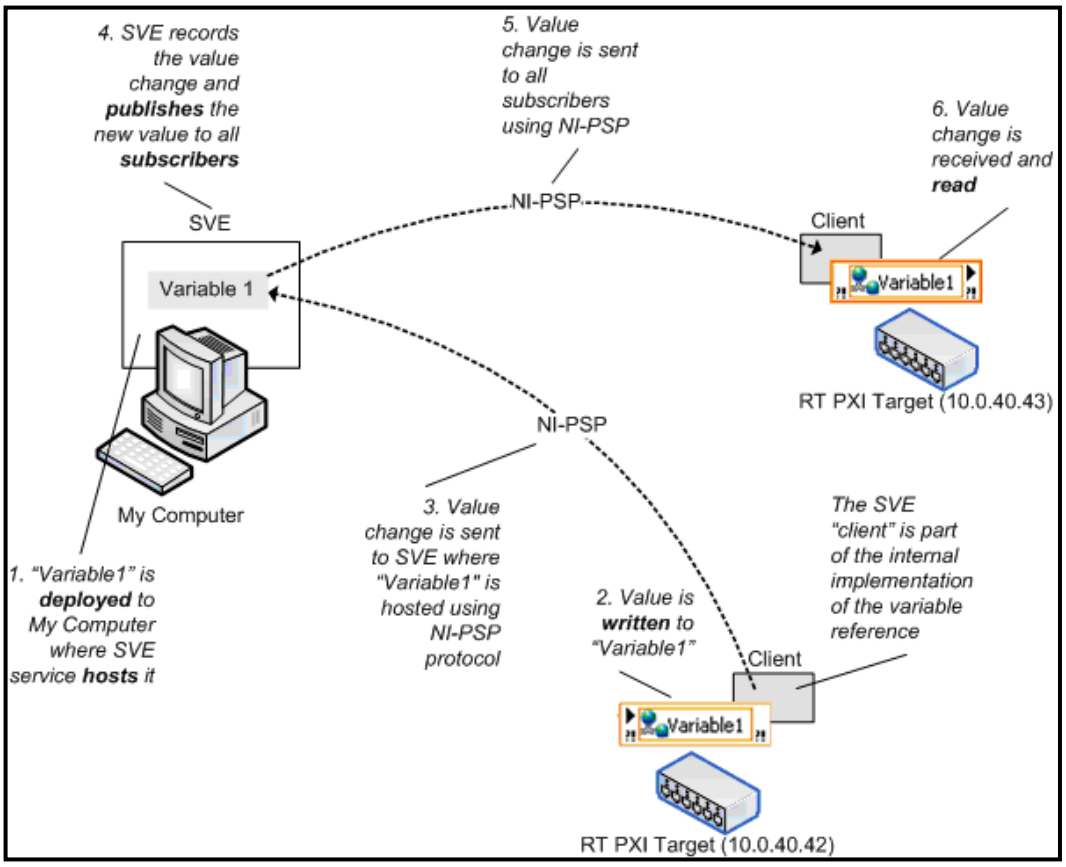


Figure 22.30: Shared Variable Engine and Network Shared Variable Value Changes

Network-Published Variables and LabVIEW Real-Time

You can enable real-time FIFOs with a network-published shared variable, but FIFO-enabled network-published shared variables have an important behavioral difference

compared to real-time, FIFO-enabled, single-process shared variables. Recall that with the single-process shared variable, all writers and readers share a single, real-time FIFO; this is not the case with the network-published shared variable. Each reader of a network-published shared variable gets its own real-time FIFO in both the single-element and multielement cases, as shown below.

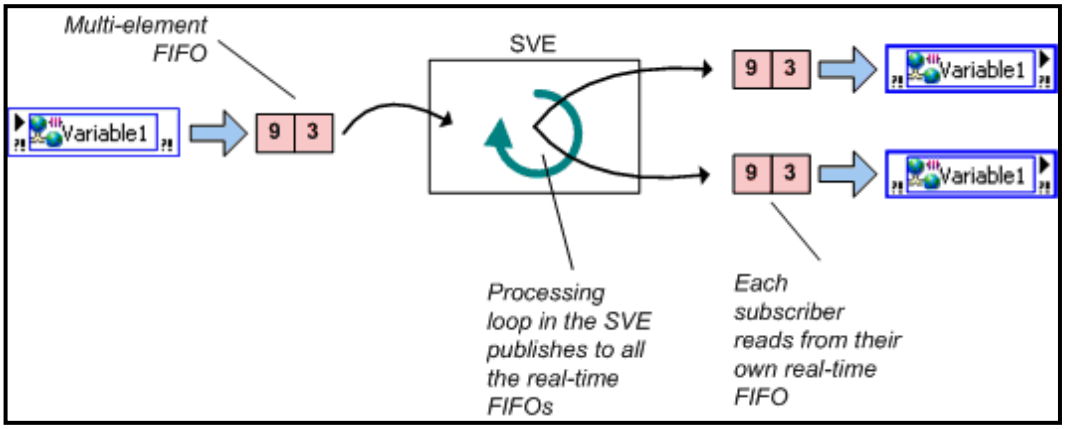


Figure 22.31: Real-Time FIFO-Enabled Network-Published Variable

Network Buffering

You can use buffering with the network-published shared variable. You can configure buffering in the **Shared Variable Properties** dialog box, as shown in Figure 22.32.

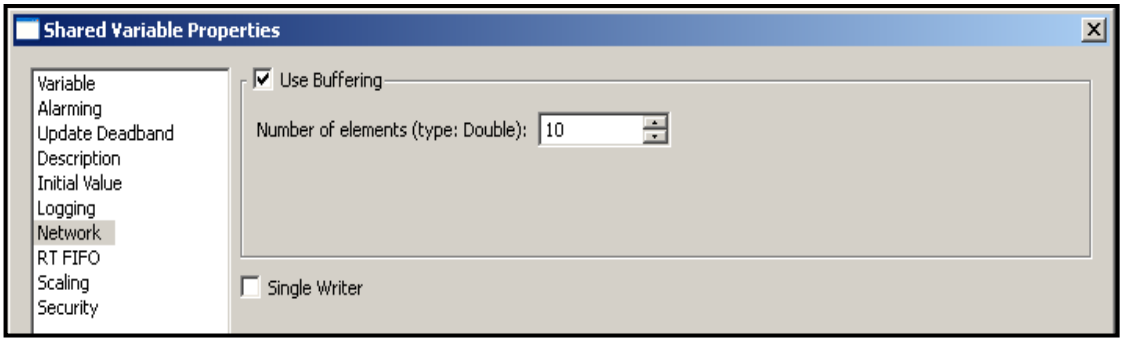


Figure 22.32: Enabling Buffering on a Networked-Published Shared Variable

With buffering enabled, you can specify the size of the buffer in units of the datatype, in this case, doubles.

With buffering, you can account for temporary fluctuations between read/write rates of a variable. Readers that occasionally read a variable slower than the writer can miss some updates. If the application can tolerate missing data points occasionally, the slower read rate does not affect the application and you do not need to enable buffering. However, if the reader must receive every update, enable buffering. You can set the size of the buffer on the **Variable** page of the **Shared Variable Properties** dialog box, so you can decide how many updates the application retains before it begins overwriting old data.

When you configure a network buffer in the dialog box above, you are actually configuring the size of *two different buffers*. The **server side buffer**, depicted as the buffer inside the box labeled Shared Variable Engine (SVE) in **figure 22.33** below is automatically created and configured to be the same size as the client side buffer, more on that buffer in a minute. The **client side buffer** is more likely the one that you logically think of when you are configuring your shared variable to have buffering enabled. The **client side buffer** (depicted on the right hand side of figure 22.33) is the buffer responsible for maintaining the queue of previous values. It is this buffer that insulates your shared variables from fluctuations in loop speed or network traffic.

In LabVIEW 8.5, network buffering has undergone a change. With the redesign of our low level NI-PSP protocol, the transmission of data becomes largely lossless. Again, in **figure 22.33** below, the buffer in the box labeled Shared Variable Engine (SVE) is the **server side buffer**. This buffer is utilized only in the case that the low level protocol gets flow-controlled. This should only happen in one of two cases:

1. If you write data into one or more shared variables faster than the client can acknowledge those incoming changes.
2. If you write data into one or more shared variables faster than the kernel driver can hand that data off to the physical layer (ethernet in most cases).

Note that both of these cases are limitations of network layers underneath (and out of the control of) the shared variable communication stack.

Unlike the real-time, FIFO-enabled, single-process variable, for which all writers and

readers share the same real-time FIFO, each reader of a network-published shared variable gets its own buffer so readers do not interact with each other.

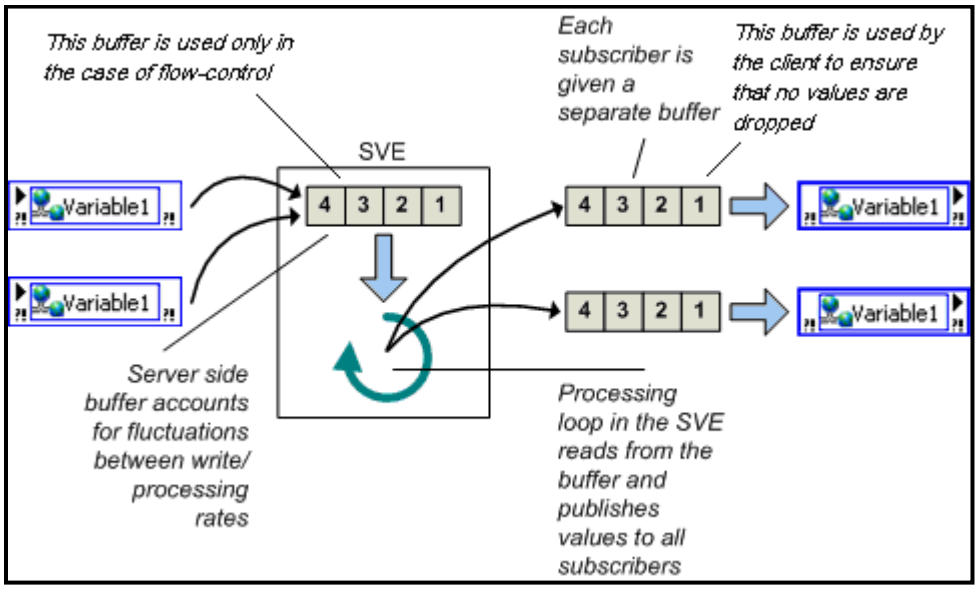


Figure 22.33: Buffering.

Buffering helps only in situations in which the read/write rates have temporary fluctuations. If the application runs for an indefinite period of time, readers that always read at a slower rate than a writer eventually lose data regardless of the size of the buffer you specify. Because buffering allocates a buffer for every subscriber, to avoid unnecessary memory usage, use buffering only when necessary.

Network and Real-Time Buffering

If you enable both network buffering and the real-time FIFO, the implementation of the shared variable includes both a network buffer and a real-time FIFO. Recall that if the real-time FIFO is enabled, a new real-time FIFO is created for each writer and reader, which means that multiple writers and readers will not block each other.

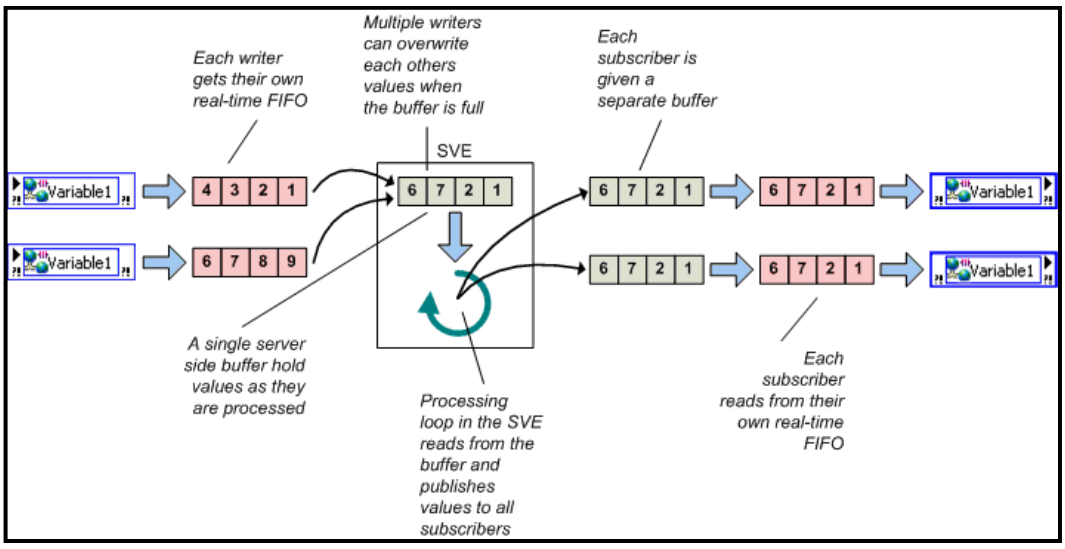


Figure 22.34: Network Buffering and Real-Time FIFOs

Although you can set the sizes of these two buffers independently, in most cases National Instruments recommends that you keep them the same size. If you enable the real-time FIFO, LabVIEW creates a new real-time FIFO for each writer and reader. Therefore, multiple writers and readers do not block each other.

Buffer Lifetime

LabVIEW creates network and real-time FIFO buffers on an initial write or read, depending on the location of the buffers. Server-side buffers are created when a writer first writes to a shared variable. Client-side buffers are created when a subscription is established. Prior to LabVIEW 8.6, this occurred when a Shared Variable read or write node executed the first time. In LabVIEW 8.6 and later, this occurs when the VI containing the Shared Variable node starts. If a writer writes data to a shared variable before a given reader subscribes to that variable, the initial data values are not available to the subscriber.

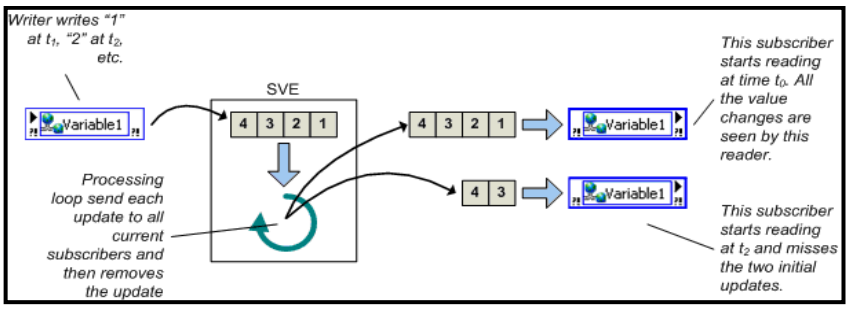


Figure 22.35: Buffer Lifetime

Buffer Overflow/Underflow

The network-published shared variable reports network buffer overflow and underflow conditions in LabVIEW 8.20 and newer. The real-time FIFO in all versions will indicate FIFO overflow/underflow by returning errors. An application in LabVIEW 8.0 or 8.0.1 can check for network buffer underflows in two ways. Because the timestamp resolution of the shared variable is 1 ms, you can compare the timestamp of a shared variable to the subsequent read timestamp to detect buffer underflows when you update the variable at less than 1 kHz. Or the reader can use a sequence number bundled with the data to notice buffer overflows/underflows. You cannot use the second approach with shared variables used inside a time-critical loop if the data type is an array because the real-time FIFO-enabled shared variables do not support the Custom Control (cluster) data type if one of the elements of the cluster is an array.

Shared Variable Lifetime

As mentioned earlier, all shared variables are part of a project library. The SVE registers project libraries and the shared variables those libraries contain whenever LabVIEW needs one of those variables. By default, the SVE deploys and publishes a library of shared variables as soon as you run a VI that references any of the contained variables. Because the SVE deploys the entire library that owns a shared variable, the SVE publishes all shared variables in the library whether or not a running VI references them all. You can deploy any project library manually at any time by right-clicking the library in the **Project Explorer** window.

Stopping the VI or rebooting the machine that hosts the shared variable does not make the variable unavailable to the network. If you need to remove the shared variable from the network, you must explicitly undeploy the library the variable is part of in the **Project Explorer** window. You also can select **Tools»Shared Variable»Variable Manager** to undeploy shared variables or entire project libraries of variables. In Labview 8.6 the Variable Manager has been replaced by the Distributed System Manager and can be found by selecting **Tools»Distributed System Manager.**

Front Panel Data Binding

An additional feature available for only network-published shared variables is front panel data binding. Drag a shared variable from the **Project Explorer** window to the front panel of a VI to create a control bound to the shared variable. When you enable data binding for a control, changing the value of the control changes the value of the shared variable to which the control is bound. While the VI is running, if the connection to the SVE is successful, a small green indicator appears next to the front panel object on the VI, as shown in Figure 22.36.

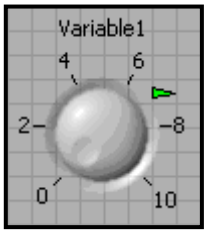


Figure 22.36: Binding a Front Panel Control to a Shared Variable

You can access and change the binding for any control or indicator on the **Data Binding** page of the **Properties** dialog box. When using the LabVIEW Real-Time Module or the LabVIEW DSC Module, you can select **Tools»Shared Variable»Front Panel Binding Mass Configuration** to display the **Front Panel Binding Mass Configuration** dialog box and create an operator interface that binds many controls and indicators to shared variables.

National Instruments does not recommend using front panel data binding for applications that run on LabVIEW Real-Time because the front panel might not be present.

Programmatic Access

As discussed above, you can create, configure, and deploy shared variables interactively using the LabVIEW Project, and you can read from and write to shared variables using the shared variable node on the block diagram or through front panel data binding. New in 2009, LabVIEW also provides programmatic access to all of this functionality.

Use VI Server to create project libraries and shared variables programmatically in applications where you need to create large numbers of shared variables. In addition, the LabVIEW DSC Module provides a comprehensive set of VIs to create and edit shared variables and project libraries programmatically as well as manage the SVE. You can create libraries of shared variables programmatically only on Windows systems; however, you can deploy these new libraries programmatically to Windows or LabVIEW Real-Time systems.

Use the programmatic Shared Variable API in applications where you need change dynamically the shared variable that a VI reads and writes, or where you need to read and write large numbers of variables. You can change a shared variable dynamically by building the URL programmatically.

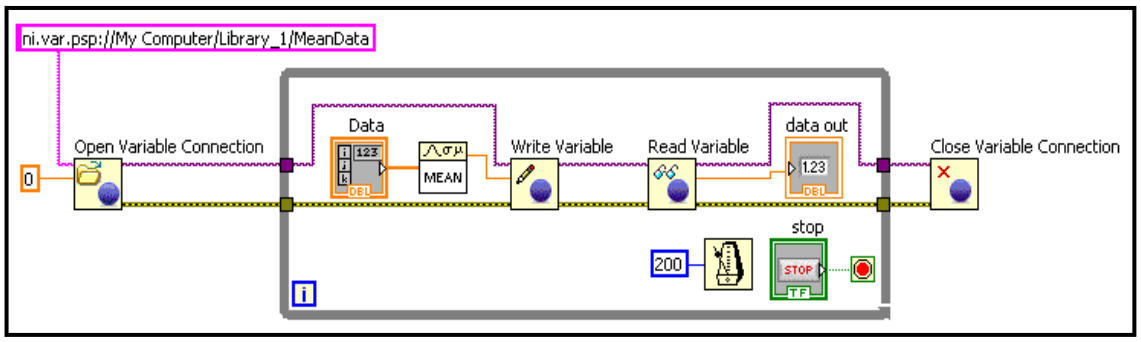


Figure 22.37: Using the programmatic Shared Variable API to Read and Write Shared Variables

In addition, with the Network Variable Library introducted in NI LabWindows/CVI 8.1 and NI Measurement Studio 8.1, you can read and write to shared variables in ANSI C, Visual Basic .NET or Visual C#.

The Shared Variable Engine

The SVE is a software framework that enables a networked-published shared variable to send values through the network. On Windows, LabVIEW configures the SVE as a service and launches the SVE at system startup. On a real-time target, the SVE is an instalable startup component that loads when the system boots.

In order to use network-published shared variables, an SVE must be running on at least one of the nodes in the distributed system. Any node on the network can read or write to shared variables that the SVE publishes. As shown in Table 22.1, nodes can reference a variable without having the SVE installed. You also might have multiple SVEs installed on multiple systems simultaneously if you need to deploy shared variables in different locations based on application requirements.

Recommendations for Shared Variable Hosting Location

You must consider a number of factors when deciding from which computing device to deploy and host network-published shared variables you use in a distributed system.

Is the computing device compatible with SVE?

The following table summarizes the platforms for which the SVE is available and identifies the platforms that can use network-published shared variables through reference nodes or the DataSocket API. National Instruments requires 32 MB of RAM and recommends 64 MB for the SVE on all applicable platforms.

New in LabVIEW 8.5

The Datasocket API now supports NI-PSP on Linux and Macintosh. Previously, we recommended a complex series of steps to get shared variable reference nodes working on those platforms. We no longer recommend this. Instead, it should be much more straightforward to simply author your client application directly on these platforms using Datasocket. Note that hosting of shared variables is still not supported on Linux nor Macintosh.

	Windows PCs	Mac OS	Linux	PXI Real-Time	Compact FieldPoint	CompactRIO	Compact Vision System	Commercial PCs with LabVIEW Real-Time ETS
SVE	✓	X	X	✓	✓	✓	✓	✓
Reference Nodes	✓	X	X	✓	✓	✓	✓	✓
DataSocket API w/ PSP	✓	✓	✓	✓	✓	✓	✓	✓

Table 22.1: Network-Published Shared Variable Compatibility Overview

Does the application require datalogging and supervisory functionality?

If you want to use the features of the LabVIEW DSC Module, you must host the shared variables on Windows. The LabVIEW DSC Module adds the following functionality to network-published shared variables:

· Historical logging to NI Citadel database.
· Networked alarms and alarm logging.
· Scaling.
· User-based security.
· Initial value.
· The ability to create custom I/O servers.
· Integration of the LabVIEW event structure with the shared variable.
· LabVIEW VIs to programmatically control all aspects of shared variables and the

shared variable engine. These VIs are especially helpful for managing large numbers of shared variables.

Does the computing device have adequate processor and memory resources?

The SVE is an additional process that requires both processing and memory resources. In order to get the best performance in a distributed system, install the SVE on machines with the most memory and processing capabilities.

Which system is always online?

If you build a distributed application in which some of the systems might go offline periodically, host the SVE on a system that is always online.

Additional Features of the Shared Variable Engine

Figure 22.38 illustrates the many responsibilities of the SVE. In addition to managing networked-published shared variables, the SVE is responsible for:

· Collecting data received from I/O servers.

· Serving up data through the OPC and the PSP servers to subscribers.

· Providing scaling, alarming, and logging services for any shared variable with those services configured. These services are available only with the LabVIEW DSC Module.

· Monitoring for alarm conditions and responding accordingly.

I/O Servers

I/O servers are plug-ins to the SVE with which programs can use the SVE to publish data. NI FieldPoint 5.0 includes an I/O server that publishes data directly from FieldPoint banks to the SVE. Because the SVE is an OPC server, the combination of the SVE and the FieldPoint I/O server serves as an FP OPC server. The FieldPoint installer

does not include the SVE; it would have to be installed by another sofware component such as LabVIEW.

NI-DAQmx 8.0 also includes an I/O server that can publish NI-DAQmx global virtual channels automatically to the SVE. This I/O server replaces the Traditional DAQ OPC server and RDA. NI-DAQmx includes the SVE and can install it when LabVIEW is not installed.

With the LabVIEW DSC Module, users can create new I/O servers.

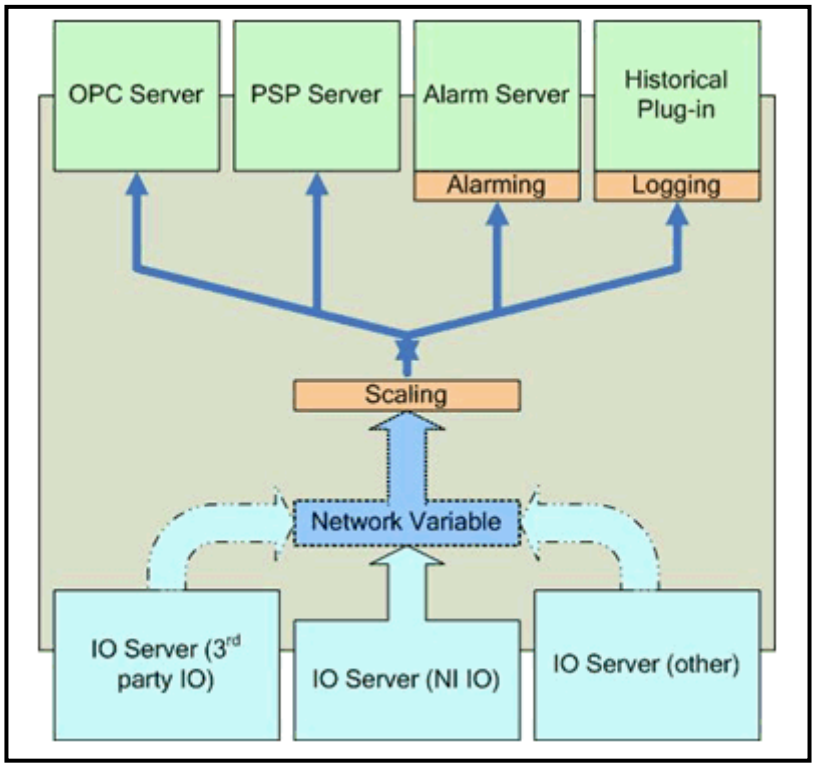


Figure 22.38: Shared Variable Engine (SVE)

OPC

The SVE is 3.0 compliant and can act as an OPC server on Windows machines. Any OPC client can write to or read from a shared variable hosted on a Windows machine. When you install the LabVIEW DSC Module on a Windows machine, the SVE also can act as an OPC client. You can bind shared variables that a Windows machine hosts to OPC data items with DSC and by writing to or reading from the variable to the OPC data ítem.

Because OPC is a technology based on COM, a Windows API, real-time targets do not work with OPC directly. As Figure 22.39 shows, you still can access OPC data items from a real-time target by hosting the shared variables on a Windows machine.

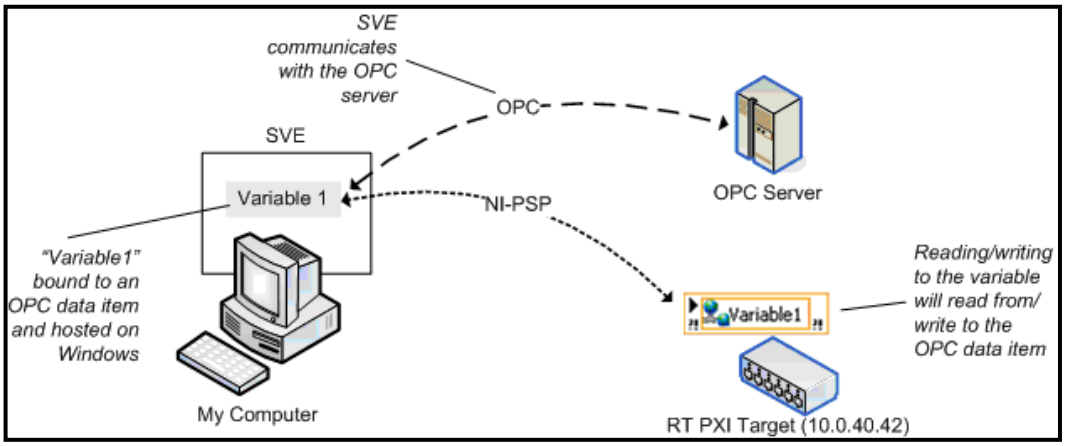


Figure 22.39: Binding to OPC Data Item

Performance

This section provides general guidelines for creating high-performing applications using the shared variable.

Because the single-process shared variable has an implementation similar to the LabVIEW global variables and real-time FIFOs, National Instruments has no particular recommendations for achieving good performance for single-process shared variables. The following sections concentrate on the network-published shared variable.

Share the Processor

The network-published shared variable simplifies LabVIEW block diagrams by hiding many of the implementation details of network programming. Applications consist of LabVIEW VIs as well as the SVE and the SVE client code. In order to get the best shared variable performance, develop the application so that it relinquishes the processor regularly for the SVE threads to run. One way to accomplish this is to place waits in the processing loops and ensure that the application does not use untimed loops. The exact amount of time you must wait is application, processor, and network-

dependent; every application requires some level of empirical tuning in order to achieve the best performance.

Consider the Location of the SVE

The *Recommendations for Shared Variable Hosting Location* section discussed a number of factors you must consider when deciding where to install the SVE. Figure 22.40 shows another factor that can greatly impact the performance of the shared variable. This example involves a real-time target, but the basic principles also apply to nonreal-time systems. Figure 22.40 shows an inefficient use of network-published shared variables: you generate data on a real-time target and need to log the data processed locally and monitor it from a remote machine. Because variable subscribers must receive data from the SVE, the latency between the write in the high-priority loop and the read in the normal-priority loop is large and involves two trips across the network.

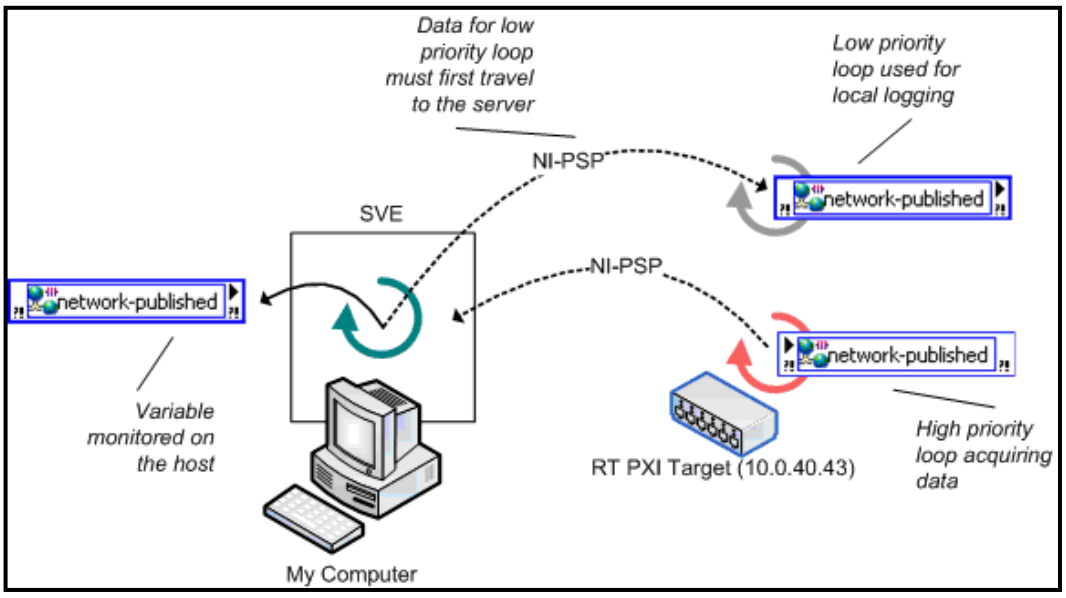


Figure 22.40: Inefficient Use of Network-Published Variables on Real-Time

Figure 22.41 shows a better architecture for this application. The application uses a single-process shared variable to transfer data between the high-priority loop and the low-priority loop, reducing the latency greatly. The low-priority loop logs the data and writes the update to a network-published shared variable for the subscriber at the host.

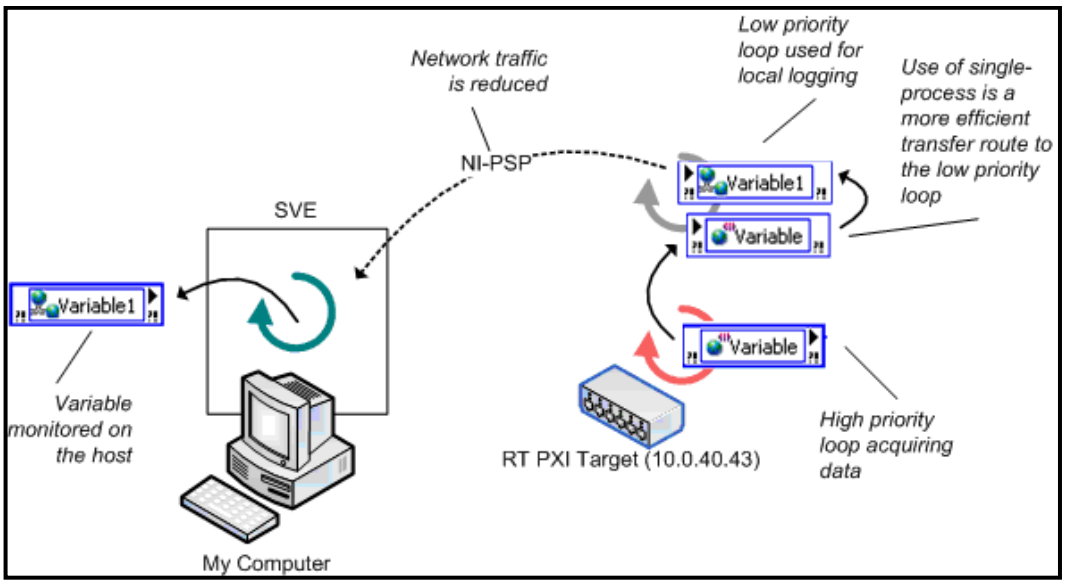


Figure 22.41: Efficient Use of Network-Published Variables on Real-Time

Benchmarks

This section compares the performance of the shared variable to other applicable data sharing methods in LabVIEW, such as the LabVIEW global variable, real-time FIFOs, and TCP/IP. The following table summarizes the tests discussed are presented in the following sections.

Test	Description	SVE Location	Notes
T1	Single-process shared variable vs. global variable	N/A	Establish maximum read/write rates.
T2	Single-process shared variable w/ real-time FIFO vs. real-time FIFO VIs	N/A	Establish maximum read/write rates when using real-time FIFOs. Determine the highest sustainable rate at which you can write to a shared variable or real-time FIFO in a timed loop and

			simultaneously read the data back from a normal priority loop.
T3	Network-published shared variable w/ real-time FIFO vs. 2-loop real-time FIFO w/ TCP	PXI running LV RT	Establish the maximum rate single-point data can be streamed across the network. Shared variable: reader VI is always on the host.RT-FIFO + TCP: similar to T2 with the addition of TCP communication./IP networking.
T4	Network-published shared variable footprint	RT Series Target	Establish memory usage of shared variables after deployment.
T5	Comparison between 8.2 Nework-published shared variables and 8.5 variables - streaming	RT Series Target	Comparing the new 8.5 implementation of NI-PSP to the 8.20 and earlier implementation. This benchmark measures throughput of an application streaming waveform data from a cRIO device to a desktop host.
T6	Comparison between 8.2 Nework-published shared variables and 8.5 variables - high channel count	RT Series Target	Comparing the new 8.5 implementation of NI-PSP to the 8.20 and earlier implementation. This benchmark measures throughput of a high channel count application on a cRIO device.

Table 22.2. Benchmark Overview

The following sections describe the code National Instruments created for each of the benchmarks, followed with the actual benchmark results. The *Methodology and Configuration* section discusses in greater detail the chosen methodology for each

benchmark and the detailed configuration used for the hardware and software on which the benchmark was run.

Single-Process Shared Variables vs. LabVIEW Global Variables

The single-process shared variable is similar to the LabVIEW global variable. In fact, the implementation of the single-process shared variable is a LabVIEW global with the added time-stamping functionality.

To compare the performance of the single-process shared variable to the LabVIEW global variable, National Instruments created benchmark VIs to measure the number of times the VI can read and write to a LabVIEW global variable or single-process shared variable every second. Figure 22.42 shows the single-process shared variable read benchmark. The single-process shared variable write benchmark and the LabVIEW global read/write benchmarks follow the same pattern.

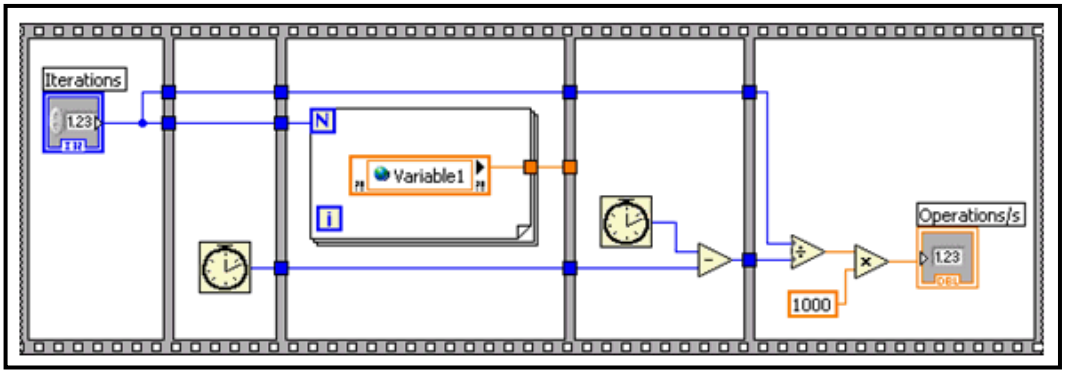


Figure 22.42: Single-Process Shared Variable Read Benchmarking VI

The combination read/write test also includes code to validate that every point written also can be read back in the same iteration of the loop with no data corruption.

T1 Test Results

Figure 22.43 shows the results of Test T1. The results show that the read performance for a single-process shared variable is lower than the read performance for a LabVIEW global variable. The write performance, and hence the read/write performance, of the single-process shared variable is slightly lower than that of the LabVIEW global variable. Performance of single-process shared variables will be affected by enabling and disabling the timestamp functionality, so it is recommended to turn the timestamp off if it is not useful.

The *Methodology and Configuration* section explains the specific benchmarking methodology and configuration details for this test set.

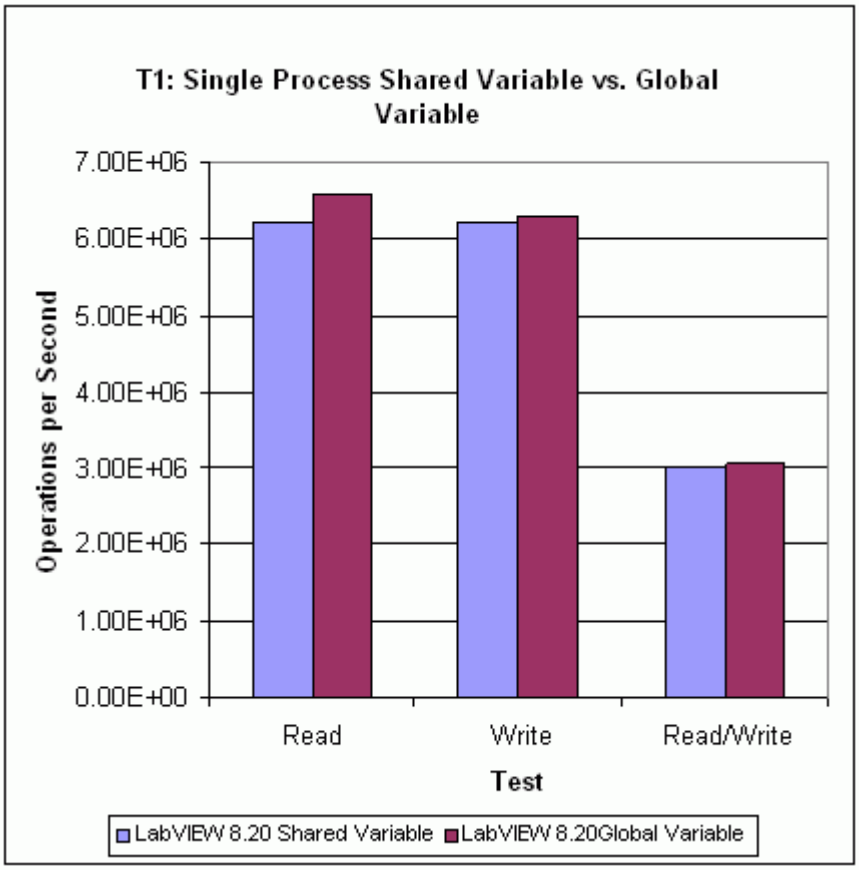


Figure 22.43: Single-Process Shared Variable vs. Global Variable Performance

Single-Process Shared Variables vs. Real-Time FIFOs

National Instruments benchmarked sustainable throughput to compare the performance of the FIFO-enabled single-process shared variable against traditional real-time FIFO VIs. The benchmark also examines the effect of the size of the transferred data, or payload, for each of the two real-time FIFO implementations.

The tests consist of a time-critical loop (TCL) generating data and a normal-priority loop (NPL) consuming the data. National Instruments determined the effect of the payload size by sweeping through a range of double-precision scalar and array data types. The scalar type determines the throughput when the payload is one double, and the array types determine the throughput for the rest of the payloads. The test records the maximum sustainable throughput by determining the maximum sustainable speed at which you can execute both loops with no data loss.

Figure 22.44 shows a simplified diagram of the real-time FIFO benchmark that omits much of the necessary code required to create and destroy the FIFOs. Note that as of LabVIEW 8.20, a new FIFO function was introduced to replace the FIFO SubVI's shown here. The FIFO functions were used for the data graphed in this paper, and perform better than their 8.0.x SubVI predecessors.

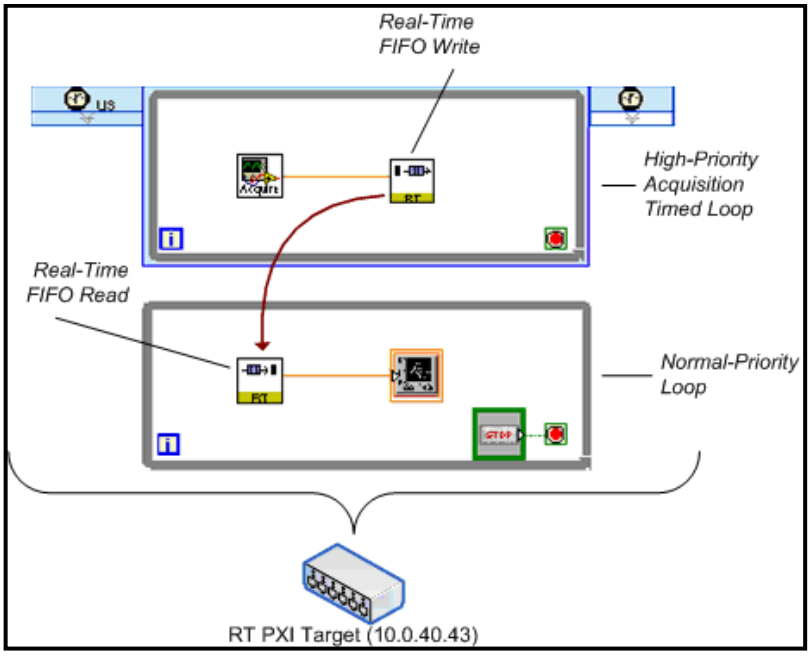


Figure 22.44: Simplified Real-Time FIFO Benchmarking VI

An equivalent version of the test uses the single-process shared variable. Figure 22.45 shows a simplified depiction of that diagram.

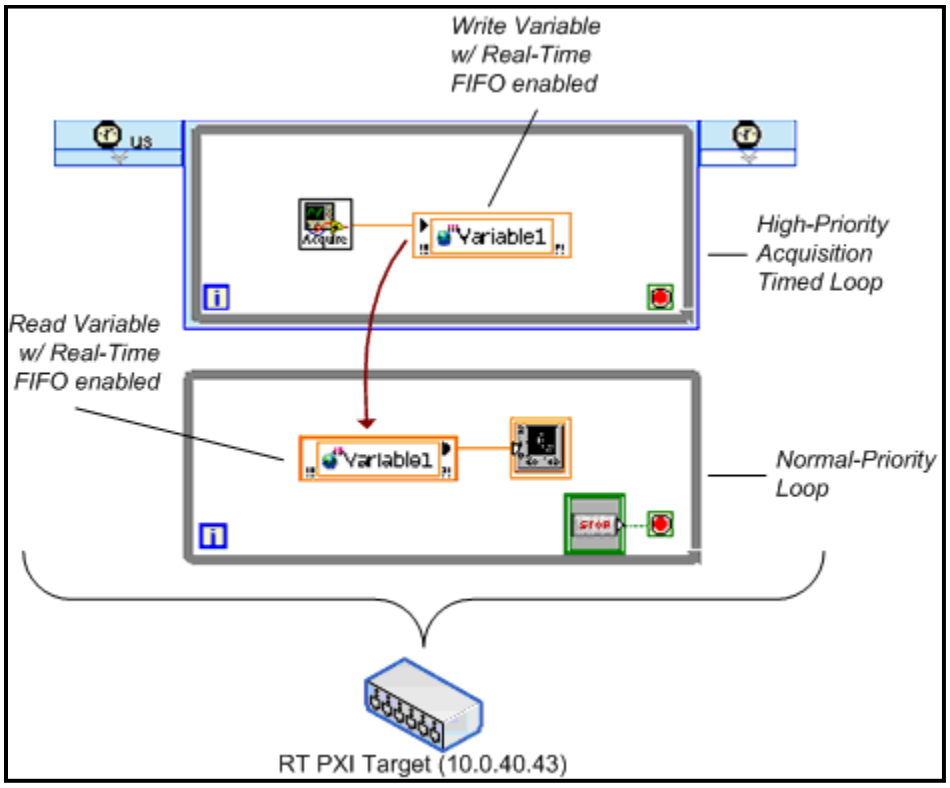


Figure 22.45: Simplified FIFO-Enabled Single-Process Shared Variable Benchmarking VI

T2 Test Results

Figure 22.46 and 22.47 shows the results of Test T2, comparing the performance of the FIFO-enabled single-process shared variable against real-time FIFO functions. The results indicate that using the single-process shared variable is marginally slower than using real-time FIFOs.

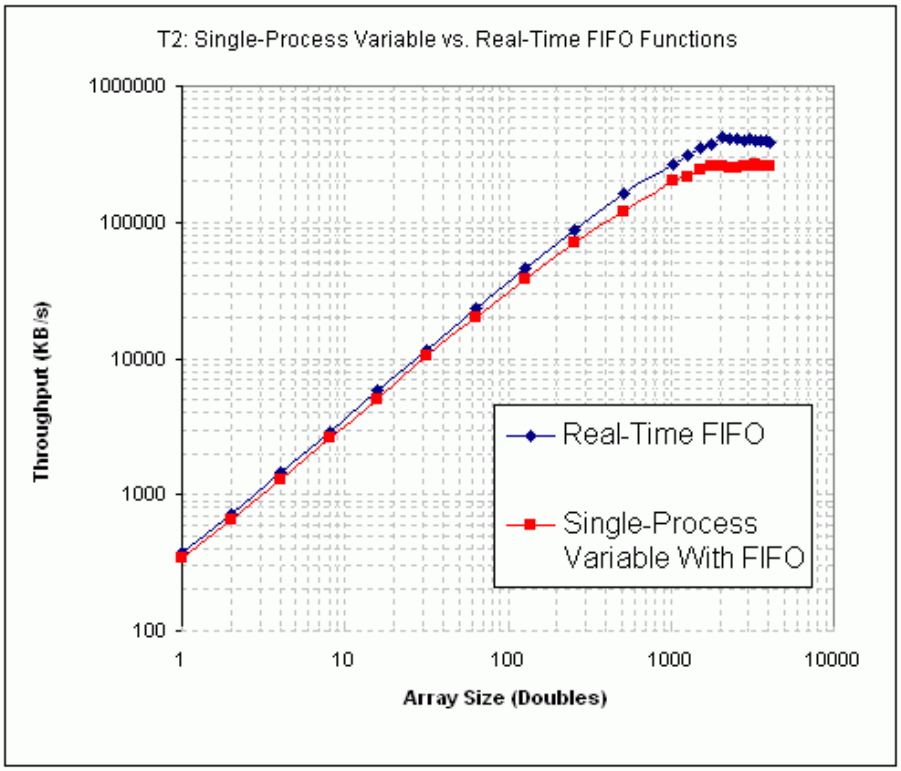


Figure 22.46: Single-Process Shared Variable vs. Real-Time FIFO VI Performance (PXI)

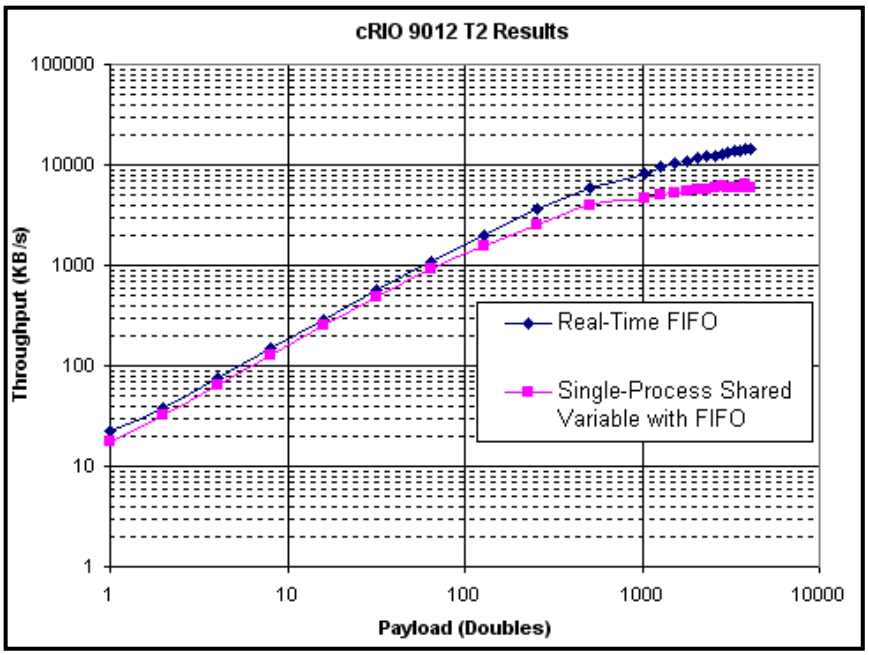


Figure 22.47: Single-Process Shared Variable vs. Real-Time FIFO VI Performance (cRIO 9012)

Network-Published Shared Variables vs. Real-Time FIFOs and TCP/IP

With the flexibility of the shared variable, you can publish a single-process shared variable quickly across the network with just a few configuration changes. For real-time applications in particular, performing the same transformation in earlier versions of LabVIEW requires the introduction of a large amount of code to read the real-time FIFOs on the RT Series controller and then send the data over the network using one of the many available networking protocols. To compare the performance of these two different approaches, National Instruments again created benchmark VIs to measure the sustainable throughput of each with no data loss across a range of payloads.

For the prevariable approach, the benchmark VI uses real-time FIFOs and TCP/IP. A TCL generates data and places it in a real-time FIFO; an NPL reads the data out of the FIFO and sends it over the network using TCP/IP. A host PC receives the data and validates that no data loss occurs.

Figure 22.48 shows a simplified diagram of the real-time FIFO and TCP/IP benchmark. Again, this diagram vastly simplifies the actual benchmark VI.

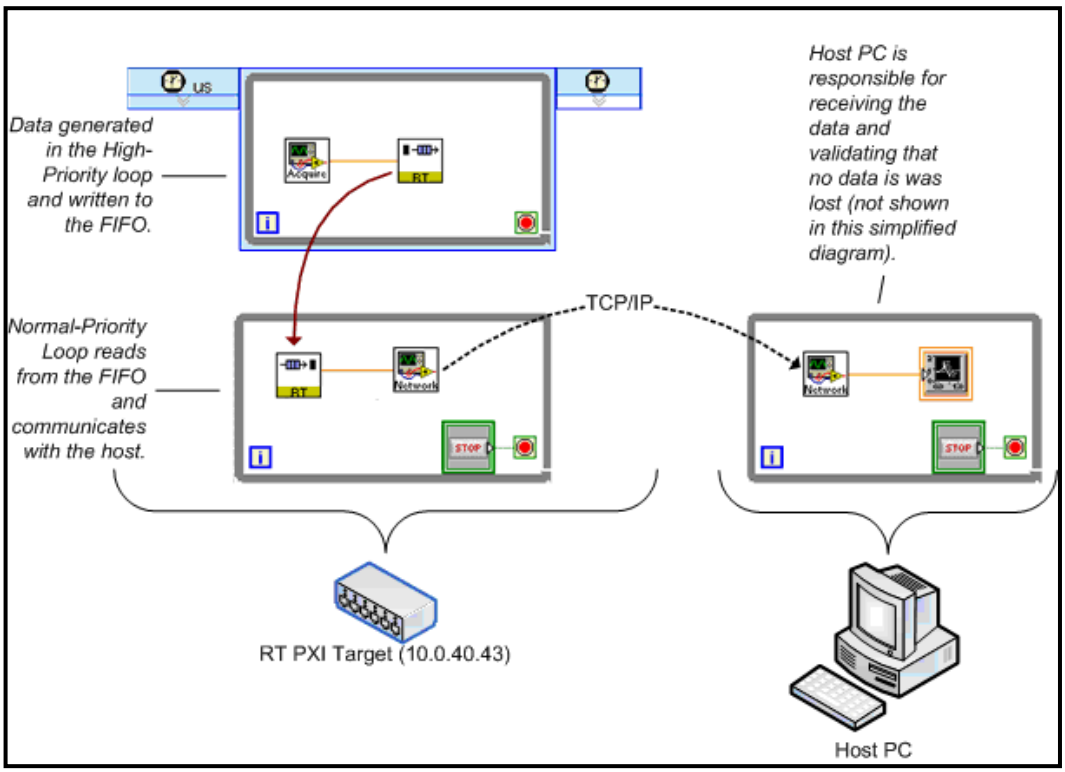


Figure 22.48: Simplified Real-Time FIFO and TCP/IP Benchmarking VI

National Instruments created an equivalent version of the test using the network-published shared variable. Figure 22.49 shows a simplified diagram.

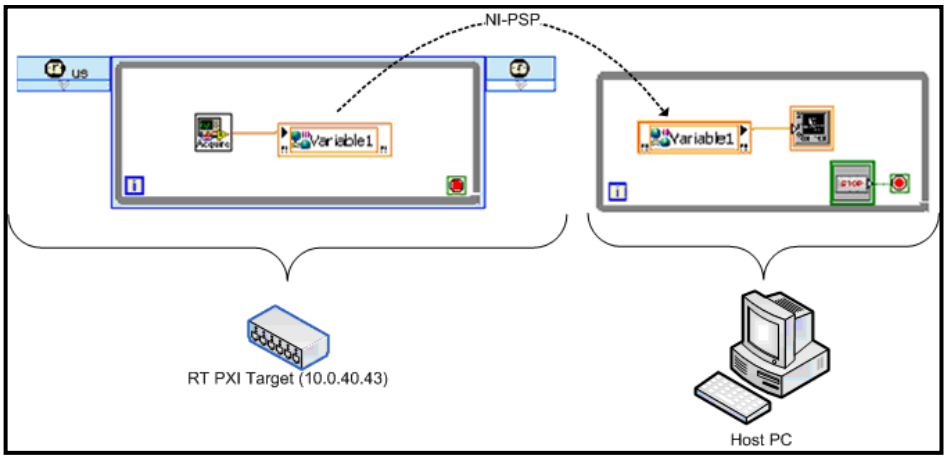


Figure 22.49: Simplified Real-Time FIFO-Enabled Network-Published Shared Variable Benchmarking VI

T3 Test Results

This section contains the results of Test T3, comparing the performance of the real-time FIFO-enabled network-published shared variable to that of equivalent code relying on real-time FIFO VIs and LabVIEW TCP/IP primitives. Figure 22.50 shows the results when the LabVIEW Real-Time target is an embedded RT Series PXI controller.

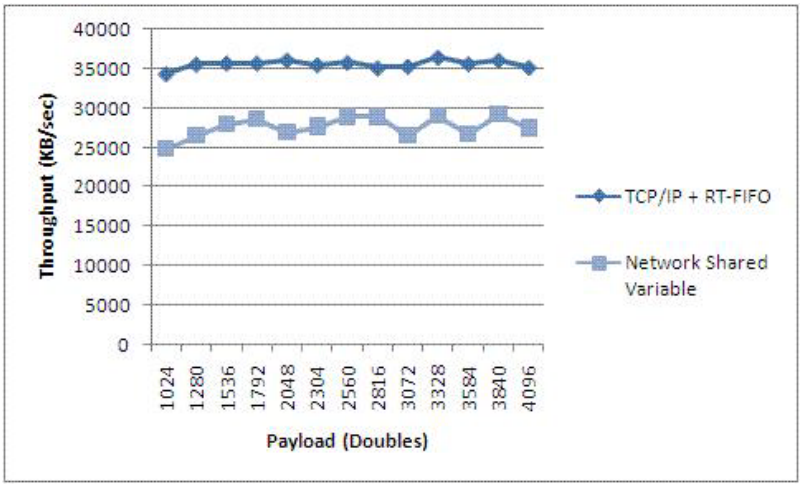


Figure 22.50: Network-Published Shared Variable vs. Real-Time FIFO and TCP VI Performance (PXI)

T3 Results indicate that throughput of Network-published Shared Variables approaches that of TCP and that both are consistent across moderate to large payload sizes. The shared variable makes your programming job easier, but that does not come without a cost. It should be noted, however, that if a naive TCP implementation is used, it could be easy to underperform shared variables, particularly with the new 8.5 implementation of NI-PSP.

T4 Test Results

Memory Footprint of the Network-Published Shared Variable

Note that no significant changes to variable footprint have been made in LabVIEW 8.5. Therefore, this benchmark was not rerun.

Determining the memory footprint of the shared variable is difficult because the memory used by the shared variable depends on the configuration. Network-published shared variables with buffering, for example, allocate memory dynamically as needed by the program. Configuring a shared variable to use real-time FIFOs also increases memory usage because in addition to network buffers, LabVIEW creates buffers for the FIFO. Therefore the benchmark results in this paper provide only a baseline measurement of memory.

Figure 22.51 shows the memory that the SVE uses after LabVIEW deploys 500 and 1000 shared variables of the specified types to it. The graph shows that the type of variable does not affect the memory usage of the deployed shared variables significantly. Note that these are nonbuffered variables.

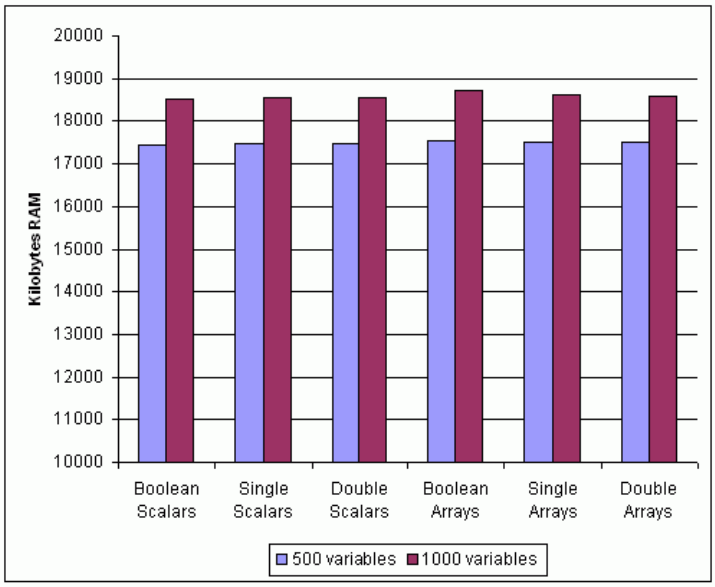


Figure 22.51: Memory Usage of Network-Published Shared Variables with Different Data Types

Figure 22.52 shows memory usage as a function of the number of shared variables deployed. This test uses only one type of variable, an empty Boolean array. The memory usage increases linearly with the variable count.

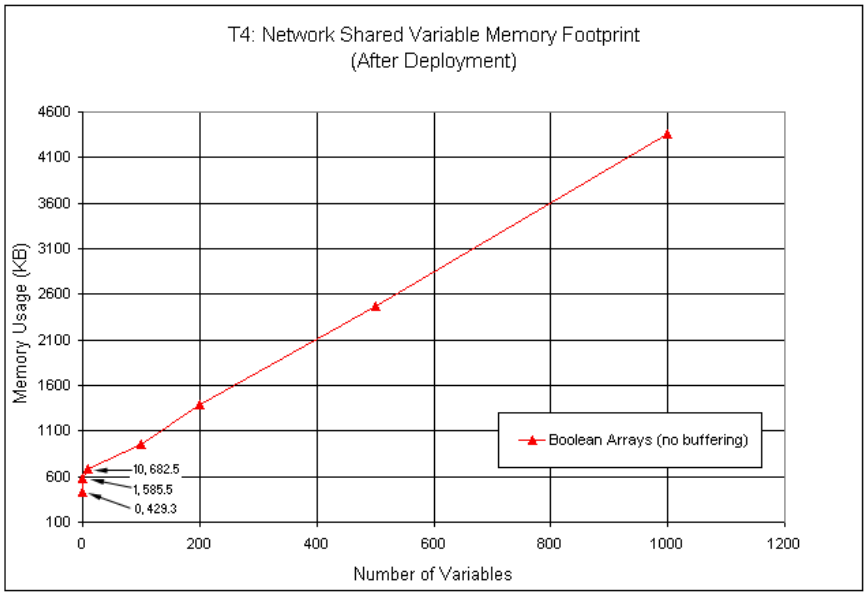


Figure 22.52: Memory Usage of Shared Variables of Different Sizes

T5 Test Results

Comparison between 8.2 Nework-published shared variables and 8.5 variables - streaming

In LabVIEW 8.5, we have reimplemented the bottom layer of the network protocol used to transport shared variable data. It offers significantly better performance.

In this case, we hosted a single variable of type **Waveform of Doubles** on a cRIO 9012. We generated all the data, then in a tight loop, transferred the data to the host which read it out of another waveform shared variable node as fast as it could.

You can see from figure 22.53, that performance has improved in LabVIEW 8.5 by over 600% for this use case.

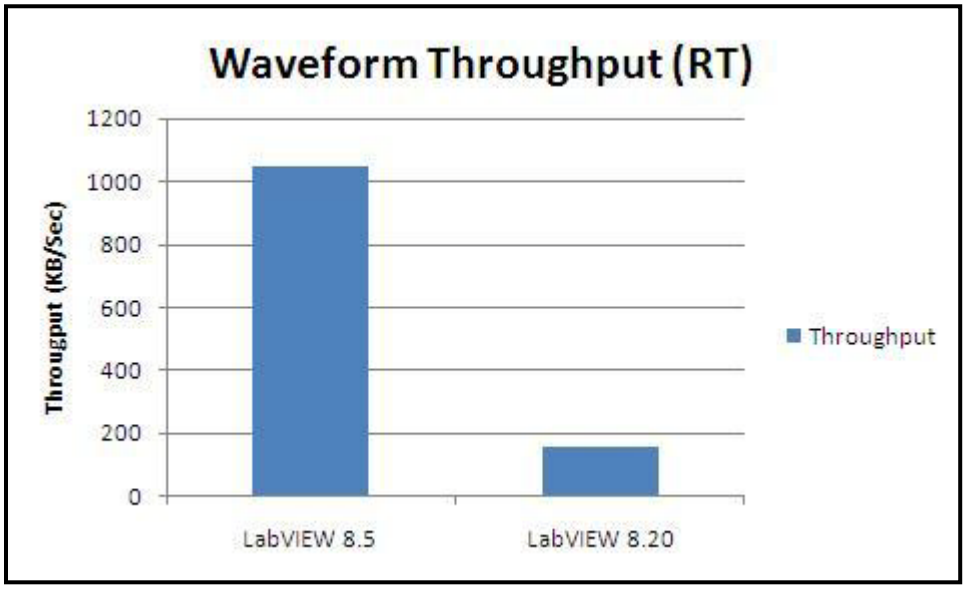


Figure 22.53: Waveform throughput comparison between LabVIEW 8.5 and LabVIEW 8.20 (and earlier)

T6 Test Results

Comparison between 8.2 Nework-published shared variables and 8.5 variables - streaming

In this test, we used the same two targets as in T5, but instead of transferring a single variable, we changed the datatype to a double and varied the number of shared variables from 1 - 1000, measuring throughput along the way. Again, all the variables were hosted on the cRIO 9012, incrementing data was generated there as well and transmitted to the host where they were read.

Figure 22.54 again shows a significant performance increase from LabVIEW 8.20 to LabVIEW 8.5. However, the throughput is significantly less in the case of many smaller variables than it is for a single large variable as in T5. This is because each variable has associated with it a fixed amount of overhead. When many variables are used, this overhead is multiplied by the nubmer of variables and becomes quite noticable.

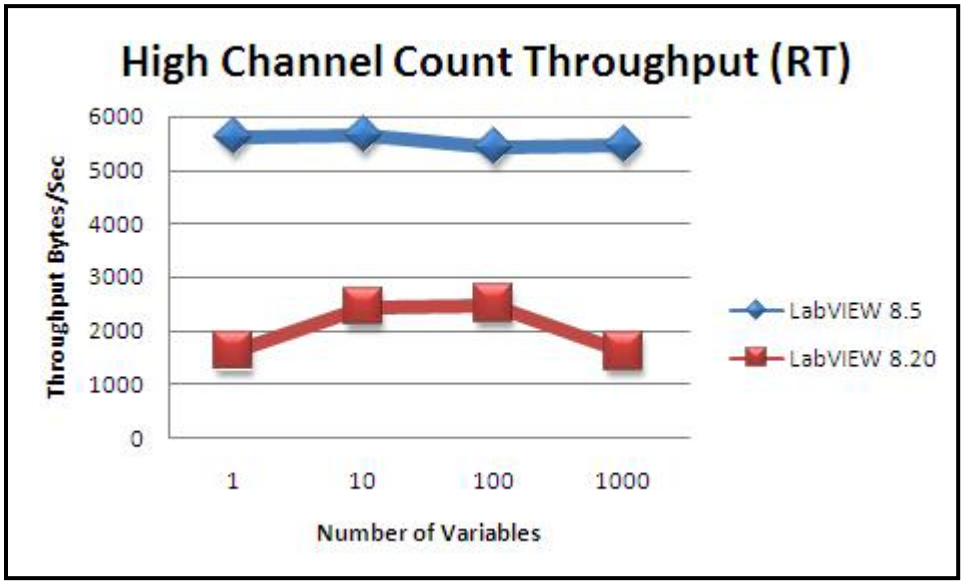


Figure 22.54: High channel count throughput comparison between LabVIEW 8.5 and LabVIEW 8.20 (and earlier)

Methodology and Configuration

This section provides detailed information regarding the benchmarking process for all the previously mentioned test sets.

T1 Methodology and Considerations

Test T1 uses a simple benchmarking template to determine read and write rates through simple averaging over a large number of iterations. Each test executed a total of 500 million iterations for total execution times in the order of one minute, timed with millisecond resolution.

T1 Hardware/Software Configuration

Host Hardware

- Dell Precision 450
- Dual Intel Xeon 2.4 GHz Pentium-class processors
- 1 GB DRAM

Host Software

- Windows XP SP2
- LabVIEW 8.20

T2 Methodology and Considerations

Test T2 measures throughput by identifying the maximum sustainable communication rate between tasks running at different priorities. A timed loop running with microsecond resolution contains the data producer. The consumer of the data is a free-running, normal-priority loop that reads from the real-time FIFO or single-process

shared variable until empty, repeating this process until a certain amount of time elapses without errors. The test result is valid only if all of the following statements are true:

- No buffer overflows occur
- Data integrity is preserved: no data losses occur and the consumer receives the data in the order it was sent
- Timed loop is able to keep up after a designated warm-up period of 1 s

The single-process shared variable receiver loop performs simple data integrity checks, such as ensuring that the expected number of data points were received and that the received message pattern does not lack intermédiate values.

National Instruments configured the buffer sizes for the real-time FIFO and shared variable FIFO buffers to be 100 elements deep for all test variations and data types involved.

T2 Hardware/Software Configuration

PXI Hardware

- NI PXI-8196 RT Series controller
- 2.0 GHz Pentium-class processor
- 256 MB DRAM
- Broadcom 57xx (1 Gb/s built-in Ethernet adapter)

PXI Software

- LabVIEW 8.20 Real-Time Module
- Network Variable Engine 1.2.0
- Variable Client Support 1.2.0
- Broadcom 57xx Gigabit Ethernet driver 2.1 configured in polling mode

CompactRIO Hardware

- NI cRIO 9012 controller

- 400 MHz processor
- 64 MB DRAM

CompactRIO Software

- LabVIEW 8.20 Real-Time Module
- Network Variable Engine 1.2.0
- Variable Client Support 1.2.0

T3 Methodology and Considerations

Test T3 measures throughput directly by recording the amount of data transmitted over the network and the overall duration of the test. A timed loop running with microsecond resolution contains the data producer, which is responsible for writing a specific data pattern to the network-published shared variable or real-time FIFO VIs.

For the network-published shared variable case, an NPL runs on the host system and reads from the variable in a free-running while loop. For the real-time FIFO VI test, the NPL runs on the real-time system, checking the state of the FIFO at a given rate, reading all available data, and sending it over the network using TCP. The benchmark results show the effect of having this polling period set to either 1 or 10 ms.

The test result is valid only if all of the following statements are true:

- No buffer overflows occur, neither FIFO nor network
- Data integrity is preserved: no data losses occur and the consumer receives the data in the order it was sent
- The timed loop in the TCL VI is able to keep up after a designated warm-up period of 1 s

After reading each data point, the NPL for the network variable test checks the data pattern for correctness. For the real-time FIFO test, the TCL is responsible for validation based on whether a real-time FIFO overflow occurred.

To help avoid data loss, National Instruments configured the buffer sizes to be no smaller than the ratio between the NPL and TCL loop periods, with a lower bound of 100 elements as the minimum size for the real-time FIFO buffers.

T3 Hardware/Software Configuration

Host Hardware

- Intel Core 2 Duo 1.8 GHz
- 2 GB DRAM
- Intel PRO/1000 (1 Gb/s Ethernet adapter)

Host Software

- Windows Vista 64
- LabVIEW 8

Network Configuration

- 1Gb/s switched network

PXI Hardware

- NI PXI-8196 RT controller
- 2.0 GHz Pentium-class processor
- 256 MB DRAM
- Broadcom 57xx (1 Gb/s built-in Ethernet adapter)

PXI Software

- LabVIEW 8.5 Real-Time Module
- Network Variable Engine 1.2.0
- Variable Client Support 1.2.0
- Broadcom 57xx Gigabit Ethernet driver 2.1

T4 Methodology and Considerations

In Test T4, National Instruments used nonbuffered, network-published shared variables with the following data types: double, single, Boolean, double array, single array, and Boolean array.

T4 Hardware/Software Configuration

PXI Hardware

- NI PXI-8196 RT Controller
- 2.0 GHz Pentium-class processor
- 256 MB DRAM
- Broadcom 57xx (1 Gb/s built-in Ethernet adapter)

PXI Software

- LabVIEW Real-Time Module 8.0
- Network Variable Engine 1.0.0
- Variable Client Support 1.0.0
- Broadcom 57xx Gigabit Ethernet driver 1.0.1.3.0

T5 and T6 Methodology and Considerations

In test T5 and T6, National Instruments used nonbuffered, network-published shared variables of the Waveform of Double datatype.

T5 and T6 Hardware/Software Configuration

Host Hardware

- 64 Bit Intel Core 2 Duo 1.8 GHz
- 2 GB RAM
- Gigabit ethernet

Host Software

- Windows Vista 64

- LabVIEW 8.20 and LabVIEW 8.5

Compact RIO Hardware

- cRIO 9012
- 64 MB RAM

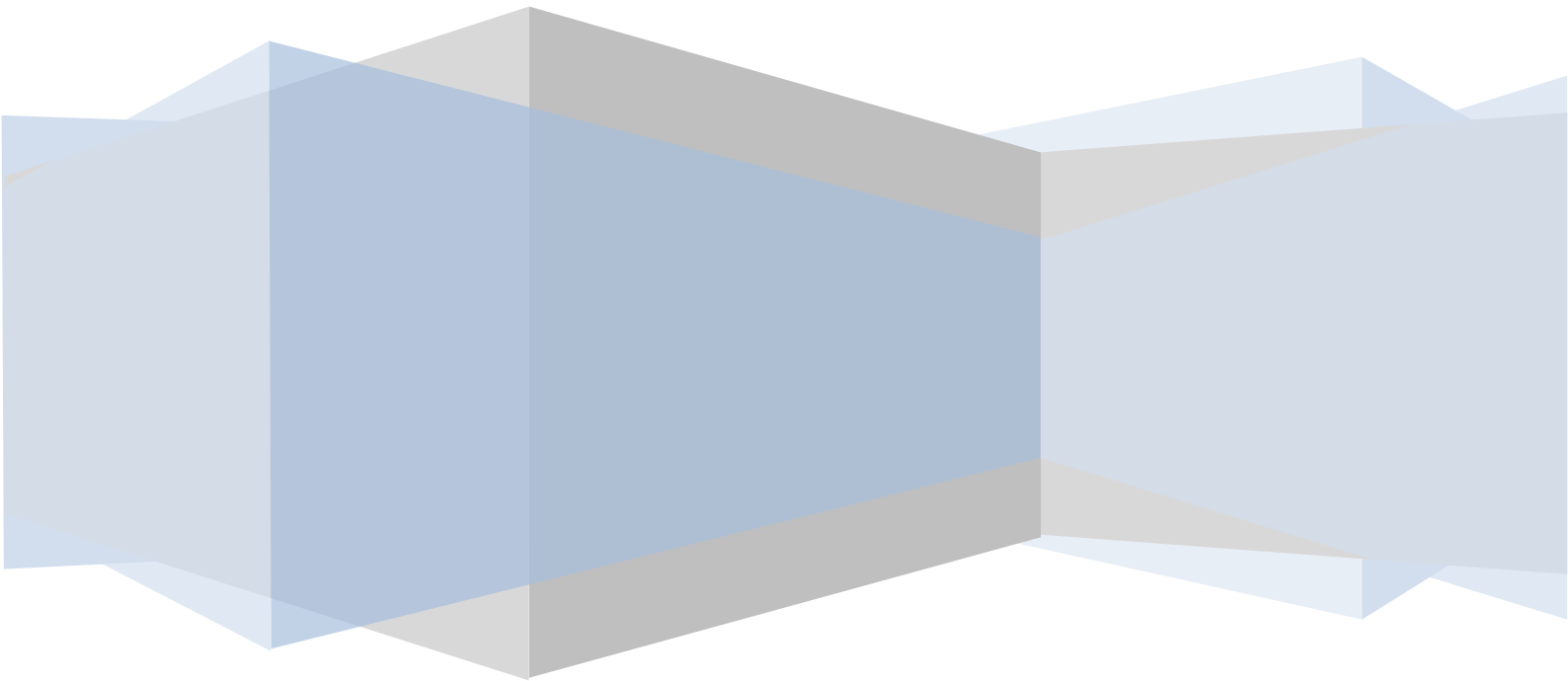
Compact RIO Software

- LabVIEW RT 8.20 and LabVIEW RT 8.5
- Network Variable Engine 1.2 (with LabVIEW 8.20) and 1.4 (with LabVIEW 8.5)
- Variable Client Support 1.0 (with LabVIEW 8.20) and 1.4 (with LabVIEW 8.5)

Legal

CONCLUSIONES Y RECOMENDACIONES

Valoración de las Prácticas



CONCLUSIONES

- ✓ Antes de comenzar a programar un equipo de electrónica compleja, se debe de estudiar la información pertinente a él. Para prevenir las condicionantes de peligro, tanto para el Técnico como para el equipo.

- ✓ El manejar un lenguaje de programación no garantiza el saber programar. Esto es, muchas veces, intuitivo en el individuo; y, tiene que desarrollarse sistemáticamente. Con ejercicios aplicativos.

- ✓ El lenguaje Grafcet facilita la programación del autómata, porque no se pierde tiempo en "bloquear" y/o "habilitar" determinado contacto. Conforme se va realizando la secuencia, se van bloqueando y habilitando intrínsecamente. Además, la programación se la puede efectuar en bloques [*networks*], pero considerar que el cambio debe tomarse como *instantáneo*.

- ✓ A pesar que el S7-200 es de gama baja, dentro de la clasificación que hace la Siemens, permite elaborar e implementar importantes aplicaciones de control. Con la modalidad de ampliación, con los módulos adecuados [máximo 9], se logra manipular dispositivos con relativa facilidad.

- ✓ Para comunicar el S7-200 con la HMI es indispensable la utilización de un OPServer. Uno de éstos es el PCAccess, que es fácil de manipular y bastante operativo.

- ✓ El movimiento simulado de algún automatismo, en el HMI, se logra con una secuencia sistemática ordenada de imágenes, que en el WinCC es de 32 imágenes máximo; éstas pueden ser editadas en el "paint" de Windows.

- ✓ El entorno "MicroWin – S7-200 – WinCC" es un ambiente agradable para desarrollar alguna aplicación de control. Facilitando la recogida de datos, obtener historiales del equipo, y la toma de decisiones importantes gerenciales de la empresa.

RECOMENDACIONES

- Antes de entrar a programar una HMI, con cualquier software comercial, se debe de conocer los lenguajes de programación en "C" y "Basic", éstos son fundamentales para determinadas aplicaciones.

- Al finalizar cada práctica se debe de efectuar una "memoria técnica", adjuntando todo lo indispensable que se tuvo que conocer e investigar para el automatismo.

- Antes de adicionar un módulo a la CPU, es aconsejable analizar si la corriente de la CPU abastece o requiere de otra fuente externa. En éste caso, es mejor que la fuente sea regulada y estable en el tiempo; para garantizar la correcta operación del módulo.

- El ordenador que contenga al SCADA deberá funcionar en tiempo real (*determinístico*). Ya que con "*Windows + aplicaciones*" se tiene un retardo en la reacción de la HMI, lo que podría dar falsas indicaciones del equipo en control.

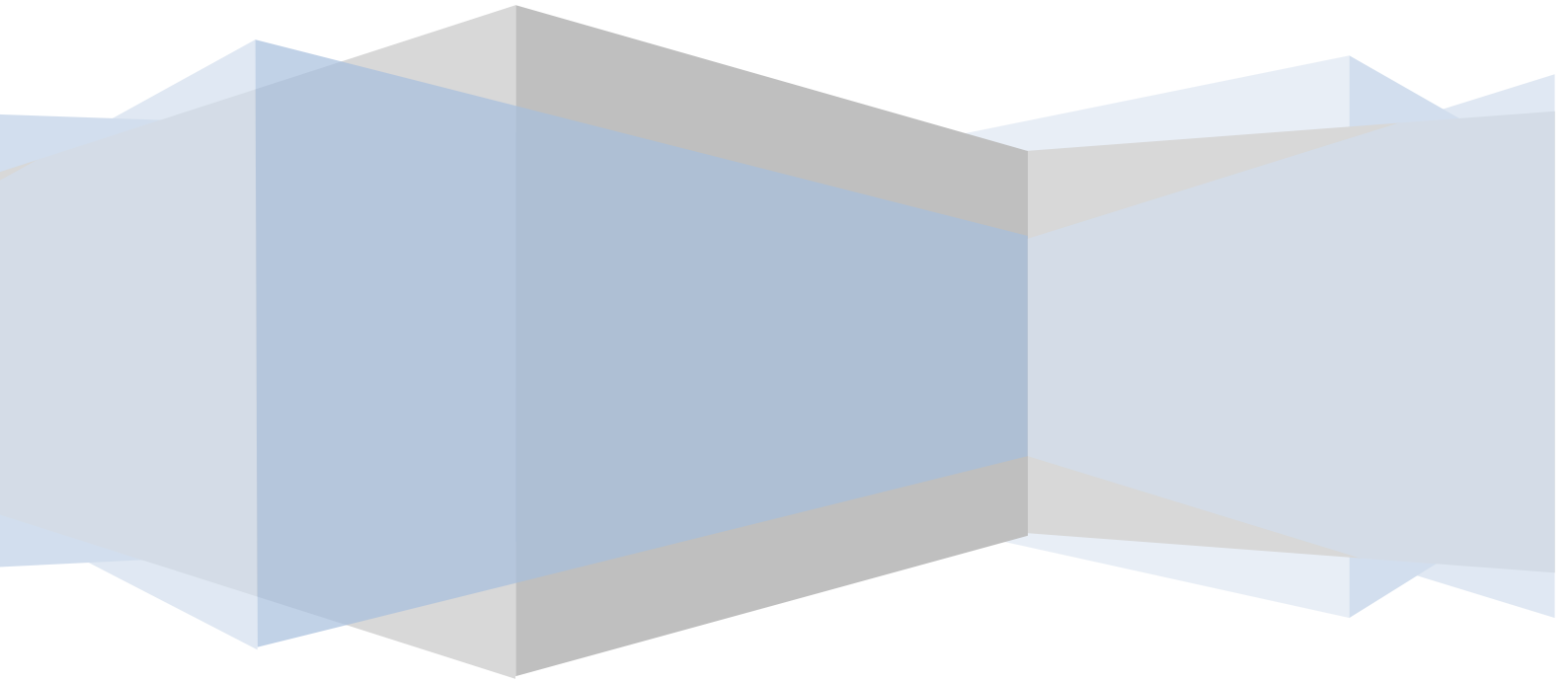
- Elaborar primero un bosquejo, basándose en las condicionantes de operación de la planta. Para luego, escribir el grafcet correspondiente con éstas. Lo que permite realizar bloques de programación, que después se unen y generan el control de la planta; facilitando la implementación y posterior corrección.

- Armar, en lo posible, el programa del MicroWin en forma modular y mediante AWL. Si se desea efectuar todo el sistema en KOP, algunas cosas serían difíciles de programar. Perdiendo tiempo y funcionalidad.

- Seguir en forma secuencial las prácticas planteadas. Lo que permitirá adquirir los conocimientos gradualmente, acorde a las indicaciones del Profesor; como también actualizar y refrescar, si es el caso, aquellos que ya se han aprendido.

BIBLIOGRAFIA

Fuentes Varias de Consulta



REFERENCIAS BIBLIOGRÁFICAS

BERGER Hans, Automating with Simatic, 2nd edition, Germany - Munich, Publicis Corporate, Siemens, 2003.

CANAL WinCC OPC, Impresión de la ayuda OnLine Canal OPC, Germany, Siemens, Marzo – 2004.

CARULLA Miguel y **LLADONOSA** Vicent, Circuitos de Neumática Básica, España, Editor Grupo AlfaOmega, 1993.

GONZALEZ Rueda Emilio, Programación de Autómatas Simatic S7-300, Lenguaje AWL, 1era. Edición, Madrid, Ediciones Ceysa, 2004.

JONES C.T., Step 7 in 7 Steps, A practical guide to implementing S7-300/S7-400 Programmable Logic Controllers, Brilliant Training, USA, 1rst edition, 2006.

KUO Benjamín, Sistemas de Control Automático, 3era. Edición, México, Prentice Hall, 1998.

MARTINEZ Torres José, WinCC V4.02 Manual de Programación, España, SIEMENS PS, 1999.

PERES M. y **LAUBWALD** E., CUCEI – Universidad de Guadalajara, paper sobre Sistemas de Depósitos Acoplados, Mexico , 2005.

SERVICE &SUPPORT, S7 Communication between S7-200 and S7-300/400, Germany, Siemens, 2008.

SICK AG, Manual del Sensor Ultrasónico D30, Alemania, 2003.

SIEMENS AG, Manual del Sistema, edición junio – 2004, Nuremberg – Germany, Nº de referencia 6ES7298 – 8FA24 – 8DH0.

SIEMENS AG, Procesador de Comunicaciones para Industrial Ethernet CP243-1, edición marzo – 2004, Nuremberg – Germany, Nº de referencia J31069 – D0428 – U001 – A2 – 7818.

SIEMENS AG, Simatic HMI: WinCC V7.0 System Description, edition september – 2008, Nuremberg – Germany, Number´s reference A1900 – L531 – B996 – X – 7600.

SIEMENS AG, Simatic HMI: WinCC V6 Documentación Estándar, edición abril – 2003, Nuremberg – Germany, Nº de referencia 6AV6594 – 1MA06 – 1AE0.

SIEMENS AG, Simatic S7-GRAPH para S7-300 / 400 Programación de Controles Secuenciales – Primeros Pasos, Nuremberg – Germany, edición octubre – 2002, Nº de referencia A5E00178399-01.

SIEMENS AG, Microsystem Simatic S7-200: El S7-200 en una hora, edición julio - 1999, Nuremberg – Germany, Nº de referencia 6ZB5310 – 0EG04 – 0BA2.

SIEMENS AG, Microsistema Simatic S7-200, El S7-200 en dos horas, edicion enero – 2000, Nuremberg – Germany, 6ZB5310 – 0FG04 – 0BA2.

SIEMENS AG, Comunicación con Simatic, edición 3, octubre – 1999, Nuremberg – Germany.

SIEMENS AG, Step by Step: Ethernet Communication between OPCServer and S7-200 including CP243-1, Nuremberg – Germany, 2003.

SIMATIC AG, Listado de Instrucciones (AWL) para S7-300 y S7-400, edición 01/ 2004, Nuremberg – Germany, Nº de referencia 6ES7810 – 4CA07 – 8DW1.

UNIVERSIDAD de Buenos Aires, Facultad de Ingeniería, Apuntes varios del Curso Control Automático y Automatización, Argentina, septiembre – 2008.

REFERENCIAS ELECTRÓNICAS

ET_AL, Grafcet (SFC) y Gemma en Internet, España, 2009. Disponible en:

http://www.electrotecnica.org/2009/09/grafcet-sfc-y-gemma-en-internet.html

E.T.S de Ingenieria de Bilbao, WinCC Programación Elemental, España, 2006. Disponible en:

http://www.disa.bi.ehu.es/spanish/ftp/material_asignaturas/Fundamentos%20de%20Automatizaci%F3n%20Industrial/Comunicaciones%20y%20Supervisi%F3n/WinCC.pdf

JIMENES Luis Miguel, Ingeniería de Sistemas Industriales, Ingeniería de Sistemas y Automática, España, período 2009. Varias publicaciones pdf que se manejan en el período de estudios. Disponibles en:

http://www.isa.umh.es/Articulos/Autómatas y Sistemas de Control -(I.Tel.).htm

MATEOS Martin Felipe, Universidad de Oviedo, España, 2007. Ejemplos resueltos sobre la programación del S7-200. Disponibles en:

http://www.isa.uniovi.es/genia/spanish/download/index.htm

MANUAL de referencia rápida: SCADA WinCC, Laboratorio de de Control por Computadora, 4º Curso de Ingenieros en Telecomunicación, España – 2004, Introducción al SCADA WinCC. Disponible en un enlace del foro:

http://www.infoplc.net/Descargas/Descargas_Siemens/Des_SiemensFiles/infoPLC_net_Introduccion_ScadaWinCC.pdf

RODRÍGUEZ José. UPCO-ICAI Departamento de Electrónica y Automática. Metodología de diseño de automatismos secuenciales: Grafcet, España, 2008. Disponible en un enlace del foro:

http://www.infoplc.net/Descargas/Descargas.htm

SALESIANOS, Autómata S7-200, España – 2009

http://www.salesianos.edu/alcoy/juanxxii/dpts/docs/automatas.pdf

SIEMENS, Ejercicios básicos de aplicación a la programación del S7-200: Microsistema Symatic S7-200, Germany, 2002. Disponible en el link del foro:

http://www.infoplc.net/Descargas/Descargas.htm

SIEMENS, WinCC V6 SP2, Germany, 2007. Disponible en:

http://www.relkocz.com/katalogy/siemens/AD4/wincc_v6_sp2_en.pdf

THOMAS J. Burke, OPC Foundation, OPC - OLE for Process Control, Versión 1.0, USA, October – 2008. Disponible en:

http://www.opcfoundation.org/

UNIVERSIDAD de Huelba, Automatización Industrial, Prácticas. 2º I.T.I. Electronica Industrial, España, 2009. Disponible en:

http://www.uhu.es/diego.lopez/Docs_ppal/PractAU2.pdf

UNIVERSIDAD de Oviedo, Informática Industrial: Resumen de Grafcet, España 2008. Disponible en:

http://www.isa.uniovi.es/genia/spanish/publicaciones/grafcet.pdf

VARIOS archivos pdf referentes a manuales del WinCC y de OPC. Disponibles en:

http://pdfdatabase.com/index.php?q=descargar+wincc+opc

REFERENCIA TESIS/TRABAJOS DE GRADUACIÓN

CORONEL P. y PERALTA P., Diseño e Implementación de un Sistema PID para el Control de Nivel de un Tanque Desarrollado con el PLC S7-200, Universidad del Azuay, Cuenca – Ecuador, Octubre – 2009.

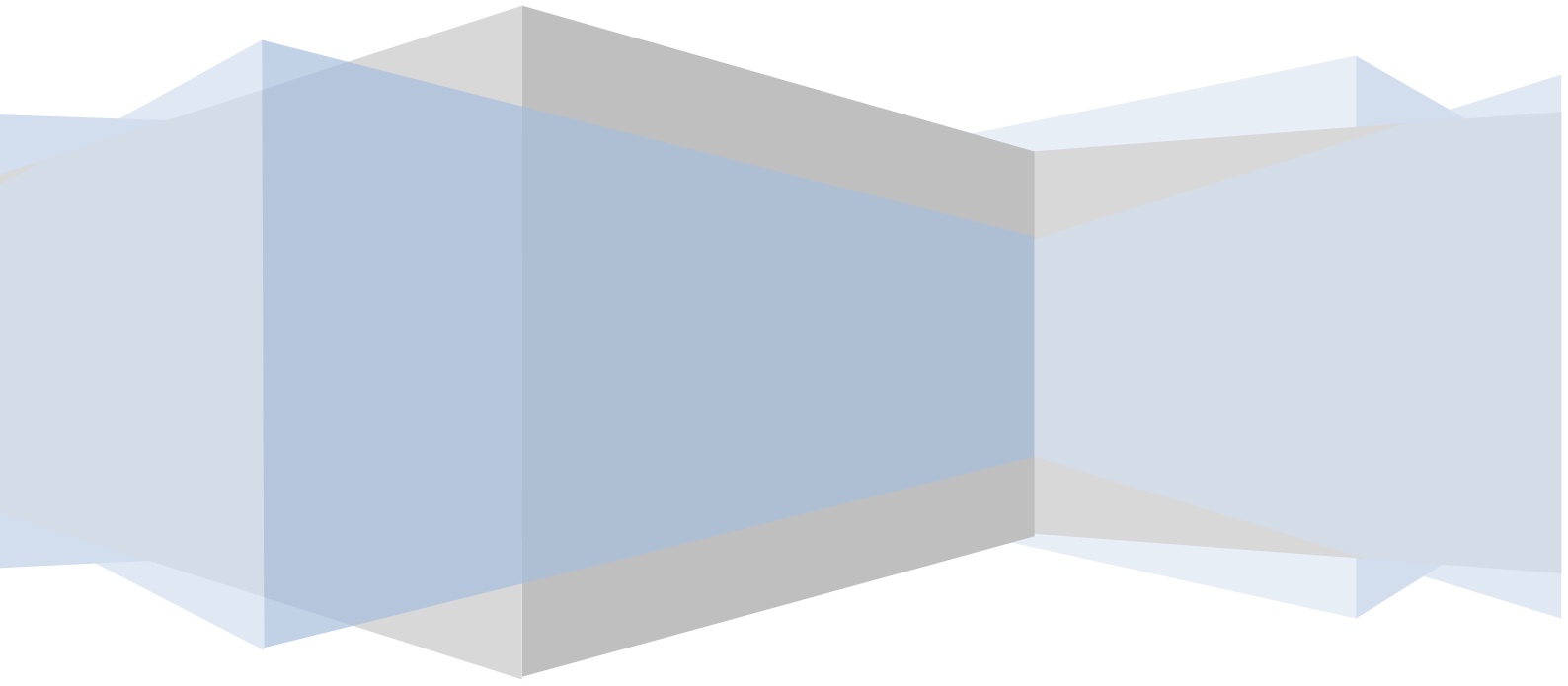
MURILLO G. y QUIROZ R., Diseño Control y Visualización para el Movimiento de un Ascensor usando un PLC y Software de Visualización, Universidad Superior Politécnica del Litoral, Guayaquil – Ecuador, 1999.

SANDOVAL Ramos Alejandro, Sistema de Control de Nivel aplicando un PID en un Modelo de un Distribuidor de Colada Contínua, Instituto Politécnico Nacional – ESIME, Mexico, Junio – 2009.

TORRES del Molino Javier, Gestión Energética y de Seguridad de una Vivienda, Universidad Pontificia Comillas, Madrid – España, junio – 2005.

ANEXOS

Información de Soporte



A1.1 Programación Estructurada

Al realizar un programa, ya sea para un ordenador o para un microcontrolador o un autómata, se tiene una serie de pasos que muchas veces se vuelven cíclicos. Por lo que se hace indispensable un procedimiento para dividir un gran *proceso* en varios *subprocesos*, que faciliten la elaboración del código.

Por lo que la programación estructurada es una técnica que permite programar de forma que se implementen algoritmos para los varios subprocesos, que a la vez puedan ser fácilmente modificados y entendidos por otras personas.

Bien, para diseñar un programa determinado se debe seguir los pasos siguientes:

1. Dependiendo del problema en particular, se debe desarrollar un algoritmo de resolución.
2. Implementar dicho algoritmo lógicamente aplicando ya sea pseudocódigo, flujogramas o lenguaje natural. Para luego someterlo a varias pruebas de escritorio para su depuración.
3. Codificar y documentar la programación. Es importante para futuras correcciones y/o adecuaciones.
4. Compilar el programa para trasladarlo a lenguaje de máquina.
5. Chequear operatividad del programa, analizar el rendimiento.

Importante es que:

La secuencia correcta que deben seguir las instrucciones en un programa, en donde no deben existir inconsistencias en el flujo de datos.

Pero que es un algoritmo:

Es una secuencia ordenada de pasos que permiten resolver una problemática determinada, eso sí con un número de pasos libre de ambigüedades.

En lo cual se debe usar:

- Procesos Secuenciales: ejecutan el verdadero procesamiento de datos.
- Procesos de Selección: es la toma de decisiones en la programación.
- Procesos de Iteración: es la repetición de una acción hasta que se cumpla determinada condición.

A1.2 Elaboración de un algoritmo

Para representar correctamente un algoritmo se debe conocer los símbolos que se emplean:

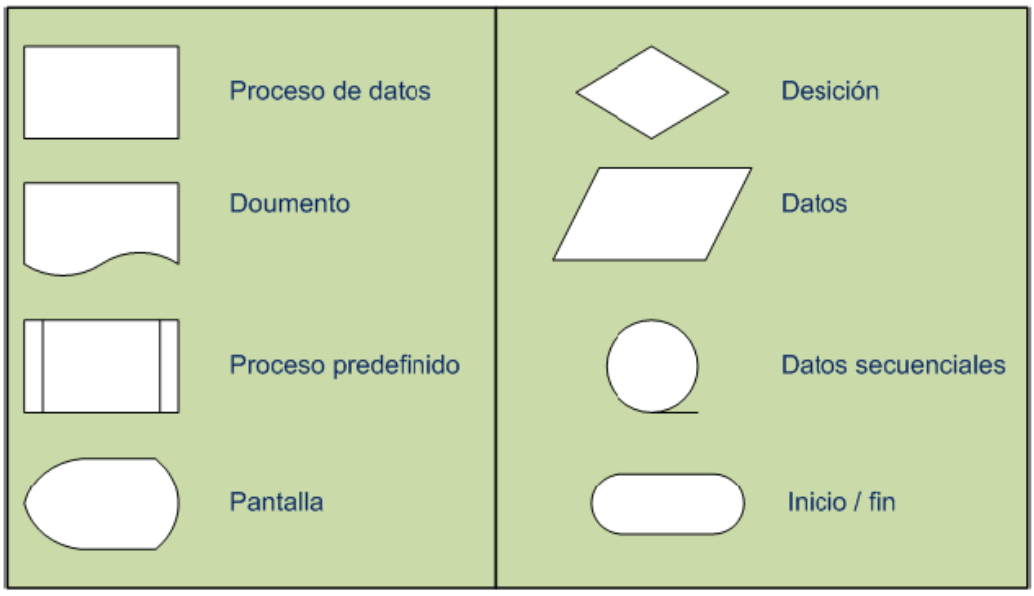


Figura A1.1: Gráficos normalizadas para elaborar un flujograma.

Fuente: Autor.

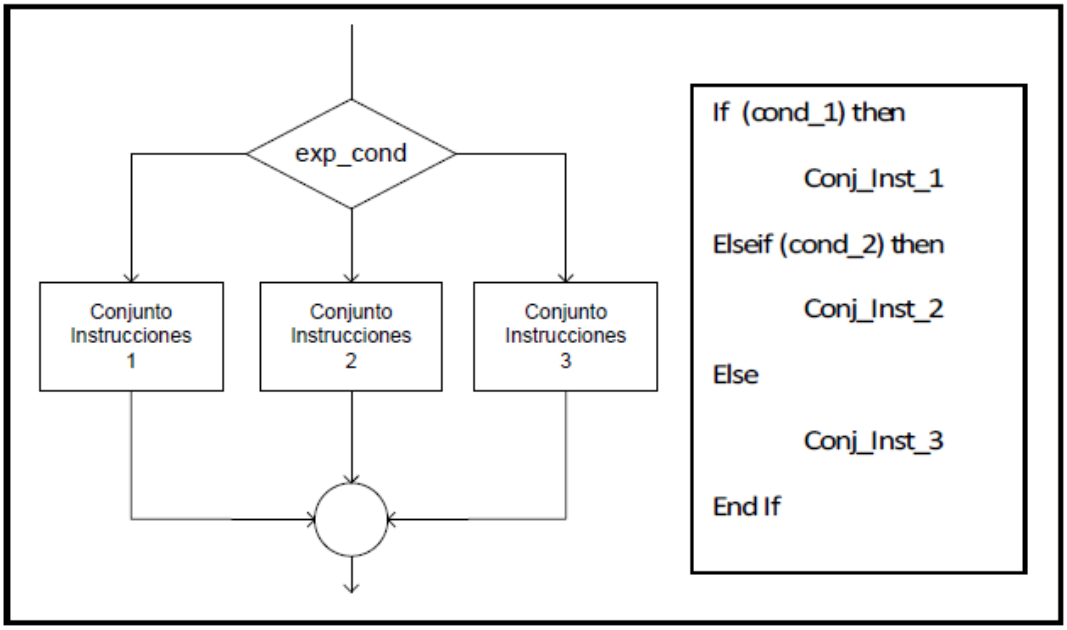
Así también, se debe conocer la reglamentación adecuada:

1. Emplear correctamente la simbología para las diferentes etapas del proceso.
2. Se usará palabras cortas y sencillas para indicar el proceso.

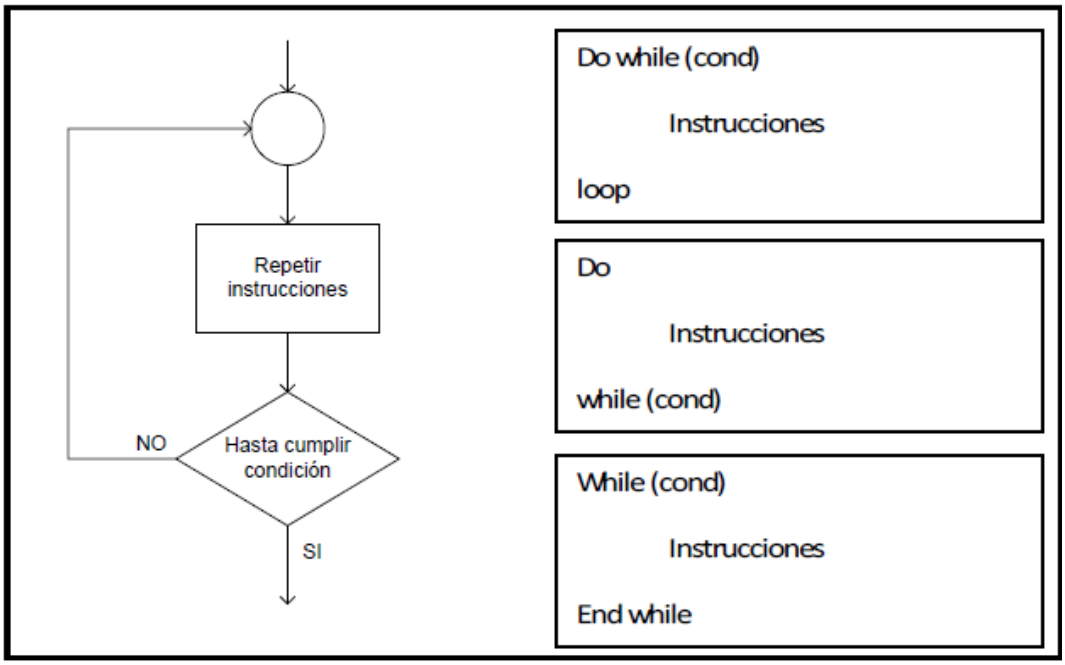
3. El direccionamiento del flujograma se lo realizara con flechas, indicando la secuencia que se debe de seguir en forma lógica y clara.
4. Al usar un condicional, siempre se tendrá dos opciones. De las cuales se deben escoger y analizar todos los casos.
5. Siempre debe de existir un inicio y un final en el diagrama.
6. Usar conectores para retornar a un determinado lugar del programa.
7. Nunca debe quedar abierto un conector.
8. Los bucles deben de cumplir siempre la condición expresada.
9. El diagrama debe ser válido para todos los casos posibles, y no para uno en particular.
10. Comprobar por medio de "pruebas de escritorio" la validez del diagrama.

A1.3 Estructuras de Control

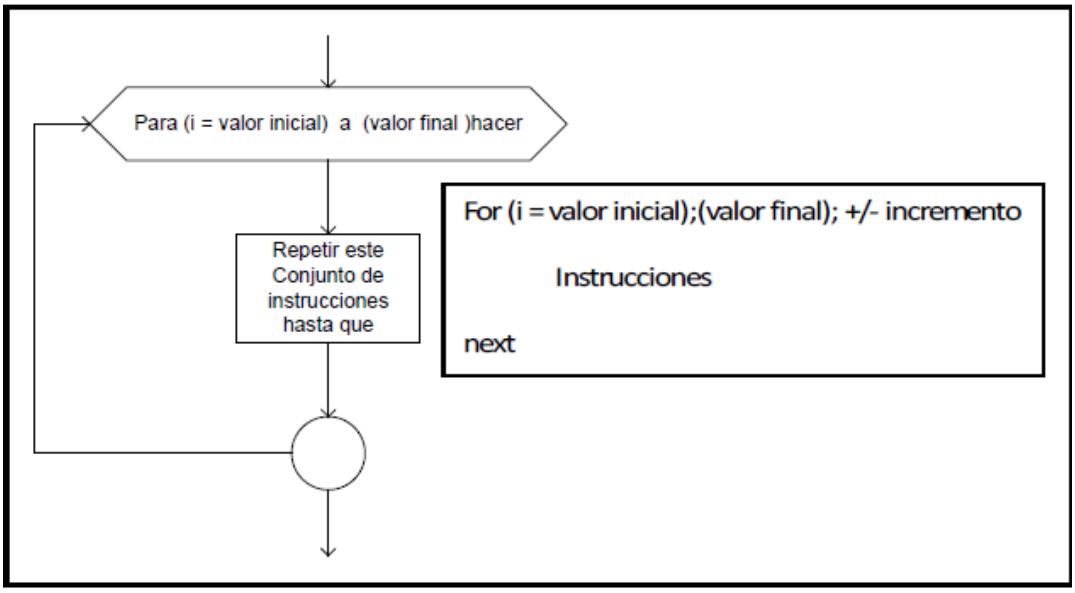
A1.3.1 Estructura Múltiple *If… elseif…end if*



A1.3.2 Estructura *Repetir hasta que…*



A1.3.3 Estructura *Para <inicio> ; <incremento>; a <valor_final> hacer…*



A2.1 Descripción

Es el encargado de realizar las comunicaciones del S7-200 a una "Industrial Ethernet"; con otro S7-200 o con un sistema S7-300 o S7-400. Como también con sistema OPC.

Se lo puede programar directamente con el MicroWin32, por lo que se lo puede reemplazar sin tener que reprogramar. Esto porque la secuencia de control se almacena en el S7-200.

La disposición física de las partes es la siguiente:

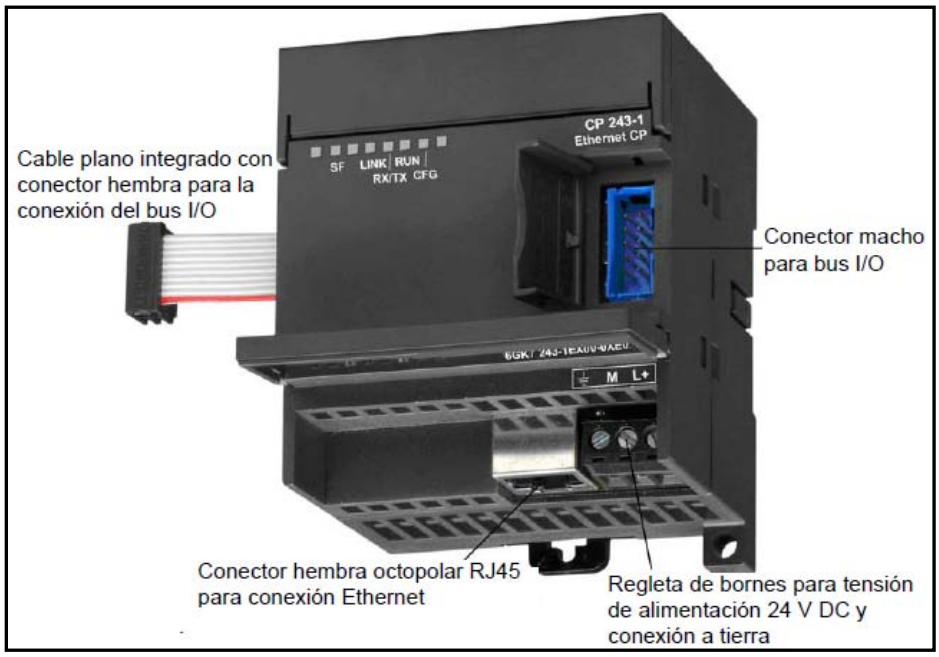


Figura A2.1: Visualización de un módulo CP243-1, y el posicionamiento de las partes que intervienen en una conexión.

Fuente: Siemens, Procesador de Comunicaciones para Industrial Ethernet, Germany, 2004.

Indicador LED	Color	Significado
SF	Rojo, luz permanente	Error de sistema: Luce cuando se ha producido un error
	Rojo, intermitente	Error de sistema: Luce intermitentemente (intervalo: aprox. 1 segundo) si la configuración es errónea y no se puede encontrar un servidor BOOTP.
LINK	Verde, luz permanente	Enlace vía interface RJ45: Se ha establecido el enlace con Ethernet
RX/TX	Verde, centelleante	Actividad de Ethernet: Se están enviando o recibiendo datos vía Ethernet **Nota**: Un paquete recibido vía Ethernet no tiene por qué ir dirigido siempre al CP 243-1. El CP 243-1 acepta en primer lugar cada paquete que se haya transmitido por Ethernet; luego decide si el paquete va dirigido a él o no. El LED RX/TX luce intermitentemente también cuando el cable Ethernet está desenchufado en cuanto el CP 243-1 intenta enviar un paquete.
RUN	Verde, luz permanente	Listo para funcionar: El CP 243-1 está listo para la comunicación
CFG	Amarillo, luz permanente	Configuración: Luce cuando STEP 7-Micro/WIN 32 está manteniendo activamente un enlace con la CPU del CPU del S7-200 a través del CP 243-1

Tabla A2.1: Significado de los leds.

Fuente: Siemens, Procesador de Comunicaciones para Industrial Ethernet, Germany, 2004.

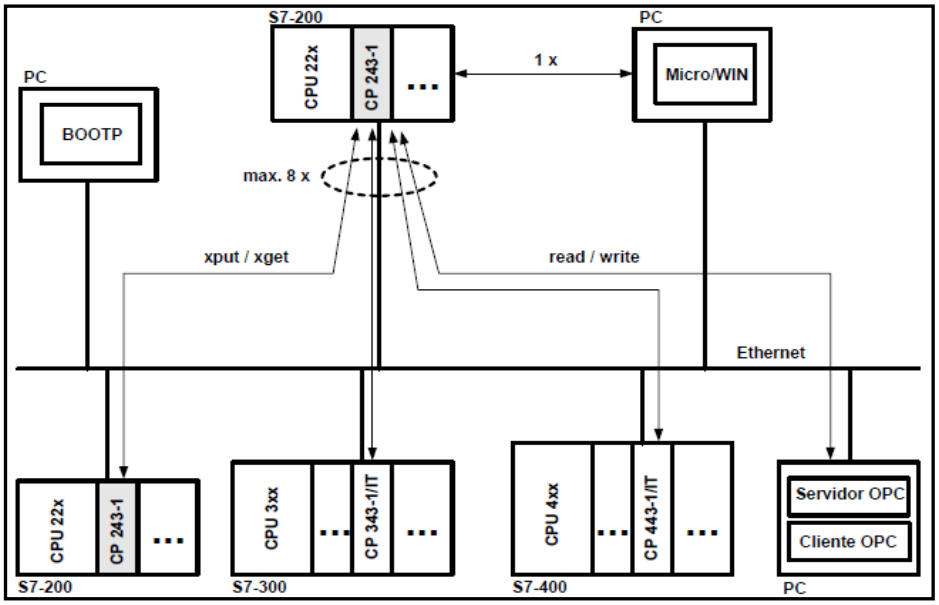


Figura A2.2: Forma de conexión en una network (según catalogo J31069-D0428-U001-A2-7818).

Fuente: Siemens, Procesador de Comunicaciones para Industrial Ethernet, Germany, 2004.

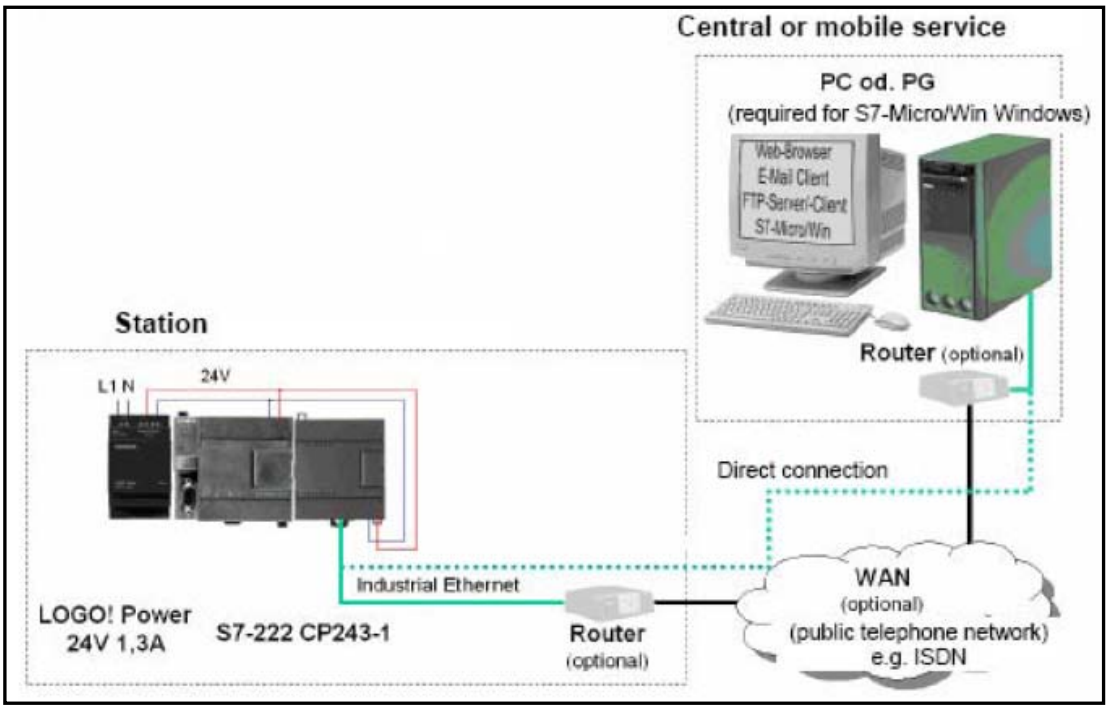


Figura A2.3: Forma de conexión en una network real.

Fuente: Siemens.- MicroAutomation Set 15.

A3.1 Datos del cable de interfaz PPI

Data Sheet
SIMATIC S7-200 RS-232/PPI Multi-Master Cable and S7-200 USB/PPI Multi-Master Cable

Thank you for purchasing the SIMATIC S7-200 Multi-Master PPI Cable.

The S7-200 RS-232/PPI Multi-Master Cable comes factory set for optimal performance with the STEP 7–Micro/WIN 3.2 Service Pack 4 (or later) programming package. The factory setting for this cable is different than for the PC/PPI cables. Refer to Figure 1 to configure the cable for your application.

You can configure the S7-200 RS-232/PPI Multi-Master Cable to operate the same as the PC/PPI cable and to be compatible with any version of a STEP 7–Micro/WIN programming package by setting Switch 5 to the PPI/Freeport setting and then selecting your required baud rate.

The USB cable requires STEP 7–Micro/WIN 3.2 Service Pack 4 (or later) programming package for operation.

Table 1 Specifications

Description Order Number	S7-200 RS-232/PPI Multi-Master Cable 6ES7 901-3CB30-0XA0	S7-200 USB/PPI Multi-Master Cable 6ES7-901-3DB30-0XA0
General Characteristics		
Supply voltage	14.4 to 28.8 VDC	14.4 to 28.8 VDC
Supply current at 24 V nominal supply	60 mA RMS max.	50 mA RMS max.
Direction change delay: RS-232 stop bit edge received to RS-485 transmission disabled	–	–
Isolation	RS-485 to RS-232: 500 VDC	RS-485 to USB: 500 VDC
RS-485 Side Electrical Characteristics		
Common mode voltage range	–7 V to +12 V, 1 second, 3 V RMS continuous	–7 V to +12 V, 1 second, 3 V RMS continuous
Receiver input impedance	5.4 K Ω min. including termination	5.4 K Ω min. including termination
Termination/bias	10K Ω to +5 V on B, PROFIBUS pin 3 10K Ω to GND on A, PROFIBUS pin 8	10K Ω to +5 V on B, PROFIBUS pin 3 10K Ω to GND on A, PROFIBUS pin 8
Receiver threshold/sensitivity	+/–0.2 V, 60 mV typical hysteresis	+/–0.2 V, 60 mV typical hysteresis
Transmitter differential output voltage	2 V min. at R_L=100 Ω, 1.5 V min. at R_L=54 Ω	2 V min. at R_L=100 Ω, 1.5 V min. at R_L=54 Ω
RS-232 Side Electrical Characteristics		
Receiver input impedance	3K Ω min.	–
Receiver threshold/sensitivty	0.8 V min. low, 2.4 V max. high 0.5 V typical hysteresis	–
Transmitter output voltage	+/– 5 V min. at R_L=3K Ω	–
USB Side Electrical Characteristics		
Full speed (12 MB/s), Human Interface Device (HID)		
Supply current at 5V	–	50 mA max.
Power down current	–	400 uA max.

Table 2 S7-200 RS-232/PPI Multi-Master Cable – Pin-outs for RS-485 to RS-232 Local Mode Connector

RS-485 Connector Pin-out		RS-232 Local Connector Pin-out	
Pin Number	Signal Description	Pin Number	Signal Description
1	No connect	1	Data Carrier Detect (DCD) (not used)
2	24 V Return (RS-485 logic ground)	2	Receive Data (RD) (output from PC/PPI cable)
3	Signal B (RxD/TxD+)	3	Transmit Data (TD) (input to PC/PPI cable)
4	RTS (TTL level)	4	Data Terminal Ready (DTR)[1]
5	No connect	5	Ground (RS-232 logic ground)
6	No connect	6	Data Set Ready (DSR)[1]
7	24 V Supply	7	Request To Send (RTS) (not used)
8	Signal A (RxD/TxD–)	8	Clear To Send (CTS) (not used)
9	Protocol select	9	Ring Indicator (RI) (not used)

[1] Pins 4 and 6 are connected internally.

Table 3 S7-200 RS-232/PPI Multi-Master Cable – Pin-outs for RS-485 to RS-232 Remote Mode Connector

RS-485 Connector Pin-out		RS-232 Remote Connector Pin-out[1]	
Pin Number	Signal Description	Pin Number	Signal Description
1	No connect	1	Data Carrier Detect (DCD) (not used)
2	24 V Return (RS-485 logic ground)	2	Receive Data (RD) (input to PC/PPI cable)
3	Signal B (RxD/TxD+)	3	Transmit Data (TD) (output from PC/PPI cable)
4	RTS (TTL level)	4	Data Terminal Ready (DTR)[2]
5	No connect	5	Ground (RS-232 logic ground)
6	No connect	6	Data Set Ready (DSR)[2]
7	24 V Supply	7	Request To Send (RTS) (output from PC/PPI cable)
8	Signal A (RxD/TxD–)	8	Clear To Send (CTS) (not used)
9	Protocol select	9	Ring Indicator (RI) (not used)

[1] A conversion from female to male, and a conversion from 9-pin to 25-pin is required for modems.
[2] Pins 4 and 6 are connected internally.

Use the S7-200 RS-232/PPI Multi-Master Cable with STEP 7-Micro/WIN as a replacement for the PC/PPI cable or for Freeport operation

For connection directly to your personal computer:

- Set the PPI/Freeport mode (Switch 5=0)
- Set the baud rate (Switches 1, 2, and 3)
- Set Local (Switch 6=0). The Local setting is the same as setting the PC/PPI cable to DCE.
- Set the 11 Bit (Switch 7=0)

For connection to a modem:

- Set the PPI/Freeport mode (Switch 5=0)
- Set the baud rate (Switches 1, 2, and 3)
- Set Remote (Switch 6=1). The Remote setting is the same as setting the PC/PPI cable to DTE.
- Set the 10 Bit or 11 Bit (Switch 7) to match the number of bits per character setting of your modem.

Use S7-200 RS-232/PPI Multi-Master Cable with STEP 7-Micro/WIN 3.2 Service Pack 4 (or later)

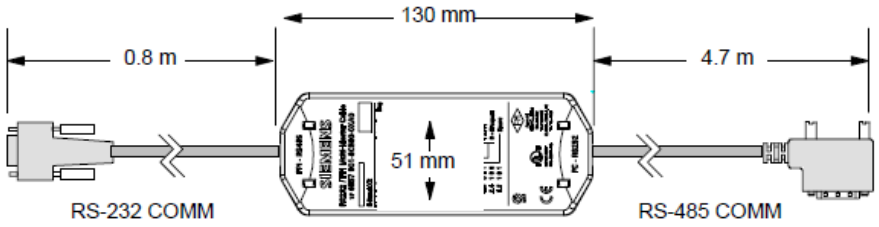
For connection directly to your personal computer:

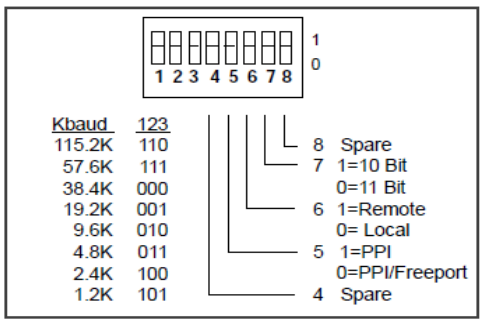
- Set the PPI mode (Switch 5=1)
- Set Local (Switch 6=0)

For connection to a modem:

- Set the PPI mode (Switch 5=1)
- Set Remote (Switch 6=1)

Figure 1 shows the S7-200 RS-232/PPI Multi-Master Cable dimensions, label and LEDs.





LED	Color	Description
Tx	Green	RS-232 transmit indicator
Rx	Green	RS-232 receive indicator
PPI	Green	RS-485 transmit indicator

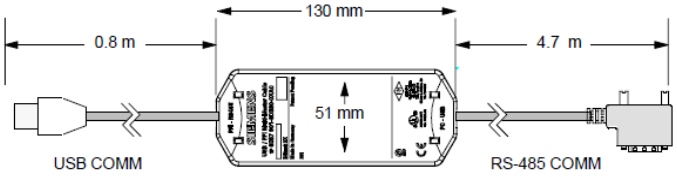
S7-200 USB/PPI Multi-Master Cable

To use the USB cable, you must have STEP 7-Micro/WIN 3.2 Service Pack 4 (or later) installed. The USB cable does not support Freeport communications.

Table 4 S7-200 USB/PPI Multi-Master Cable – Pin-outs for the RS-485 to USB Series "A" Connector

RS-485 Connector Pin-out		USB Connector Pin-out	
Pin Number	Signal Description	Pin Number	Signal Description
1	No connect	1	USB – DataP
2	24 V Return (RS-485 logic ground)	2	USB – DataM
3	Signal B (RxD/TxD+)	3	USB 5V
4	RTS (TTL level)	4	USB logic ground
5	No connect		
6	No connect		
7	24 V Supply		
8	Signal A (RxD/TxD–)		
9	Protocol select (low = 10 bit)		

Figure 2 shows the S7-200 USB/PPI Multi-Master Cable dimensions and LEDs.



LED	Color	Description
Tx	Green	USB transmit indicator
Rx	Green	USB receive indicator
PPI	Green	RS-485 transmit indicator

Figure 2 S7-200 USB/PPI Multi-Master Cable Dimensions and LEDs

A4.1 Juego de Operaciones para la CPU en Simatic.

CPU	221			222			224				226		
Versión	1.0	1.1	1.2	1.00	1.1	1.2	1.0	1.1	1.2	2.0	1.00	1.2	2.0
Operaciones lógicas con bits KOP / FUP (SIMATIC)													
Entradas KOP													
--| |--, --| /|--	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
--| P |--, --| N |--	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
--|NOT|--	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
--|P|--, --|N|--	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Salidas KOP													
--( )	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
--(#)	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
--(S), –(R)	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
--(SI),–(RI)	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
NOP	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SR	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
RS	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Entradas FUP													
AND, OR	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
P,N	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Salidas FUP													
=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
=#	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
S, R	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SI, RI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SR	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
RS	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Operaciones de reloj KOP / FUP (SIMATIC)													
READ_RTC	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SET_RTC	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
READ_RTCX										✓			✓
SET_RTCX										✓			✓
Operaciones de comunicación KOP / FUP (SIMATIC)													
XMT	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
RCV	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
NETR,NETW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
GET_ADDR	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SET_ADDR	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Operaciones de comparación KOP / FUP (SIMATIC)													
Bytes y palabras dobles													
==B, ==D	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
<>B, <>D	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
>=B, >=D	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
<=B, <=D	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
>B, >D	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
<B, <D	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

CPU	221			222			224				226		
Versión	1.0	1.1	1.2	1.00	1.1	1.2	1.0	1.1	1.2	2.0	1.00	1.2	2.0
Palabras													
==I	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
<>I	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
>=I	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
<=I	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
>I	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
<I	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Números reales													
==R	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
<>R	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
>=R	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
<=R	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
>R	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
<R	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Operaciones con cadenas													
--|==S|--			✓			✓			✓	✓		✓	✓
--|<>S|--			✓			✓			✓	✓		✓	✓

Operaciones de conversión KOP / FUP (SIMATIC)													
B_I	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
I_B	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
I_DI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DI_I	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DI_R	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BCD_I	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
I_BCD	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ROUND	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
TRUNC	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ITA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DTA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
RTA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DECO	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ENCO	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ATH, HTA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SEG	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
I_S			✓			✓			✓	✓		✓	✓
DI_S			✓			✓			✓	✓		✓	✓
R_S			✓			✓			✓	✓		✓	✓
S_I			✓			✓			✓	✓		✓	✓
S_DI			✓			✓			✓	✓		✓	✓
S_R			✓			✓			✓	✓		✓	✓

Operaciones de contaje KOP / FUP (SIMATIC)													
CTU	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CTD	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CTUD	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
HDEF, HSC	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
PLS	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

CPU	221			222			224				226		
Versión	1.0	1.1	1.2	1.00	1.1	1.2	1.0	1.1	1.2	2.0	1.00	1.2	2.0
Operaciones aritméticas en coma flotante KOP / FUP (SIMATIC)													
ADD_R	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SUB_R	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MUL_R	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DIV_R	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SQRT	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
PID	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SIN			✓			✓		✓	✓	✓	✓	✓	✓
COS			✓			✓		✓	✓	✓	✓	✓	✓
TAN			✓			✓		✓	✓	✓	✓	✓	✓
LN			✓			✓		✓	✓	✓	✓	✓	✓
EXP			✓			✓		✓	✓	✓	✓	✓	✓
Operaciones aritméticas en coma fija KOP / FUP (SIMATIC)													
ADD_I	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ADD_DI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SUB_I	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SUB_DI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MUL,DIV	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MUL_I	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MUL_DI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DIV_I	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DIV_DI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
INC_B	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
INC_W	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
INC_DW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DEC_B	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DEC_W	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DEC_DW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Operaciones de interrupción KOP / FUP (SIMATIC)													
INT **	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ENI,DISI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
RETI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ATCH,DTCH	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CLR_EVNT										✓			✓
Operaciones lógicas KOP / FUP (SIMATIC)													
INV_B	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
INV_W	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
INV_DW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
WAND_B	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
WAND_W	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
WAND_DW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
WOR_B	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
WOR_W	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
WOR_DW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
WXOR_B	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
WXOR_W	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
WXOR_DW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

CPU	221			222			224				226		
Versión	1.0	1.1	1.2	1.00	1.1	1.2	1.0	1.1	1.2	2.0	1.00	1.2	2.0
Cartucho de memoria KOP / FUP (SIMATIC)													
LOAD_RCP										✓			✓
STORE_RCP										✓			✓
STORE_DL										✓			✓
Operaciones de transferencia KOP / FUP (SIMATIC)													
MOV_B	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MOV_W	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MOV_DW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MOV_R	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLKMOV_B	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLKMOV_W	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BLKMOV_D	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SWAP	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MOV_BIR		✓	✓		✓	✓		✓	✓	✓	✓	✓	✓
MOV_BIW		✓	✓		✓	✓		✓	✓	✓	✓	✓	✓

Operaciones de control del programa KOP / FUP (SIMATIC)													
FOR,NEXT	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
JMP,LBL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SCR	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SCRT	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SCRE	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
RET	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
FIN	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
STOP	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
WDR	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DIAG_LED										✓			✓

CPU	221			222			224				226		
Versión	1.0	1.1	1.2	1.00	1.1	1.2	1.0	1.1	1.2	2.0	1.00	1.2	2.0
Operaciones de desplazamiento y rotación KOP / FUP (SIMATIC)													
SHL_B	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SHL_W	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SHL_DW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SHR_B	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SHR_W	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SHR_DW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ROL_B	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ROL_W	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ROL_DW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ROR_B	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ROR_W	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ROR_DW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SHRB	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Operaciones de tabla KOP / FUP (SIMATIC)													
LIFO,FIFO	✓	✓	✓	✓	✓	✓	✓	✓	✓			✓	✓
AD_T_TBL	✓	✓	✓	✓	✓	✓	✓	✓	✓			✓	✓
FILL_N	✓	✓	✓	✓	✓	✓	✓	✓	✓			✓	✓
TBL_FIND	✓	✓	✓	✓	✓	✓	✓	✓	✓			✓	✓

CPU	221			222			224				226		
Versión	1.0	1.1	1.2	1.00	1.1	1.2	1.0	1.1	1.2	2.0	1.00	1.2	2.0
Operaciones de temporización KOP / FUP (SIMATIC)													
TON	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
TONR	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
TOF	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BGN_ITIME										✓			✓
CAL_ITIME										✓			✓
Operaciones con subrutinas KOP / FUP (SIMATIC)													
SBR	✓	✓	✓	✓	✓	✓	✓	✓	✓			✓	✓
Operaciones con cadenas KOP / FUP (SIMATIC)													
STR_LEN			✓			✓			✓	✓		✓	✓
STR_CPY			✓			✓			✓	✓		✓	✓
SSTR_CPY			✓			✓			✓	✓		✓	✓
STR_CAT			✓			✓			✓	✓		✓	✓
STR_FIND			✓			✓			✓	✓		✓	✓
CHR_STR			✓			✓			✓	✓		✓	✓
--|==S|--			✓			✓			✓	✓		✓	✓
--|<>S|--			✓			✓			✓	✓		✓	✓
I_S			✓			✓			✓	✓		✓	✓
DI_S			✓			✓			✓	✓		✓	✓
R_S			✓			✓			✓	✓		✓	✓
S_I			✓			✓			✓	✓		✓	✓
S_DI			✓			✓			✓	✓		✓	✓
S_R			✓			✓			✓	✓		✓	✓

A4.2 Listado de instrucciones.

CPU	221			222			224				226		
Versión	1.0	1.1	1.2	1.00	1.1	1.2	1.0	1.1	1.2	2.0	1.00	1.2	2.0
Operaciones lógicas con bits AWL (SIMATIC)													
Entradas													
LD[N]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
A[N]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
O[N]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LDI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
AI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
OI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LDNI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ANI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ONI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Pila													
NOT	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
EU,ED	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ALD,OLD	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LPS	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LRD	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LPP	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Salidas													
=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
=I	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
S,R	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SI,RI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
NOP	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
AENO	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LDS	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Operaciones de reloj AWL (SIMATIC)													
TODR	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
TODW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
TODRX										✓			✓
TODWX										✓			✓
Operaciones de comunicación AWL (SIMATIC)													
XMT	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
RCV	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
NETR,NETW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
GPA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SPA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

CPU	221			222			224				226		
Versión	1.0	1.1	1.2	1.00	1.1	1.2	1.0	1.1	1.2	2.0	1.00	1.2	2.0
Operaciones de comparación AWL (SIMATIC)													
Bytes y palabras dobles													
LD[B | D]=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
A[B | D]=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
O[B | D]=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LD[B | D]<>	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
A[B | D]<>	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
O[B | D]<>	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LD[B | D]>=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
A[B | D]>=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
O[B | D]>=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LD[B | D]<=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
A[B | D]<=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
O[B | D]<=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LD[B | D]>	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
A[B | D]>	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
O[B | D]>	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LD[B | D]<	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
A[B | D]<	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
O[B | D]<	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Palabras													
LDW=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
AW=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
OW =	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LDW <>	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
AW <>	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
OW <>	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LDW >=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
AW > =	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
OW > =	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LDW <=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
AW < =	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
OW < =	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LDW >	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
AW >	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
OW >	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LDW <	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
AW <	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
OW <	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

CPU	221			222			224				226		
Versión	1.0	1.1	1.2	1.00	1.1	1.2	1.0	1.1	1.2	2.0	1.00	1.2	2.0
Operaciones de comparación (continuación) AWL (SIMATIC)													
Números reales													
LDR=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
AR=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
OR=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LDR <>	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
AR <>	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
OR<>	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LDR>=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
AR>=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
OR>=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LDR<=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
AR<=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
OR<=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LDR >	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
AR >	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
OR>	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LDR <	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
AR <	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
OR<	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Operaciones con cadenas													
LDS=			✓			✓			✓	✓		✓	✓
AS=			✓			✓			✓	✓		✓	✓
OS=			✓			✓			✓	✓		✓	✓
LDS<>			✓			✓			✓	✓		✓	✓
AS<>			✓			✓			✓	✓		✓	✓
OS<>			✓			✓			✓	✓		✓	✓

Operaciones de conversión AWL (SIMATIC)													
BTI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ITB	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ITD	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DTI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DTR	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ROUND	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
TRUNC	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BCDI,IBCD	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ITA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DTA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
RTA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DECO	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ENCO	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ATH,HTA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SEG	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ITS			✓			✓			✓	✓		✓	✓
DTS			✓			✓			✓	✓		✓	✓
RTS			✓			✓			✓	✓		✓	✓
STI			✓			✓			✓	✓		✓	✓
STD			✓			✓			✓	✓		✓	✓
STR			✓			✓			✓	✓		✓	✓

CPU	221			222			224				226		
Versión	1.0	1.1	1.2	1.00	1.1	1.2	1.0	1.1	1.2	2.0	1.00	1.2	2.0
Operaciones de contaje AWL (SIMATIC)													
CTU	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CTD	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CTUD	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
HDEF,HSC	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
PLS	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

CPU	221			222			224				226		
Versión	1.0	1.1	1.2	1.00	1.1	1.2	1.0	1.1	1.2	2.0	1.00	1.2	2.0
Operaciones aritméticas en coma flotante AWL (SIMATIC)													
+R,-R,*R,/R	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SQRT	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
PID	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SIN			✓			✓		✓	✓	✓	✓	✓	✓
COS			✓			✓		✓	✓	✓	✓	✓	✓
TAN			✓			✓		✓	✓	✓	✓	✓	✓
LN			✓			✓		✓	✓	✓	✓	✓	✓
EXP			✓			✓		✓	✓	✓	✓	✓	✓
Operaciones aritméticas en coma fija AWL (SIMATIC)													
+I,+D,-I,-D	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MUL,DIV	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
*I, *D, /I, /D	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
INCB, DECB	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
INCW,DECW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
INCD,DECD	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Operaciones de interrupción AWL (SIMATIC)													
INT **	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CRETI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
RETI **	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ENI,DISI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ATCH,DTCH	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CEVNT										✓			✓
Operaciones lógicas AWL (SIMATIC)													
INVB	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
INVW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
INVD	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ANDB	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ANDW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ANDD	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ORB	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ORW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ORD	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
XORB	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
XORW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
XORD	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

CPU	221			222			224				226		
Versión	1.0	1.1	1.2	1.00	1.1	1.2	1.0	1.1	1.2	2.0	1.00	1.2	2.0
Cartucho de memoria AWL (SIMATIC)													
LRCP										✓			✓
SRCP										✓			✓
SDL										✓			✓

CPU	221			222			224				226		
Versión	1.0	1.1	1.2	1.00	1.1	1.2	1.0	1.1	1.2	2.0	1.00	1.2	2.0
Operaciones de transferencia AWL (SIMATIC)													
MOVB	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MOVW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MOVD	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MOVR	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BMB	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BMW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BMD	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SWAP	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
RBI		✓	✓		✓	✓		✓	✓	✓	✓	✓	✓
WBI		✓	✓		✓	✓		✓	✓	✓	✓	✓	✓
Operaciones de control del programa AWL (SIMATIC)													
FOR,NEXT	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
JMP,LBL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LSCR	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SCRT	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SCRE	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CSCRE			✓			✓			✓	✓		✓	✓
CRET	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
RET **	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
FIN	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MEND **	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
STOP	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
WDR	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DLED										✓			✓
Operaciones de desplazamiento y rotación AWL (SIMATIC)													
SLB,SRB	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SLW,SRW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SLD,SRD	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
RLB,RRB	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
RLW,RRW	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
RLD,RRD	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SHRB	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Operaciones de tabla AWL (SIMATIC)													
FILL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ATT	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
FND =	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
FND <>	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
FND >	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
FND <	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LIFO,FIFO	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Operaciones de temporización AWL (SIMATIC)													
TON	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
TONR	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
TOF	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BITIM										✓			✓
CITIM										✓			✓
Operaciones con subrutinas AWL (SIMATIC)													
SBR **	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CALL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

CPU	221			222			224				226		
Versión	1.0	1.1	1.2	1.00	1.1	1.2	1.0	1.1	1.2	2.0	1.00	1.2	2.0
Operaciones con cadenas AWL (SIMATIC)													
SLEN			✓			✓			✓	✓		✓	✓
SCPY			✓			✓			✓	✓		✓	✓
SSCPY			✓			✓			✓	✓		✓	✓
SCAT			✓			✓			✓	✓		✓	✓
SFND			✓			✓			✓	✓		✓	✓
CFND			✓			✓			✓	✓		✓	✓
LDS=			✓			✓			✓	✓		✓	✓
AS=			✓			✓			✓	✓		✓	✓
OS=			✓			✓			✓	✓		✓	✓
LDS<>			✓			✓			✓	✓		✓	✓
AS<>			✓			✓			✓	✓		✓	✓
OS<>			✓			✓			✓	✓		✓	✓
ITS			✓			✓			✓	✓		✓	✓
DTS			✓			✓			✓	✓		✓	✓
RTS			✓			✓			✓	✓		✓	✓
STI			✓			✓			✓	✓		✓	✓
STD			✓			✓			✓	✓		✓	✓
STR			✓			✓			✓	✓		✓	✓

A5.1 Área de memoria y funciones del S7-200.

Descripción	CPU 221	CPU 222	CPU 224	CPU 224XP	CPU 226
Tamaño del programa de usuario con edición en modo RUN sin edición en modo RUN	 4096 bytes 4096 bytes	 4096 bytes 4096 bytes	 8192 bytes 12288 bytes	 12288 bytes 16384 bytes	 16384 bytes 24576 bytes
Tamaño de los datos de usuario	2048 bytes	2048 bytes	8192 bytes	10240 bytes	10240 bytes
Imagen del proceso de las entradas	I0.0 a I15.7	I0.0 a I15.7	I0.0 a I15.7	I0.0 a I15.7	I0.0 a I15.7
Imagen del proceso de las salidas	Q0.0 a Q15.7	Q0.0 a Q15.7	Q0.0 a Q15.7	Q0.0 a Q15.7	Q0.0 a Q15.7
Entradas analógicas (sólo lectura)	AIW0 a AIW30	AIW0 a AIW30	AIW0 a AIW62	AIW0 a AIW62	AIW0 a AIW62
Salidas analógicas (sólo escritura)	AQW0 a AQW30	AQW0 a AQW30	AQW0 a AQW62	AQW0 a AQW62	AQW0 a AQW62
Memoria de variables (V)	VB0 a VB2047	VB0 a VB2047	VB0 a VB8191	VB0 a VB10239	VB0 a VB10239
Memoria local (L)[1]	LB0 a LB63	LB0 a LB63	LB0 a LB63	LB0 a LB63	LB0 a LB63
Área de marcas (M)	M0.0 a M31.7	M0.0 a M31.7	M0.0 a M31.7	M0.0 a M31.7	M0.0 a M31.7
Marcas especiales (SM)	SM0.0 a SM179.7	SM0.0 a SM299.7	SM0.0 a SM549.7	SM0.0 a SM549.7	SM0.0 a SM549.7
Sólo lectura	SM0.0 a SM29.7	SM0.0 a SM29.7	SM0.0 a SM29.7	SM0.0 a SM29.7	SM0.0 a SM29.7
Temporizadores	256 (T0 a T255)	256 (T0 a T255)	256 (T0 a T255)	256 (T0 a T255)	256 (T0 a T255)
Retardo a la conexión con memoria					
1 ms	T0, T64	T0, T64	T0, T64	T0, T64	T0, T64
10 ms	T1 a T4 y T65 a T68	T1 a T4 y T65 a T68	T1 a T4 y T65 a T68	T1 a T4 y T65 a T68	T1 a T4 y T65 a T68
100 ms	T5 a T31 y T69 a T95	T5 a T31 y T69 a T95	T5 a T31 y T69 a T95	T5 a T31 y T69 a T95	T5 a T31 y T69 a T95
Retardo a la conexión/desconexión					
1 ms	T32, T96	T32, T96	T32, T96	T32, T96	T32, T96
10 ms	T33 a T36 y T97 a T100	T33 a T36 y T97 a T100	T33 a T36 y T97 a T100	T33 a T36 y T97 a T100	T33 a T36 y T97 a T100
100 ms	T37 a T63 y T101 a T255	T37 a T63 y T101 a T255	T37 a T63 y T101 a T255	T37 a T63 y T101 a T255	T37 a T63 y T101 a T255
Contadores	C0 a C255	C0 a C255	C0 a C255	C0 a C255	C0 a C255
Contadores rápidos	HC0 a HC5	HC0 a HC5	HC0 a HC5	HC0 a HC5	HC0 a HC5
Relés de control secuencial (S)	S0.0 a S31.7	S0.0 a S31.7	S0.0 a S31.7	S0.0 a S31.7	S0.0 a S31.7
Acumuladores	AC0 a AC3	AC0 a AC3	AC0 a AC3	AC0 a AC3	AC0 a AC3
Saltos a metas	0 a 255	0 a 255	0 a 255	0 a 255	0 a 255
Llamadas a subrutinas	0 a 63	0 a 63	0 a 63	0 a 63	0 a 127
Rutinas de interrupción	0 a 127	0 a 127	0 a 127	0 a 127	0 a 127
Detectar flanco positivo/negativo	256	256	256	256	256
Lazos PID	0 a 7	0 a 7	0 a 7	0 a 7	0 a 7
Puertos	Puerto 0	Puerto 0	Puerto 0	Puerto 0, puerto 1	Puerto 0, puerto 1

[1] STEP 7-Micro/WIN (versión 3.0 o posterior) reserva LB60 a LB63.

A5.2 Rangos de los operandos del S7-200.

Tipo de acceso		CPU 221	CPU 222	CPU 224	CPU 224XP	CPU 226
Bit (byte.bit)	I	0.0 a 15.7	0.0 a 15.7	0.0 a 15.7	0.0 a 15.7	0.0 a 15.7
	Q	0.0 a 15.7	0.0 a 15.7	0.0 a 15.7	0.0 a 15.7	0.0 a 15.7
	V	0.0 a 2047.7	0.0 a 2047.7	0.0 a 8191.7	0.0 a 10239.7	0.0 a 10239.7
	M	0.0 a 31.7	0.0 a 31.7	0.0 a 31.7	0.0 a 31.7	0.0 a 31.7
	SM	0.0 a 165.7	0.0 a 299.7	0.0 a 549.7	0.0 a 549.7	0.0 a 549.7
	S	0.0 a 31.7	0.0 a 31.7	0.0 a 31.7	0.0 a 31.7	0.0 a 31.7
	T	0 a 255	0 a 255	0 a 255	0 a 255	0 a 255
	C	0 a 255	0 a 255	0 a 255	0 a 255	0 a 255
	L	0.0 a 63.7	0.0 a 63.7	0.0 a 63.7	0.0 a 63.7	0.0 a 63.7
Byte	IB	0 a 15	0 a 15	0 a 15	0 a 15	0 a 15
	QB	0 a 15	0 a 15	0 a 15	0 a 15	0 a 15
	VB	0 a 2047	0 a 2047	0 a 8191	0 a 10239	0 a 10239
	MB	0 y 31	0 y 31	0 y 31	0 y 31	0 y 31
	SMB	0 a 165	0 a 299	0 a 549	0 a 549	0 a 549
	SB	0 y 31	0 y 31	0 y 31	0 y 31	0 y 31
	LB	0 a 63	0 a 63	0 a 63	0 a 63	0 a 63
	AC	0 a 3	0 a 3	0 a 3	0 a 255	0 a 255
	KB (constante)	KB (constante)	KB (constante)	KB (constante)	KB (constante)	KB (constante)
Palabra	IW	0 a 14	0 a 14	0 a 14	0 a 14	0 a 14
	QW	0 a 14	0 a 14	0 a 14	0 a 14	0 a 14
	VW	0 a 2046	0 a 2046	0 a 8190	0 a 10238	0 a 10238
	MW	0 a 30	0 a 30	0 a 30	0 a 30	0 a 30
	SMW	0 a 164	0 a 298	0 a 548	0 a 548	0 a 548
	SW	0 a 30	0 a 30	0 a 30	0 a 30	0 a 30
	T	0 a 255	0 a 255	0 a 255	0 a 255	0 a 255
	C	0 a 255	0 a 255	0 a 255	0 a 255	0 a 255
	LW	0 a 62	0 a 62	0 a 62	0 a 62	0 a 62
	AC	0 a 3	0 a 3	0 a 3	0 a 3	0 a 3
	AIW	0 a 30	0 a 30	0 a 62	0 a 62	0 a 62
	AQW	0 a 30	0 a 30	0 a 62	0 a 62	0 a 62
	KW (constante)	KW (constante)	KW (constante)	KW (constante)	KW (constante)	KW (constante)
Palabra doble	ID	0 a 12	0 a 12	0 a 12	0 a 12	0 a 12
	QD	0 a 12	0 a 12	0 a 12	0 a 12	0 a 12
	VD	0 a 2044	0 a 2044	0 a 8188	0 a 10236	0 a 10236
	MD	0 a 28	0 a 28	0 a 28	0 a 28	0 a 28
	SMD	0 a 162	0 a 296	0 a 546	0 a 546	0 a 546
	SD	0 a 28	0 a 28	0 a 28	0 a 28	0 a 28
	LD	0 a 60	0 a 60	0 a 60	0 a 60	0 a 60
	AC	0 a 3	0 a 3	0 a 3	0 a 3	0 a 3
	HC	0 a 5	0 a 5	0 a 5	0 a 5	0 a 5
	KD (constante)	KD (constante)	KD (constante)	KD (constante)	KD (constante)	KD (constante)

A6.1 Referencias de las CPUs.

Nº de referencia	Modelo de CPU	Alimentación (nominal)	Entradas digitales	Salidas digitales	Puertos COM	Entradas analógicas	Salidas analógicas	Bloque de terminales extraíble
6ES7 211-0AA23-0XB0	CPU 221	24 V c.c.	6 x 24 V c.c.	4 x 24 V c.c.	1	No	No	No
6ES7 211-0BA23-0XB0	CPU 221	120 a 240 V c.a.	6 x 24 V c.c.	4 salidas de relé	1	No	No	No
6ES7 212-1AB23-0XB0	CPU 222	24 V c.c.	8 x 24 V c.c.	6 x 24 V c.c.	1	No	No	No
6ES7 212-1BB23-0XB0	CPU 222	120 a 240 V c.a.	8 x 24 V c.c.	6 salidas de relé	1	No	No	No
6ES7 214-1AD23-0XB0	CPU 224	24 V c.c.	14 x 24 V c.c.	10 x 24 V c.c.	1	No	No	Sí
6ES7 214-1BD23-0XB0	CPU 224	120 a 240 V c.a.	14 x 24 V c.c.	10 salidas de relé	1	No	No	Sí
6ES7 214-2AD23-0XB0	CPU 224XP	24 V c.c.	14 x 24 V c.c.	10 x 24 V c.c.	2	2	1	Sí
6ES7 214-2BD23-0XB0	CPU 224XP	120 a 240 V c.a.	14 x 24 V c.c.	10 salidas de relé	2	2	1	Sí
6ES7 216-2AD23-0XB0	CPU 226	24 V c.c.	24 x 24 V c.c.	16 x 24 V c.c.	2	No	No	Sí
6ES7 216-2BD23-0XB0	CPU 226	120 a 240 V c.a.	24 x 24 V c.c.	16 salidas de relé	2	No	No	Sí

A6.2 Generales.

Nº de referencia	Nombre y descripción de la CPU	Dimensiones en mm (l x a x p)	Peso	Disipación	Tensión c.c. disponible	
					+5 V c.c.	+24 V c.c.[1]
6ES7 211-0AA23-0XB0	CPU 221 DC/DC/DC 6 entradas/4 salidas	90 x 80 x 62	270 g	3 W	0 mA	180 mA
6ES7 211-0BA23-0XB0	CPU 221 AC/DC/relé 6 entradas/4 salidas de relé	90 x 80 x 62	310 g	6 W	0 mA	180 mA
6ES7 212-1AB23-0XB0	CPU 222 DC/DC/DC 8 entradas/6 salidas	90 x 80 x 62	270 g	5 W	340 mA	180 mA
6ES7 212-1BB23-0XB0	CPU 222 AC/DC/relé 8 entradas/6 salidas de relé	90 x 80 x 62	310 g	7 W	340 mA	180 mA
6ES7 214-1AD23-0XB0	CPU 224 DC/DC/DC 14 entradas/10 salidas	120,5 x 80 x 62	360 g	7 W	660 mA	280 mA
6ES7 214-1BD23-0XB0	CPU 224 AC/DC/relé 14 entradas/10 salidas de relé	120,5 x 80 x 62	410 g	10 W	660 mA	280 mA
6ES7 214-2AD23-0XB0	CPU 224XP DC/DC/DC 14 entradas/10 salidas	140 x 80 x 62	390 g	8 W	660 mA	280 mA
6ES7 214-2BD23-0XB0	CPU 224XP AC/DC/relé 14 entradas/10 salidas de relé	140 x 80 x 62	440 g	11 W	660 mA	280 mA
6ES7 216-2AD23-0XB0	CPU 226 DC/DC/DC 24 entradas/16 salidas	196 x 80 x 62	550 g	11 W	1000 mA	400 mA
6ES7 216-2BD23-0XB0	CPU 226 AC/DC/relé 24 entradas/16 salidas de relé	196 x 80 x 62	660 g	17 W	1000 mA	400 mA

1 Esta es la alimentación de sensores de 24 V c.c. disponible tras tenerse en cuenta la alimentación interna de bobinas de relé y los requisitos de corriente de 24 V c.c. del puerto de comunicación.

	CPU 221	CPU 222	CPU 224	CPU 224XP	CPU 226
Memoria					
Tamaño del programa de usuario (EEPROM) con edición en modo RUN sin edición en modo RUN	 4096 bytes 4096 bytes		 8192 bytes 12288 bytes	 12288 bytes 16384 bytes	 16384 bytes 24576 bytes
Datos de usuario (EEPROM)	2048 bytes (remanentes)		8192 bytes (remanentes)	10240 bytes (remanentes)	10240 bytes (remanentes)
Respaldo (condensador de alto rendimiento) (pila opcional)	Típ. 50 h (mín. 8 h a 40° C) Típ. 200 días		Típ. 100 h (mín. 70 h a 40° C) Típ. 200 días	Típ. 100 horas (mín. 70 horas a 40° C) Típ. 200 días	
Entradas y salidas (E/S)					
E/S de ampliación	6 E/4 S	8 E/6 S	14 E/10 S	14 E/10 S	24 E/16 S
E/S analógicas	Ninguna			2 E/1 S	Ninguna
Tamaño de la imagen de E/S digitales	256 (128 E/128 S)				
Tamaño de la imagen de E/S analógicas	Ninguno	32 (16 E/16 S)	64 (32 E/32 S)		
Nº máx. de módulos de ampliación	Ninguno	2 módulos[1]	7 módulos[1]		
Nº máx. de módulos inteligentes	Ninguno	2 módulos[1]	7 módulos[1]		
Entradas de captura de impulsos	6	8	14		24
Contadores rápidos Fase simple Dos fases	4 contadores (total) 4 a 30 kHz 2 a 20 kHz		6 contadores (total) 6 a 30 kHz 4 a 20 kHz	6 contadores (total) 4 a 30 kHz 2 a 200 kHz 3 a 20 kHz 1 a 100 kHz	6 contadores (total) 6 a 30 kHz 4 a 20 kHz
Salidas de impulsos	2 a 20 kHz (sólo en salidas c.c.)			2 a 100 kHz (sólo en salidas c.c.)	2 a 20 kHz (sólo en salidas c.c.)
Datos generales					
Temporizadores	256 temporizadores en total: 4 temporizadores de 1 ms, 16 temporizadores de 10 ms y 236 temporizadores de 100 ms				
Contadores	256 (respaldo por condensador de alto rendimiento o pila)				
Marcas internas almacenadas al desconectar la CPU	256 (respaldo por condensador de alto rendimiento o pila) 112 (almacenamiento en EEPROM)				
Interrupciones temporizadas	2 con resolución de 1 ms				
Interrupciones de flanco	4 flancos positivos y/o 4 flancos negativos				
Potenciómetros analógicos	1 con resolución de 8 bits		2 con resolución de 8 bits		
Velocidad de ejecución booleana	0,22 µs por operación				
Reloj de tiempo real	Cartucho opcional		Incorporado		
Cartuchos opcionales	Memoria, pila y reloj de tiempo real		Memoria y pila		
Comunicación integrada					
Puertos (potencia limitada)	1 puerto RS-485			2 puertos RS-485	
Velocidades de transferencia PPI, DP/T	9,6, 19,2 y 187,5 kbit/s				
Velocidades de transferencia Freeport	1.2 kbit/s a 115.2 kbit/s				
Longitud máx. del cable por segmento	Con repetidor aislado: 1000 m hasta 187,5 kbit/s, 1200 m hasta 38,4 kbit/s Sin repetidor aislado: 50 m				
Nº máximo de estaciones	32 por segmento, 126 por red				
Nº máximo de maestros	32				
Punto a punto (modo maestro PPI)	Sí (NETR/NETW)				
Enlaces MPI	4 en total, 2 reservados (1 para una PG y 1 para un OP)				

1 Es preciso calcular la corriente necesaria para determinar cuánta energía puede suministrar la CPU S7-200 a la configuración deseada. Si se excede la corriente necesaria para la CPU, es posible que no se pueda conectar el número máximo de módulos. Consulte el anexo A para obtener información acerca de los requisitos de alimentación de la CPU y de los módulos de ampliación, así como el anexo B para calcular la corriente necesaria.

A6.3 Alimentación.

	Corriente continua		Corriente alterna	
Potencia de entrada				
Tensión de entrada	20,4 a 28,8 V c.c.		85 V a 264 V c.a., 47 a 63 Hz	
Intensidad de entrada	CPU sólo a 24 V c.c.	Carga máx. a 24 V c.c.	sólo CPU	Carga máx.
CPU 221	80 mA	450 mA	30/15 mA a 120/240 V c.a.	120/60 mA a 120/240 V c.a.
CPU 222	85 mA	500 mA	40/20 mA a 120/240 V c.a.	140/70 mA a 120/240 V c.a.
CPU 224	110 mA	700 mA	60/30 mA a 120/240 V c.a.	200/100 mA a 120/240 V c.a.
CPU 224XP	120 mA	900 mA	70/35 mA a 120/240 V c.a.	220/100 mA a 120/240 V c.a.
CPU 226	150 mA	1050 mA	80/40 mA a 120/240 V c.a.	320/160 mA a 120/240 V c.a.
Corriente de irrupción	12 A a 28,8 V c.c.		20 A a 264 V c.a.	
Aislamiento (campo a circuito lógico)	Sin aislamiento		1500 V c.a.	
Tiempo de retardo (desde la pérdida de corriente)	10 ms a 24 V c.c.		20/80 ms a 120/240 V c.a.	
Fusible (no reemplazable)	3 A, 250 V, de acción lenta		2 A, 250 V, de acción lenta	
Alimentación de sensores 24 V c.c.				
Tensión de sensores (potencia limitada)	L+ menos 5 V		20,4 a 28,8 V c.c.	
Intensidad límite	1,5 A pico, límite térmico no destructivo (v. tabla A-3, carga nominal)			
Rizado/corriente parásita	Derivado de potencia de entrada		Menos de 1 V pico a pico	
Aislamiento (sensor a circuito lógico)	Sin aislamiento			

A6.4 De las entradas digitales.

Datos generales	Entrada de 24 V c.c. (CPU 221, CPU 222, CPU 224, CPU 226)		Entrada de 24 V c.c. (CPU 224XP)
Tipo de datos	Sumidero de corriente/fuente (tipo 1 IEC con sumidero de corriente)		Sumidero de corriente/fuente (tipo 1 IEC, excepto I0.3 a I0.5)
Tensión nominal	Típ. 24 V c.c. a 4 mA		Típ. 24 V c.c. a 4 mA
Tensión continua máx. admisible	30 V c.c.		
Sobretensión	35 V c.c., 0,5 s		
Señal 1 lógica (mín.)	15 V c.c. a 2,5 mA		15 V c.c. a 2,5 mA (I0.0 a I0.2 e I0.6 a I1.5) 4 V c.c. a 8 mA (I0.3 a I0.5)
Señal 0 lógica (máx.)	5 V c.c. a 1 mA		5 V c.c. a 1 mA (I0.0 a I0.2 e I0.6 a I1.5) 1 V c.c. a 1 mA (I0.3 a I0.5)
Retardo de entrada	Seleccionable (0,2 a 12,8 ms)		
Conexión de sensor de proximidad de 2 hilos (Bero)			
Corriente de fuga admisible (máx.)	1 mA		
Aislamiento (campo a circuito lógico)	Sí		
Separación galvánica	500 V c.a., 1 minuto		
Grupos de aislamiento	Consulte el diagrama de cableado		
Frecuencia de entrada de los contadores rápidos (HSC)			
Entradas HSC	Señal 1 lógica	Fase simple	Dos fases
Todos los HSC	15 a 30 V c.c.	20 kHz	10 kHz
Todos los HSC	15 a 26 V c.c.	30 kHz	20 kHz
HC4, HC5 (sólo CPU 224XP)	> 4 V c.c.	200 kHz	100 kHz
Entradas ON simultáneamente	Todas		Todas Sólo CPU 224XP AC/DC/relé: Todas a 55° C con entradas c.c a 26 V c.c. máx. Todas a 50° C con entradas c.c a 30 V c.c. máx.
Longitud del cable (máx.)			
Apantallado	500 m para las entradas normales, 50 m para las entradas HSC[1]		
No apantallado	300 m para las entradas normales		

1 Para las entradas HSC se recomienda utilizar cables apantallados de par trenzado.

A6.5 De las salidas digitales.

Datos generales	Salida de 24 V c.c. (CPU 221, CPU 222, CPU 224, CPU 226)	Salida de 24 V c.c. (CPU 224XP)	Salidas de relé
Tipo de datos	Estado sólido-MOSFET[1] (fuente)		Contacto de baja potencia
Tensión nominal	24 V c.c.	24 V c.c.	24 V c.c. ó 250 V c.a.
Rango de tensión	20,4 a 28,8 V c.c.	5 a 28,8 V c.c. (Q0.0 a Q0.4) 20,4 a 28,8 V c.c. (Q0.5 a Q1.1)	5 a 30 V c.c. ó 5 a 250 V c.a.
Sobreintensidad (máx.)	8 A, 100 ms		5 A durante 4 s c/u 10% de ciclo de trabajo
Señal 1 lógica (mín.)	20 V c.c. a intensidad máx.	L+ menos 0,4 V a intensidad máx.	-
Señal 0 lógica (máx.)	0,1 V c.c. con 10 K Ω de carga		-
Intensidad nominal por salida (máx.)	0,75 A		2,0 A
Intensidad nominal por neutro (máx.)	6 A	3,75 A	10 A
Corriente de fuga (máx.)	10 µ A		-
Carga de lámparas (máx.)	5 W		30 W c.c.; 200 W c.a.[3, 4]
Tensión de bloqueo inductiva	L+ menos 48 V c.c., disipación de 1 W		-
Resistencia en estado ON (contactos)	Típ. 0,3 Ω (0.6 Ω máx.)		0,2 Ω (máx. si son nuevas)
Separación galvánica Separación galvánica (campo a circuito lógico) Circuito lógico a contacto Resistencia (circuito lógico a contacto) Grupos de aislamiento	 500 V c.a., 1 minuto - - Consulte el diagrama de cableado		 - 1500 V c.a., 1 minuto 100 MΩ Consulte el diagrama de cableado
Retardo (máx.) OFF a ON (µs) ON a OFF (µs) Conmutación	 2µs (Q0.0, Q0.1), 15µs (todas las demás) 10µs (Q0.0, Q0.1), 130µs (todas las demás) -	 0,5µs (Q0.0, Q0.1), 15µs (todas las demás) 1,5µs (Q0.0, Q0.1), 130µs (todas las demás) -	 - 10 ms
Frecuencia de impulsos (máx.)	20 kHz[2] (Q0.0 y Q0.1)	100 kHz[2] (Q0.0 y Q0.1)	1 Hz
Vida útil mecánica	-	-	10.000.000 (sin carga)
Vida útil de los contactos	-	-	100.000 (carga nominal)
Salidas ON simultáneamente	Todas a 55° C (horizontal), todas a 45° C (vertical)		
Conexión de dos salidas en paralelo	Si, sólo salidas de un mismo grupo		No
Longitud del cable (máx.) Apantallado No apantallado	 500 m 150 m		

1 Cuando un contacto mecánico aplica tensión a una CPU S7-200, o bien a un módulo de ampliación digital, envía una señal "1" a las salidas digitales durante aproximadamente 50 microsegundos. Considere ésto especialmente si desea utilizar aparatos que reaccionen a impulsos de breve duración.

2 En función del receptor de impulsos y del cable, un resistor de carga externo (al menos 10% de la intensidad nominal) puede mejorar la calidad de señal de los impulsos y la inmunidad a interferencias.

3 La vida útil de los relés con carga de lámparas se reducirá en 75%, a menos que la sobrecorriente al conectar se reduzca por debajo de la sobrecorriente límite de la salida.

4 El vatiaje límite de la carga de lámparas es aplicable a la tensión nominal. Reduzca el vatiaje límite proporcionalmente a la tensión conmutada (p. ej. 120 V c.a. - 100 W).

A6.6 Cálculo de la corriente

Todas las CPUs S7-200 ofrecen alimentación tanto en 5 V c.c. como 24 V c.c.:

- Todas las CPUs disponen de una fuente de alimentación para sensores de 24 V c.c. que puede suministrar tensión para las entradas locales o para las bobinas de relés en los módulos de ampliación. Si el consumo de 24 V c.c. excede la corriente que puede aportar la CPU, entonces puede agregarse una fuente de alimentación externa de 24 V c.c. para abastecer con 24 V c.c. a los módulos de ampliación. La alimentación de 24 V c.c. se debe conectar manualmente a esas entradas o bobinas de relé.
- La CPU alimenta también con 5 V c.c. los módulos de ampliación cuando se conectan al módulo base. Si el consumo de 5 V c.c. de los módulos de ampliación excede la corriente que puede aportar la CPU, entonces es necesario desconectar tantos módulos de ampliación como sean necesarios para no superar la corriente suministrable por la CPU.

Las hojas de datos técnicos que se incluyen en el anexo A informan sobre las corrientes suministrables por las CPUs y sobre el consumo de los módulos de ampliación.

La tabla B-1 muestra un ejemplo de cálculo de los requisitos de alimentación de un S7-200 compuesto de los módulos siguientes:

- CPU S7-200 224 AC/DC/relé
- 3 módulos de ampliación EM 223, 8 entradas DC / 8 salidas de relé
- 1 módulo de ampliación EM 221, 8 entradas DC

Esta configuración tiene un total de 46 entradas y 34 salidas.

La CPU S7-200 de este ejemplo suministra suficiente corriente (5 V c.c.) a los módulos de ampliación, pero la alimentación de sensores no suministra suficiente corriente de 24 V c.c. a todas las entradas y salidas de relé. Las E/S requieren 400 mA, pero la CPU S7-200 sólo puede suministrar 280 mA. Para esta configuración se necesita una fuente adicional de alimentación de 120 mA (como mínimo) a 24 V c.c. para que todas las entradas y salidas puedan funcionar correctamente.

Tabla B-1 Cálculo de requisitos de alimentación en una configuración de ejemplo

Corriente de la CPU	**5 V c.c.**	**24 V c.c.**
CPU 224 AC/DC/relé	660 mA	280 mA

menos

Consumo del sistema	**5 V c.c.**	**24 V c.c.**
CPU 224, 14 entradas		14 * 4 mA = 56 mA
3 EM 223, alimentación necesaria de 5 V	3 * 80 mA = 240 mA	
1 EM 221, alimentación necesaria de 5 V	1 * 30 mA = 30 mA	
3 EM 223, 8 entradas c/u		3 * 8 * 4 mA = 96 mA
3 EM 223, 8 salidas de relé c/u		3 * 8 * 9 mA = 216 mA
1 EM 221, 8 entradas		8 * 4 mA = 32 mA
Consumo total	270 mA	400 mA

igual a

Balance de corriente	**5 V c.c.**	**24 V c.c.**
Balance total de corriente	390 mA	[120 mA]