ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

ESCUELA DE INGENIERIA EN SISTEMAS

"PROPUESTA METODOLÓGICA PARA LA PROGRAMACIÓN DE PLC EN GRAFCET PARA LAS COMPETENCIAS DE MECATRÓNICA WORLD SKILL, CASO PRÁCTICO LABORATORIO DE MECATRÓNICA (EIS)"

TESIS DE GRADO

Previa obtención del Título de

INGENIERO EN SISTEMAS INFORMÁTICOS

Presentado por:

GUIDO DARWIN VILEMA GUIJARRO

RIOBAMBA – ECUADOR
2011

AGRADECIMIENTO

Deseo expresar mi más profundo agradecimiento a la Escuela Superior Politécnica de Chimborazo y a sus dignas autoridades por darnos la oportunidad de forjarnos académica y humanamente.

Un reconocimiento especial al Ing. Marco Viteri, quien más que mi Director es un amigo, quien por su guía, apoyo y paciencia contribuyó en gran manera al desarrollo de este trabajo.

También agradezco de todo corazón al Ing. Danny Velasco por su tiempo energías y esfuerzos brindados en la culminación de este trabajo.

DEDICATORIA

Este trabajo de Investigación está dedicado a mi Creador Jehová Dios, a mi esposita Ana Inés Llamuca Moyota y a mi hijita Ginger Anahí Vilema Llamuca, quienes con su apoyo incondicional me han dado la fortaleza necesaria para luchar y alcanzar una meta más en mi vida.

A mis amados padres Guido Hermel Vilema Orozco, y Delia Petita Guijarro, y a mis hermanos René, David, Alejandra y Álvaro, por sus palabras de aliento en los momentos en que mas los necesite.

Guido Darwin Vilema Guijarro

FIRMAS DE RESPONSABILIDAD

NOMBRE	FIRMA	FECHA
Ing. Iván Menes DECANO FACULTAD DE INFORMÁTICA Y ELECTRÓNICA	……	……
Ing. Raúl Rosero DIRECTOR DE LA ESCUELA INGENIERÍA EN SISTEMAS	……	……
Ing. Marco Viteri DIRECTOR DE TESIS	……	……
Ing. Danny Velasco MIEMBRO DEL TRIBUNAL	……	……
Tlgo. Carlos Rodríguez DIRECTOR DEL CENTRO DE DOCUMENTACIÓN	……	……
NOTA DE LA TESIS	……	

“Yo, Guido Darwin Vilema Guijarro, soy responsable de las ideas, doctrinas y resultados expuestos en esta Tesis, y el patrimonio intelectual de la Tesis de Grado pertenece a la “**ESCUELA SUPERIOR POLITECNICA DEL CHIMBORAZO**”.

……

Guido Darwin Vilema Guijarro

INDICE DE ABREVIATURAS

CPU	Central Process Unit Unidad Central de Procesos
GRAFCET	GRAFica de Control de Etapas de Transición
IEEE	Institute of Electronic and Electrical Engineers Instituto de ingenieros eléctricos y electrónicos
IP	Internet Protocol Protocolo de Internet
I/O	Input/Output Entrada/Salida
LAN	Local Área Network Red de área local
MMS	“Manufacturing Message Specification” (acorde al ISO/IEC 9506).
PLC	Programmable Logic Controllers Controlador Lógico Programable
RAM	Random Access Memory Memoria de Acceso rápido
SCADA	Supervisory control And Data Adquisition Control Supervisión y adquisición de Datos
SFC	Sequential Function Charts Gráficas de funciones secuenciales

INDICE GENERAL

CAPÍTULO III

INDICE DE FIGURAS

INDICE DE TABLAS

INTRODUCCION

En la actualidad ya no es una excepción que tanto en el campo industrial como en el campo educativo se hable de transformar los procesos manuales en procesos automáticos. Sin embargo es importante contar con estándares de automatización que permitan realizar dichas transformaciones.

La Ingeniería de la Automatización Industrial ha efectuado un enorme progreso en las últimas décadas. Elementos de hardware cada día más potentes, la incorporación de nuevas funcionalidades, y el desarrollo de las redes de comunicación industriales, permiten realizar excelentes sistemas de Automatización Industrial en tiempos mínimos.

Cabe indicar que en el campo educativo, se ha estado capacitando por algunos años a los jóvenes para que alcancen la excelencia en automatizar procesos en un tiempo mínimo, y con ese fin WorldSkills Internacional ha sido la pionera en alcanzar dichos objetivos.

Sin embargo la programación secuencial sigue siendo larga de ejecutar con los diferentes pasos que se debe cumplir para poder automatizar cualquier equipo, y es por ello que se dará una metodología de programación que reduzca aun más los tiempos de programación.

CAPÍTULO I

1. MARCO REFERENCIAL

1.1. Antecedentes

La necesidad de automatizar procesos viene desde hace mucho tiempo, aunque se da mayor importancia en el siglo XX con la aparición del ordenador o computador en la década de 1960, y de ahí la creación de mecanismos sensoriales para recepción de datos y mecanismos actuadores de procesos.

De hecho la automatización como una disciplina de la ingeniería es más amplia que un mero sistema de control, abarca la instrumentación industrial, que incluye los sensores y transmisores de campo, los sistemas de control y supervisión, los sistema de transmisión y recolección de datos y las aplicaciones de software en tiempo real para supervisar y controlar las operaciones de plantas o procesos industriales.

Podemos decir que, como consecuencia a este hecho nace la Mecatrónica, término que fue utilizado como tal en Japón a principios de los años ochenta´s y comenzó a ser usado en Europa y USA un poco después. Hoy en día la Mecatrónica es un término que une distintas tecnologías – mecánica, electrónica y programación – para crear procesos secuenciales.

A su vez con el paso del tiempo, se ha ido ampliando y perfeccionando la Mecatrónica mediante el concurso de habilidades y destrezas WorldSkills que busca promover el intercambio entre jóvenes profesionales de varias regiones del mundo, intercambio de habilidades, experiencias e innovaciones tecnológicas y despertar el espíritu deportivo en los profesionales.

La ESPOCH ha sido la pionera en promover la especialización a sus estudiantes y en el campo de la Mecatrónica no ha sido la excepción. En el año 2007 fueron invitados por World Skills Internacional con el auspicio de la empresa TECHNA.

Sin embargo como primera experiencia vivida, requerían de una metodología de programación que ayude a aprovechar al máximo el tiempo en cuanto a programación secuencial se refiere y es por ello que en este trabajo se propone dicha metodología aplicada al laboratorio de Automatización Industrial de la EIS en la ESPOCH.

1.2. Justificación

1.2.1. Justificación teórica

Tanto la Automatización Industrial como la Mecatrónica ha evolucionado a un ritmo acelerado, y la ESPOCH como Institución educativa ha procurado incluir como materia en las respectivas Escuelas según su malla curricular.

Dentro de la ejecución de procesos ya sea de automatización industrial como mecatrónicos, al programar cualquier PLC's normalmente se realizan los siguientes pasos: plantear una secuencia de instrucciones, usar la representación gráfica, definir las ecuaciones necesarias, establecer la lista de asignaciones y ejecutar el programa, lo cual conlleva un considerable tiempo.

Normalmente existen instrucciones de programación que son muy generalizadas, lo cual en muchos casos confunden en vez de entender y programar el PLC's. El presente trabajo investigativo pretende complementar de una manera clara y entendible cómo programar y más aún, hacerlo en un tiempo mínimo de tal modo que pueda fácilmente aplicarlo a cualquier proyecto mecatrónico.

Esta información también servirá de herramienta evaluativa en caso que el programador requiera aplicar cualquier proceso secuencial y ejecutarlo en el PLC que esté más familiarizado y con el lenguaje de programación apropiado.

También a los docentes que cada vez conocen más del tema será provechoso disponer de un manual de consulta para las clases de Automatización Industrial y Mecatrónica, logrando llegar al estudiante y así fomentar el estudio y la investigación de programación de PLC ya sea en automatización industrial o mecatrónica.

1.2.2. Justificación Práctica Aplicativa

Los estudiantes desean profesionalizarse y alcanzar estándares de excelencia en cualquier campo, y en el campo de la mecatrónica no es la excepción. Recientemente se está difundiendo en nuestro país las competencias de habilidades y destrezas WorldSkills debido a que la empresa TECNA fue aceptada en representación de Ecuador como parte del equipo WorldSkills.

En el 2007 tuvieron su primera experiencia los estudiantes y su entrenador en Shizuoka – Japón, y en cuanto a ensamblaje del equipo les fue bien, pero en cuanto a la programación hubo demoras hasta realizar las respectivas pruebas, aunque no existe límite de tiempo en el concurso para que se cumpla el objetivo de armar y funcionar determinadas MSP.

La Propuesta Metodológica para la programación de PLC en Grafcet permitirá reducir dichos tiempos y más aún si lo aplicamos a las competencias de Mecatrónica WorldSkills en los cuales se debe cumplir objetivos en el menor tiempo posible.

En competencias de Mecatrónica auspiciadas por FESTO, las reglas de competencia dan enfoque en el funcionamiento de las diferentes MPS entregadas a los equipos aplicando los estándares en cuanto a instalación hardware se refiere, pero en cuanto a software es criterio propio del equipo de trabajo. Es por ello que se puede lograr optimizar los tiempos de programación mediante el uso del Grafcet como lenguaje.

También en el laboratorio de la ESPOCH se podrá notar la diferencia especialmente en casos de estudio donde esté involucrado muchas tareas.

Por tanto, las competencias World Skills son una buena plataforma para tener experiencia en el campo de la Mecatrónica, y para los estudiantes que participan en la misma esta investigación de tesis será una buena herramienta de trabajo en cuanto a optimización de tiempos se refiere.

También a los docentes que cada vez conocen más del tema será provechoso disponer de un manual de consulta para las clases de Automatización Industrial y Mecatrónica. El presente proyecto de tesis nos permitirá plasmar todos los conocimientos adquiridos en clase y servirá con fines de aprendizaje para los estudiantes, ya que muchos de ellos desearán alcanzar la excelencia en la optimización de recursos y tiempo, y como premio a su esfuerzo tener mejores oportunidades de trabajo ya que no solamente se requiere en nuestro medio Ingenieros que estén capacitados en lo referente a la Automatización Industrial, sino también que sean los mejores.

Figura I.1. Equipo de competencia WorldSkills Ecuador.

1.3. Objetivos

1.3.1. Objetivo General

Realizar una propuesta metodológica para la programación de PLC's en grafcet para las competencias de Mecatrónica World Skills con un tiempo de programación mínimo.

1.3.2. Objetivos Específicos

- Reconocer los diferentes elementos hardware usados en competencias de World Skills para establecer secuencias de programación.
- Definir mecanismos de comunicación de componentes para la implementación de los equipos mecatrónicos.
- Analizar los diferentes métodos de programación de PLC's para la evaluación de tiempos de programación.

- Determinar varios software para PLC’s. como herramientas de programación para implantar el lenguaje Grafcet.
- Determinar las ventajas y los resultados para la implementación metodológica de programación de PLC’s.

1.4. Hipótesis

“El diseño de una propuesta metodológica de programación de PLC’s en Grafcet para las competencias de Mecatrónica World Skills permitirá disminuir el tiempo de programación de procesos secuenciales”

CAPÍTULO II

2. MARCO TEORICO

2.1. Competencias WorldSkills International

WorldSkills Internacional constituye un medio único de intercambio y comparación de las normas de competencia de clase mundial en los oficios industriales y de servicios de la economía mundial.

El crecimiento continuo de WorldSkills Internacional atestigua el hecho de que el comercio tradicional y las habilidades artesanales, junto con las vocaciones, nueva tecnología polivalente, hacen una contribución esencial al desarrollo económico y el bienestar social de los pueblos en todas partes.

En forma independiente y no político, WorldSkills International proporciona un medio rentable para el gobierno y la cooperación internacionales de la industria en el logro de estándares más altos y la situación de la educación y la formación profesional a nivel mundial.

2.2. Objetivos

Los objetivos que persigue WorldSkills International son:

- Colocar a las competencias WorldSkills como el evento mundial de primera para el reconocimiento y la promoción de habilidades
- Desarrollar una nueva identidad moderna y una estructura flexible para apoyar las actividades globales de WorldSkills Internacional.
- Desarrollar alianzas estratégicas con determinados, gobierno corporativo y las organizaciones no gubernamentales para promover los objetivos de WorldSkill Internacional.
- Difundir información y compartir conocimientos sobre las normas de la habilidad y el rendimiento de referencia de WorldSkills, especialmente a través de la World Wide Web.
- Facilitar la creación de redes entre WorldSkills Internacional de Expertos para desarrollar nuevas oportunidades para el desarrollo de habilidades y la innovación.
- Favorecer la transferencia de habilidades, conocimientos y el intercambio cultural entre los participantes en WorldSkills y otros jóvenes de todo el mundo.

2.3. Reglas de competencia

En cada competencia promocionada por WorldSkills en cualquier parte del mundo, elaboran sus reglas para los participantes, tanto a nivel de alumnos, como entrenadores, etc. En el apéndice veremos con más detalle un modelo de dichas reglas.

Por lo general, existen reglas aplicadas tanto a competidores, como jueces y expertos en el área en el cual están demostrando las habilidades.

2.4. Categorías y Habilidades

En WorldSkills se ha logrado establecer diferentes categorías de formación profesional y cada categoría dispone de un determinado número de habilidades. Las categorías son:

- ***Transporte y Logística***

  La categoría "Transporte y Logística" cubre todos los ámbitos calificados que están relacionados con el mundo del transporte. Esto incluye la creación, reparación y mantenimiento de los vehículos de transporte.

- ***Construcción y Tecnología de la Construcción***

  La categoría "Construcción y Tecnología de la Construcción", cubre todos los ámbitos calificados que están relacionados con el mundo de la

construcción. Esto abarca todo, desde los cimientos, la construcción, acabados y mantenimiento de todo tipo de edificios.

- ***Fabricación y Tecnología de la Ingeniería***

  La categoría “Fabricación y Tecnología de la Ingeniería” cubre todos los ámbitos calificados que están relacionados con el desarrollo industrial y la creación.

  Esto abarca desde el diseño, creación, fabricación y mantenimiento de cualquier cosa que implica la electrónica y las máquinas.

- ***Tecnología de Información y Comunicación***

  La categoría "Información y Tecnología de la Comunicación" cubre todos los ámbitos calificados que están relacionados con los servicios de información. Esto abarca desde la creación y mantenimiento de redes para el desarrollo y acabado de tecnologías de la información.

- ***Arte Creativo y Moda***

  La categoría "Arte Creativo y de moda" cubre todos los ámbitos calificados que están relacionados con las artes plásticas y diseño de moda y la creación. Esto abarca todo, desde varios medios de comunicación, la decoración interior creativo y la moda.

- ***Sociales y Servicios Personales***

  La categoría "Social y Servicios Personales" cubre todos los ámbitos calificados que están relacionados con la industria de servicios. Esto incluye servicios relacionados con la industria de bebidas y alimentos, así como la hospitalidad y el cuidado personal.

2.4.1. Habilidades Oficiales de Competencia.

Dentro de cada categoría se demuestran las habilidades oficiales de competencia, las mismas que han ido aumentando con el tiempo. Estas son:

- **Transporte y Logística**
  - Mantenimiento de Aeronaves
  - Reparación de Carrocerías
  - Tecnología del automóvil
  - Pintura de coches
- **Construcción y Tecnología de la Construcción**
  - Albañíleria
  - Ebanistería
  - Carpintería
  - Instalaciones electricas
  - Carpintería
  - Jardinería Paisajista
  - Cubiertas de metal
  - Pintura y decoración
  - Revestimiento de yeso y Sistemas de Yeso
  - Servicios de fontanería y calefacción
  - Refrigeración
  - solados y alicatados
- **Fabricación y Tecnología de la Ingeniería**
  - Fresado CNC
  - Torno CNC

- Trabajos de construcción de metal
- Modelos creativos
- Electrónica
- Control Industrial
- Fabricación de Equipo Desafío
- Ingeniería de Diseño Mecánico – CAD
- Mecatrónica
- Robotica Movil
- Fabricación de moldes
- Automatización de poli mecánicos
- Hoja de Tecnólogos de metal
- Soldadura

➢ **Tecnología de Información y Comunicación**

- Red de Información de Cableado
- IT/ PC Red de Apoyo
- IT software de aplicaciones
- Impresión Offset
- Diseño Web

➢ **Arte Creativo y Moda**

- Tecnología de la Moda
- Floristería
- Tecnología del Diseño Gráfico
- Joyería
- Comerciante Visual

- **Sociales y Servicios Personales**
    - Pelo afro-caribeña
    - Terapia de belleza
    - Cuidado
    - Pastelero
    - Cocina
    - Damas y Caballeros de peluquería
    - Servicio de restaurante

Es interesante notar que la Habilidad Oficial de ***Mecatrónica*** está dentro de la categoría ***Fabricación y Tecnología de la Ingeniería***.

2.4.2. Habilidades de Demostración.

Las habilidades de demostración cumplen un fin complementario en las competencias WorldSkills puesto que no están en la lista de competencias para ser calificadas y más bien son de tipo educativo. Esto sucede cuando cierta habilidad no tiene un determinado número mínimo de países participantes para convertirse en una habilidad de competencia.

2.4.3. Habilidades del Miembro Anfitrión.

Con respecto a las habilidades del miembro anfitrión, cada país que organiza las competencias WorldSkills tiene un determinado número de habilidades de demostración que han desarrollado los jóvenes locales, dando realce al evento.

2.5. La Mecatrónica en WorldSkills

Mecatrónica es una tecnología híbrida que cubre los campos de la ingeniería mecánica, electrónica e informática Los alcances de la mecatrónica incluye productos tales como lap top, cámara de video grabación, o maquinas expendedoras de boletos en estacionamientos públicos o puertas de acceso en aeropuertos por mencionar algunos.

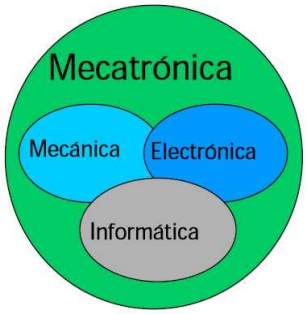


Figura II.2 Relación de la Mecatrónica con otras ciencias

A manera de realizar un aprendizaje y mejoramiento continuo, los especialistas en mecatrónica deben trabajar en equipo, para lo que se requiere personal altamente calificado y habilidades de integración social. Mientras que al final el patrón o dueño espera que el especialista de mecatrónica como un individuo que domina todas las tecnologías necesarias, esta funcionando dentro de un buen ambiente con grupo motivado que facilita la difusión de conocimientos e ideas. Este ambiente permite a los especialistas auto desarrollar sus habilidades a un nivel mucho más lejos del nivel que se pudiera obtener mediante un entrenamiento convencional.

Poco a poco se han integrado algunos países para perfeccionar en el campo de la mecatrónica, tal como se muestra en la siguiente tabla.

No	País	Ámsterdam 91	Taipei 93	Lyón 95	St Gallen 97	Montreal 99	Corea 01
1	Australia	X	X	X	X	X	X
2	Austria						X
3	Brasil				X	X	X
4	Canadá		X	X	X	X	X
5	Finlandia		X	X	X	X	X
6	Francia			X		X	X
7	Alemania		X	X	X	X	X
8	Corea			X	X	X	X
9	Hong Kong				X	X	
10	Japón					X	X
11	Singapur			X	X	X	X
12	Suecia			X		X	X
13	Suiza				X	X	X
14	Filipinas					X	X
15	Taiwán			X	X	X	X
16	Tailandia					X	X
17	Holanda	X	X	X	X	X	X
18	Marruecos						X
19	Nueva Zelanda			X	X	X	
20	Noruega		X	X	X	X	X
21	Reino unido			X	X	X	X
22	Emiratos Árabes Unidos					X	X
	Competidores en el pasado						
	Sudáfrica			X			

Tabla II.I Países participantes en Mecatrónica

Dentro de poco tendremos una generación que priorice la automatización de procesos secuenciales en diferentes campos.

2.5.1. Condiciones generales de competencia

Mecatrónica tiene la ventaja que puede ser evaluada al 100 % con un criterio objetivo, en donde los principales objetivos son ***funcionamiento y tiempo***. Lo anterior significa que el sistema de evaluación es transparente para todos –

competidores, expertos y visitantes – lo que deduce una competencia justa. Por otra parte, los resultados están disponibles inmediatamente cuando el equipo ha terminado, lo que origina emoción en los asistentes al evento.

Los competidores trabajan y son evaluados en grupos de dos personas. Esto refleja la tendencia de la organización del trabajo en muchos países. Mientras las competencias sociales y personales no son evaluadas de manera separada, son esenciales para un excelente desempeño y obtener un buen resultado. Los resultados se muestran al término de cada sección.

Como es sabido de todos el PLC se ha convertido en una herramienta indispensable en los sistemas mecatrónicos, razón por la cual los competidores pueden traer su propio PLC y el software necesario para programarlo, tomando en cuenta que las ventajas o desventajas que esto tenga dependerán del equipo seleccionado

CAPÍTULO III

3. SISTEMAS DE PROGRAMACION

3.1. Introducción al PLC

Los controladores lógicos programables o PLC (*Programmable Logic Controller*) son dispositivos electrónicos muy usados en automatización industrial. Su historia se remonta a finales de la década de 1960, cuando la industria buscó en las nuevas tecnologías electrónicas una solución más eficiente para reemplazar los sistemas de control basados en circuitos eléctricos con relés, interruptores y otros componentes comúnmente utilizados para el control de los sistemas de lógica combinacional.

Hoy en día, los PLC's no sólo controlan la lógica de funcionamiento de máquinas, plantas y procesos industriales, sino que también pueden realizar operaciones aritméticas, manejar señales analógicas para realizar estrategias de control, tales como controladores PID (Proporcional Integral y Derivativo).

Los PLC's actuales pueden comunicarse con otros controladores y computadoras en redes de área local, y son una parte fundamental de los modernos sistemas de control distribuido.

3.2. Componentes básicos de un PLC

La estructura básica de cualquier autómata es la siguiente.

- CPU
- Memoria del autómata
- Interfaces de E/S
- Fuente de alimentación

Figura III.3 Diagrama de bloques del PLC

3.2.1. CPU

La Unidad Central de Procesos es el auténtico cerebro del sistema. Se encarga

de recibir las ordenes, del operario por medio de la consola de programación y el modulo de entradas. Posteriormente las procesa para enviar respuestas al módulo de salidas. En su memoria se encuentra residente el programa destinado a controlar el proceso

La CPU es el corazón del autómata programable. Es la encargada de ejecutar el programa de usuario mediante el programa del sistema (es decir, el programa de usuario es interpretado por el programa del sistema).

Sus funciones son:

- Vigilar que el tiempo de ejecución del programa de usuario no excede un determinado tiempo máximo (tiempo de ciclo máximo). A esta función se le suele denominar Watchdog (perro guardián).
- Ejecutar el programa de usuario.
- Crear una imagen de las entradas, ya que el programa de usuario no debe acceder directamente a dichas entradas.
- Renovar el estado de las salidas en función de la imagen de las mismas obtenida al final del ciclo de ejecución del programa de usuario
- Chequeo del sistema.

3.2.2. Memoria del Autómata

Dentro de la CPU vamos a disponer de un área de memoria, la cual emplearemos para diversas funciones:

- Memoria del programa de usuario: aquí introduciremos el programa que el autómata va a ejecutar cíclicamente.
- Memoria de la tabla de datos: se suele subdividir en zonas según el tipo de datos (como marcas de memoria, temporizadores, contadores, etc.).
- Memoria del sistema: aquí se encuentra el programa en código máquina que monitorea el sistema (programa del sistema o firmware). Este programa es ejecutado directamente por el microprocesador/microcontrolador que posea el autómata.
- Memoria de almacenamiento: se trata de memoria externa que empleamos para almacenar el programa de usuario, y en ciertos casos parte de la memoria de la tabla de datos. Suele ser de uno de los siguientes tipos: EPROM, EEPROM, o FLASH.

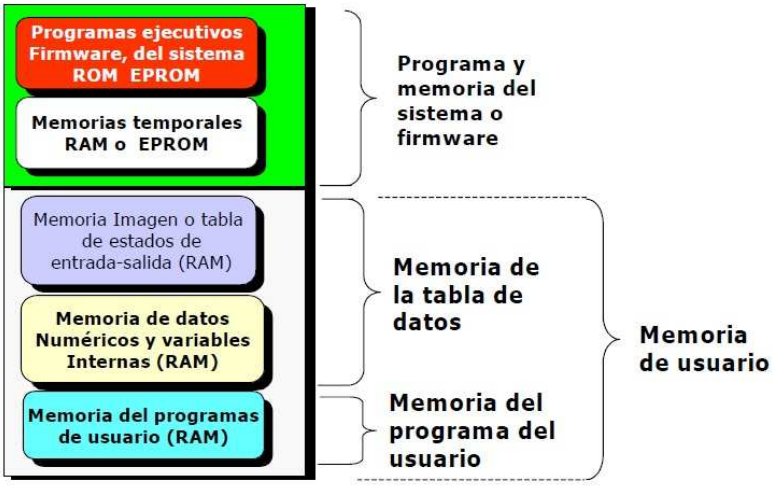


Figura III.4 Memorias de un PLC

Cada autómata divide su memoria de esta forma genérica, haciendo subdivisiones específicas según el modelo y fabricante, tal como se muestra en la Fig. III.4.

Físicamente las memorias constituyen elementos electrónicos que hacen posible usar imágenes de datos como entradas, salidas, temporizadores, contadores, etc. Mas adelante veremos la manera de acceder a dichos datos usando la sintaxis que ha implantado el estándar IEC 61131-3.

3.2.3. Interfaces de E/S

Sección de entradas: se trata de líneas de entrada, las cuales pueden ser de tipo digital o analógico. En ambos casos tenemos unos rangos de tensión característicos, los cuales se encuentran en las hojas de características del fabricante. A estas líneas conectaremos los sensores.

Sección de salidas: son una serie de líneas de salida, que también pueden ser de carácter digital o analógico. A estas líneas conectaremos los actuadores.

Tanto las entradas como las salidas están aisladas de la CPU según el tipo de autómata que utilicemos. Normalmente se suelen emplear opto acopladores en las entradas y relevadores/optoa copladores en las salidas.

A este módulo se unen eléctricamente los sensores (interruptores, finales de carrera, pulsadores,...). La información recibida en él, es enviada a la CPU para ser procesada de acuerdo la programación residente.

Se pueden diferenciar dos tipos de captadores conectables al módulo de entradas: los Pasivos y los Activos.

Los Captadores Pasivos son aquellos que cambian su estado lógico, activado - no activado, por medio de una acción mecánica. Estos son los Interruptores, pulsadores, finales de carrera, etc.

Los Captadores Activos son dispositivos electrónicos que necesitan ser alimentados por una tensión para que varíen su estado lógico. Este es el caso de los diferentes tipos de detectores (Inductivos, Capacitivos, Fotoeléctricos). Muchos de estos aparatos pueden ser alimentados por la propia fuente de alimentación del autómata.

Las condiciones de entrada se almacenan en una parte de la memoria del procesador denominada archivo de imágenes de entrada. Esto es que cada terminal del modulo de entrada tiene su lugar dentro del archivo de imágenes de entrada, y este lugar solo sirve para guardar el ultimo registro de estado de un terminal de entrada (1 logico o HI 0 logico o LO) High input o Low output

Las condiciones de salida se almacenan de manera similar en el archivo de imágenes de salida, y cuya función es idéntica al archivo de imágenes de entrada. La única diferencia es el flujo de informaciñon, mientras que en la salida, el flujo es del archivo de imágenes de salida al módulo de salida; en la entrada el flujo es del modulo de entrada al archivo de imágenes de entrada.

Cada terminal de entrada o salida, tiene su lugar el archivo de imágenes correspondiente, ubicados en localidades e identificados respectivamente por

direcciones, el sistema de direcciones para cada terminal salida y entrada varía según el fabricante para el PLC, por ejemplo una terminal de entrada se denomina I1.0 y una terminal de salida se denomina Q2.0

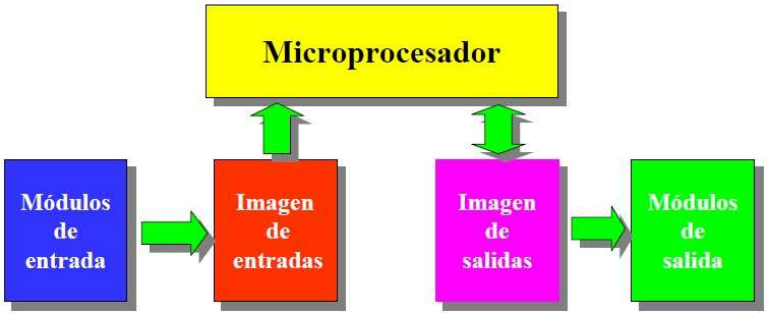


Figura III.5 Componentes de un PLC

3.2.4. Fuente de alimentación

Es la encargada de convertir la tensión de la red, 117 v c.a. ó 220v c.a., a baja tensión de c.c, normalmente 24 v. Siendo esta la tensión de trabajo en los circuitos electrónicos que forma el Autómata.

3.3. Lenguajes de programación

Los lenguajes de programación son necesarios para la comunicación entre el usuario, sea programador u operario de la máquina o proceso donde se encuentre el PLC y el PLC. La interacción que tiene el usuario con el PLC la puede realizar por medio de la utilización de un cargador de programa también reconocida como consola de programación o por medio de un PC.

En procesos grandes o en ambientes industriales el PLC recibe el nombre también de API (Autómata Programable Industrial) y utiliza como interfase para el usuario pantallas de plasma, pantallas de contacto (touch screen) o sistemas SCADA (sistemas para la adquisición de datos, supervisión, monitoreo y control de los procesos).

Para que la forma de programación sea unforme y entendible, existe un organismo internacional de programación y especialmente en el campo Mecatrónico que ha establecido estándares

3.4. Clasificación

Los lenguajes de programación para PLC son de dos tipos, visuales y escritos. Los visuales admiten estructurar el programa por medio de símbolos gráficos, similares a los que se han venido utilizando para describir los sistemas de automatización, planos esquemáticos y diagramas de bloques. Los escritos son listados de sentencias que describen las funciones a ejecutar.

Los programadores de PLC poseen formación en múltiples disciplinas y esto determina que exista diversidad de lenguajes. Los programadores de aplicaciones familiarizados con el área industrial prefieren lenguajes visuales, por su parte quienes tienen formación en electrónica e informática optan, inicialmente por los lenguajes escritos.

3.5. Niveles de los Lenguajes

Los lenguajes de programación de sistemas basados en microprocesadores, como es el caso de los PLC, se clasifican en niveles; al microprocesador le corresponde el nivel más bajo, y al usuario el más alto.

3.5.1. Lenguajes de Bajo Nivel

Lenguaje de Máquina: Código binario encargado de la ejecución del programa directamente en el microprocesador.

Lenguaje Ensamblador: Lenguaje sintético de sentencias que representan cada una de las instrucciones que puede ejecutar el microprocesador. Una vez diseñado un programa en lenguaje ensamblador es necesario, para cargarlo en el sistema, convertirlo o compilarlo a lenguaje de máquina. Los programadores de lenguajes de bajo nivel deben estar especializados en microprocesadores y demás circuitos que conforman el sistema.

3.5.2. Lenguajes de Alto Nivel

Se basan en la construcción de sentencias orientadas a la estructura lógica de lo deseado; una sentencia de lenguaje de alto nivel representa varias de bajo; cabe la posibilidad que las sentencias de un lenguaje de alto nivel no cubran todas las instrucciones del lenguaje de bajo nivel, lo que limita el control sobre la máquina. Para que un lenguaje de alto nivel sea legible por el sistema, debe traducirse a lenguaje ensamblador y posteriormente a lenguaje de máquina.

3.6. El Estándar IEC

3.6.1. Definición

La normalización o estandarización es la redacción y aprobación de normas que se establecen para garantizar el acoplamiento de elementos construidos en forma independiente de tal modo que exista intercomunicación tanto a nivel de hardware como a nivel de software.

En el campo de la Automatización Industrial y la Mecatrónica, el organismo que establece los estándares es la Comisión Electrotécnica Internacional (o IEC por sus siglas en inglés, International Electrotechnical Commission); organización de normalización dedicada en los campos eléctrico, electrónico y tecnologías relacionadas. En 1938, el organismo publicó el primer diccionario internacional (International Electrotechnical Vocabulary) con el propósito de unificar la terminología eléctrica, esfuerzo que se ha mantenido durante el transcurso del tiempo, siendo el Vocabulario Electrotécnico Internacional un importante referente para las empresas del sector. A continuación analizaremos el estándar IEC 61131 para la programación de PLC.

3.6.2. IEC 61131

El estándar internacional IEC 61131 es una colección completa de estándares referentes a controladores programables y sus periféricos asociados. Consiste de las siguientes partes:

- Parte 1: Información general

- Parte 2: Requisitos del equipo y pruebas
- Parte 3: Los lenguajes de programación
- Parte 4: Guía del usuario
- Parte 5: Especificación del servicio de mensajería
- Parte 6: Comunicaciones a través de bus de campo
- Parte 7: Control difuso de programación
- Parte 8: Directrices para la aplicación e implementación de lenguajes de programación

➢ ***Parte1: Información General***

Establece las definiciones e identifica las principales características significativas a la selección y aplicación de los controladores programables y sus periféricos asociados.

➢ ***Parte 2: Requerimientos del equipo y pruebas***

Especifica los requisitos del equipo y pruebas relacionadas para los controladores programables (PLC) y sus periféricos asociados.

➢ ***Parte 3: Lenguajes de Programación***

Define como un conjunto mínimo, los elementos básicos de programación. Reglas sintácticas y semánticas para los lenguajes de programación usados más comúnmente, incluyendo los lenguajes gráficos de Diagrama de Escalera y Diagrama de Bloques de Funciones y los lenguajes textuales de Lista de Instrucciones y Texto estructurado; así como sus principales campos de aplicación, pruebas aplicables y los medios por los cuales los fabricantes pueden expandir o adaptar esos

conjuntos básicos a sus propias implementaciones de controlador programable.

- ***Parte 4: Guías de Usuario.***

  Un reporte técnico que proporciona una vista general y guías de aplicación del estándar para los usuarios finales de los controladores programables.

- **Parte 5: Especificación del servicio de Mensajería.**

  Define la comunicación de datos entre controladores programables y otros sistemas electrónicos usando el “Manufacturing Message Specification” (MMS, acorde al ISO/IEC 9506).

- **Parte 6: Comunicaciones a través de bus de campo.**

  Proporciona información referente a la comunicación por medio de Fieldbus o bus de campo. Consiste en un sistema de redes industriales de control distribuido de tiempo real y puede adoptar diferentes estructuras de red (cadena,, estrella, anillo, ramas, árboles y topologías de red)

- **Parte 7: Programación en lógica difusa.**

  Define los elementos básicos de programación de “lógica difusa” para su uso en Controladores programables. Esto está relacionado en inteligencia artificial y sistemas expertos en los cuales las decisiones tomadas por el PLC se asemejan al mundo real.

- **Parte 8: Guías para aplicación e implementación de lenguajes de programación.**

  Proporciona una guía para los desarrolladores de software referente a los lenguajes de programación definidos en la parte 3.

3.7. Estándar de programación IEC 61131-3

IEC 61131-3 es el primer esfuerzo real para estandarizar los lenguajes de programación usados en para la automatización industrial. Con su soporte mundial, es independiente de una sola compañía. Esta parte de programación, es la tercera del estándar 61131.

Una forma conveniente de verlo, es dividiendo el estándar en 2 partes:

- Elementos Comunes.
- Lenguajes de Programación.

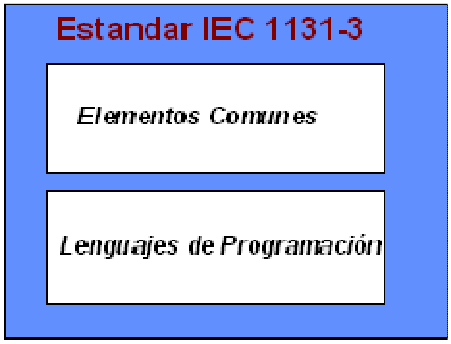


Figura III.6 Estandar IEC

3.8. Elementos Comunes

3.8.1. Tipos de Datos

Dentro de los elementos comunes se definen los tipos de datos. La tipificación de los datos previene errores en una etapa temprana. Se usa para definir el tipo de cualquier parámetro usado. Esto evita que por ejemplo se divida una fecha entre un entero.

Los tipos de datos comunes son: Boolean, Integer, Real, Byte y Word. También Date, Time_of_Day y String. Basado en ellos, uno puede definir sus propios tipos de datos, llamados “tipos de datos derivados”.

3.8.2. Variables.

Las variables son únicamente asignadas a direcciones de hardware explicitas (por ejemplo entradas y salidas) en la configuración, recursos o programas. De esta manera se le da a los programas una independencia de alto nivel del hardware, soportando el re-uso del software.

El enfoque (visibilidad) de las variables es normalmente limitado a la unidad de organización en la cual son declaradas (por ejemplo: local). Esto significa que sus nombres pueden ser usados nuevamente sin ningún conflicto en otras partes, eliminando otra fuente de errores. Si las variables requieren un alcance global, deben ser declaradas como tales. Los parámetros pueden recibir un valor inicial al arranque y al reinicio “en frió”, con objeto de asegurar su valor correcto al inicio de la ejecución de los programas.

3.8.3. Configuración, recursos y tareas.

Para entender mejor esto, es conveniente ver el modelo de software, tal como se define en la siguiente figura.

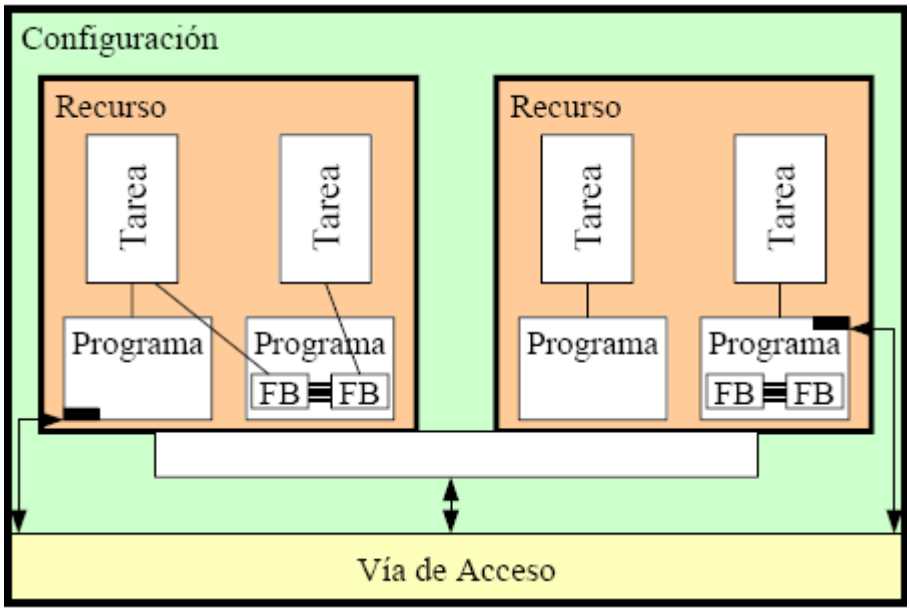


Figura III.7 Configuración y tareas de un PLC

Al nivel mas alto, el software completo que se requiere para solucionar un problema de control particular puede ser formulado como una configuración. Una configuración es específica a un sistema de control particular, incluyendo el arreglo del hardware, recursos de procesamiento, direcciones de memoria para los canales de entrada/salida y otras capacidades del sistema.

Dentro de una configuración, se pueden definir una o más tareas. Estas tareas controlan la ejecución de un conjunto de programas y/o bloques de función. Las

tareas pueden ser ejecutadas periódicamente o a la ocurrencia de algún evento disparador, por ejemplo el cambio en una variable.

Los programas están constituidos por diferentes elementos de software escritos en cualquiera de los lenguajes definidos por IEC. Típicamente un programa consiste de una red (network) o funciones y bloques de función que son capaces de intercambiar datos. Las funciones y los bloques de función son los bloques de construcción básicos y contienen una estructura de datos y un algoritmo.

Comparemos lo anterior con un PLC convencional: Este contiene recursos corriendo una tarea, corriendo un programa. IEC 61131-3 le agrega a esto mucho mas, haciéndolo abierto a mayores capacidades tales como multiprocesamiento y conducción por sucesos.

3.8.4. Unidades de organización del programa.

La norma define tres formas distintas de “presentar” o crear programas de control para PLCs, a saber:

- Funciones
- Bloques funcionales
- Programas

En IEC 61131-3 los Programas, Bloques de Función y Funciones son llamados Unidades de Organización de Programa (program organization units o POUs).

3.8.4.1. Funciones

IEC define Funciones Estándar y Funciones Definidas por el Usuario. Las funciones estándar son por ejemplo: ADD (suma), ABS (absoluto), SQRT (cuadrado) SIN (seno), etc. Las funciones definidas por el usuario (basadas en las funciones estándar), una vez definidas pueden ser re-usadas una y otra vez.

3.8.4.2. Bloques de Función (Function Blocks FBs)

Los bloques funcionales son los equivalentes de los circuitos integrados usados en electrónica, IC´s, que representan funciones de control especializadas. Los FB´s contienen tanto datos como instrucciones, pudiendo guardar los valores de dichas variables entre sucesivas ejecuciones (que es una de las diferencias con las funciones). Se dice por tanto que los FBs tienen "memoria", característica que les confiere un gran potencial de uso. Presentan una interfaz de entradas y salidas bien definida y un código interno oculto, como un circuito integrado o una caja negra. De este modo, establecen una clara separación entre los diferentes niveles de programadores o personal de mantenimiento.

Un lazo de control de temperatura, o un PID es un excelente ejemplo de un Bloque de Función. Una vez definido puede ser usado una y otra ves en el mismo programa, en diferentes programas o en diferentes proyectos, es decir son reusables.

Existen FB´s estándares como por ejemplo: biestables, detección de flancos, contadores, etc.; así como definidos por el usuario, basados en los existentes, obteniéndose así los Bloques de Función derivados.

Otra de las diferencias fundamentales con respecto a las funciones y que les confiere gran potencia de uso, es la posibilidad de crear tantas copias como se desee de un mismo FB. A cada copia se le llama instancia. Cada instancia llevará asociado un identificador y una estructura de datos que contenga sus variables de entrada, de salida e internas separada del resto de instancias.

3.8.4.3. Programas

La norma define un programa como el "conjunto lógico de todos los elementos y construcciones que son necesarios para el tratamiento de señales que se requiere para el control de una máquina o proceso mediante un PLC". Es decir, que un programa puede contener la declaración de tipos de datos, variables e instancias de bloques funcionales junto con el conjunto de instrucciones (código o programa propiamente dicho) necesario para llevar a cabo el control deseado del proceso o máquina.

En resumen, un Programa es una Red de Funciones, Bloques de Función, variables etc. el mismo que puede ser escrito en cualquiera de los lenguajes de programación definidos en el estándar.

3.8.5. Herramienta de modelado SFC

El SFC – **S**equential **F**unction **C**hart – describe gráficamente el comportamiento secuencial de un programa de control. El SFC estructura la organización

interna de un programa y ayuda a descomponerlo en partes más fácilmente manejables, mientras mantiene la visión general.

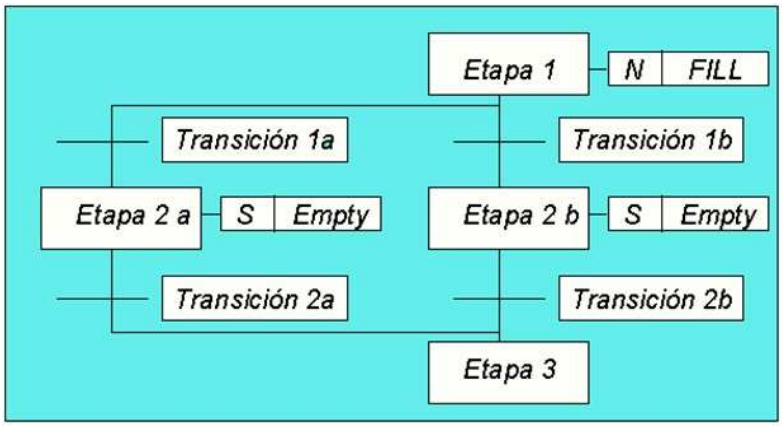


Figura III. 8 Herramienta de modelado SFC

Por su estructura general, SFC proporciona un medio de comunicación o entendimiento entre personas con diferentes especialidades. En el capitulo siguiente veremos con más detalle las características y bondades de este lenguaje.

3.9. Lenguajes de Programación

Dentro del estándar se definen 4 lenguajes de programación, los mismos que poseen su sintaxis y semántica propia. Una vez que usted los aprendió, puede aplicarlos a una gran cantidad de sistemas que están basados en estos estándares. Los lenguajes son 2 de tipo textual y 2 de tipo grafico:

- ***Textuales:***
  - Lista de Instrucciones (Instruction List – IL)
  - Texto estructurado (Structured Tex – ST)

- ***Gráficos:***
  - Diagrama de Bloques de Funciones (Function Block Diagram – FBD)
  - Diagrama de Escalera (Ladder Diagram – LD)

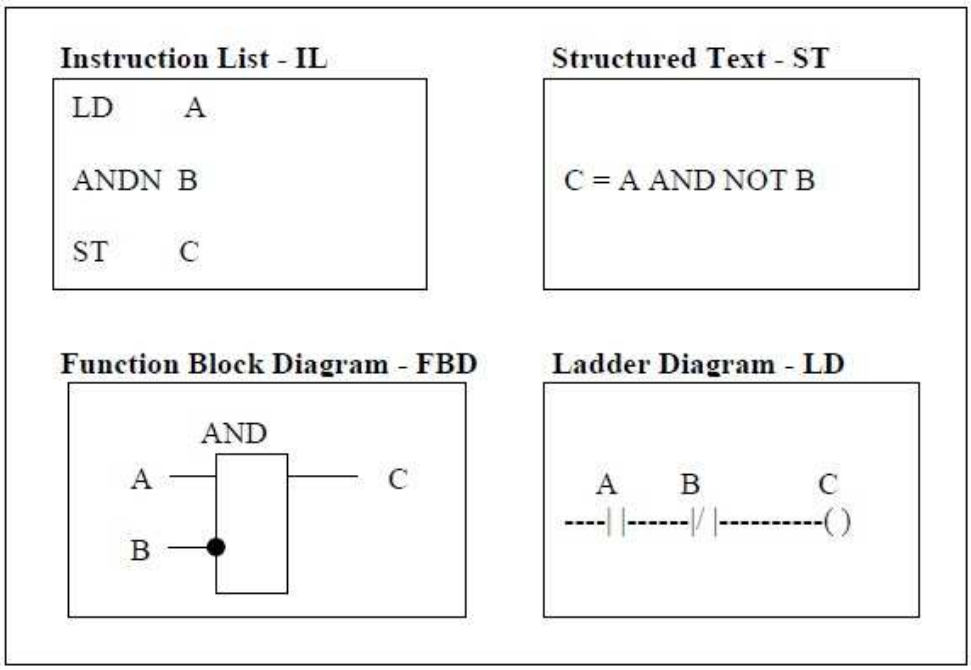


Figura III.9 Tipos de lenguaje

En la figura superior, los cuatro programas describen la misma acción. La elección del lenguaje de programación depende de:

- Los conocimientos del programador,
- El problema a tratar,
- El nivel de descripción del proceso,
- La estructura del sistema de control,
- La coordinación con otras personas o departamentos.

Los cuatros lenguajes están interrelacionados y permiten su empleo para resolver conjuntamente un problema común según la experiencia del usuario.

3.9.1. IL

Lista de Instrucciones (IL) es el modelo de lenguaje ensamblador basado un acumulador o pila simple; procede del alemán “Anweisungliste” (AWL)

3.9.2. ST

El lenguaje Texto estructurado (ST) es un lenguaje de alto nivel con orígenes en el Ada, Pascal y ´C´. Puede ser utilizado para codificar expresiones complejas e instrucciones anidadas mediante instrucciones para bucles (REPEAT-UNTIL; WHILE-DO), ejecución condicional (IF-THEN-ELSE; CASE), funciones (SQRT, SIN, etc.), etc. Este lenguaje resulta excelente para la definición de bloques de función complejos que pueden ser usados en cualquiera de los otros lenguajes.

3.9.3. FBD

El Diagramas de Bloques Funcionales (FBD) es muy común en aplicaciones que implican flujo de información o datos entre componentes de control. Las funciones y bloques funcionales aparecen como circuitos integrados y es ampliamente utilizado en Europa.

3.9.4. LD

El Diagrama de escalera (LD) también conocido como “lenguaje de contactos” tiene sus orígenes en los Estados Unidos. Está basado en la representación gráfica de la lógica de relés (automatismos eléctricos).

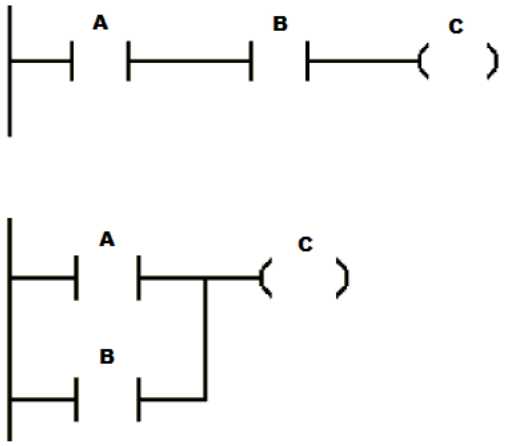


Figura III.10 Diagrama Ladder

3.10. Memoria de Datos

En términos generales, dentro de los PLC existe una unidad de memoria de datos, el mismo que se divide en dos partes:

- Área de datos
- Objetos

- El ***área de datos*** contiene
  - V : Memoria de variables
  - I : Imagen de proceso de las entradas
  - Q: imagen de proceso de las salidas

- M; Marcas internas
- SM: marcas especiales

- Los ***objetos*** pueden ser
  - T: temporizadores
  - C: Contadores
  - AI: entradas analógicas
  - AC: Acumuladores
  - HSC: Valores actuales de los contadores rápidos

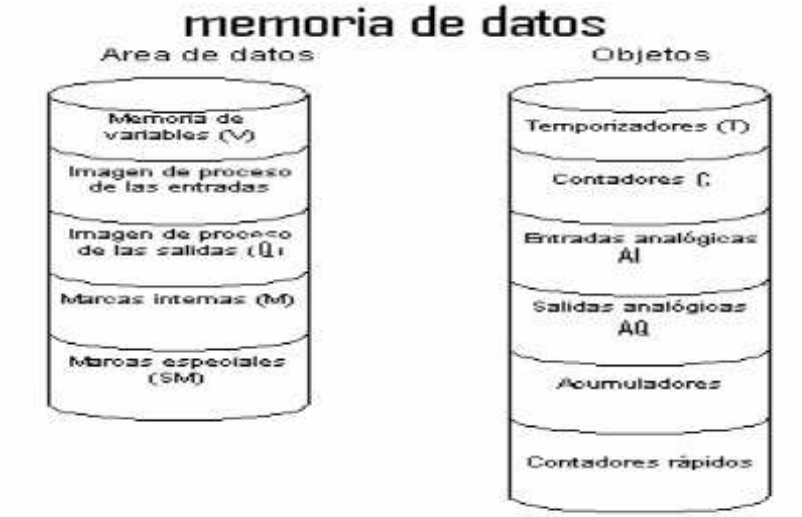


Figura III.11 Memoria de datos

3.11. Direccionamiento

La estructura de direccionamiento para PLC básicamente está conformada por un identificador de área, una dirección del byte y el número del bit; cuyo formato se establece de la siguiente manera:

Identificador_área dirección_byte . dirección_bit

Gráficamente, sería lo siguiente:

Figura III.12 Formato de las direcciones de memoria

Esto significa que en memoria estoy utilizando el bit 5 del byte 2. Sin embargo, para generalizar la sintaxis del direccionamiento de entradas, salidas y memorias; en algunos PLC y sus respectivos programas el despliegue directo de las células de memoria individual (direcciones absolutas) de acuerdo con la IEC 61131-3, está representado mediante strings especiales como se observa en la siguiente tabla.

Caracter	Significado	Observaciones
%	Carácter de inicio de direcciones absolutas	
I Q M	Entradas (Input) Salidas (Output) Memorias (Flag)	Prefijos de área de distribución
X* B W D	Bit simple Byte (8 bits) Word (16 bits) Double Word (32 bits)	Prefijo de tamaño en memoria
0.0 0.1 …….. 1.0 1.1 …….. ……	Direcciones (Según el tipo de dato que representan: X,B,W,D)	Direcciones de Byte y bit. Dependerá de las características que disponga el PLC.
* El caracter X para bits puede obviarse		

Tabla III.II Tipos de Variables

Por tanto, para la declaración de datos tanto de entrada, salida y memoria se denomina así: (% I 0.0). La visualización directa de las posiciones de memoria individual se realiza mediante el uso de secuencias de caracteres especiales. Estas secuencias son una concatenación del signo de porcentaje "%", un prefijo de identificador de área, un prefijo de tamaño y uno o más números naturales separados por espacios en blanco.

Los siguientes prefijos de ***identificador de área*** son compatibles:

- **I:** O datos de entrada
- **Q:** O datos de salida
- **M:** O la localización de memoria

Los siguientes prefijos de ***tamaño*** son compatibles:

- **X:** de un solo bit
- **B:** Byte (8 bits)
- **W:** Word (16 bits)
- **D:** Double Word (32 bits). Si no existe ninguna letra, se considerará en memoria un solo bit

Ejemplos:

Ejemplo	Significado
%QX7.5 y %Q7.5	Bit de salida 7.5
%IW215	Entrada de tipo Word 215
% QB7	Salida tipo Byte de 7
%MD48	Memoria tipo Double Word en la posición de memoria 48
%IW2.5.7.1	Dependiendo de la configuración del PLC

Tabla III.III Ejemplos de direcciones de memoria

Tomemos en cuenta que los valores booleanos se asignarán byte a byte. Si no hay una dirección explícita de un solo bit se debe especificar.

Ejemplo: Un cambio en el valor de una variable boolena %QW0 afecta el rango de QX0.0 a QX0.7 que son de un solo bit, y producirá un error de tipo de datos no compatibles.

3.12. Programación en Diagrama de Contactos (LD)

El lenguaje Ladder o Diagrama de contactos es un lenguaje gráfico, derivado del lenguaje de relés. Mediante símbolos representa contactos, bobinas, etc. Su principal ventaja es que los símbolos básicos están normalizados según el estándar IEC y son empleados por todos los fabricantes.

Los símbolos básicos son:

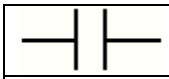
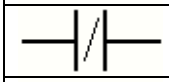
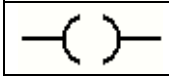
	CONTACTO NORMALMENTE ABIERTO
	CONTACTO NORMALMENTE CERRADO
	ASIGNACIÓN DE SALIDA

Tabla III.IV Elementos del Diagrama Ladder

Un programa en lenguaje ladder es una serie de ramas de circuito. Una rama (network) está compuesta de una serie de contactos, conectados en serie o en

paralelo, que dan origen a una salida (activación de una bobina o de una función especial).

Figura III.13 Diagrama sencillo de Ladder

Las ramas de circuitos tienen origen en una barra vertical puesta a la izquierda del diagrama. El flujo de la señal va de izquierda a derecha y de arriba abajo. A una rama de circuito en LD corresponde una secuencia de instrucciones en forma nemónica. Sin embargo, existen situaciones que no se pueden implementar en lenguaje ladder:

- Una bobina no puede venir conectada directamente a la barra de inicio, en tal caso es necesario interponer un contacto siempre cerrado.
- A la derecha de una bobina no es posible programar ningún contacto.
- El número de contactos posibles en serie o en paralelo es prácticamente ilimitado.

3.12.1. Elementos Básicos

Para programar un autómata con LADDER, es necesario conocer cada uno de los elementos de que consta este lenguaje. A continuación se describen de modo general los más comunes.

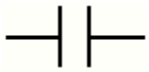
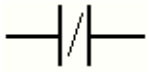
Símbolo	Nombre	Descripción
	Contacto NA	Se activa cuando hay un uno lógico en el elemento que representa, esto es, una entrada (para captar información del proceso a controlar), una variable interna o un bit de sistema.
	Contacto NC	Su función es similar al contacto NA anterior, pero en este caso se activa cuando hay un cero lógico, cosa que deberá de tenerse muy en cuenta a la hora de su utilización.
	Bobina NA	Se activa cuando la combinación que hay a su entrada (izquierda) da un uno lógico. Su activación equivale a decir que tiene un uno lógico. Suele representar elementos de salida, aunque a veces puede hacer el papel de variable interna.
	Bobina NC	Se activa cuando la combinación que hay a su entrada (izquierda) da un cero lógico. Su activación equivale a decir que tiene un cero lógico. Su comportamiento es complementario al de la bobina NA.
	Bobina SET	Una vez activa (puesta a 1) no se puede desactivar (puesta a 0) si no es por su correspondiente bobina en RESET. Sirve para memorizar bits y usada junto con la bobina RESET dan una enorme potencia en la programación.
	Bobina RESET	Permite desactivar una bobina SET previamente activada.

Tabla III.V Elementos Básicos Ladder

Además existe una gran variedad de elementos adicionales como temporizadores, contadores, operaciones lógicas, etc., su nomenclatura es muy diversa, dependiendo siempre del tipo de autómata y fabricante.

Una vez conocidos los elementos que LADDER proporciona para su programación, resulta importante resaltar cómo se estructura un programa y cuál es el orden de ejecución

3.12.2. Distribución de un programa

En cuanto a su equivalencia eléctrica, podemos imaginar que la línea vertical de la izquierda representa el terminal de alimentación, mientras que la línea

vertical de la derecha representa el terminal de masa. El orden de ejecución es generalmente de arriba hacia abajo y de izquierda a derecha, primero los contactos y luego las bobinas, de manera que al llegar a éstas ya se conoce el valor de los contactos y se activan si procede. El orden de ejecución puede variar de un autómata a otro, pero siempre se respetará el orden de introducción del programa, de manera que se ejecuta primero lo que primero se introduce.

3.12.3. Sistemas Combinacionales

Aunque en los sistemas industriales la programación se centra en procesos secuenciales, no teniendo demasiado interés los procesos combinacionales, es necesario conocer la lógica combinacional ya que en muchas ocasiones es necesaria en la programación secuencial.

Una vez obtenida la función lógica de un problema combinacional, el paso a LADDER o esquema de contactos es muy sencillo. De acuerdo con el álgebra de Boole aplicada a la conmutación, las sumas serán contactos en paralelo, los productos contactos en serie y las negaciones contactos normalmente cerrados.

Veamos en la siguiente tabla con detalle el algebra booleana aplicado a lenguaje ladder.

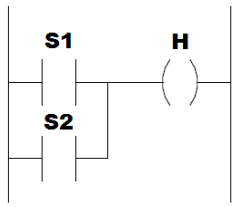
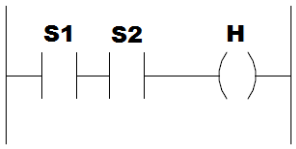
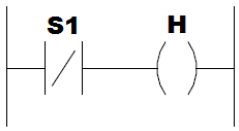
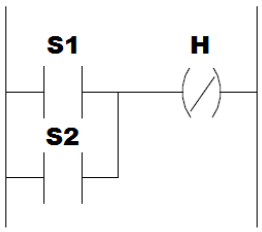
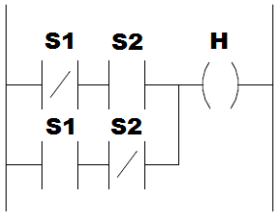
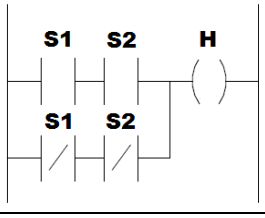
OPERA-DOR	DESCRIPCION	ECUACION	DIAGRAMA LADDER
OR "O"	Se representa con el signo **+** Son dos interruptores en paralelo Implica activar una señal de dos lugares diferentes	H = S1+S2	
AND "Y"	Se representa con el signo • Son dos interruptores en serie Implica activar una señal solo si activo dos interruptores a la vez	H=S1•S2	
NOT "NO"	Se representa con el signo **—** en la parte superior de la variable Es la negación de una señal Es decir si activo una señal la salida se desactiva	H=$\overline{\text{ }}$ S1	
AND NOT (NO "Y")	Es la negación de una operación AND Primero realizo la operación AND y luego la niego	H=$\overline{\text{ }}$ S1•S2	
NOT OR (NOR) (No "O")	Es la negación de una operación OR Primero realizo la operación OR y luego la niego	H=$\overline{\text{ }}$ S1+S2	
OR EXCLUSIVO	Es la conexión de dos conmutadores Es decir puedo prender o apagar una señal solo si activo o desactivo solo una señal de entrada	H=$\overline{\text{ }}$ S1•S2 +S1•$\overline{\text{ }}$ S2	
Nor exclusivo	Es la negación de un circuito OR exclusivo	H=S1•S2+ $\overline{\text{ }}$ S1•$\overline{\text{ }}$ S2	

Tabla III.VI Operadores del Diagrama Ladder

En la siguiente figura se muestra un ejemplo de esquema LADDER para una determinada ecuación.

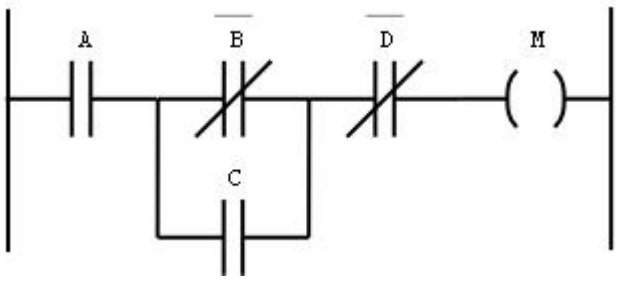


Figura III.14 Ejemplo de Esquema Ladder

LADDER para la función: **M = A(B'+C)D'**

3.12.4. Enclavamiento o Memorización

El término de enclavamiento viene de la conexión tradicional para realizar una función de memoria en los circuitos con relés, es el llamado circuito con auto alimentación. Esto se consigue mediante la conexión de un contacto NA del relé (o contactor) en paralelo con el pulsador de marcha. A continuación puede observarse las dos variantes de este circuito: con prioridad a la descoconexión (figura a) y con prioridad a la conexión (figura b). Se los denomina así puesto que, si el pulsador de paro S0 se mantiene pulsado, en el primer caso no se podría volver a alimentar a la bobina del contactor y se prioriza que esté el circuito desconectado; en cambio si en la segunda figura se mantiene pulsado el pulsador de paro S0, se alimentará la bobina cada vez que se active el pulsador de marcha S1.

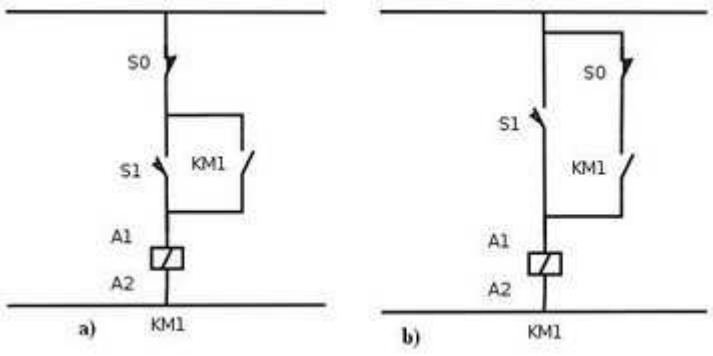


Figura III.15 Circuitos con auto alimentación con prioridad a la desconexión **a)** y a la conexión **b)**

En la siguiente figura se pueden observar los sus esquemas equivalentes en LADDER:

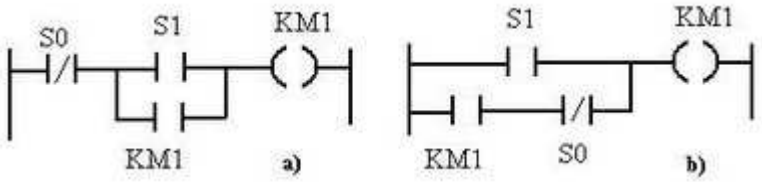


Figura III.16 Circuitos LADDER con auto alimentación

Las ecuaciones asociadas serían:

a) KM1 = $\underline{S0}$ (S1+KM1)

b) KM1= S1+KM1·$\underline{S0}$

Tomemos muy en cuenta el esquema b) puesto que ese será el patrón que seguiremos al implementar la programación de los problemas de automatización.

3.13. Programación en Grafcet (SFC)

3.13.1. Introducción al SFC

Los primeros métodos para el desarrollo de automatismos eran puramente intuitivos, llevados a términos por expertos y desarrollados basándose en la experiencia.

En la actualidad se utilizan métodos más sistemáticos con lo que no es necesario ser un experto en automatismos para llevarlos a término.

En un principio se empleaba como método de resolución de problemas el GRAFCET (**GRAF**ica de **C**ontrol de **E**tapas de **T**ransición), que consiste en un grafo o diagrama funcional normalizado, que permite hacer un modelo del proceso a automatizar, contemplando entradas, acciones a realizar, y los procesos intermedios que provocan estas acciones. Inicialmente fue propuesto para documentar la etapa secuencial de los sistemas de control de procesos a eventos discretos. No fue concebido como un lenguaje de programación de autómatas, sino un tipo de Grafo para elaborar el modelo pensando en la ejecución directa del automatismo o programa de autómata.

Con el tiempo, se ha ido estandarizando y hoy el SFC (**S**equential **F**unction **C**hart) es un diagrama funcional normalizado según la norma IEC 61131-3, que permite hacer una descripción o modelo del proceso a automatizar,

contemplando entradas, acciones a realizar, y los procesos intermedios que provocan estas acciones.

Varios fabricantes en sus autómatas de gama alta hacen este paso directo, lo que lo ha convertido en un potente lenguaje gráfico de programación para autómatas, adaptado a la resolución de sistemas secuenciales. En la actualidad no tiene una amplia difusión como lenguaje, puesto que la mayoría de los autómatas no pueden programarse directamente en este lenguaje, a diferencia del Lenguaje Ladder. Pero se ha universalizado como herramienta de modelado que permite el paso directo a programación mediante el lenguaje Ladder o lenguaje de contactos.

A continuación detallaremos algunas características del SFC o GRAFCET como método de resolución de problemas mecatrónicos o de automatización industrial.

3.13.2. Elementos de programación

Para una comprensión didáctica, resumiremos los elementos básicos de programación que encontraremos en el modelo GRAFCET.

- ETAPA: Define un estado en el que se encuentra el automatismo. Las etapas de inicio se marcan con un doble cuadrado.
- ACCIÓN ASOCIADA: Define la acción que va a realizar la etapa, por ejemplo conectar un contactor, desconectar una bobina, etc.

- TRANSICIÓN : Es la condición o condiciones que, conjuntamente con la etapa anterior, hacen evolucionar el GRAFCET de una etapa a la siguiente, por ejemplo un pulsador, un detector, un temporizador, etc.

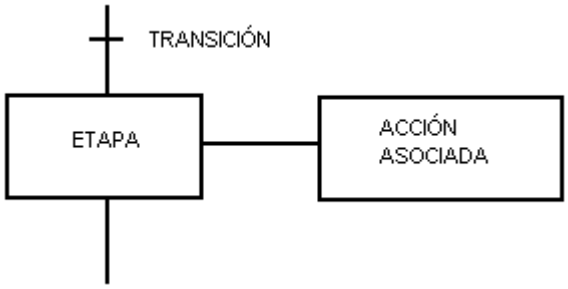
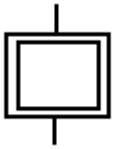
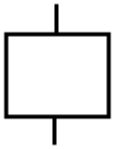


Figura III.17 Elementos básicos del grafcet

Sin embargo, existen otros elementos adicionales que hacen más entendible este modelo de programación:

Elementos GRAFCET de programación		
Símbolo	**Nombre**	**Descripción**
	Etapa inicial	Indica el comienzo del esquema SFC y se activa al poner en RUN el autómata. Por lo general suele haber una sola etapa de este tipo.
	Etapa	Su activación lleva consigo una acción o una espera.
	Unión	Las uniones se utilizan para unir entre sí varias etapas.
	Transición	Condición para desactivarse la etapa en curso y activarse la siguiente etapa, Se indica con un trazo perpendicular a una unión.

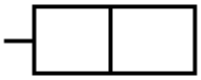
	Direcciona-miento	Indica la activación de una u otra etapa en función de la condición que se cumpla.
	Proceso simultáneo	Muestra la activación o desactivación de varias etapas a la vez.
	Acciones asociadas	Acciones que se realizan al activarse la etapa a la que pertenecen.

Tabla III. Elementos grafcet de programación

3.13.3. Reglas y/o principios básicos

Para realizar el programa correspondiente a un ciclo de trabajo en GRAFCET, se deberán tener en cuenta las siguientes reglas:

- 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.

Podemos describir en conjunto como principios básicos; que un GRAFCET es una sucesión de **etapas**. Cada etapa tiene sus **acciones** asociadas de forma que cuando aquella etapa está activa se realizan las correspondientes

acciones; pero estas acciones no podrán ejecutarse nunca si la etapa no está activa.

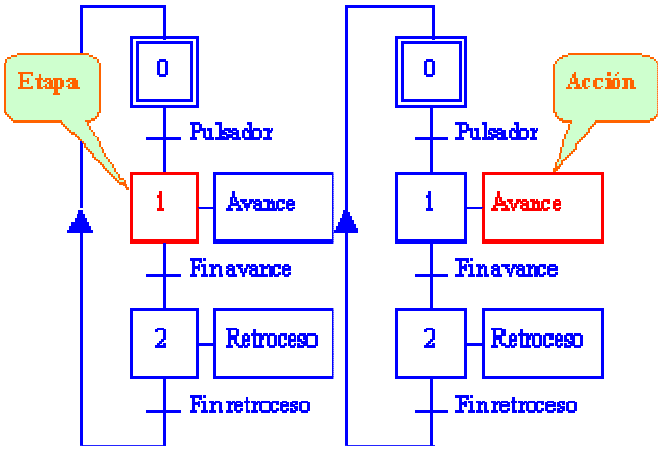


Figura III.18 Etapa - Acción

Entre dos etapas hay una **transición**. A cada transición le corresponde una **receptividad**, es decir una condición que se ha de cumplir para poder pasar la transición. Una transición es **válida** cuando la etapa inmediatamente anterior a ella está activa. Cuando una transición es válida y su receptividad asociada se cumple se dice que la transición es **franqueable**.

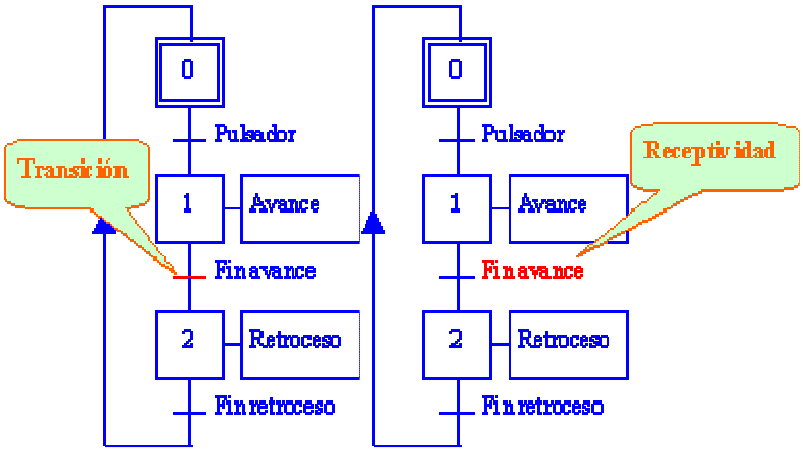


Figura III.19 Transición - Receptividad

Al franquear una transición se desactivan sus etapas anteriores y se activan las posteriores. Las etapas iniciales, que se representan con línea doble, se activan en la puesta en marcha.

Es importante determinar que dependiendo del problema se pueden tener diferentes tipos de secuencias con diferentes resultados o acciones. Por tanto veremos la clasificación de las secuencias y de las acciones.

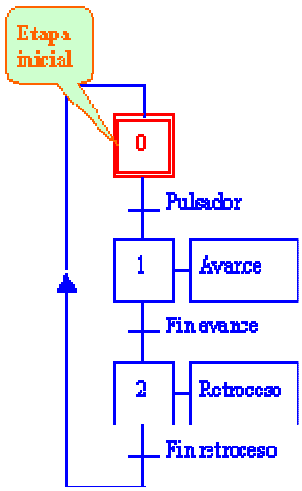


Figura III.20 Etapa inicial

3.13.4. Clasificación de las secuencias

En Grafcet podemos encontrarnos con tres tipos de secuencias:

- Lineales
- Con direccionamientos o alternativa
- Simultáneas

3.13.4.1. Lineales

En las secuencias lineales el ciclo lo componen una sucesión lineal de etapas como se refleja en el siguiente ejemplo:

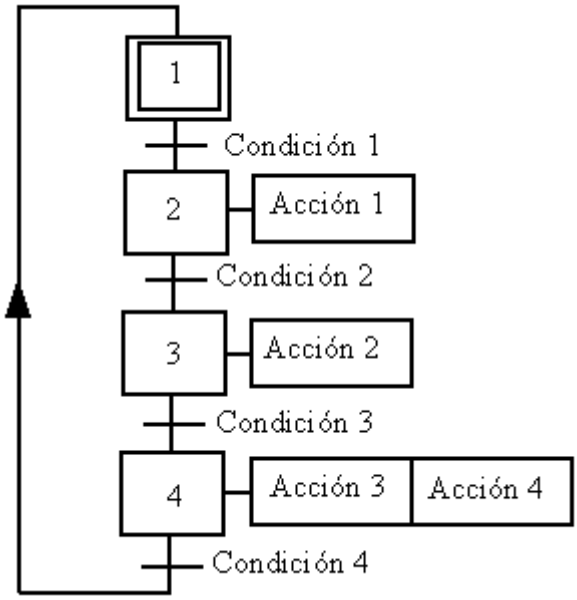


Figura III.21 Secuencia Lineal

El programa irá activando cada una de las etapas y desactivando la anterior conforme se vayan cumpliendo cada una de las condiciones. Las acciones se realizarán en función de la etapa activa a la que están asociadas. Por ejemplo, con la etapa 1 activa tras arrancar el programa, al cumplirse la "Condición 1", se activará la etapa 2, se desactivará la 1, y se realizará la "Acción 1", al cumplirse la "Condición 2", se activará la etapa 3, se desactivará la 2, y se realizará la "Acción 2", al cumplirse la "Condición 3", se activará la etapa 4, se desactivará la 3, y se realizará simultáneamente la "Acción 3" y la “Acción 4”, al

cumplirse la "Condición 4", se activará la etapa 1, se desactivará la 4, completando la secuencia y volviendo a repetirse el ciclo.

3.13.4.2. Con direccionamiento

Si tenemos implantado el GRAFCET con direccionamiento, el ciclo puede variar en función de la condición que se cumpla. En el siguiente ejemplo a partir de la etapa inicial se pueden seguir tres ciclos diferentes dependiendo de cual de las tres condiciones (1, 2 ó 3) se cumpla, (sólo una de ellas puede cumplirse mientras la etapa 1 esté activa):

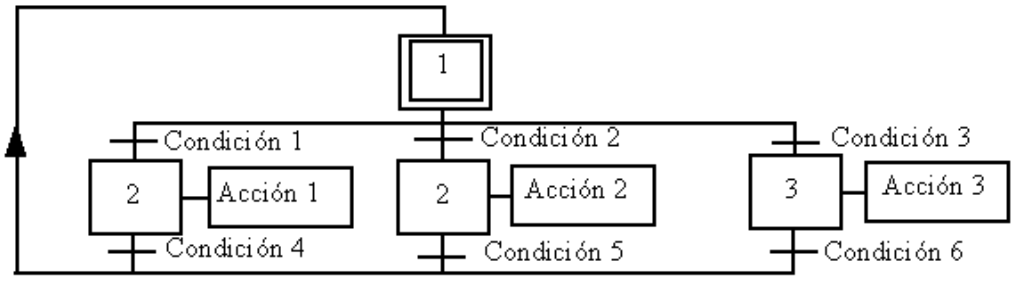


Figura III.22 Secuencia con direccionamiento

3.13.4.3. Simultáneas

En las secuencias simultáneas varios ciclos pueden estar funcionando a la vez por activación simultánea de etapas. En el siguiente ejemplo, cuando se cumple la condición 1 las etapas 2, 3 y 4 se activan simultáneamente:

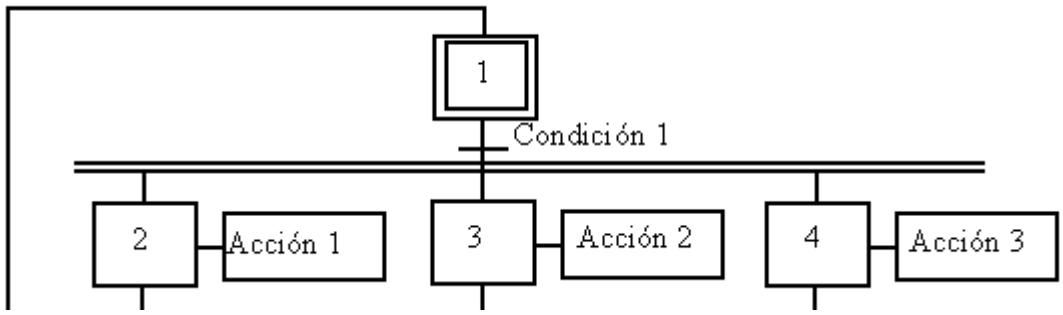


Figura III.23 Secuencia simultánea

Es importante determinar que las secuencias tanto de direccionamiento como las simultáneas son estructuras anidadas, por tanto no deben existir secuencias desconectadas y deben tener un punto en común para volver a la secuencia lineal y completar el ciclo de etapas.

3.13.5. Clasificación de las acciones

En GRAFCET podemos encontrar con alguna o varias de las acciones asociadas a una etapa que se describen seguidamente.

3.13.5.1. Acciones asociadas a varias etapas

Una misma acción puede estar asociada a etapas distintas. Así en el siguiente ejemplo la acción A se realiza cuando está activa la etapa 21 ó la 23 (función **O**):

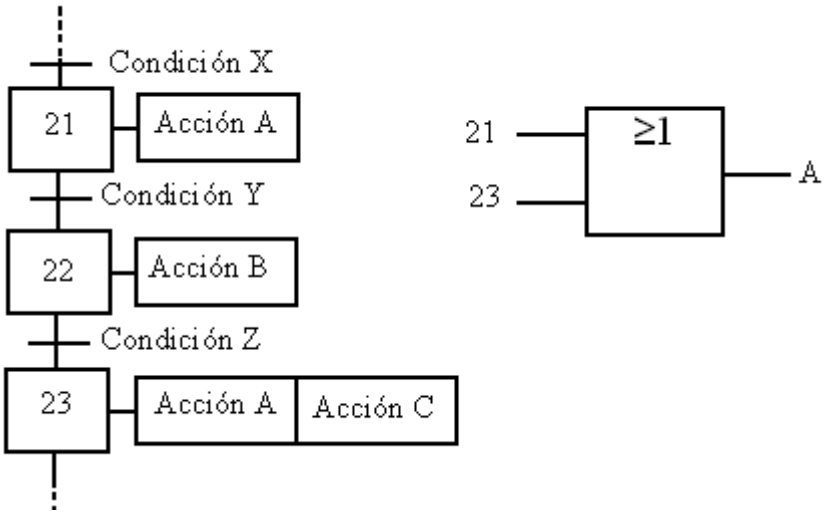


Figura III.24 Acciones asociadas a varias etapas

3.13.5.2. Acciones condicionadas

La ejecución de la acción se produce cuando además de encontrarse activa la etapa a la que está asociada, se debe verificar una condición lógica suplementaria (función Y):

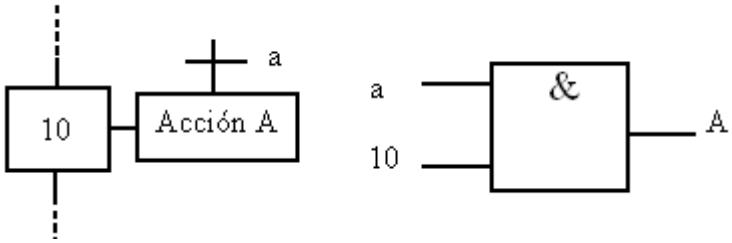


Figura III.25 Acciones condicionadas

3.13.5.3. Acciones temporizadas o retardadas

Es un caso particular de las acciones condicionadas que se encuentran en multitud de aplicaciones. En este caso, el tiempo interviene como una condición lógica más. En el siguiente ejemplo la acción A se realizará durante 10 segundos:

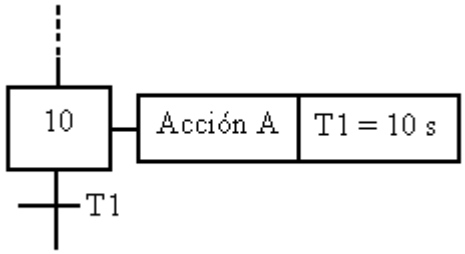


Figura III.26 Acciones temporizadas o retardadas

3.13.6. Niveles del GRAFCET

El GRAFCET puede utilizarse para describir los tres niveles de especificaciones de un automatismo. Estos tres niveles son los que habitualmente se utilizan para diseñar y para describir un automatismo.

3.13.6.1. GRAFCET de nivel 1: Descripción funcional

En el primer nivel interesa una descripción global (normalmente poco detallada) del automatismo que permita comprender rápidamente su función. Es el tipo de descripción que haríamos para explicar lo que queremos que haga la máquina a la persona que la ha de diseñar o el que utilizaríamos para justificar, a las personas con poder de decisión en la empresa, la necesidad de esta máquina.

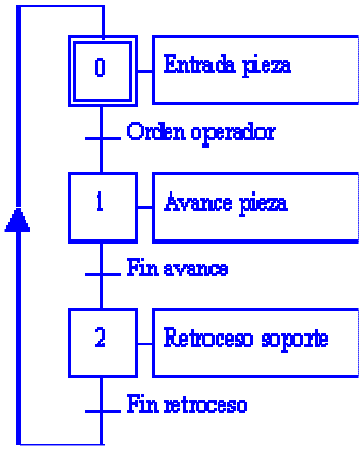


Figura III.27 Grafcet de Nivel 1

Este GRAFCET no debe contener ninguna referencia a las tecnologías utilizadas; es decir no se especifica cómo hacemos avanzar la pieza (cilindro neumático, motor y cadena, cinta transportadora, etc.), ni cómo detectamos su posición (fin de carrera, detector capacitivo, detector fotoeléctrico, etc.), ni tan

solo el tipo de automatismo utilizado (autómata programable, neumática, ordenador industrial, etc.).

3.13.6.2. GRAFCET de nivel 2: Descripción tecnológica

En este nivel se hace una descripción a nivel tecnológico y operativo del automatismo. Quedan perfectamente definidas las diferentes tecnologías utilizadas para cada función. El GRAFCET describe las tareas que han de realizar los elementos escogidos. En este nivel completamos la estructura de la máquina y nos falta el automatismo que la controla.

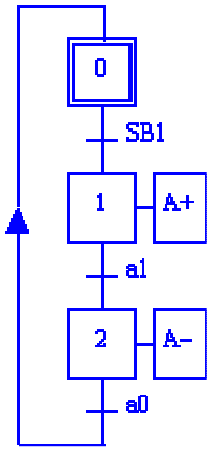


Figura III.28 Grafcet de Nivel 2

3.13.6.3. GRAFCET de nivel 3: Descripción operativa

En este nivel se implementa el automatismo. El GRAFCET definirá la secuencia de actuaciones que realizará este automatismo. En el caso de que se trate, por ejemplo, de un autómata programable, definirá la evolución del automatismo y la activación de las salidas en función de la evolución de las entradas.

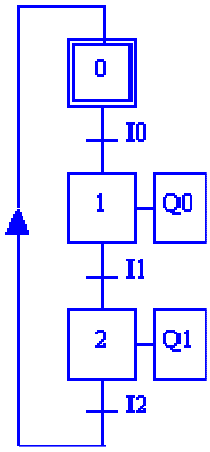


Figura III.29 Grafcet de Nivel 3

3.13.7. Campos de aplicación

El GRAFCET es un poderoso modelo de programación que es aplicado en todo campo en donde se pueda identificar secuencias. Es bien conocido que especialmente en las industrias se implantan sistemas usando esta herramienta de modelado, generando otros sistemas como por ejemplo el sistema SCADA.

El SCADA (**S**upervisory **C**ontrol **A**nd **D**ata **A**cquisition) es una aplicación de software especialmente diseñada para funcionar sobre ordenadores (computadores) en el control de producción, proporcionando comunicación con los dispositivos de campo (controladores autónomos) y controlando el proceso de forma automática desde la pantalla del ordenador. También provee de toda la información que se genera en el proceso productivo a diversos usuarios, tanto del mismo nivel como de otros usuarios supervisores dentro de la

empresa (supervisión, control calidad, control de producción, almacenamiento de datos, etc.).

También se ha empleado el modelo GRAFCET para la resolución de problemas enfocados a la automatización de procesos mecatrónicos. En nuestro medio se ha enfocado más al campo didáctico, pero en otros países se constituye en una disciplina integradora que utiliza las tecnologías de la mecánica, electrónica y tecnología de información para proveernos de productos, procesos y sistemas mejorados.

La Mecatrónica también sirve para la resolución de problemas industriales, científicos y tecnológicos de una manera integral, es decir utilizando un enfoque tecnológico integral y un Ingeniero Mecatrónico desarrolla dichos sistemas incluso si existe innovación tecnológica.

CAPÍTULO IV

4. ESTUDIO DE LOS ELEMENTOS QUE PERMITA LA APLICACIÓN DE LA PROGRAMACION GRAFCET

4.1. Introducción

En todo proceso automatizado o mecatrónico se requerirá tanto la parte software para la programación del proceso como también la parte Hardware para la visualización de los procesos. Esto lo lograremos mediante el PLC Wago 750-842 y su programa o software WAGO I/O PRO 32.

También requeriremos especialmente en los caso de estudio, un equipo mecatrónico. En nuestro caso es la Estación de Distribución que proporciona la marca FESTO para la práctica didáctica de los estudiantes. Describiremos tanto sus componentes, accesorios y secuencias de procesos para una mejor comprensión.

4.2. Introducción al PLC WAGO 750-842

Una de las herramientas para la automatización de procesos en PLC es el WAGO, modelo 750 - 842, cuyos creadores son la empresa **WAGO Kontakttechnik GmbH & Co. KG**, siendo una empresa alemana que fabrica componentes para la conexión eléctrica de tecnología y componentes para la descentralización de tecnología de automatización. Su alcance tanto en

El PLC Wago consiste de varios componentes que son: módulos y aplicaciones especificas para nodos fieldbus. Un nodo fieldbus está compuesto de tres partes: al principio un acoplador fieldbus (Coupler/Controller) o un controlador fieldbus programable, en el medio se encuentran varios módulos de E/S, y al final un modulo de terminación o fin, tal como muestra la siguiente figura.

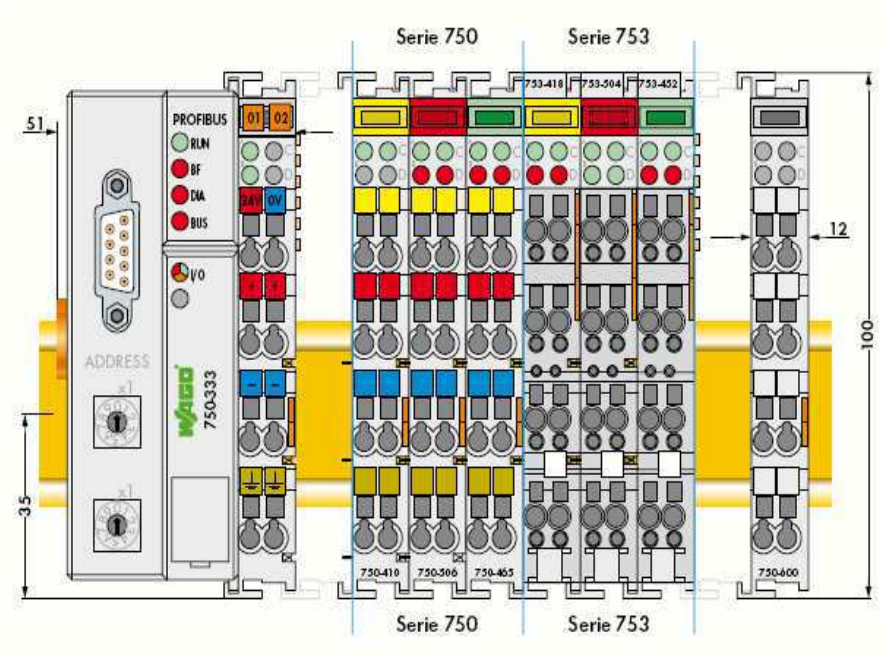


Figura IV.30 Partes del PLC WAGO 750-842

Describamos a continuación cada una de sus partes.

4.2.1. Controlador Fieldbus

El Controlador Fieldbus Programable [1] forma el vínculo entre los módulos de entrada/salida y los dispositivos de campo. Todas las funciones de control de entrada/salida deben llevarse a cabo en el Controlador Fieldbus Programable para su correcto funcionamiento.

El Controlador Fieldbus Programable Wago 750-842 o también llamado PFC, combina la funcionalidad de Ethernet TCP/IP del Controlador Fieldbus Programable 750-342 con la funcionalidad de un PLC. La programación de las aplicaciones es realizada con WAGO- I/O-PRO 32 en conformidad con IEC 61131-3

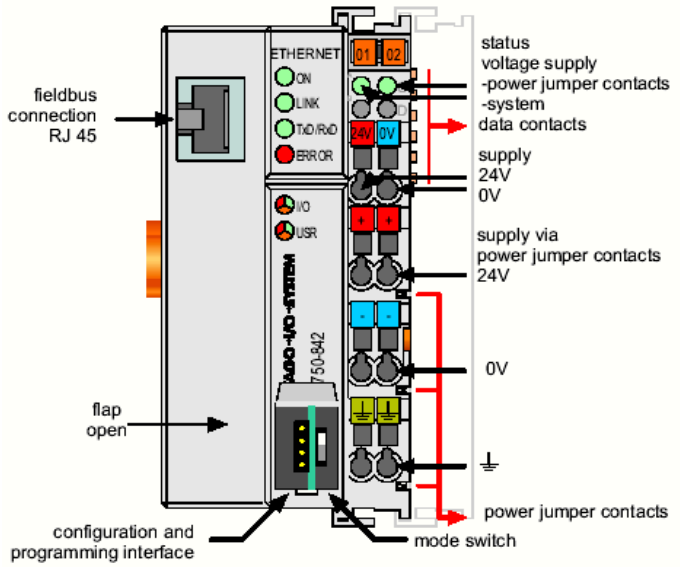


Figura IV.31 Controlador Fieldbus

El controlador Fieldbus consta de:

- El suministro de energía del dispositivo está compuesto por un módulo de suministro de energía interno para la alimentación del sistema, así como de jumpers de contacto para suministrar energía a los módulos ensamblados de entrada y salida.
- Una Interfaz fieldbus para la conexión del bus.
- LED's para visualizar el estado de una operación, la comunicación del bus, el voltaje y para diagnosticar los mensajes enviados y recibidos.
- Interfaz de configuración y programación.
- Un switch para realizar operaciones especiales.
- Bus interno para la comunicación de los módulos E/S con la Interfaz fieldbus.

4.2.2. Módulos E/S.

En los módulos de E/S [2], los datos del proceso entrante son convertidos de acuerdo a los diferentes requerimientos de los módulos. Hay entradas y salidas, digitales y analógicas, y además existen módulos de E/S especiales que son utilizables para una variedad de funciones específicas.

4.2.3. Módulo de terminación

El módulo de terminación o fin del módulo [3], es necesario para el correcto funcionamiento del fieldbus Wago. La terminación del módulo siempre se coloca al final de los módulos de entrada/salida, para obtener una

terminación del fieldbus Wago. Este módulo no tiene ninguna función de entrada/salida. Todas las señales de entrada de los sensores están agrupadas en el controlador. Según la IEC 61131-3, el tratamiento de los datos del proceso ocurre localmente en el PFC.

Los resultados de este tratamiento de datos del proceso pueden ser publicados directamente en los actuadores o pueden ser transmitidos a un sistema de control superior a través del bus.

4.2.4. Instalación Hardware del PLC

Se procede a enclavar tal como se muestra en la figura para las extensiones tanto de entrada como de salida de información.

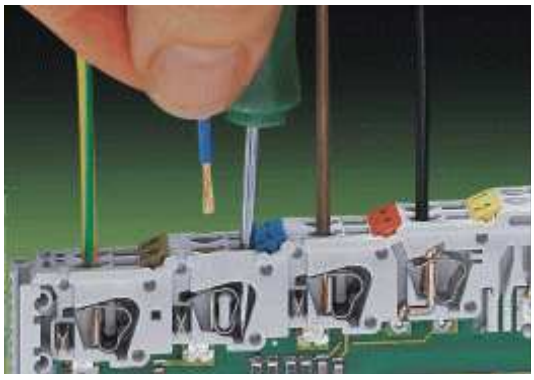


Figura IV.32 Instalación de cables

4.2.5. Instalación Software y configuración del PLC

El software que empleamos para configurar el PLC Wago 750-842 es el WAGO-IO-PRO 32, Versión 2.2. A continuación describiremos los pasos para la instalación:

1. Localizar la carpeta I O Pro32 y acceder al ejecutable SETUP.

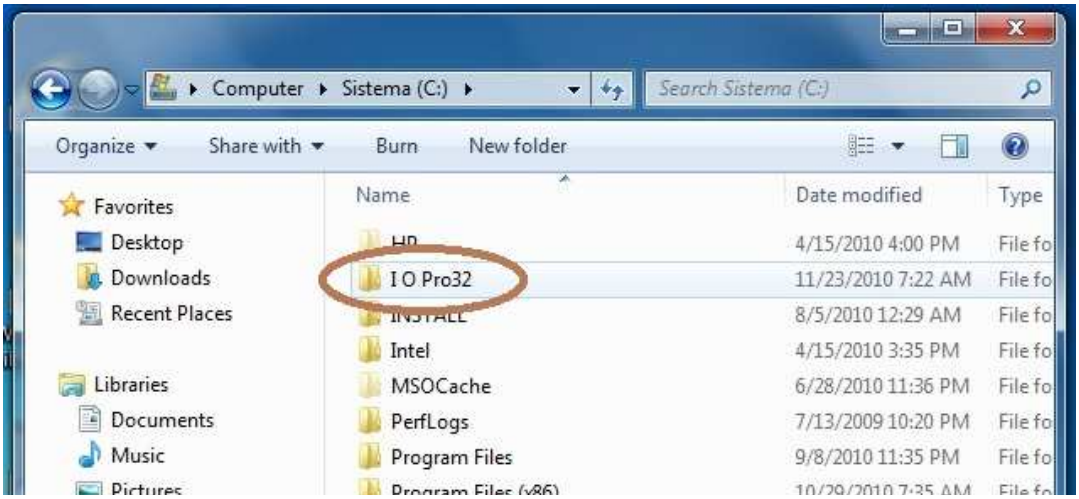


Figura IV.33 Instalación de I O Pro

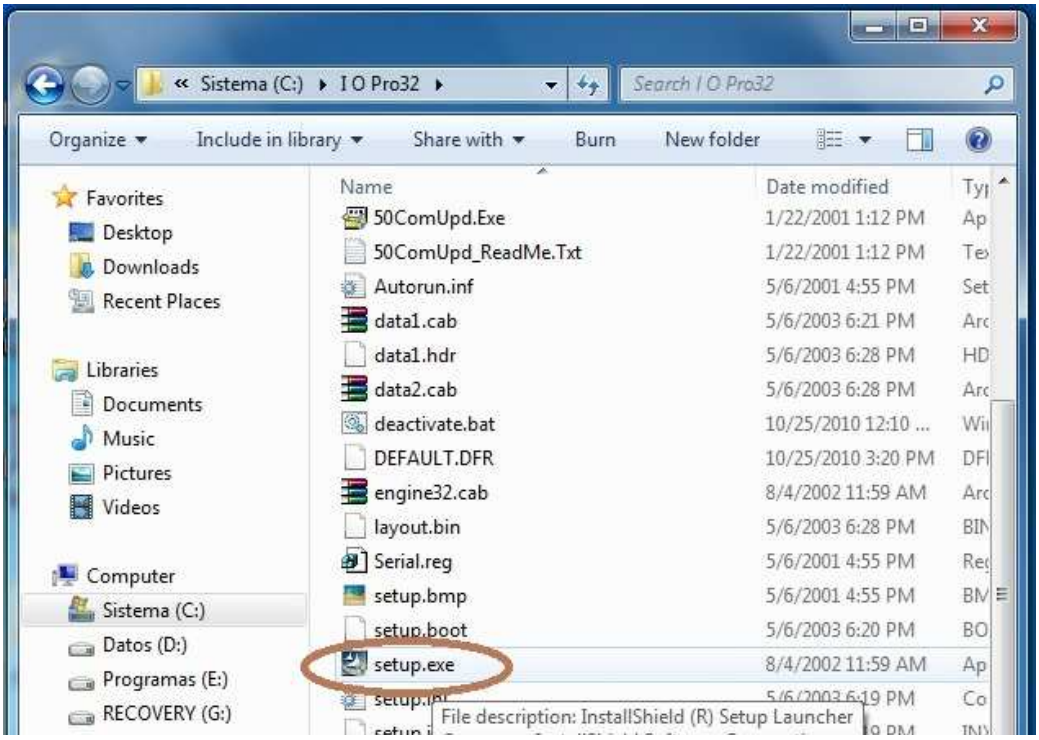


Figura IV.34 El programa SETUP de WAGO

2. Al pulsar se presentará la pantalla en donde debemos escoger el idioma en el cual se instalará el programa. En nuestro caso escogemos el idioma INGLES.

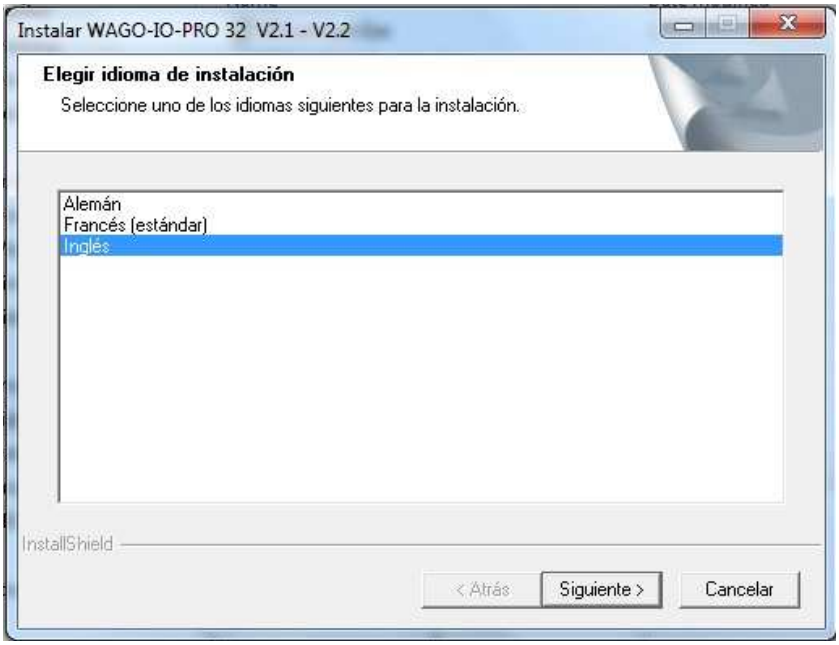


Figura IV.35 Instalación WAGO 2

3. A continuación se presenta el logo de bienvenida del programa WAGO IO PRO 32 y los primeros pasos para continuar con la instalación.

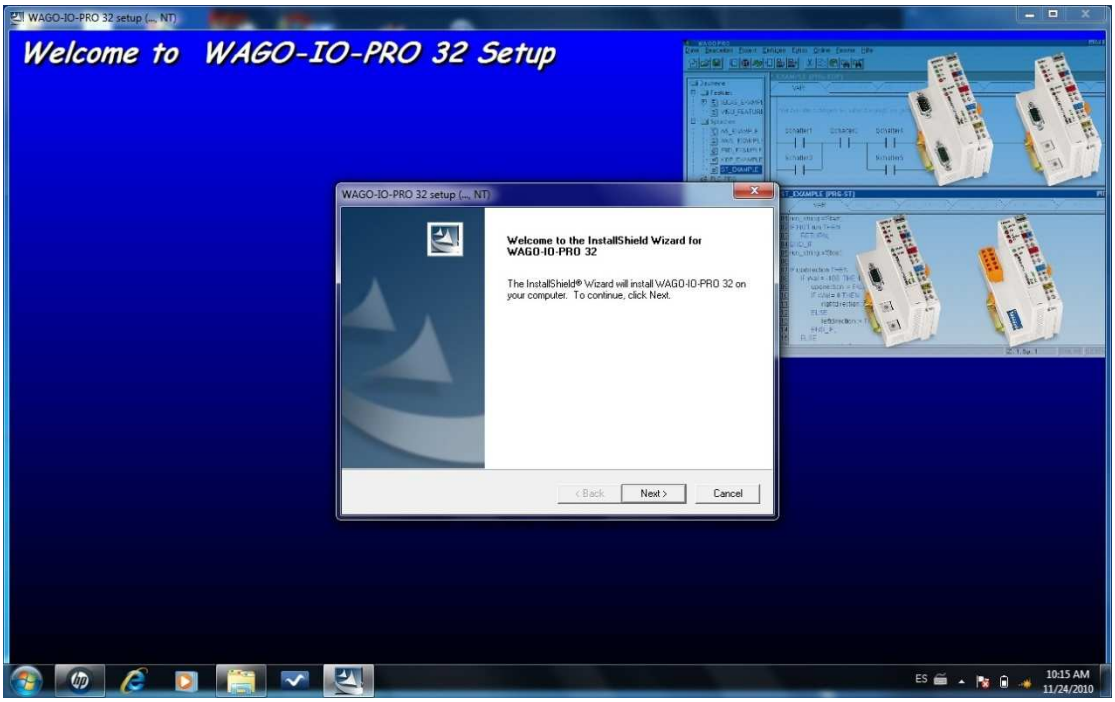


Figura IV.36 Instalación WAGO 3

4. Pulsamos Siguiente y debemos llenar los datos requeridos y el serial que está en un archivo en formato .rtf, el mismo que se encuentra en la carpeta I O Pro.

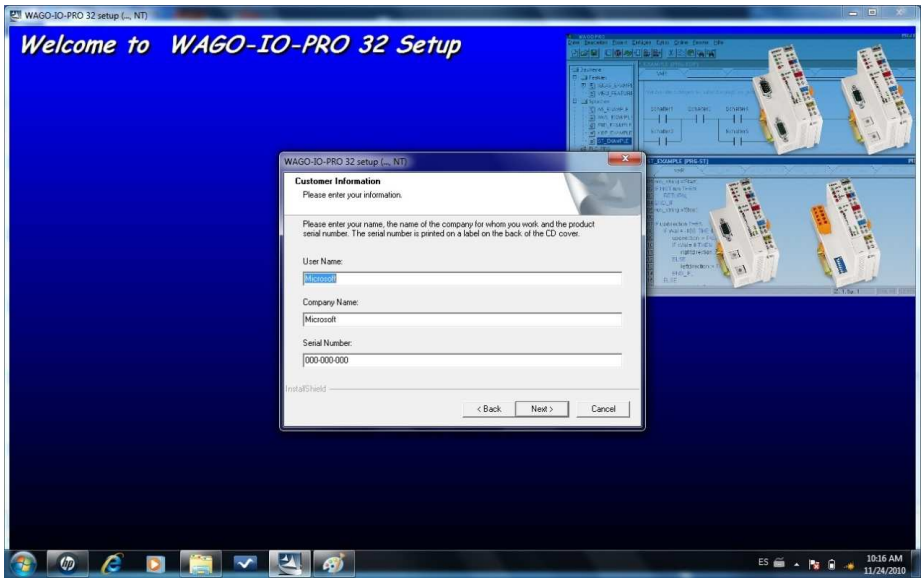


Figura IV.37 Instalación WAGO 4

5. Después de pulsar Siguiente, confirmamos los datos que están los correctos.

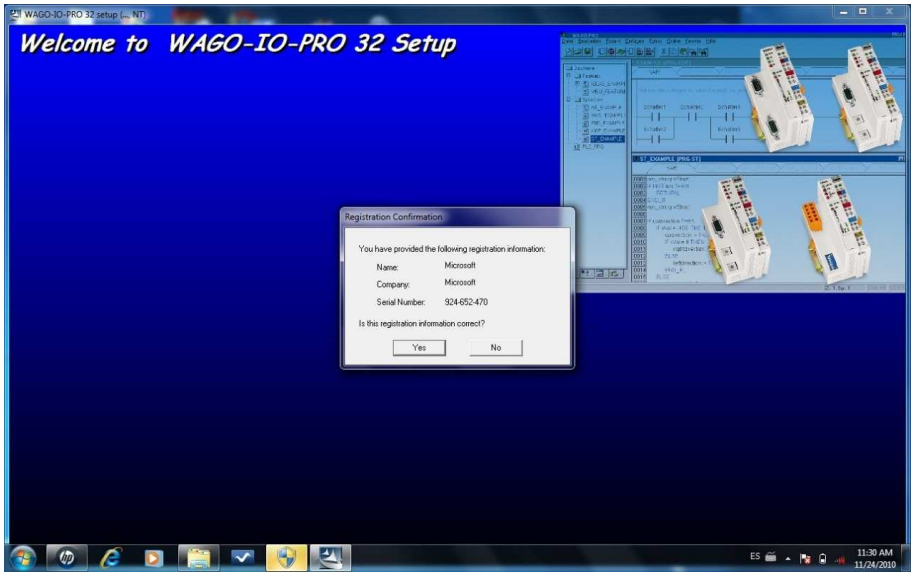


Figura IV.38 Instalacion WAGO 5

6. Al pulsar siguiente nos identifica en donde se va a instalar el programa por defecto, o si deseáramos ubicarlo en otro sitio lo podemos hacer. En nuestro caso conservamos la ubicación predeterminada del programa.

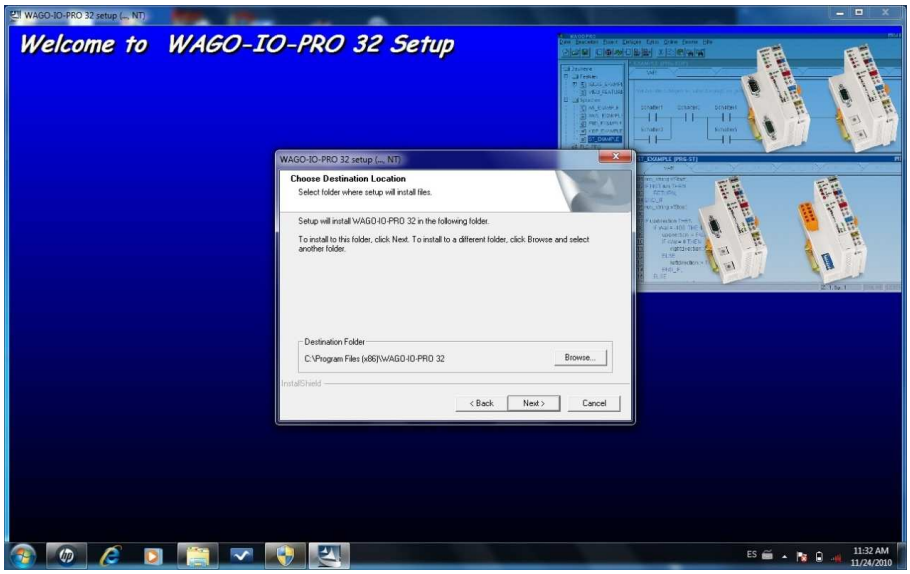


Figura IV.39 Instalación WAGO 6

7. En la siguiente pantalla se ve tres opciones de instalación. Para nosotros, queremos solo instalar el programa, por lo tanto escogemos la opción Completar nueva instalación.

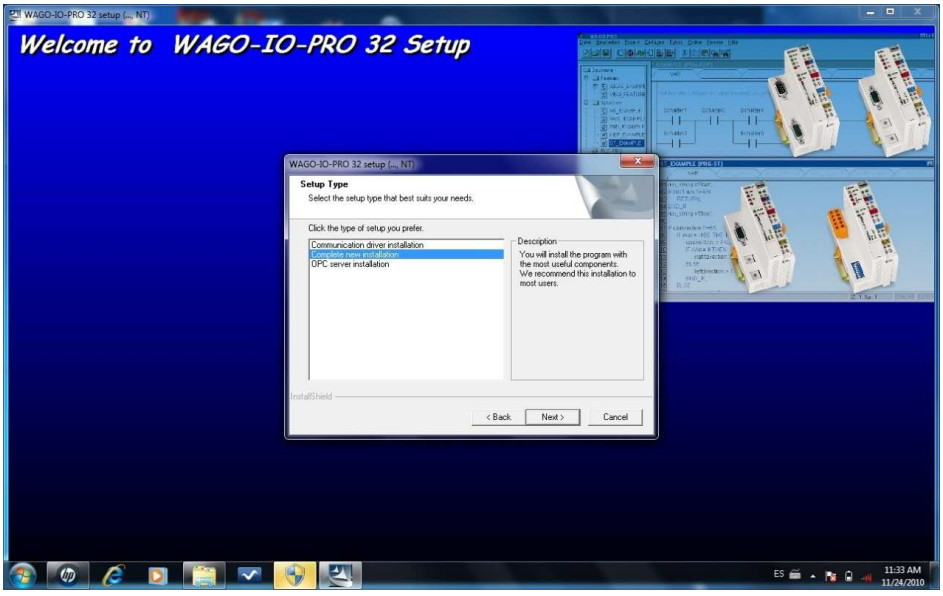


Figura IV.40 Instalación WAGO 7

8. De aquí en adelante solo nos pide la versión en la cual vamos a trabajar, escogemos la versión 2.2 y nos indica los accesos directos, nos advierte que debemos reiniciar el equipo para no tener ningún problema y concluimos con la instalación.

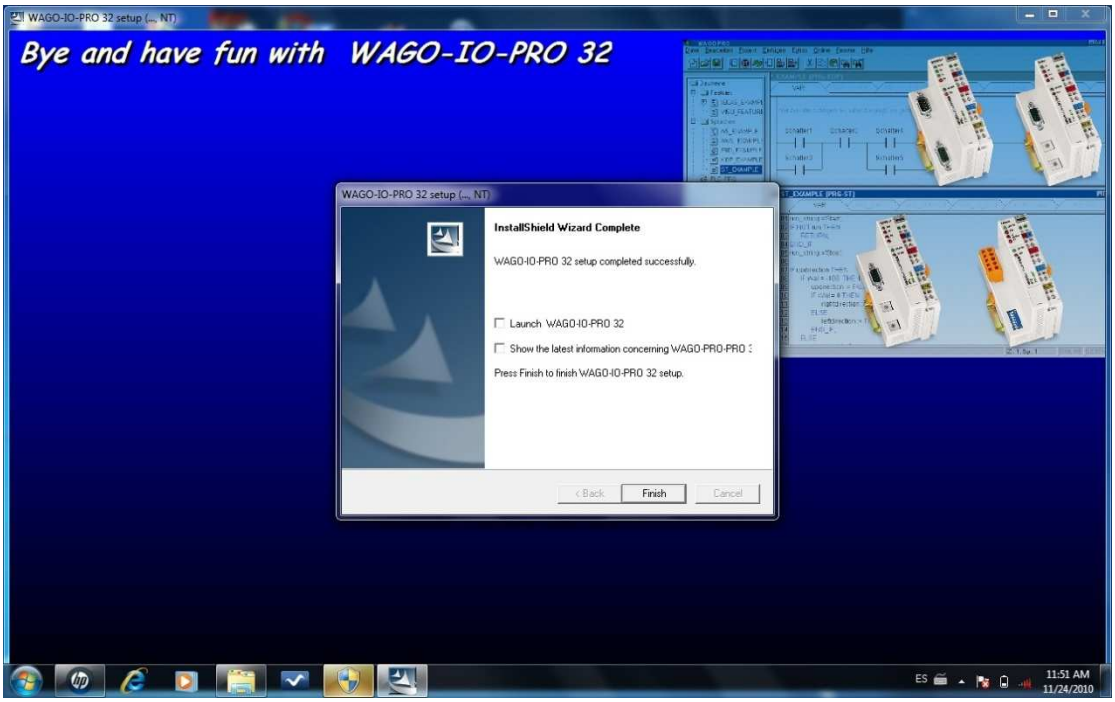


Figura IV.41 Instalación WAGO 8

La configuración del PLC se realiza cada vez que queramos grabar o simular la aplicación. Por lo tanto indicaremos en los casos de estudio.

4.3. Introducción al Equipo Mecatrónico

Tanto es el interés generado por perfeccionar técnicas de programación que inclusive empresas dedicadas a fabricar equipos mecatrónicos e industriales han capacitado para poder promocionar sus equipos.

Tal es el caso de la Compañía Festo, proveedor mundial de soluciones de automatización mediante tecnología neumática, electrónica y de redes para todo tipo de procesos y actividades industriales. Suministran desde componentes independientes a sistemas completos, así como asesoramiento y formación tecnológica, empresarial y didáctica.

4.3.1. Sistemas MPS®

Los sistemas MPS® 200 de Festo son un conjunto de estaciones, PLCs, accesorios y Software que permiten capacitar a los estudiantes en diferentes ámbitos como; Mecánica, Neumática, electrónica, programación. Esto garantiza una formación efectiva ya desde el comienzo. Van desde pequeños sistemas completos hasta equipos enteros de laboratorios de Mecatrónica.

Figura IV.42 Estaciones MPS

Generalmente tenemos los siguientes elementos:

- Estaciones MPS®

- Todos los accesorios necesarios, tales como mesas rodantes, fuentes de alimentación, consolas de control, juegos de piezas de trabajo, etc.
- Paquete de control con software de programación y cables
- Software de visualización y simulación

Uno de los componentes principales del sistema MPS® 200 es la Estación de Distribución.

4.3.2. Estación de Distribución MPS®

El objetivo de la estación de Distribución es de separar piezas. Se encarga de receptar piezas cilíndricas, separarlas y transportarlas a otra estación o proceso de producción. El equipo mecatrónico será empleado en el segundo caso de estudio. Detallemos primeramente los accesorios y elementos principales de la estación de distribución.

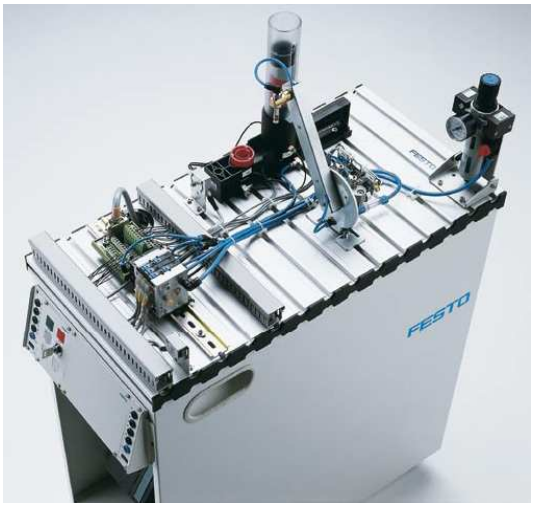


Figura IV.43 Estación de Distribución MPS 200

4.3.2.1. Elementos principales

Los elementos más importantes de la estación de distribución son:

- **Placa perfilada de aluminio**

  La placa perfilada en aluminio anodizado forma la base de todos los equipos de formación de Festo Didactic. Todos los componentes de fijan de forma segura en las ranuras de la placa perfilada. Hay ranuras en cada laso y, si es necesario, pueden montarse componentes en ambos lados. Las ranuras son compatibles con el sistema de perfiles ITEM. Tapetas laterales incluidas. Dimensiones de retícula: 50 mm.

Figura IV.44 Placa de aluminio

Para instalación sobre mesas, recomendamos usar los correspondientes pies de goma. Tamaños 350 x 1100 mm y 350 x 250 mm suministradas sin tapetas laterales.

- **Módulo almacén apilador**

  Separa piezas de un almacén. Un cilindro de doble efecto empuja la pieza de la parte inferior sacándola del almacén por gravedad, contra un tope mecánico. La posición del cilindro es detectada por sensores inductivos.

Figura IV.45 Modulo almacen apilador

En la Estación de Distribución el módulo de almacén Apilador, separa piezas de un almacén. Un cilindro de doble efecto empuja la pieza de la parte inferior sacándola del almacén por gravedad, contra un tope mecánico. La posición del cilindro es detectada por sensores inductivos. Las velocidades de avance y retroceso pueden ajustarse por medio de válvulas reguladoras de caudal de un sólo sentido.

- **Módulo cambiador**

Es un dispositivo manipulador neumático. Se utiliza una ventosa para tomar piezas y recolocarlas a posiciones de 0° a 180° uti lizando un actuador semigiratorio. La posición final se detecta por medio de sensores.

Figura IV.46 Módulo cambiador

- **Pinza de aspiración**

La pinza de aspiración del módulo cambiador sujeta la pieza. El vacío es generado en la placa de vacío del terminal de válvulas CP por medio del principio Venturi y es supervisada por un presostato.

Figura IV.47 Aspirador

- **Terminal E/S (SysLink)**

El terminal de E/S es la unidad central del concepto MPS® SysLink. Permite el cableado de 8 entradas y 8 salidas a un conector. Los LEDs están montados en los terminales de entrada y salida para indicación del estado del circuito y localización de averías sistemática.

Figura IV.48 Terminal E/S

El terminal puede montarse en un raíl DIN.

- Terminales para entradas: 8

- Terminales para salidas: 8

➢ **Vacuostato**

Vacuostato tipo mecánico con punto de conmutación, es ajustable y posee indicador de estado (LED).

Figura IV.49 Vacuóstato

➢ **Válvula de cierre con filtro regulador**

Filtro regulador con manómetro, válvula de cierre, racores rápidos y acoplamientos rápidos, montados en un soporte basculante.

Figura IV.50 Válvula de cierre

4.3.2.2. Accesorios requeridos

Para complementar la Estación de distribución, requerimos los siguientes accesorios

- **Juego de herramientas**

  El juego de herramientas facilita el trabajo en las estaciones. Un práctico Minisystainer incluye:

Figura IV.51 Herramientas

- Regla de acero 200 mm
- Llaves fijas 7, 8, 9, 10
- Llave inglesa
- Alicate de corte oblicuo
- Alicate desforrador
- Alicate de montaje de terminales

- Juego de destornilladores hexagonales, 1.5 – 6
- Destornillador hexagonal, 0,9; 1.3
- Destornillador de estrella PZ02 – corto
- Destornillador plano 2,5 x 75; 4,0 x 100
- Destornillador plano 1.2 – 1.6
- Cortatubos
- Cortador de fibras ópticas
- Juego de piezas de trabajo, rojo, negro, plata
- 100 bridas 2,5 x 100
- 100 terminales de cable 0,25
- 100 terminales de cable 0,75

➢ **Fuente de alimentación de sobremesa**

Son reguladores que permiten dar energía al equipo. Algunas de sus características son:

Figura IV.52 Fuente de alimentación

- Tensión de entrada: 85 – 265 V AC (47 – 63 Hz)
- Tensión de salida: 24 V DC, a prueba de cortocircuitos
- Corriente de salida: máx. 4,5 A

- Dimensiones: 115 x 155 x 200 mm

➢ **Mesa rodante**

La mesa rodante convierte una estación MPS® en una unidad móvil y compacta. Es fácil montar la estación en la mesa rodante. También puede montarse el EduTrainer® Universal. Orificios en las paredes laterales y posterior facilitan el tendido ordenado de los cables.

Figura IV.53 Mesa Rodante

La parte frontal está equipada con fijaciones para el panel de control. La mesa se suministra completa, con ruedas.

- Altura (incl. ruedas, hasta el borde superior de la placa perfilada): 750 mm
- Anchura: 350 mm

- Fondo: 700 mm

➢ **Consola de control MPS® para SysLink**

La consola de control MPS® facilita el funcionamiento de la estación MPS®. SysLink o AS-interface – los diversos interfaces aseguran versatilidad de uso. Completamente montada con panel de operador, paneles de comunicación, panel de reserva y bastidor de montaje con conector SysLink.

Figura IV.54 Consola de control

➢ **Juego de piezas cilíndricas**

El juego de piezas comprende 4 cuerpos de material sintético negros, 4 cuerpos de material sintético rojos y 4 cuerpos de aluminio del cilindro.

- Diámetro exterior: 40 mm
- Alto (negras): 22,5 mm
- Alto (rojas y de aluminio): 25 mm

Figura IV.55 Juego de piezas cilíndricas

4.3.2.3. Accesorios complementarios

Los accesorios complementarios consisten en tubos, terminales, bridas sujeta cables etc. que permitan complementar la terminación de la estación y cumplir con el estándar de calidad requerido especialmente en las competencias de Mecatrónica.

4.3.2.4. Secuencia del proceso

La secuencia del proceso es la siguiente: existen algunas piezas en el tubo del módulo almacén apilador. Un cilindro de doble efecto expulsa las piezas individualmente. El módulo cambiador sujeta la pieza individual por medio de una ventosa. El brazo del cambiador, que es accionado por un actuador giratorio, transporta la pieza al punto de transferencia de la estación posterior.

CAPITULO V

5. ANALISIS Y DISEÑO DE CASOS DE ESTUDIO

5.1. Introducción

En automatización industrial es común ver secuencias sencillas, con contactores, timers, etc, y por tanto ha sido una herramienta principal el uso del método ladder o lenguaje de contactos para resolver dichos problemas. Es por ello que inclusive tanto los PLC como sus respectivos softwares y programas didácticos están enfocados a dicho lenguaje.

El inconveniente es cuando se dispone de problemas complejos. El programa se ha convertido en un pequeño lío de símbolos ladder. Cuando se requieren hacer secuencias complejas por un autómata o PLC, el programa en ladder crece y resulta bastante ininteligible, sobre todo si uno no es quién lo ha creado. Por eso, es necesario hacer las cosas de otra forma, más ordenada y además que permita entender el programa más fácilmente. Una posibilidad es

el uso de la metodología tradicional incluyendo al lenguaje Grafcet como método, el mismo que es ampliamente utilizado para resolver problemas.

5.2. Pasos de la metodología tradicional

Básicamente el método tradicional consiste en los siguientes pasos:

- Establecer la secuencia del problema planteado, ya sea con un Grafcet funcional o con un bosquejo gráfico del problema. (grafcet de nivel 1)
- Elaborar el grafcet descriptivo de etapas y transiciones (grafcet de nivel 2).
- Diseñar la tabla de asignaciones
- Elaborar el grafcet operativo (grafcet de nivel 3)
- Extraer las ecuaciones necesarias, tomando en cuenta ciertas reglas.
- Trasladar a lenguaje ladder, y
- Depurar el programa.

Detallaremos a continuación cómo se resuelve normalmente un problema de secuencias combinando el lenguaje Grafcet como método y luego implementando en lenguaje de contactos para su posterior programación. A continuación plantearemos dos casos de estudio en los cuales se programará con dicha metodología habitual, para posteriormente demostrar que la propuesta metodológica de programación de PLC planteada en este trabajo investigativo es eficaz y que se resuelve en poco tiempo.

5.3. Implementación de la metodología habitual

5.3.1. Metodología Tradicional

Al analizar la metodología que usualmente involucra la resolución de problemas de automatización y de mecatrónica generalmente se emplea el uso del grafcet como método inicial y el lenguaje ladder como método de implantación del problema, debido a que en la mayoría de los PLC solo soportan el lenguaje ladder.

Antes de seguir los pasos para resolver problemas de automatización, recordemos los elementos que tiene el grafcet:

- ETAPA: Define un estado en el que se encuentra el automatismo. Las etapas de inicio se marcan con un doble cuadrado. Este elemento lo asociamos con las memorias y lo denominaremos con una **M**.
- ACCIÓN ASOCIADA: Define la acción que va a realizar la etapa, por ejemplo conectar un contactor, desconectar una bobina, etc. Este elemento lo asociamos con las salidas y lo denominaremos con una **Q**.
- TRANSICIÓN: Es la condición o condiciones que, conjuntamente con la etapa anterior, hacen evolucionar el GRAFCET de una etapa a la siguiente, por ejemplo un pulsador, un detector, un temporizador, etc. Este elemento lo asociamos con las entradas y lo denominaremos con una **I**. (en otros PLC lo denominan con la **E**).-

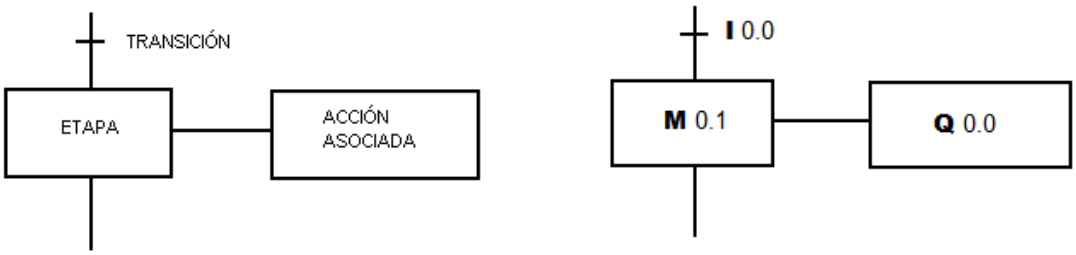


Figura V.56 Diagrama Grafcet Nivel 1 y Nivel 3

Para establecer las **ecuaciones** a partir del grafcet y de ahí a ladder, vamos a detallar la siguiente fórmula:

Etapa_Actual = Etapa_Anterior x Condición + Etapa_Actual x Etapa_Siguiente_Negada

Que implica lo siguiente: ***"La etapa o memoria actual es igual a la etapa o memoria anterior por la condición anterior más la etapa o memoria actual por la etapa siguiente negada".*** En conclusión, constituye el mismo concepto de enclavamiento con prioridad a la conexión:

Ecuación

$$M\_act = \underbrace{M\_ant * Cond}_{\text{ENCENDIDO}} + \underbrace{M\_act * \overline{M\_sig}}_{\text{MEMORIA APAGADO}}$$

Ladder

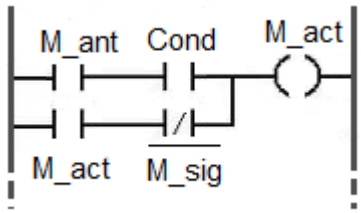


Figura V.57 Diagrama ladder de Grafcet Nivel 1

Notamos que al igual que el enclavamiento con prioridad a la conexión, dispone de ENCENDIDO, MEMORIA Y APAGADO. Las salidas que disponga el grafcet solamente se deben asignar con la etapa o memoria en la cual están enlazadas. Por ejemplo, si tenemos este grafcet:

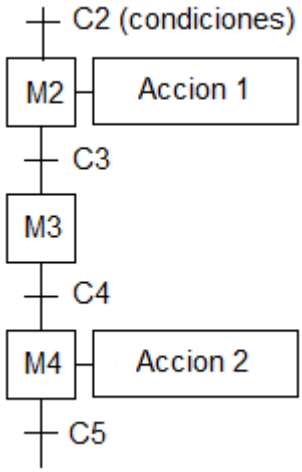


Figura V.58 Diagrama Grafcet demo

Estas serían las ecuaciones:

$$M2 = M1 \cdot C2 + M2 \cdot M3$$

$$M2 = M1 \cdot C2 + M2 \cdot M3$$

$$M2 = M1 \cdot C2 + M2 \cdot M3$$

$$Accion1 = M2$$

$$Accion2 = M4$$

Y a su vez el programa en ladder sería:

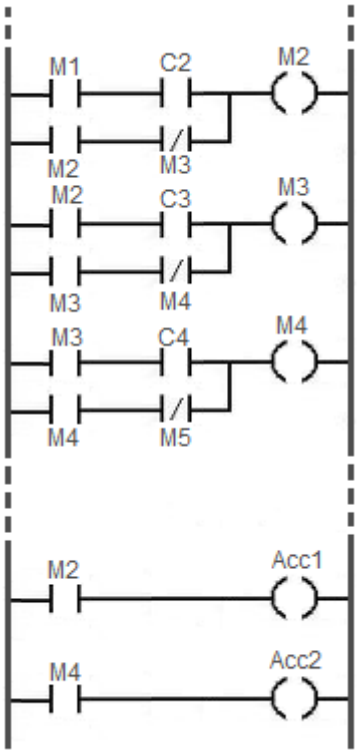


Figura V.59 Diagrama Ladder demo

5.3.2. Método de Programación

El método de programación tradicional dispone de los siguientes pasos:

a) Establecer la secuencia del problema que se quiere resolver mediante el grafcet de nivel 1.

b) Elaborar el grafcet de nivel 2 con todos los elementos técnicos y simbologías que se requiera.

c) Crear la tabla de asignaciones.

d) Elaborar el grafcet de nivel 3.

e) Extraer las ecuaciones a partir del grafcet de nivel 2 o 3.

f) Programar en lenguaje ladder a partir de las ecuaciones.

g) Trasladar al software del PLC

h) Depurar el programa.

5.3.3. Casos de Estudio

A continuación veremos los siguientes casos de estudio:

- ***Caso de Estudio 1:*** Control automático de un semáforo.
- ***Caso de Estudio 2:*** Estación de Distribución MPS.

CASO DE ESTUDIO 1

Se desea realizar el control automático de un semáforo para peatones. El peatón al llegar a la bocacalle observa el semáforo de peatón en rojo, pero a su vez el semáforo de la calle está en verde, el mismo que dura 30 s, después se observa que el semáforo de la calle está en amarillo durante 5 s y posteriormente cambia a rojo, cambiando el semáforo del peatón a verde, después de 20 s el semáforo de la calle cambia a verde y el del peatón cambia a rojo, comenzando un nuevo ciclo.

SOLUCION

a) Establecer la secuencia del problema mediante el grafcet de nivel 1.

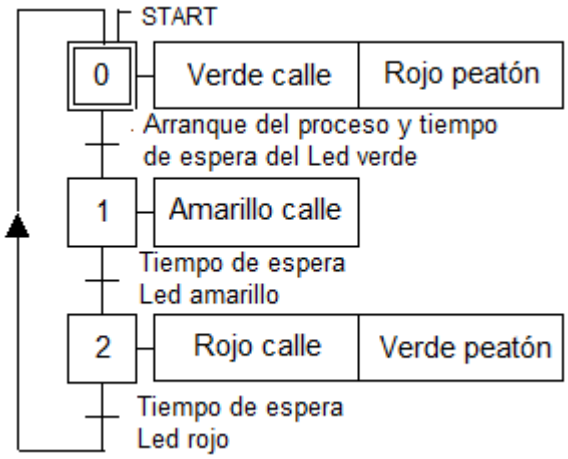


Figura V.60 Grafcet Nivel 1 Semáforo

b) Elaborar el grafcet de nivel 2 con todos los elementos técnicos y simbologías que se requiera.

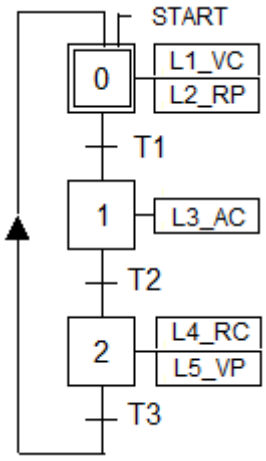


Figura V.61 Grafcet Nivel 2 Semáforo

c) Crear la tabla de asignaciones.

ENTRADAS	SALIDAS	MEMORIAS	TEMPORIZADORES
I0.0=START	Q0.0=L1_VC	M0.0=M0	t1=T1=30s
	Q0.1=L2_RP	M0.1=M1	t2=T2=5s
	Q0.2=L3_AC	M0.2=M2	t3=T3=20s
	Q0.3=L4_RC	M0.3=M3	
	Q0.4=L5_VP		

Tabla V.VIII Tabla de asignaciones – Semáforo

En el caso de los temporizadores y otras variables, dependerá del programa del PLC que se disponga para ver su sintaxis, sin embargo generalizaremos con variables t1, t2, etc. y sus respectivos valores de tiempo, tal como se describe en la tabla anterior.

d) Elaborar el grafcet de nivel 3.

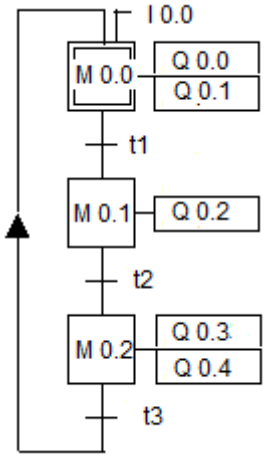


Figura V.62 Grafcet Nivel 3 Semáforo

e) Extraer las ecuaciones a partir del grafcet de nivel 2 o 3.

Para ello debemos usar la siguiente ecuación

$$M\_act = \underbrace{M\_ant * Cond}_{\text{ENCENDIDO}} + \underbrace{M\_act * \overline{M\_sig}}_{\text{MEMORIA APAGADO}}$$

Estas son las ecuaciones si extraemos del grafcet de nivel 2

$$M0 = START + M2 \cdot T3 + M0 \cdot \overline{M1}$$

$$M1 = M0 \cdot T1 + M1 \cdot \overline{M2}$$

$$M2 = M1 \cdot T2 + M2 \cdot \overline{M0}$$

$$L1\_VC = M0$$

$$L2\_RP = M0$$

$$L3\_AC = M1$$

$$L4\_RC = M2$$

$$L5\_VP = M2$$

Estas son las ecuaciones si extraemos del grafcet de nivel 3

$$M0.0 = I\,0.0 + M0.2 \cdot t3 + M0.0 \cdot \overline{M0.1}$$
$$M0.1 = M0.0 \cdot t1 + M0.1 \cdot \overline{M0.2}$$
$$M0.2 = M0.1 \cdot t2 + M0.2 \cdot \overline{M0.0}$$
$$Q0.0 = M0.0$$
$$Q0.1 = M0.0$$
$$Q0.2 = M0.1$$
$$Q0.3 = M0.2$$
$$Q0.4 = M0.2$$

El objetivo de extraer los dos tipos de ecuaciones es porque existen programas que trabajan simultáneamente las direcciones de memoria (%I 0.0; %Q 0.2; etc), con las variables de cada dirección; y en algunos casos se requerirá saber qué variable es o qué dirección es la que estoy necesitando o manipulando. Sin duda dependerá de la habilidad del programador para optar por cualquiera de las dos ecuaciones.

f) Programar en lenguaje ladder a partir de las ecuaciones.

Por didáctica, utilizaremos las ecuaciones tomadas del grafcet de nivel 2 para programar en lenguaje ladder, así:

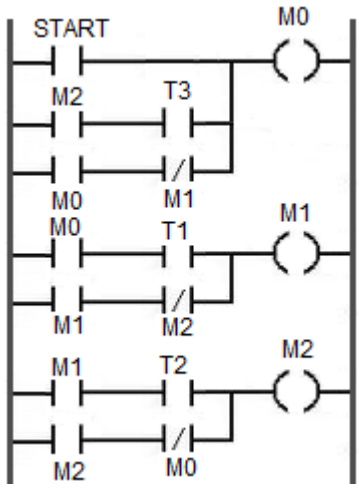


Figura V.63 Ladder Semáforo

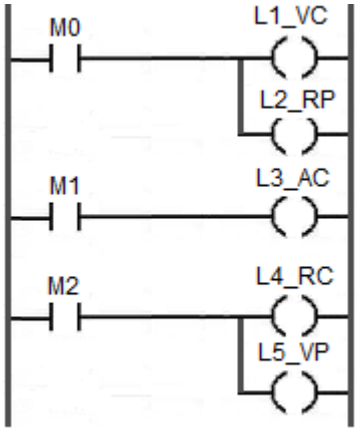


Figura V.63 Ladder Semáforo

g) Trasladar al software del PLC

En nuestro caso, utilizaremos el programa ***WAGO-IO-PRO 32 V2.2***, para la respectiva programación y posterior depuración del programa.

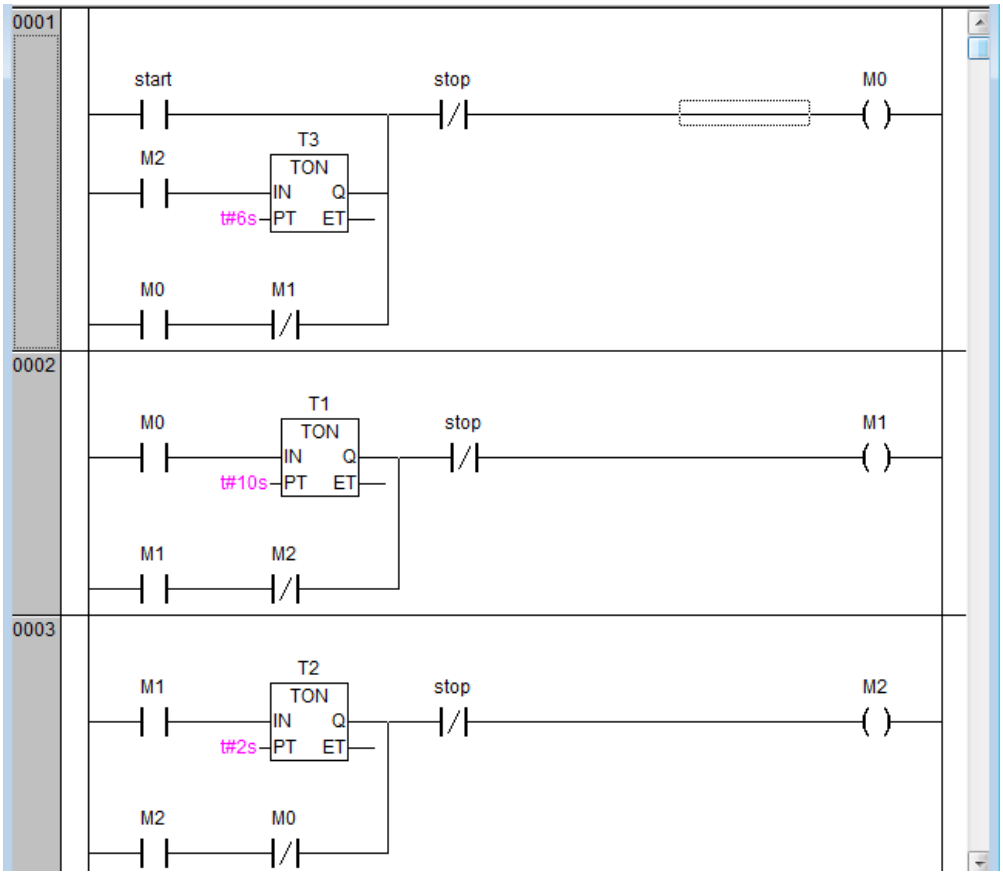


Figura V.64 Ladder Semáforo en programa WAGO

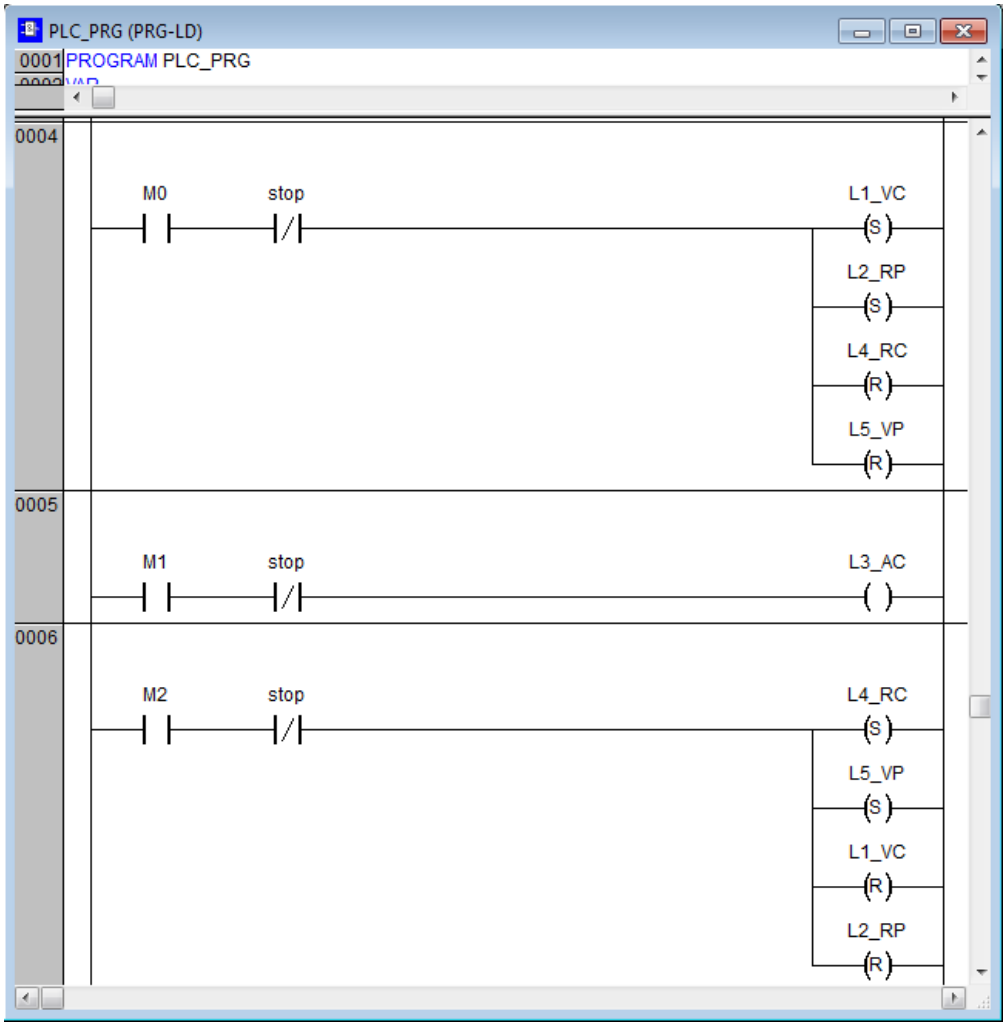


Figura V.64 Ladder Semáforo en programa WAGO

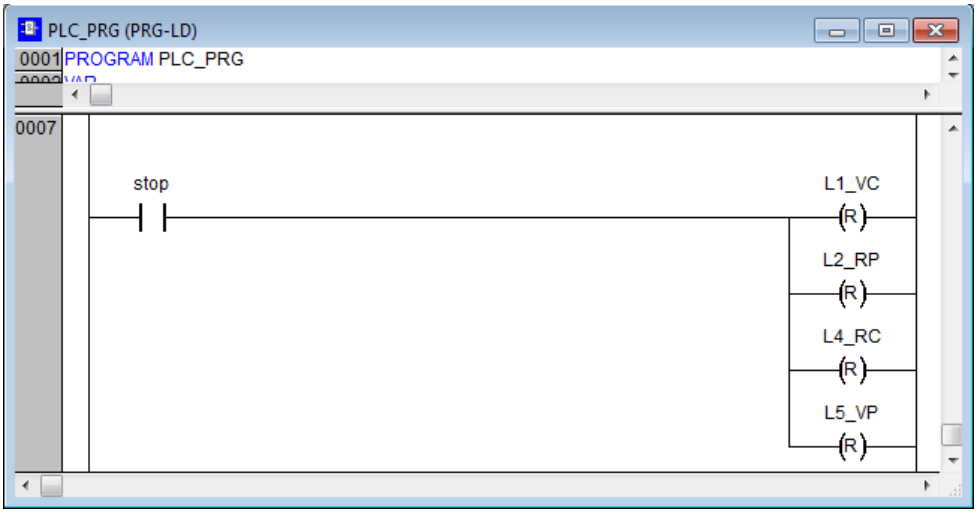


Figura V.64 Ladder Semáforo en programa WAGO

h) Depurar el programa.

En este caso, como es un sistema de señalización y que no implica movimiento de actuadores, no hace falta depurarlo, excepto por el control de paro STOP que está ubicado en cada red de la escalera y en forma cerrada.

Como se ha visto en este primer caso, se sigue con la metodología tradicional, normalmente en el momento de analizar el problema lleva un cierto tiempo aproximadamente 3 minutos para tener la idea, en el momento de la implantación en lenguaje grafcet de nivel 1 llevará unos 2 minutos, es importante seguir con el grafcet de nivel 2 y hasta cierto modo darlo como opcional la programación en grafcet de nivel 3 puesto que son las mismas ecuaciones que se extrae, llevando al programador a no tardar más de un minuto en disponer del grafcet de nivel 2. Después de establecer la tabla de asignaciones y extraer las ecuaciones, para luego trasladar a lenguaje ladder se llevará aproximadamente 5 minutos y por ende, dependiendo de la familiaridad con el programa que se va a implantar y con las respectivas correcciones se tomará unos 15 minutos, dando un total de 26 minutos aproximadamente en resolver el problema.

CASO DE ESTUDIO 2

Se desea realizar el control automático de una estación de Distribución, la misma que separa piezas. Un cilindro de doble efecto expulsa las piezas individualmente. El módulo Cambiador o brazo sujeta la pieza separada por

medio de una ventosa o succión. El brazo del cambiador, que es accionado por un actuador giratorio, transporta la pieza al punto de transferencia de la estación posterior.

SOLUCION

a) Establecer la secuencia del problema mediante el grafcet de nivel 1.

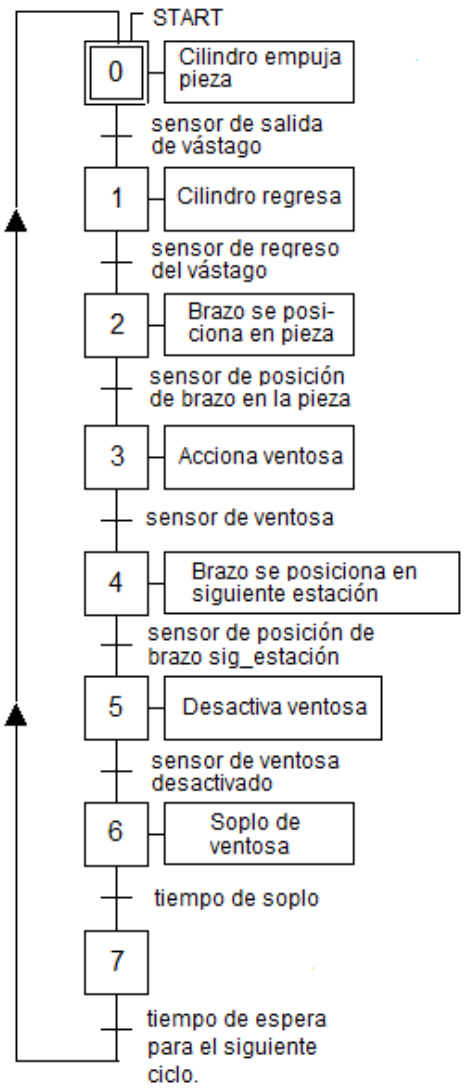


Figura V.65 Grafcet Nivel 1 Estación de distribución

b) Elaborar el grafcet de nivel 2 con todos los elementos técnicos y simbologías que se requiera.

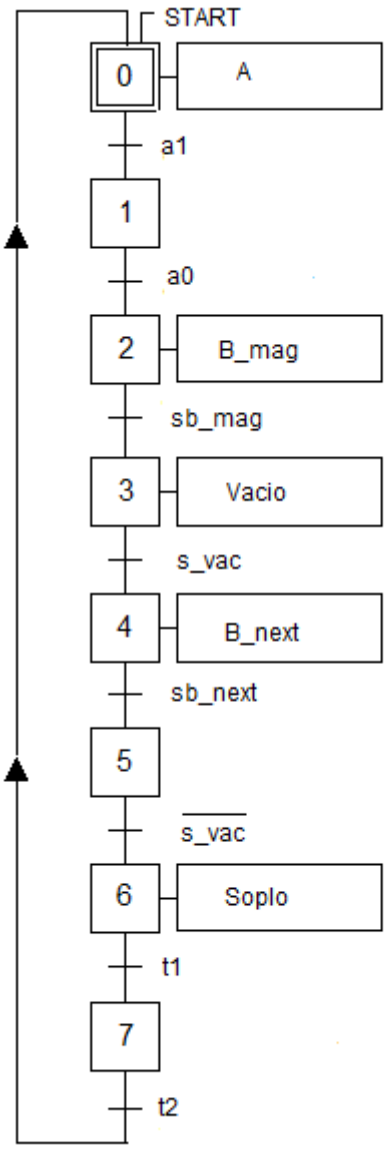


Figura V.66 Grafcet Nivel 2 Estación de distribución

c) Crear la tabla de asignaciones.

ENTRADAS	SALIDAS	MEMORIAS	TEMPORIZADORES
I0.0=START	Q0.0=L1_VC	M0.0=M0	t1=T1=2s
I0.1=a1	Q0.1=L2_RP	M0.1=M1	t2=T2=6s
I0.2=a0	Q0.2=L3_AC	M0.2=M2	
I0.3=s_vac	I0.3=L4_RC	M0.3=M3	
I0.4=sb_mag	I0.4=L5_VP	M0.4=M4	
I0.5=sb_next		M0.5=M5	
		M0.6=M6	
		M0.7=M7	

Tabla V.IX Tabla de asignaciones – Estación de distribución

En este caso, también utilizaremos temporizadores tanto para expulsar el objeto absorbido por el brazo actuador como también para esperar un cierto tiempo hasta que capture la siguiente figura o pieza circular. Esto se lo refleja en la tabla anterior con dos temporizadores.

Recordemos que no es indispensable programar en el grafcet de nivel 3 por otra razón. Normalmente en las competiciones de mecatrónica se requiere aprovechar al máximo el tiempo, así que el programador está consciente qué variable corresponde con cada entrada o salida y no hace falta programarlo, sino mas bien directamente grabar en el PLC y

hacer las pruebas. Por didáctica y para completar los pasos, vamos a seguir con el grafcet de nivel 3.

d) Elaborar el grafcet de nivel 3.

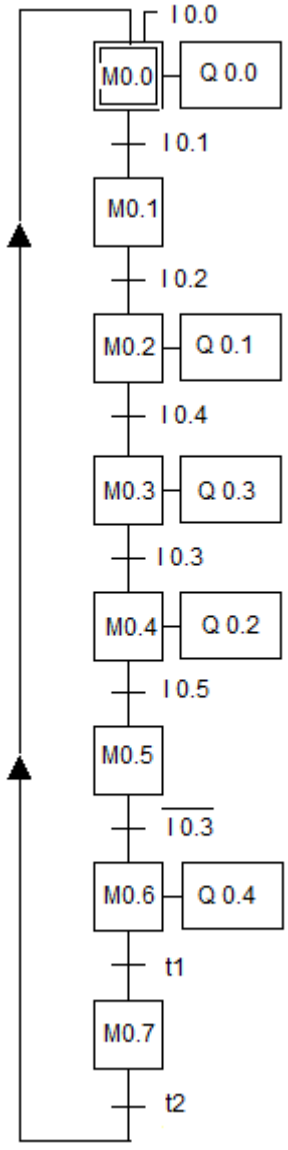


Figura V.67 Grafcet Nivel 3 Estación de distribución

e) Extraer las ecuaciones a partir del grafcet de nivel 2 o 3.

Para ello debemos usar la siguiente ecuación

$$M\_act = \underbrace{M\_ant * Cond}_{ENCENDIDO} + \underbrace{M\_act}_{MEMORIA} * \underbrace{\overline{M\_sig}}_{APAGADO}$$

Estas son las ecuaciones si extraemos del grafcet de nivel 2

$$M0 = START + M7\ t2 + M0\ \overline{M1}$$
$$M1 = M0\ a1 + M1\ \overline{M2}$$
$$M2 = M1\ a0 + M2\ \overline{M3}$$
$$M3 = M2\ sb\_mag + M3\ \overline{M4}$$
$$M4 = M3\ s\_vac + M4\ \overline{M5}$$
$$M5 = M4\ sb\_next + M5\ \overline{M6}$$
$$M6 = M5\ \overline{s\_vac} + M6\ \overline{M7}$$
$$M7 = M6\ t1 + M7\ \overline{M0}$$

$$A = M0$$
$$B\_mag = M2$$
$$Vacio = M3$$
$$B\_next = M4$$
$$Soplo = M6$$

Estas son las ecuaciones si extraemos del grafcet de nivel 3

$$M0.0 = I0.0 + M0.7\ t2 + M0.0\ \overline{M0.1}$$
$$M0.1 = M0.0\ I0.1 + M0.1\ \overline{M0.2}$$
$$M0.2 = M0.1\ I0.2 + M0.2\ \overline{M0.3}$$
$$M0.3 = M0.2\ I0.4 + M0.3\ \overline{M0.4}$$
$$M0.4 = M0.3\ I0.3 + M0.4\ \overline{M0.5}$$
$$M0.5 = M0.4\ I0.5 + M0.5\ \overline{M0.6}$$
$$M0.6 = M0.5\ \overline{I0.3} + M0.6\ \overline{M0.7}$$
$$M0.7 = M0.6\ t1 + M0.7\ \overline{M0.0}$$

$$Q0.0 = M0.0$$
$$Q0.1 = M0.2$$
$$Q0.3 = M0.3$$
$$Q0.2 = M0.4$$
$$Q0.4 = M0.6$$

f) Programar en lenguaje ladder a partir de las ecuaciones.

Por didáctica, utilizaremos las ecuaciones tomadas del grafcet de nivel 2 para programar en lenguaje ladder, así:

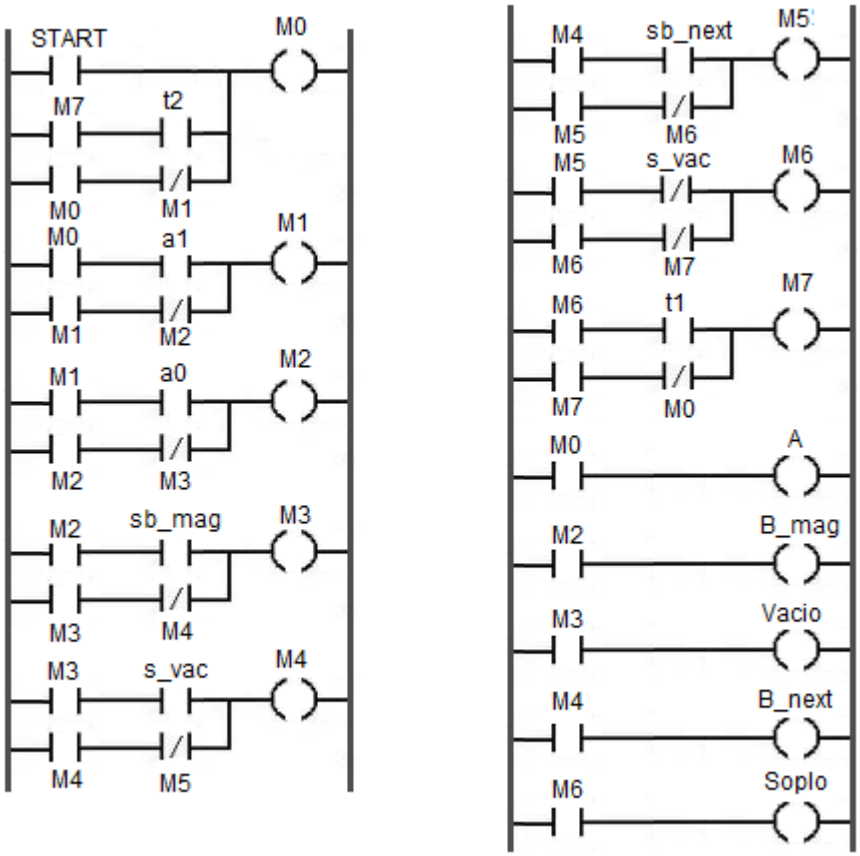


Figura V.68 Ladder Estación de distribución

g) Trasladar al software del PLC

En este caso, el problema es muy grande por lo que lo presentaremos de forma seccionada. Esto es solo una muestra de que, para presentar como programa el leguaje ladder, el proceso es muy largo, aproximadamente unas 13 redes y sin tomar en cuenta las respectivas depuraciones. Esto lo presentamos en cuatro imágenes.

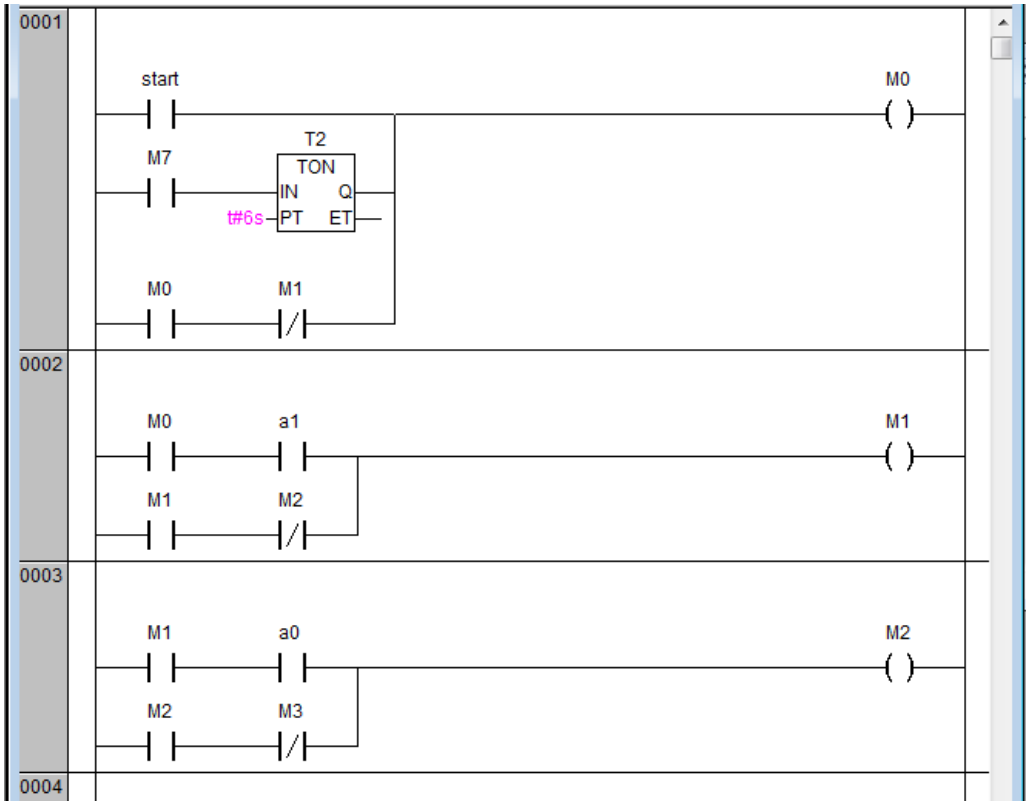


Figura V.69 Ladder Estación de distribución en programa WAGO

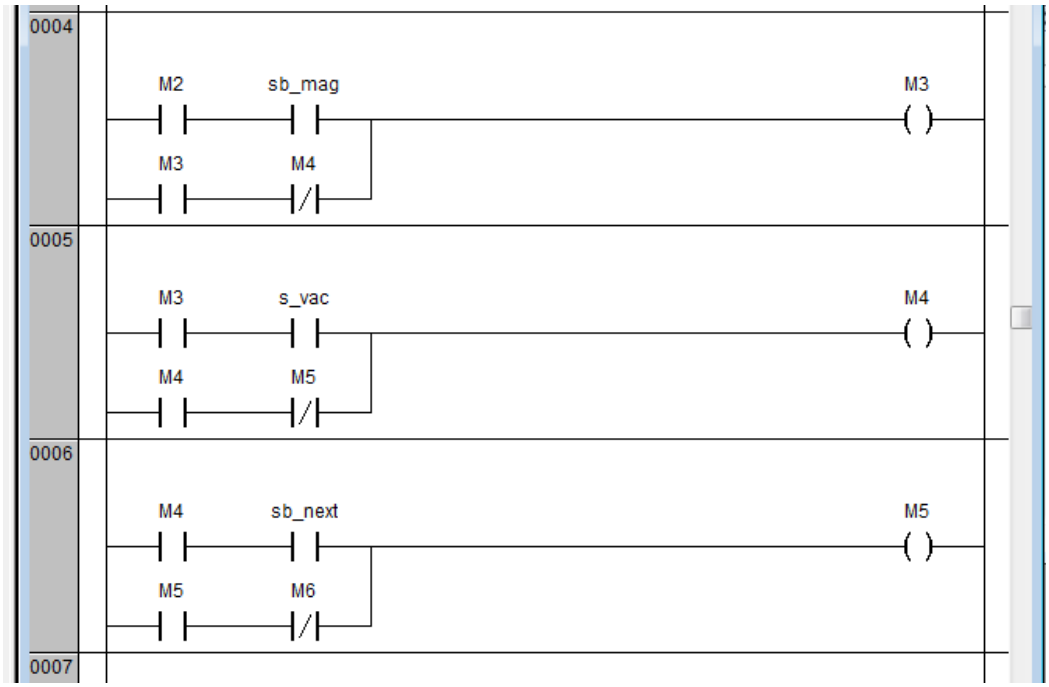


Figura V.69 Ladder Estación de distribución en programa WAGO

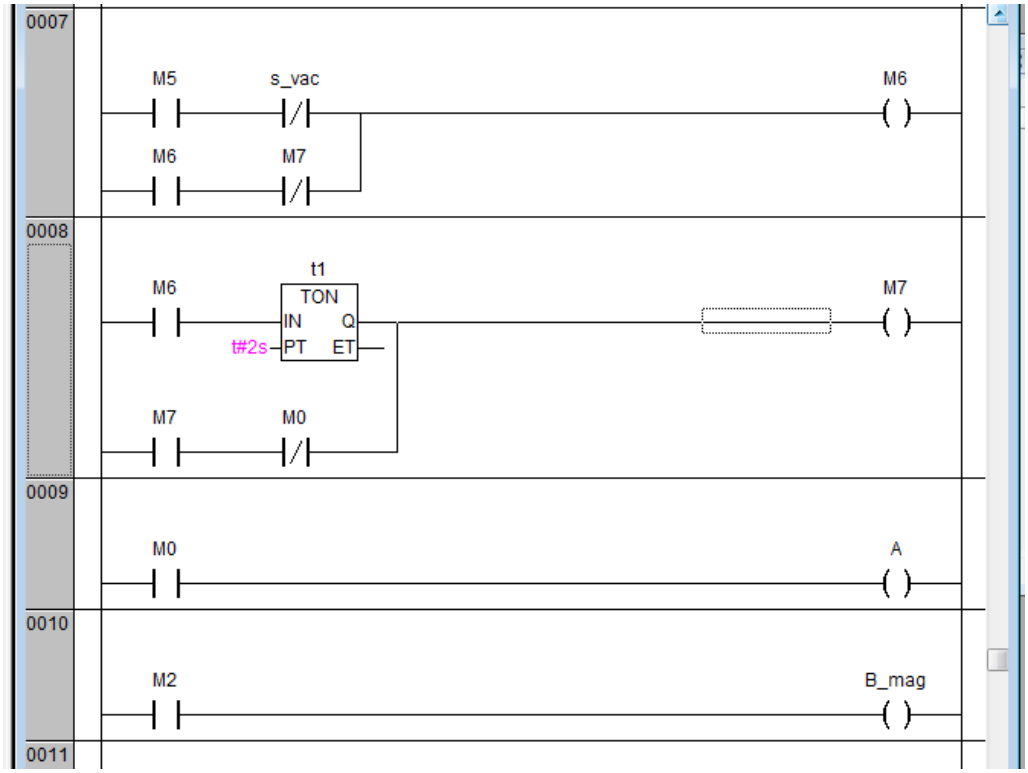


Figura V.69 Ladder Estación de distribución en programa WAGO

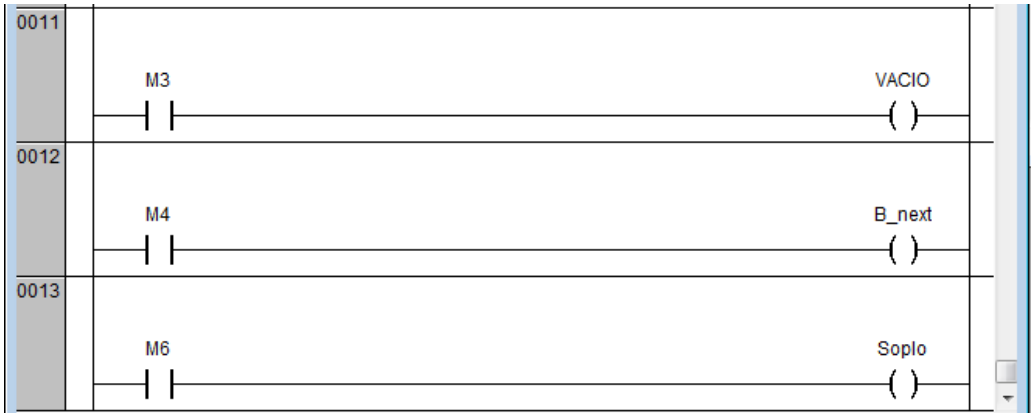


Figura V.69 Ladder Estación de distribución en programa WAGO

h) Depurar el programa.

En este caso, sí se requiere hacer determinados controles. A más del habitual paro de emergencia, también habrá el caso si el brazo de la estación no está en la posición inicial para programar, por tanto se pondría una sencilla condición: a más del START en la primera rama, se

colocaría junto al mismo el sensor abierto sb_next para asegurarnos que está en dirección a la otra estación y de ahí proceder con la secuencia, así:

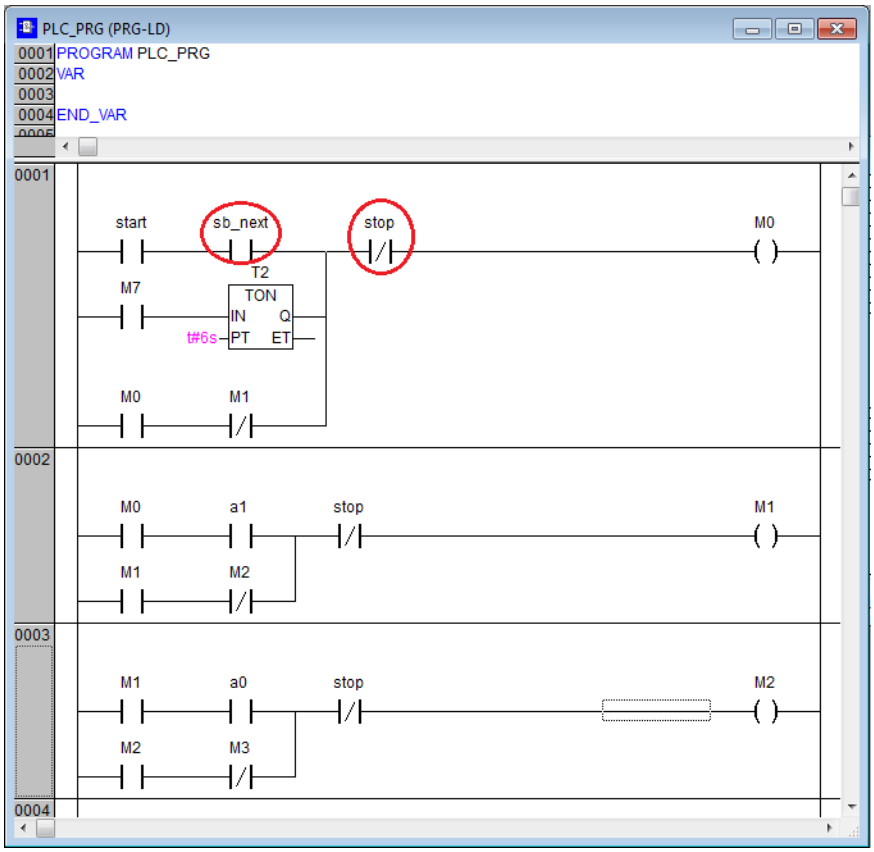


Figura V.70 Ladder Estación de distribución con controles adicionales

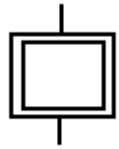
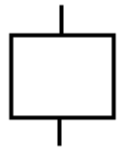
Recordemos que a diferencia del caso de estudio anterior, este fue relativamente más largo y cada vez que el problema se haga más complejo, el tiempo de programación también crecerá.

Relativamente el tiempo que llevó a entender el problema, realizar el respectivo grafcet, la tabla de asignaciones y las respectivas ecuaciones con su correspondiente lenguaje en ladder sin mencionar las respectivas pruebas y depuraciones, se llevo alrededor de 40 minutos.

5.4. Implementación de la propuesta metodológica para la programación de PLC en Grafcet

En este apartado, se determinará que, es posible implementar la metodología de programación propuesta, recalcando que se usará sólo los pasos necesarios para optimizar el tiempo de programación. Para ello requeriremos que el programador tenga los conocimientos necesarios de la metodología de programación en SFC o GRAFCET detallados en el capítulo IV de este documento.

De igual manera, recordemos los siguientes elementos en la siguiente tabla.

Elementos GRAFCET de programación		
Símbolo	**Nombre**	**Descripción**
	Etapa inicial	Indica el comienzo del esquema SFC y se activa al poner en RUN el autómata. Por lo general suele haber una sola etapa de este tipo.
	Etapa	Su activación lleva consigo una acción o una espera.
	Unión	Las uniones se utilizan para unir entre sí varias etapas.
	Transición	Condición para desactivarse la etapa en curso y activarse la siguiente etapa, Se indica con un trazo perpendicular a una unión.

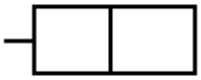
	Direcciona-miento	Indica la activación de una u otra etapa en función de la condición que se cumpla.
	Proceso simultáneo	Muestra la activación o desactivación de varias etapas a la vez.
	Acciones asociadas	Acciones que se realizan al activarse la etapa a la que pertenecen.

Tabla V.X Elementos grafcet de programación

Será importante recordar que se puede programar simultáneamente procesos, todo dependerá de la complejidad de programación que tengamos. Dichos métodos están detallados en el Capítulo IV.

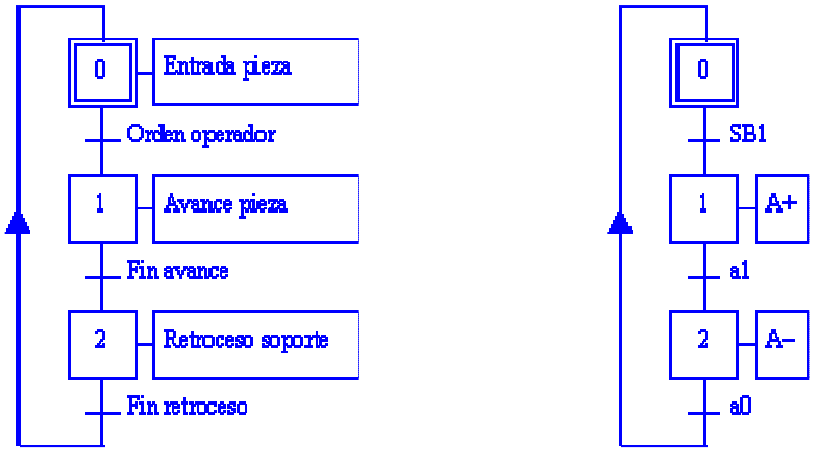
5.4.1. Método de Programación Propuesto

El método de programación propuesto de PLC en GRAFCET dispone de los siguientes pasos:

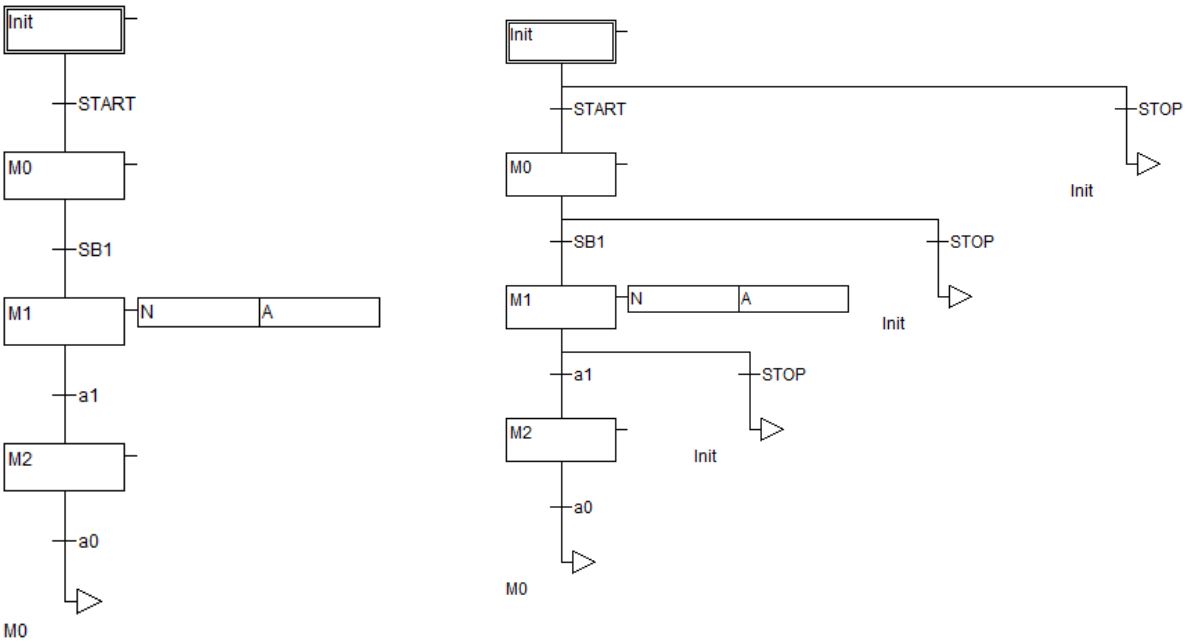
a) Establecer la secuencia del problema que se quiere resolver mediante el grafcet de nivel 1.
b) Elaborar el grafcet de nivel 2 con todos los elementos técnicos y simbologías que se requiera.
c) Programar en lenguaje GRAFCET a partir del grafcet de nivel 2
d) Depurar el programa.

Gráficamente sería lo siguiente:

a) Establezco secuencias de solución b) Elaboro el Grafcet de Nivel 2



c) Programo en Lenguaje Grafcet d) Depuro el programa



Para demostración de los pasos de la metodología, emplearemos los mismos casos de estudio empleados con la metodología tradicional.

5.4.2. Casos de Estudio

CASO DE ESTUDIO 1

Se desea realizar el control automático de un semáforo para peatones. El peatón al llegar a la bocacalle observa el semáforo de peatón en rojo, pero a su vez el semáforo de la calle está en verde, el mismo que dura 30 s, después se observa que el semáforo de la calle está en amarillo durante 5 s y posteriormente cambia a rojo, cambiando el semáforo del peatón a verde, después de 20 s el semáforo de la calle cambia a verde y el del peatón cambia a rojo, comenzando un nuevo ciclo.

SOLUCION

a) Establecer la secuencia del problema que se quiere resolver mediante el grafcet de nivel 1.

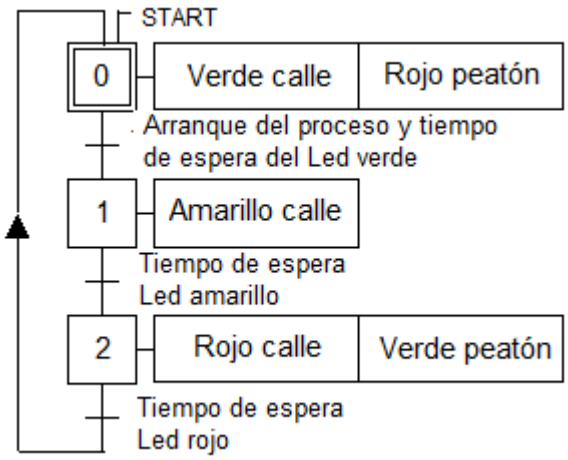


Figura V.71 Grafcet Nivel 1 Semáforo

b) Elaborar el grafcet de nivel 2 con todos los elementos técnicos y simbologías que se requiera.

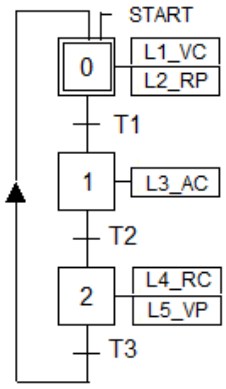


Figura V.72 Grafcet Nivel 2 Semáforo

c) Programar en lenguaje GRAFCET a partir del grafcet de nivel 2

En nuestro caso, utilizaremos el programa ***WAGO-IO-PRO 32 V2.2***, para la respectiva programación y posterior depuración del programa.

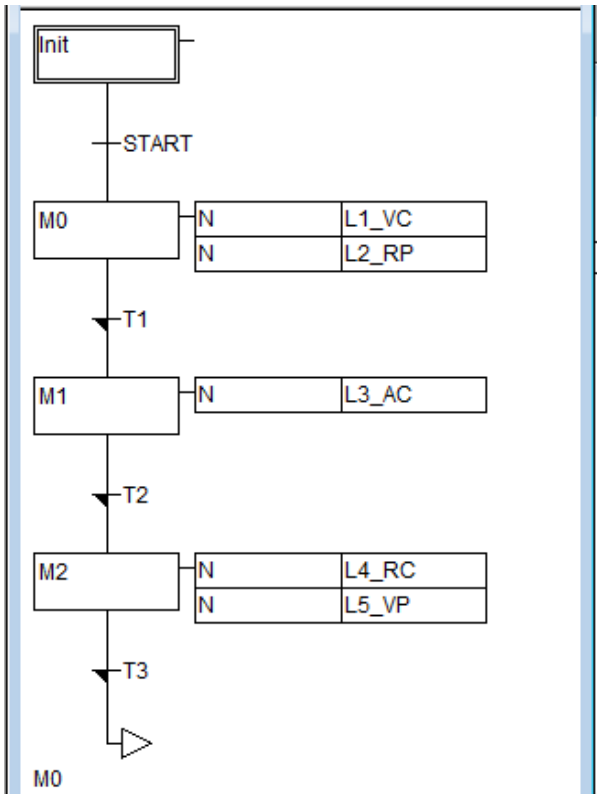


Figura V.73 Grafcet Nivel 2 de Semáforo en programa WAGO

En las transiciones están programadas el tiempo que requerimos para el cambio de luces en el semáforo, tal como muestra el siguiente ejemplo en T1:

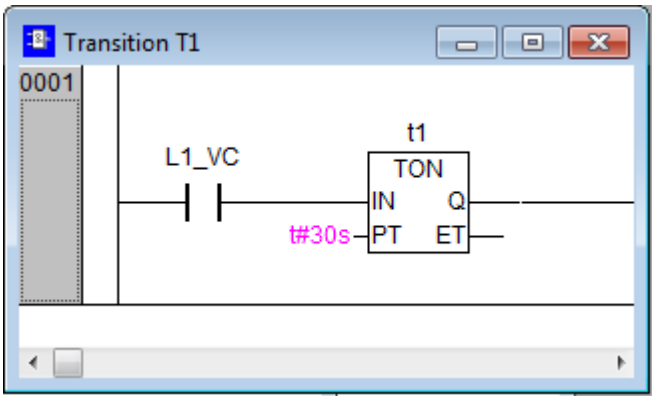


Figura V.74 Lenguaje ladder en transición de tiempo

d) Depurar el programa.

En nuestro caso, se realizó una restricción o señal de paro, para que cuando se active, ningún LED o lámpara esté encendido. Esto se logra mediante bifurcaciones de tal modo que si hay una señal de paro o stop, se dirijan al estado INIT.

Notemos la diferencia tanto en el programa ladder del caso anterior como el programa actual, prácticamente el programa es sencillo, pequeño y de fácil corrección.

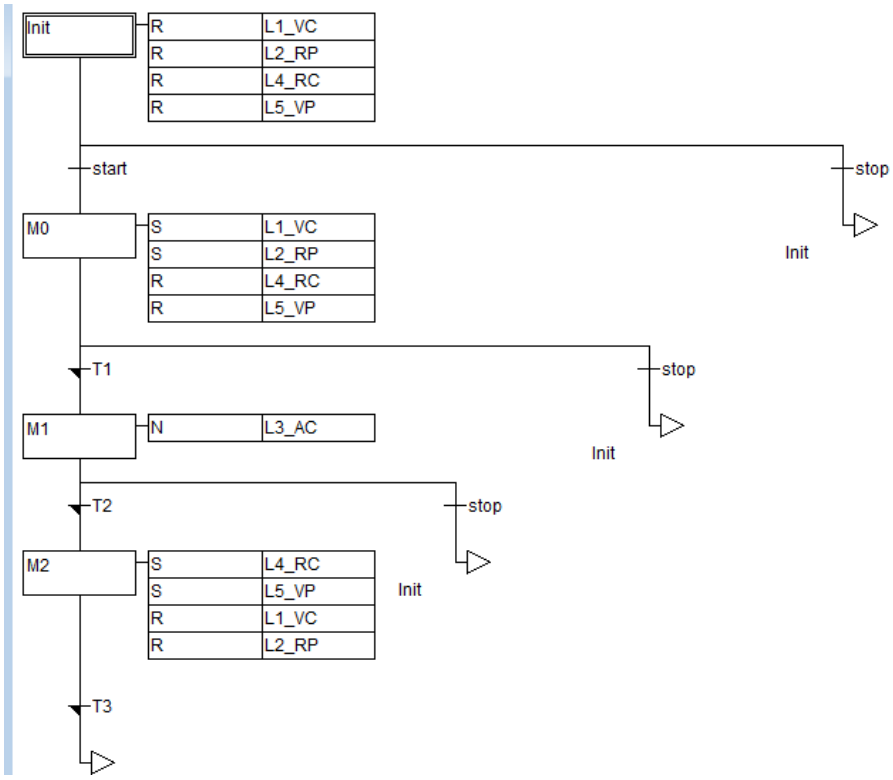


Figura V.75 Grafcet Nivel 2 Semáforo con controles

Es demostrable que a diferencia del método tradicional, que lleva alrededor de 26 minutos y con las debidas pruebas, con esta metodología se logró realizar en 8 minutos aproximadamente.

CASO DE ESTUDIO 2

Se desea realizar el control automático de una estación de Distribución, la misma que separa piezas. Un cilindro de doble efecto expulsa las piezas individualmente. El módulo Cambiador o brazo sujeta la pieza separada por medio de una ventosa o succión. El brazo del cambiador, que es accionado por un actuador giratorio, transporta la pieza al punto de transferencia de la estación posterior

SOLUCION

a) Establecer la secuencia del problema que se quiere resolver mediante el grafcet de nivel 1.

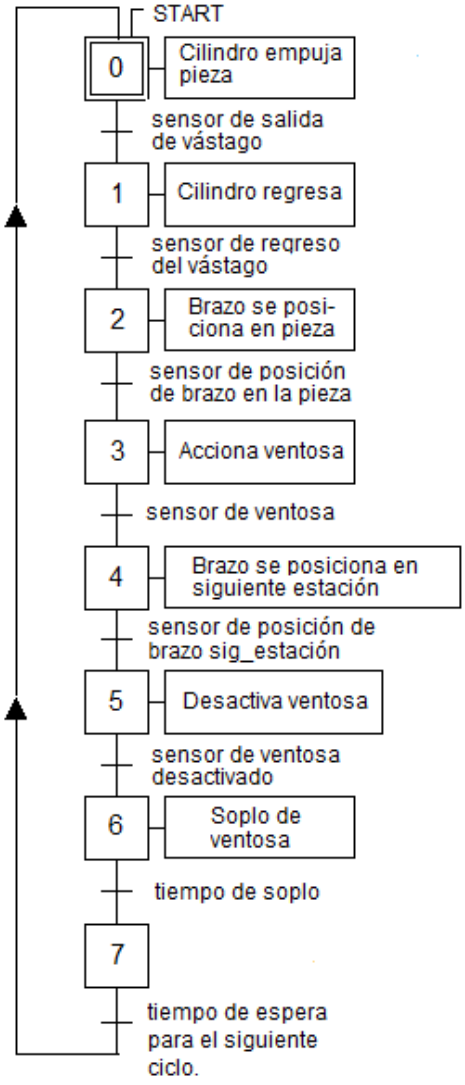


Figura V.76 Grafcet Nivel 1 Estación de distribución

b) Elaborar el grafcet de nivel 2 con todos los elementos técnicos y simbologías que se requiera.

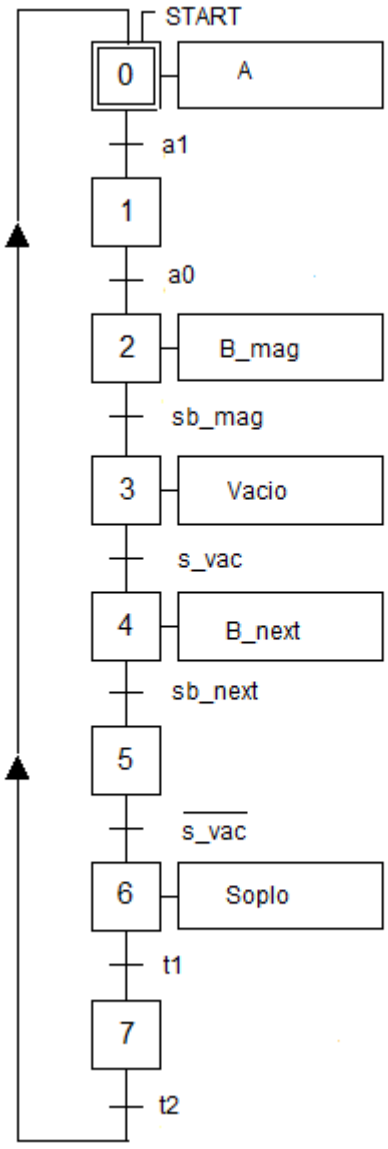


Figura V.77 Grafcet Nivel 2 Estación de distribución

c) Programar en lenguaje GRAFCET a partir del grafcet de nivel 2

En este caso nuevamente utilizaremos el programa ***WAGO-IO-PRO 32 V2.2***, para la respectiva programación y posterior depuración del programa. Para ello el programa lo seccionaremos en partes.

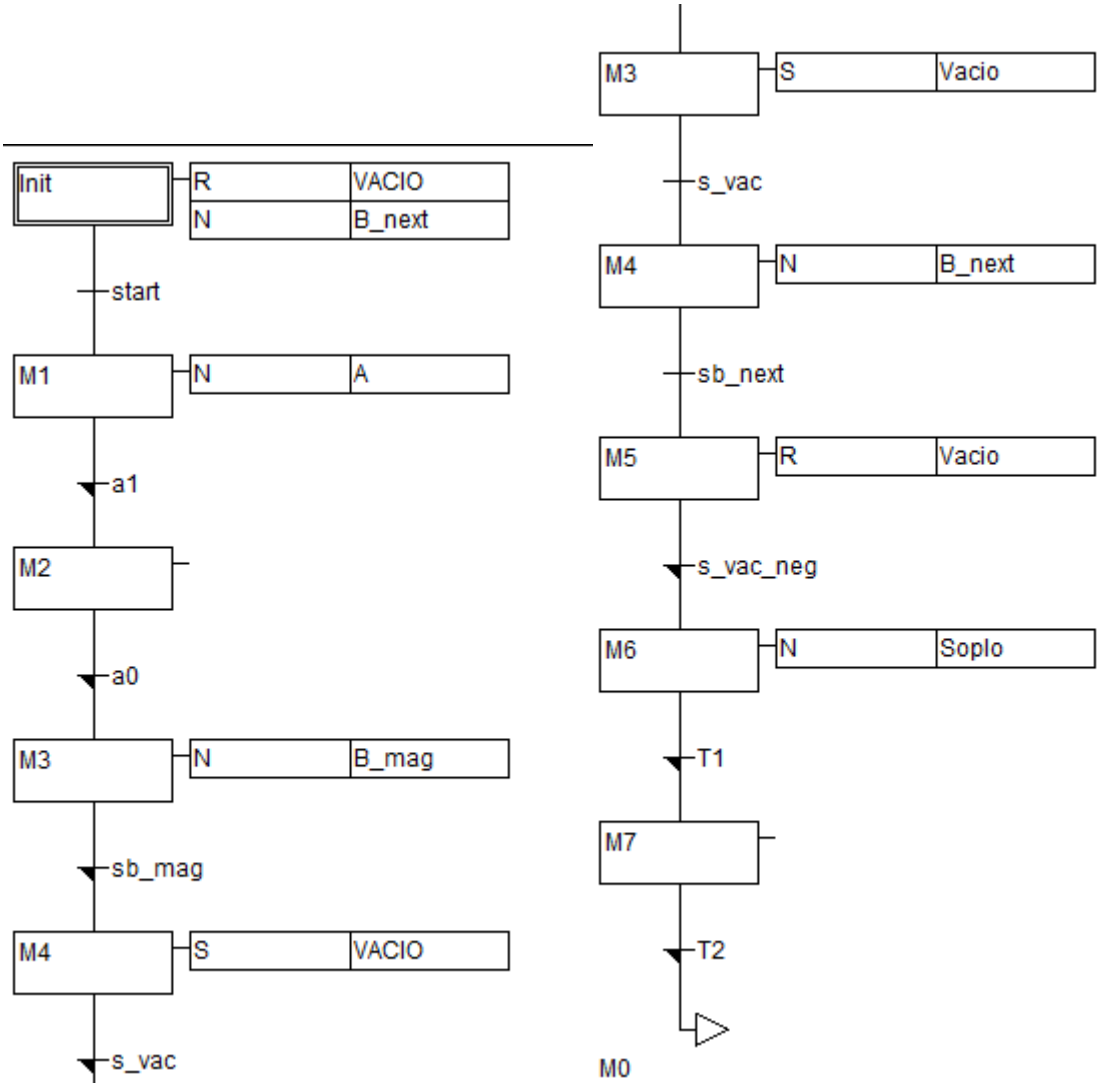


Figura V.78 Grafcet Nivel 2 Estación de distribución en programa WAGO

A pesar de que en este problema hay más etapas, sigue siendo muy rápido programar, demorando 13 minutos.

d) Depurar el programa.

En nuestro caso, se realizó una restricción o señal de paro, para que cuando se active, el sensor de vacío deje de funcionar y que el brazo esté en la posición de enlace a otra estación, es decir tal como se

plantea en el problema. Para estar seguros, en el momento que haya señal de paro, situarse en la posición inicial, y si no lo está, colocar junto al sensor de START el sensor sb_next.

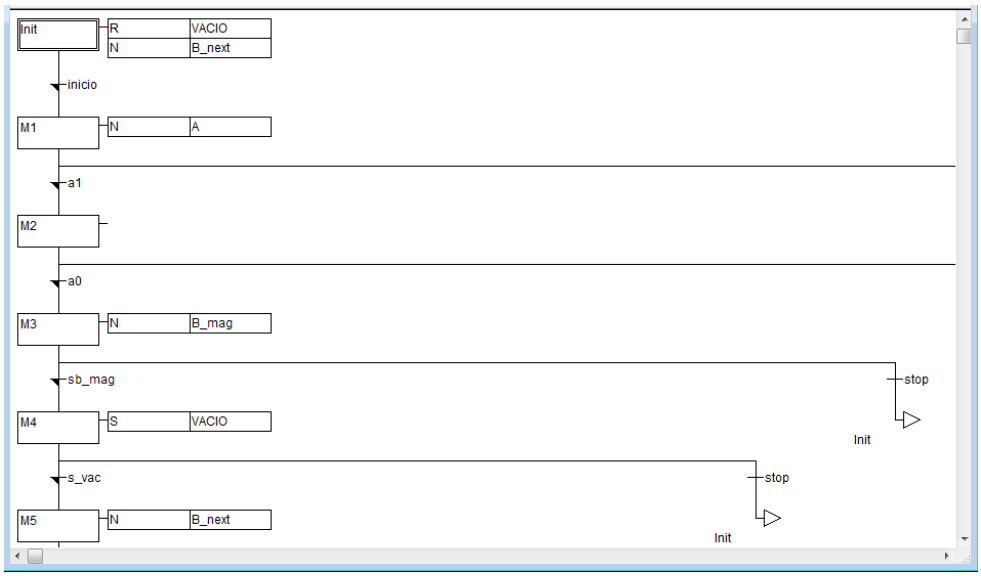


Figura V.79 Grafcet Nivel 2 Estación de distribución con controles

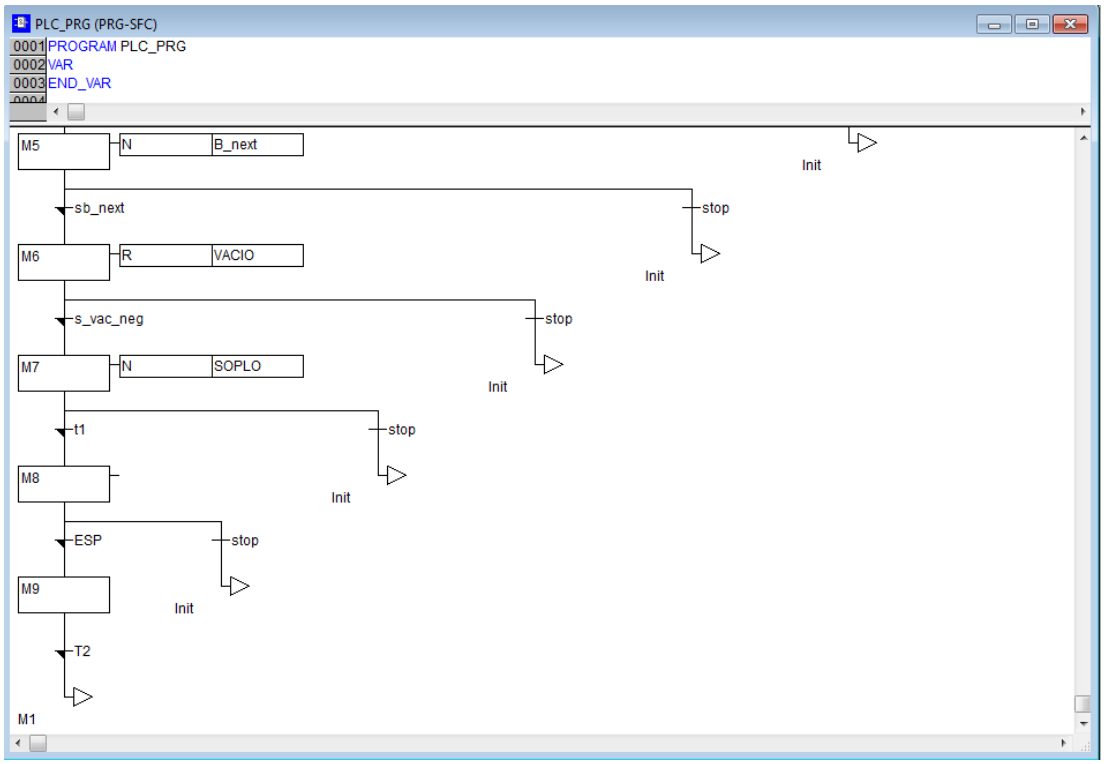


Figura V.79 Grafcet Nivel 2 Estación de distribución con controles

Por tanto, desde el análisis hasta la programación en sí, se demoraría alrededor de 15 minutos con todas las correcciones y las respectivas pruebas. Tanto en tamaño de programación como en tiempo es mínimo y si el problema es más complejo, no se extenderá tanto el programa, demostrando así las bondades de utilizar este método, para realizar en corto tiempo un problema de mecatrónica.

5.5. ANALISIS DE RESULTADOS

A continuación se realiza el estudio y análisis de resultados obtenidos a lo largo de la investigación tanto de la metodología tradicional para la resolución de problemas secuenciales como también la metodología propuesta aplicada a cualquier proceso ya sea automatizado como de Mecatrónica, para lograr la comprobación de la hipótesis.

Se procede a identificar las diferentes categorías que corresponden con las variables independientes y dependientes. Para la variable independiente, se realiza una comparación entre sus categorías, siendo la base de la hipótesis; para que en las siguientes categorías –conocimiento, medición de tiempo y confiabilidad de resultados– realizar las graficas comparativas y estadísticas de cada categoría en función de las categorías de la variable independiente. Los datos son obtenidos de una muestra de 4 alumnos de la ESPOCH familiarizados con la programación secuencial en Automatización Industrial y Mecatrónica.

5.5.1. Descripción de Hipótesis

Hipótesis investigativa: "El diseño de una propuesta metodológica de programación de PLC's en Grafcet para las competencias de Mecatrónica World Skills permitirá disminuir el tiempo de programación de procesos secuenciales".

5.5.1.1. Operacionalización Conceptual de Variables

VARIABLE	TIPO DE VARIABLE
El diseño de una propuesta metodológica de programación de PLC's en Grafcet	V. Independiente
disminuir el tiempo de programación de procesos secuenciales	V. Dependiente

Tabla V.XI Operacionalización Conceptual de Variables

5.5.1.2. Operacionalización Metodológica de Variables

VARIABLES	CATEGORIAS	INDICADORES	INDICES
El diseño de una propuesta metodológica de programación de PLC's en Grafcet	Metodología Tradicional	• Análisis • Resolución manual • Resolución en lenguaje de programación	Conocimiento Grafcet como Método Lenguaje ladder
	Metodología Grafcet	• Análisis • Resolución manual • Resolución en lenguaje de programación	Conocimiento Grafcet como Método Diagramas SFC
disminuir el tiempo de programación de procesos secuenciales	Velocidad	Tiempo de programación	Cronómetro
	Conocimiento	Depuraciones	Depurador paso a paso
	Confiabilidad	Pruebas de los equipos	Implementación física de sensores y actuadores Implementación lógica – Simulador Programar Cargar al PLC Depurar

Tabla V.XII Operacionalización Metodológica de Variables

5.5.2. Determinación de Muestras y Escalas

La muestra tomada en consideración para analizar esta variable es un grupo de cuatro estudiantes de la Escuela de Ingeniería en Sistemas, a los cuales se les evalúa en función de la Metodología Tradicional vs la Metodología Propuesta. Cada estudiante está siendo evaluado en función del tiempo las diferentesº

Tabla V.XIII Escala de valores

5.5.3. Comprobación de Hipótesis

Para realizar la comprobación de la hipótesis, se partirá de la demostración de las variables tanto independientes como dependientes, realizando cuadros comparativos entre la Metodología Tradicional y la Metodología Propuesta. Recordemos que todos los alumnos son evaluados al presentar los diferentes casos de uso para su resolución.

A continuación un análisis comparativo cuantitativo entre los componentes de la variable independiente, que teoriza la hipótesis planteada en función de la opinión del estudiante, realizando preguntas de punto de vista.

CATEGORIA	CUESTIONARIO	Resp. Estud. (Escala)				PROM
		1º	2º	3º	4º	
Metodología Tradicional	¿Se requiere considerable tiempo para programar en etapas y posteriormente trasladar a lenguaje ladder?	3	4	2	4	3,25
Metodología Grafcet	¿Sería corto el tiempo si se lograría usar los diagramas SFC como programa?	1	2	2	1	1,50

Tabla V.XIV Cuadro comparativo de la variable independiente

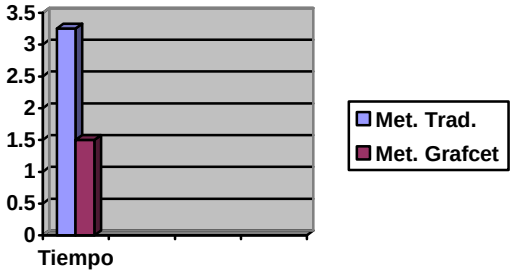


Figura V.80 Comparación gráfica de tiempos entre las metodologías

Análisis:

Se puede comprobar que, en función de la encuesta realizada a los estudiantes, coinciden que se requiere un considerable tiempo para poder resolver problemas, y que en comparación con la posibilidad de programar en SFC como programa, también coinciden que el tiempo sería menor, reafirmando que la hipótesis es comprobable. Si el tiempo que actualmente se requiere representa un 81,25%; al encuestar sobre la nueva metodología se estima que el tiempo requerido abarcaría un 37,5%, se lograría reducir un 43,75% del tiempo que se requiere para programar. Analicemos a continuación las variables: ***conocimiento, rapidez y confiabilidad***.

CONOCIMIENTO

En la Tabla siguiente, se muestra el análisis comparativo del conocimiento que demuestran los alumnos a la hora de ejecutar el problema.

Metodología	Conocimientos previos	Conocimientos recibidos	Escala (Prom)
Tradicional	4	2	3
Grafcet	4	4	4

Tabla V.XV Cuadro comparativo del conocimiento

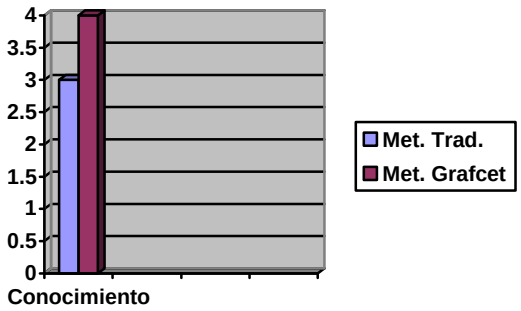


Figura Nº V.81 Gráfico estadístico del conocimiento

Análisis:

Se puede apreciar que en conocimientos previos, tienen buena base para programar en lenguaje tradicional, y por tanto no requerirán tanto conocimiento a posterior, dando un rendimiento en conocimiento en escala de 3, pero en el método grafcet, a más del conocimiento previo se les instruye la metodología propuesta en un 100%, dando una escalabilidad de 4.

VELOCIDAD

En la siguiente tabla se verá que tan rápido resuelven los alumnos aplicando la metodología tradicional y la metodología propuesta.

Metodología	Tiempo (min)	Escala
Tradicional	28	2
Grafcet	12	4

Tabla V.XVI Cuadro Comparativo de la velocidad

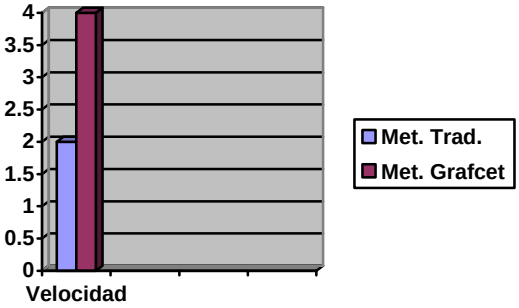


Figura Nº V.82 Gráfico estadístico de la rapidez

Análisis:

De acuerdo a los datos, en el instante que se ponen a programar, si emplean la metodología habitual se demoran en función del tiempo, alrededor de 37 minutos como promedio, de aquello se califica con un valor de escala 2. Al aplicar la metodología grafcet, usaron como promedio 12 minutos, dando un valor de escala 4.

Confiabilidad

Al programar con el diagrama ladder en la metodología habitual, existe la probabilidad del 90% que el programa funcionará apenas sea instalado en su soporte Hardware, en cambio al emplear la metodología grafcet, la confiabilidad es del 100%. En escala de valores, como no hay diferencia en conocimientos académicos, se les asignará el valor 4.

Metodología	Funcionalidad	Escala
Tradicional	90%	4
Grafcet	100%	4

Tabla V.XVII Cuadro comparativo de la confiabilidad

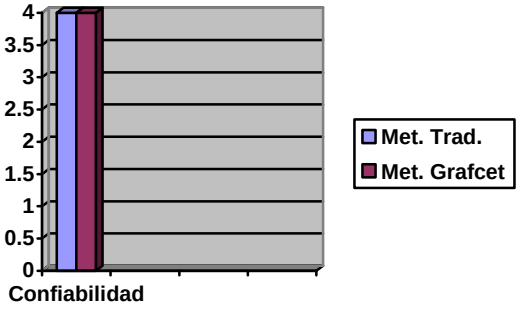


Figura Nº V.83 Gráfico estadístico de la Confiabilidad

Análisis:

En el caso actual, ambas metodologías bien aplicadas tienen un indicio de confiabilidad a la hora de cargar al PLC. Por tanto se deduce que se puede aplicar cualquiera de las dos dependiendo de la habilidad del estudiante.

Comparación Global de las categorías

Con los datos obtenidos se realizó una tabla comparativa entre las variables de los dos tipos de métodos, tomando en consideración las escalas establecidas en la Tabla.

PARAMETROS DE COMPARACION	METODOLOGÍA HABITUAL	METODOLOGÍA PROPUESTA
Conocimiento	3	4
Rapidez	2	4
Confiabilidad	4	4

Tabla V.XVIII Tabla comparativa de las categorías

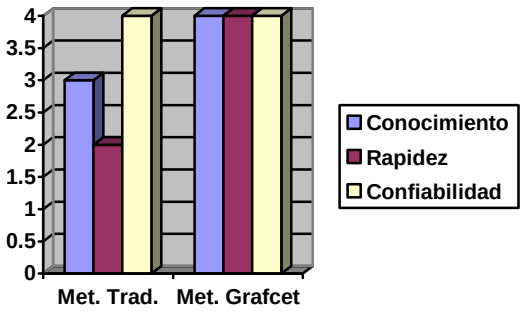


Figura Nº V.84 Gráfico estadístico de las categorías

Con los datos obtenidos, se desprende que la metodología propuesta permite incrementar su conocimiento y habilidades, se realiza rápidamente el proceso y tiene un alto grado de confiabilidad al aplicar la metodología grafcet, siendo mucho más eficiente en un 100% que la metodología tradicional. Se ha comprobado además que, en función de la complejidad del problema, si es menor, entonces el tiempo no es tan notorio, pero si el ejercicio es más complejo, mejor se notará el tiempo que se ahorra programando directamente en diagrama SFC como lenguaje.

CONCLUSIONES

El realizar un estudio de los mecanismos para optimizar procesos industriales mediante la programación en Grafcet, fue un asunto de peso al momento de elegir el camino de solución al problema planteado.

Nos hemos dado cuenta de la gran necesidad de programar en competencias de Mecatrónica lo más rápido posible, puesto que ahí se demuestra las habilidades del estudiante.

Al analizar los tiempos de programación estadísticamente, vemos que el tiempo para el análisis y diseño del problema son similares, dependiendo de la agilidad mental que el competidor tenga, pero al momento de programar, el ahorro de la metodología tradicional y la metodología propuesta es entre 40% y 65%, dependiendo de la complejidad del problema.

El estudio y la propuesta de las aplicaciones Grafcet, ayudó en gran manera el ahorro de tiempo y recursos inclusive es reutilizable en otras áreas de estudio.

El aplicar dicha metodología a la estación de distribución nos amplia el horizonte para mejorar nuestros procesos en cuanto a programación se refiere.

RECOMENDACIONES

Para el estudio de esta metodología se recomienda conocer el mecanismo de programación en grafcet como método para luego aplicarlo como programa.

Para la aplicación en cualquier estación mecatrónica o de automatización industrial, se recomienda profundizar los conocimientos de programación mediante el WAGO I/O PRO

Se recomienda tomar en consideración las características técnicas que cada tipo de red industrial presenta, que tienen diferentes maneras de diseño, y por tanto implicaría una demora lo cual no se quisiera perder valioso tiempo.

Se sugiere efectuar un estudio previo en cuanto a la disponibilidad de otros programas similares que puedan aplicarse esta metodología.

Se aconseja a los estudiantes de la Facultad, que apoyen su investigación en proyectos similares pues esa información puede sentar la base para el desarrollo de su trabajo.

RESUMEN

La metodología propuesta en esta investigación permitió programar estaciones de automatización industrial en corto tiempo, aplicado a competencias de programación, y se lo realizó con el propósito de disminuir el tiempo en programación. Se aplicó en la Estación de Distribución marca FESTO, del Laboratorio de Automatización Industrial de la Escuela de Ingeniería en Sistemas de la ESPOCH.

Se empleó Hardware PLC Wago 7582 y la Estación de distribución Festo; como paquete informático Wago I/O Pro 32. Además se empleó el manual técnico de PLC Wago, aplicándose método inductivo, experimental y técnicas de programación Ladder vs Grafcet.

Con el método Ladder se analizó, diseñó, implementó y programó con el estándar IEC 61131-3, el tiempo empleado fue de 45 minutos. En cambio con la metodología propuesta, siendo una técnica gráfica, se usó el SFC –**S**equential **F**unction **C**hart– analítico y programado, empleando 21 minutos, obteniendo un ahorro del 53.33%, con un margen de error de 0.05.

En conclusión, se alcanzó el objetivo en un 95% en cuanto a velocidad de programación usando la metodología propuesta. Se recomienda el uso práctico de la programación Grafcet lo que permitirá adquirir destrezas en su manejo.

SUMMARY

The methodology proposed in this investigation allowed to program stations of industrial automation in short time, applied to programming competitions, and she was carried with the purpose of diminishing the time in programming. It was applied in the Station of Distribution FESTO it marks, of the Laboratory of Industrial Automation of the School of Engineering in Systems of the ESPOCH.

Hardware PLC Wago 7582 was used and the distribution Station Festo; as software package Wago I/O Pro 32. Also the technical manual of PLC Wago was used, being applied inductive method, experimental and technical of programming Ladder vs Grafcet.

With the method Ladder was analyzed, it designed, it implemented and it programmed with the standard IEC 61131-3, the used time was of 45 minutes. On the other hand with the proposed methodology, being a graphic technique, the SFC -**S**equential **F**unction **C**hart- was used analytic and programmed, using 21 minutes, obtaining a saving of 53.33%, with a margin of error of 0.05.

In conclusion, the objective was reached in 95% as for programming fast using the proposed methodology. The practical use of the programming Grafcet that is recommended that will allow to acquire skills in management.

GLOSARIO DE TERMINOS

AUTÓMATA

Controlador programable Twido. Existen dos tipos de controladores: compacto y modular.

AUTOMATIZACIÓN

Automatización es la tecnología que trata de la aplicación de sistemas mecánicos, electrónicos y de bases computacionales para operar y controlar la producción.

GRAFCET

Grafcet permite representar gráficamente y de forma estructurada el funcionamiento de una operación secuencial. Método analítico que divide cualquier sistema de control secuencial en una serie de pasos a los que se asocian acciones, transiciones y condiciones.

LENGUAJE LADDER

Programa escrito en lenguaje Ladder compuesto por una representación gráfica de instrucciones de un programa de controlador con símbolos para contactos, bobinas y bloques en una serie de escalones ejecutados de forma secuencial por un controlador.

PLC

Son dispositivos electrónicos creados específicamente para el control de procesos secuenciales, con el fin de lograr que una máquina o cualquier otro dispositivo funcionen de forma automática. Puesto que están pensados para aplicaciones de control industrial, su diseño les confiere una especial robustez.

BIBLIOGRAFIA

BIBLIOGRAFÍA GENERAL

ALCIATORE, DAVID y HISTAND MICHAEL B. Introducción a la Mecatrónica y a los Sistemas de Medición. 3a ed. California: Mcgraw-Hill, 2008.

CASTRO GIL, ALONSO y DÍAZ, FERNANDO. Comunicaciones Industriales: sistemas distribuidos y aplicaciones. Madrid: Paraninfo, 2007.

PIEDRAFITA MORENO, RAMÓN. Ingeniería de la Automatización Industrial. 2a. Edición. México D.F.: Ra-ma, 2004.

EQUIPOS MECATRÓNICOS

http://www.festo-didactic.com/int-es/product-search/?search=GSM.
200805

http://www.festo-didactic.com/int-es/learning-systems/mps-sistema-de-producci-n-modular/mps-200/mps-203-it-mantenimiento-y-diagnosis-remotos-con-paquete-de-visualizaci-n-y-mechatronics-assistant.htm
200805.

http://www.wago.com/wagoweb/documentation/app_note/a1119.pdf.
200805

http://www.automation.siemens.com/tia/index_00.htm

200805.

http://www.automation.siemens.com/download/internet/cache/3/1378398/pub/es/k_schrift_es-0406.pdf

200805

REDES INDUSTRIALES

http://www.textoscientificos.com/redes/ethernet

200806

http://tec.upc.es/ie/practi/Sistemas.pdf

200807

http://www.profibus.com/metanavigation/weiche/pb/pb.html

200807

http://en.wikipedia.org/wiki/Profibus

200808

CONTROL LÓGICO PROGRAMABLE (PLC)

http://www.isa.uniovi.es/~felipe/files/infindII/documentos/Pres%20Genia%20IEC%201131-123.pdf

2009-11-21

http://es.wikipedia.org/wiki/Estandarizaci%C3%B3n.

2009-11-21

http://isa.uniovi.es/docencia/IngdeAutom/transparencias/iec1131-3%20espa%F1ol.pdf

2009-12-23

ANEXOS

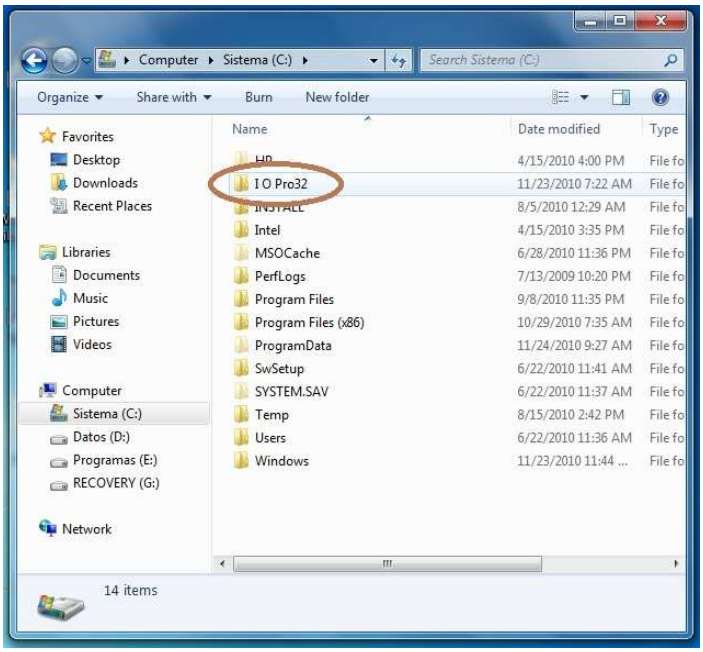
ANEXO 1

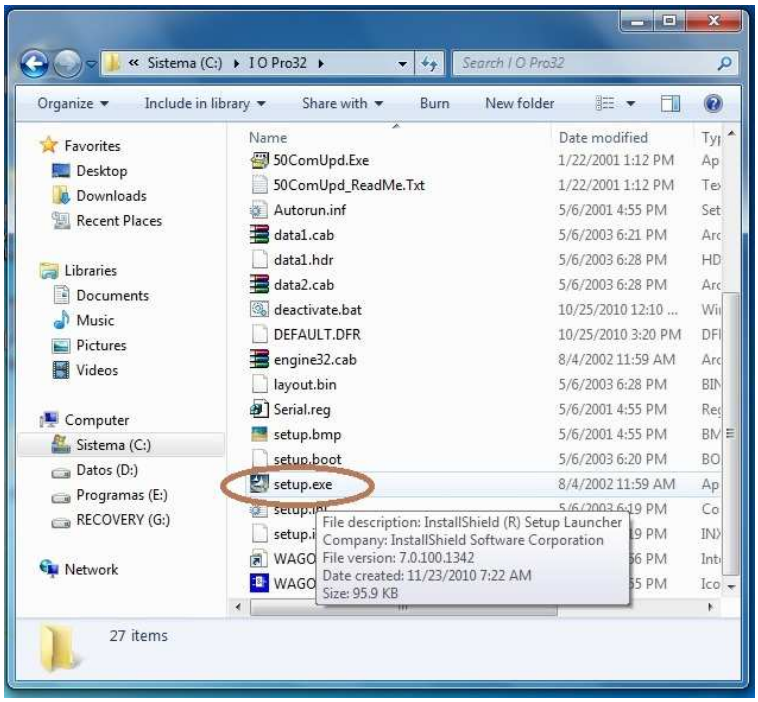
MANUAL BASICO DE PROGRAMACION DE WAGO I/O PRO 32

Proceso de instalación

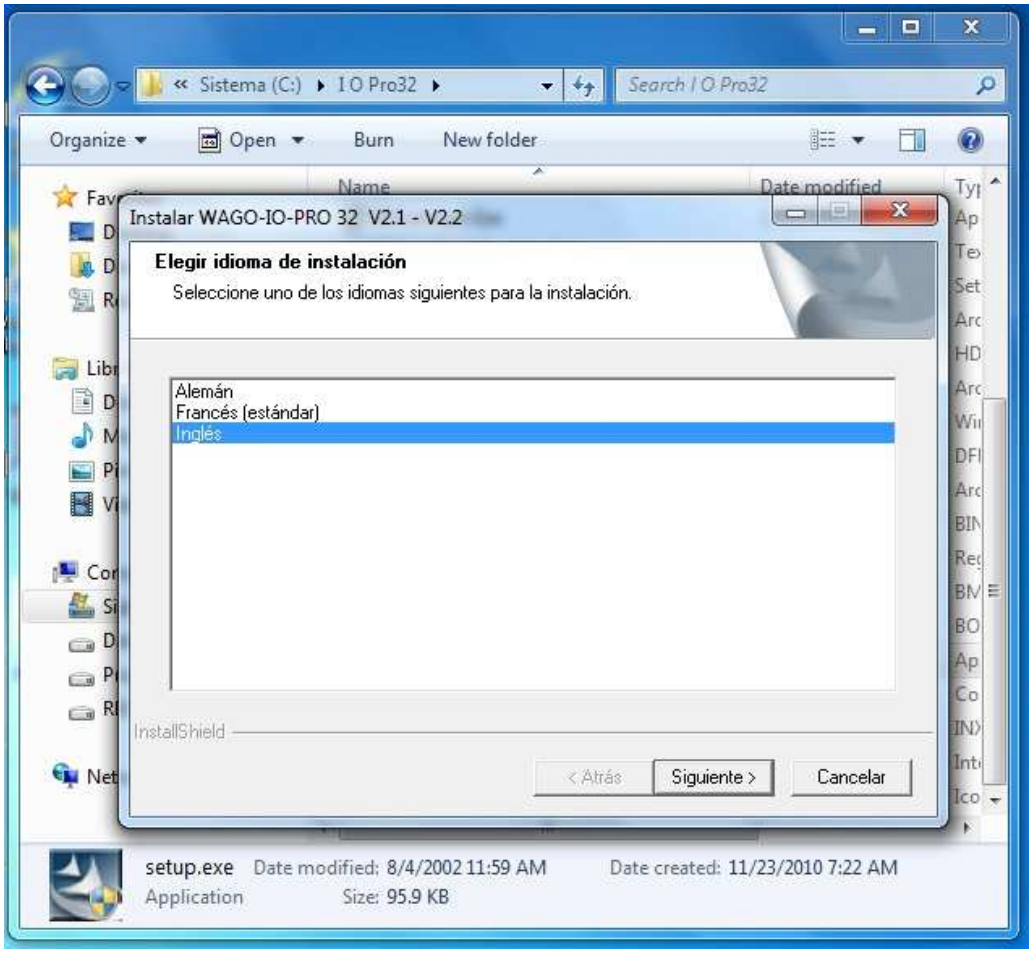
Para instalar el programa procedemos lo siguiente

1. Localizar la carpeta I O Pro32 y acceder al ejecutable SETUP.

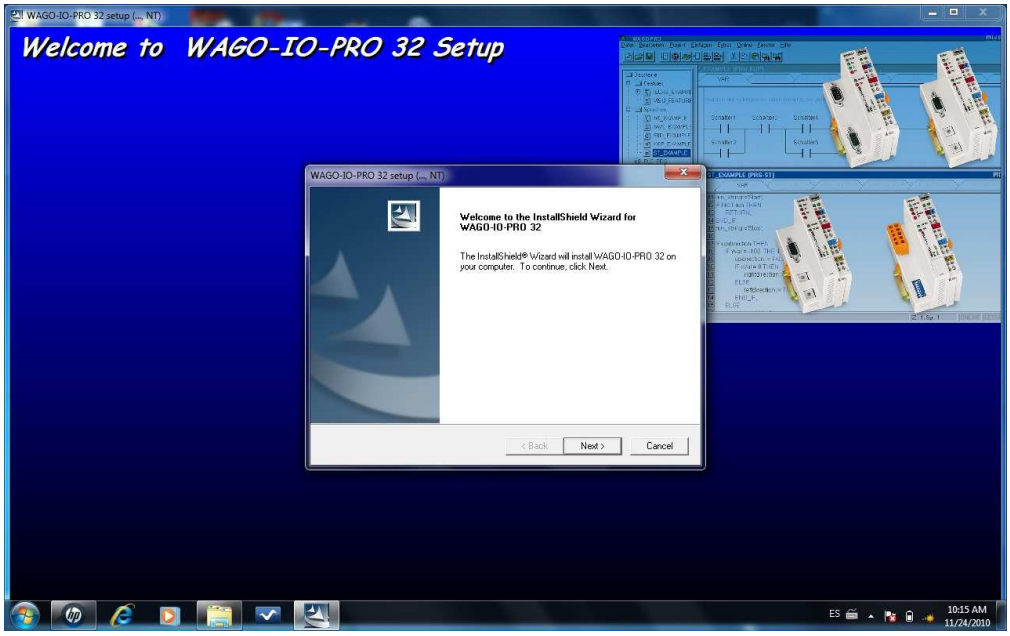




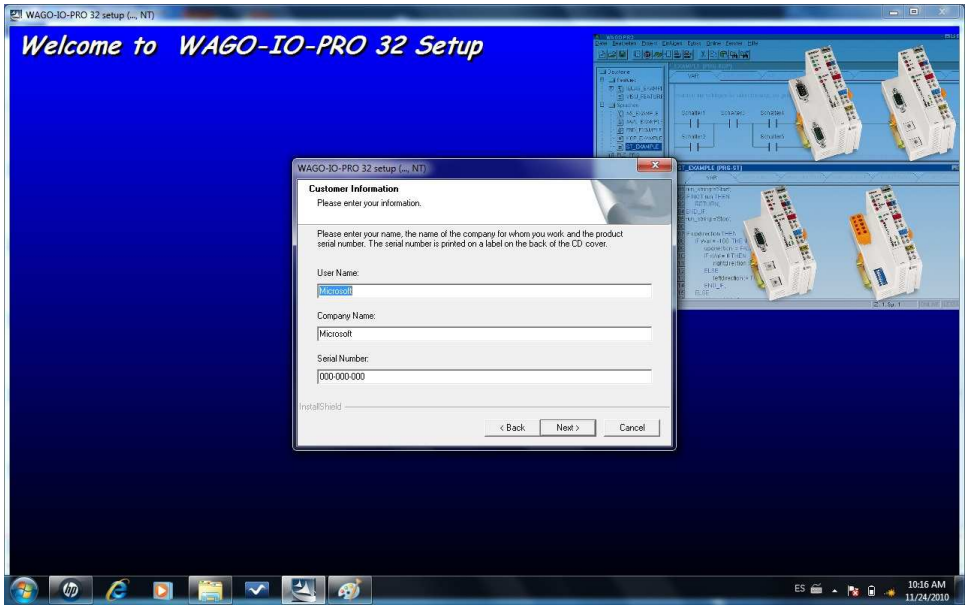
2. Al pulsar se presentará la pantalla en donde debemos escoger el idioma en el cual se instalará el programa. En nuestro caso escogemos el idioma INGLES.

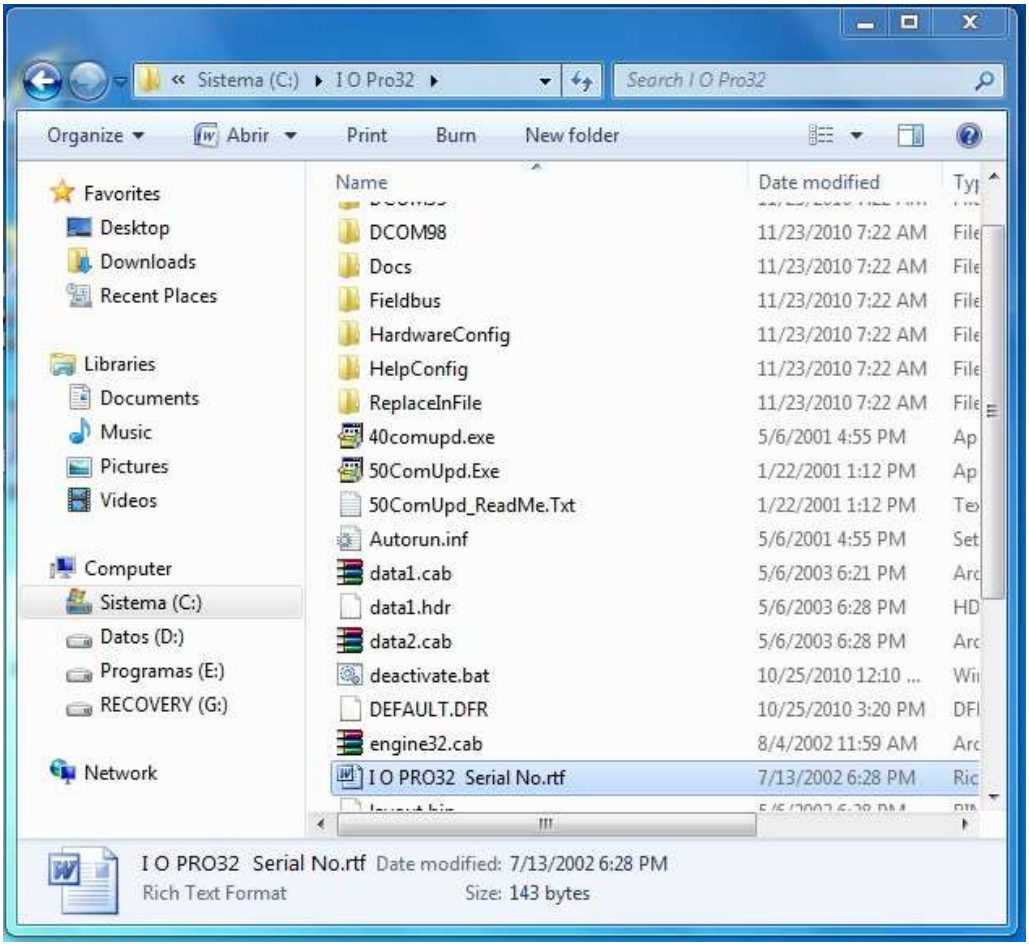


3. A continuación se presenta el logo de bienvenida del programa WAGO IO PRO 32 y los primeros pasos para continuar con la instalación.

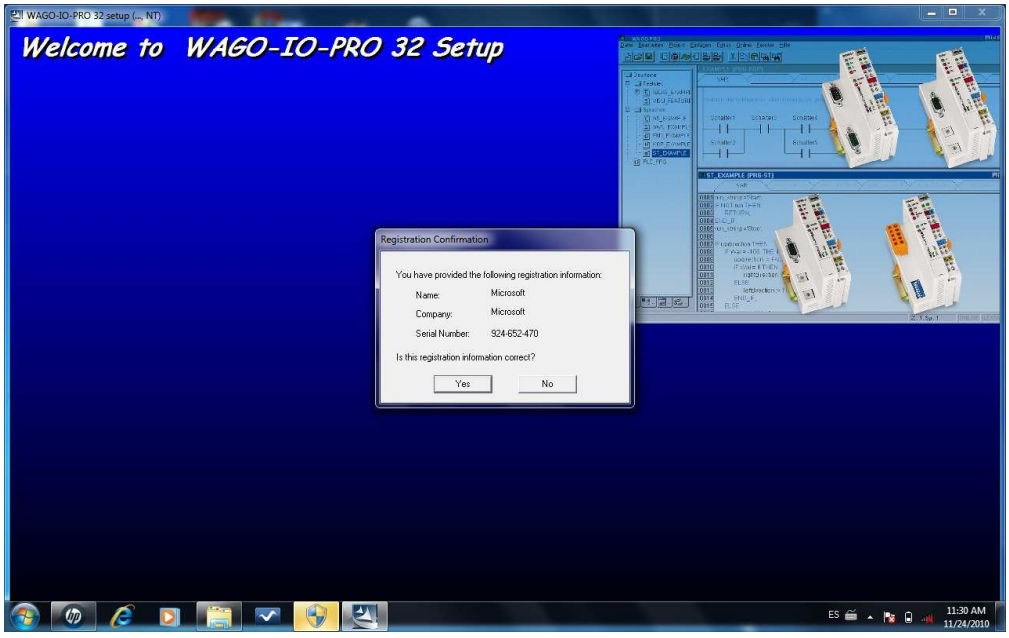


4. Pulsamos Siguiente y debemos llenar los datos requeridos y el serial que está en un archivo en formato .rtf, el mismo que se encuentra en la carpeta I O Pro.

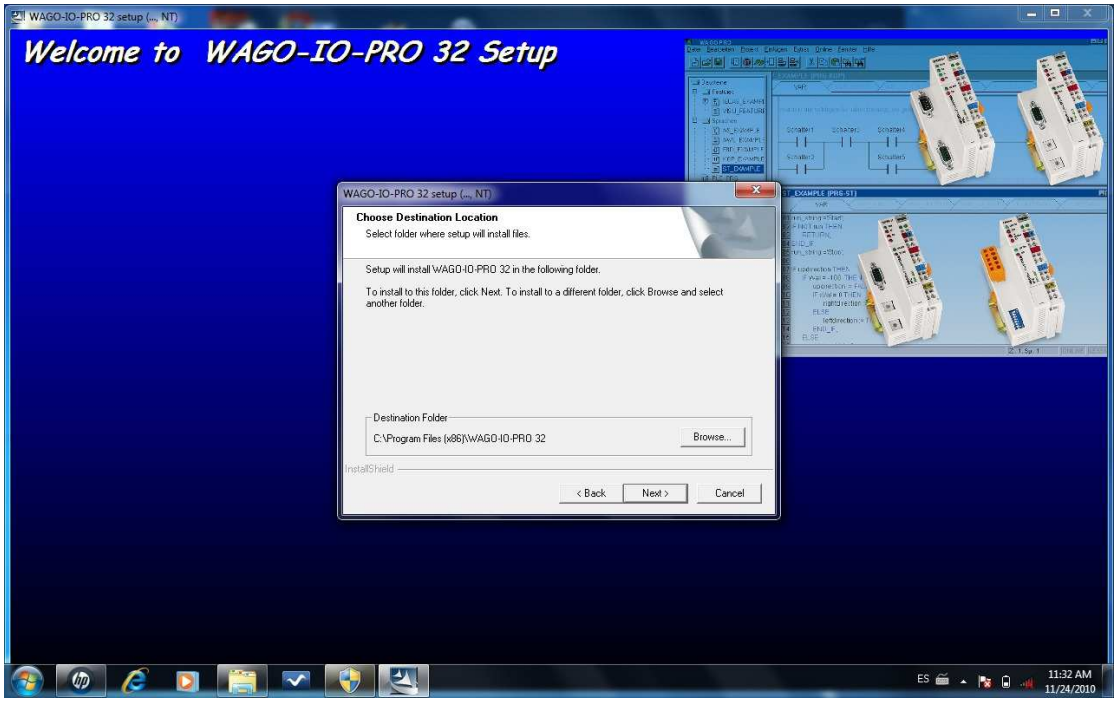




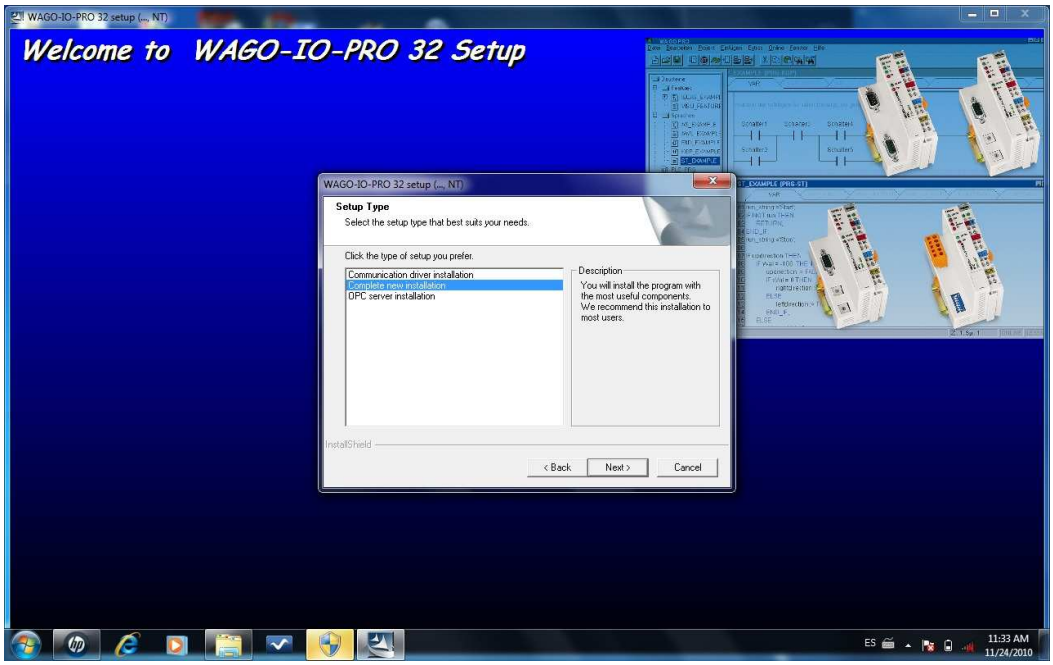
5. Después de pulsar Siguiente, confirmamos los datos que están los correctos.



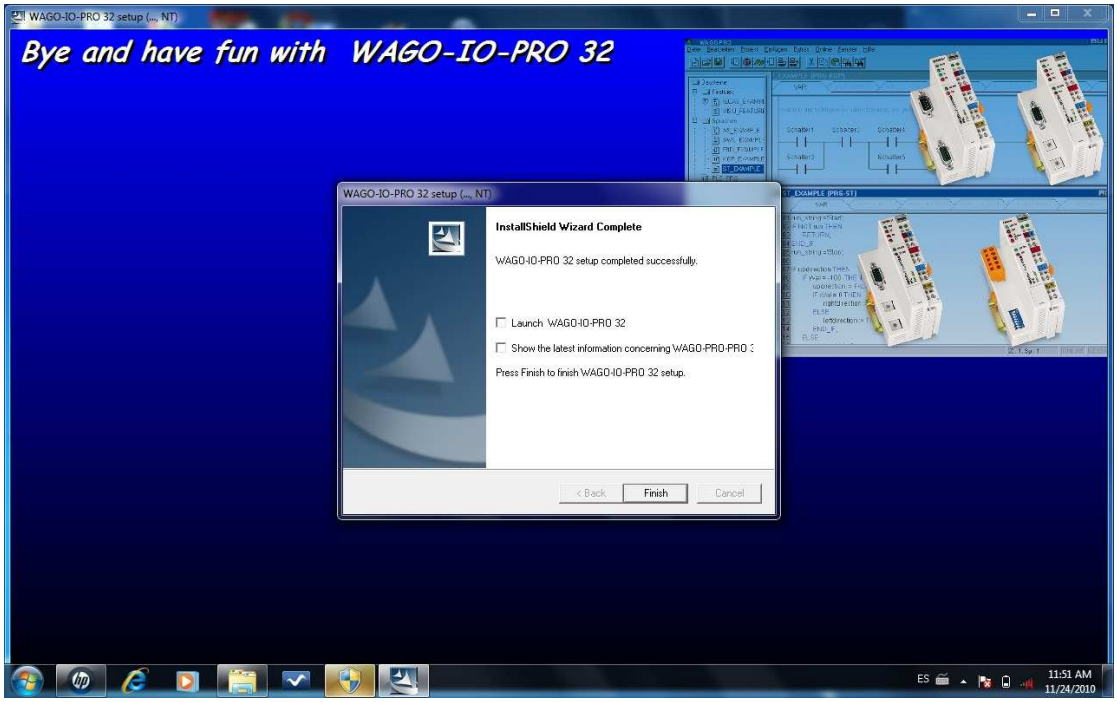
6. Al pulsar siguiente nos identifica en donde se va a instalar el programa por defecto, o si deseáramos ubicarlo en otro sitio lo podemos hacer. En nuestro caso conservamos la ubicación predeterminada del programa.



7. En la siguiente pantalla se ve tres opciones de instalación. Para nosotros, queremos solo instalar el programa, por lo tanto escogemos la opción Completar nueva instalación.



8. De aquí en adelante solo nos pide la versión en la cual vamos a trabajar, escogemos la versión 2.2 y nos indica los accesos directos, nos advierte que debemos reiniciar el equipo para no tener ningún problema y concluimos con la instalación.

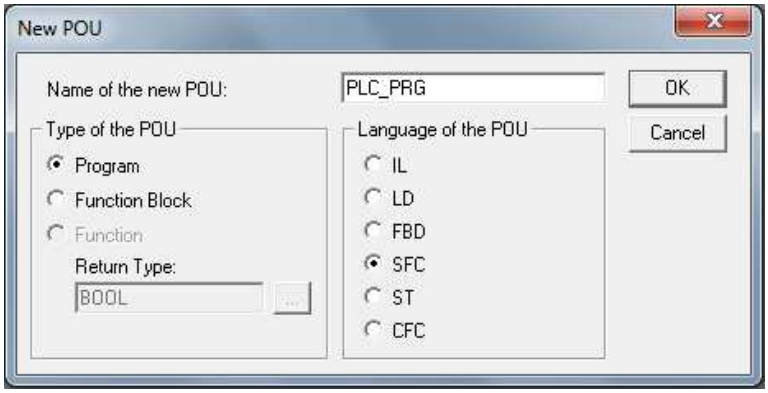


Creación del Proyecto

Antes de programar, recordemos que siempre que se accede al programa, este por defecto abrirá una ventana PLC_PRG ya sea del programa que se inició o del último programa utilizado, por tanto en el momento de ir al menú FILE se eliminará el archivo y crearemos uno nuevo o modificaremos el que requeramos.

Al acceder al menú FILE tenemos las siguientes opciones de programación: Lista de Instrucciones (IL), Escalera (LD), Diagrama de bloques funcionales (FBD), Gráfico funcional secuencial (SFC), Texto estructurado (ST) y Gráfico continuo de funciones (CFC). Escogeremos la opción SFC.

Explicaremos la creación de simulaciones para PLC usando el lenguaje Grafcet, para esto se selecciona SFC como se muestra en la imagen siguiente.



Guardar el proyecto

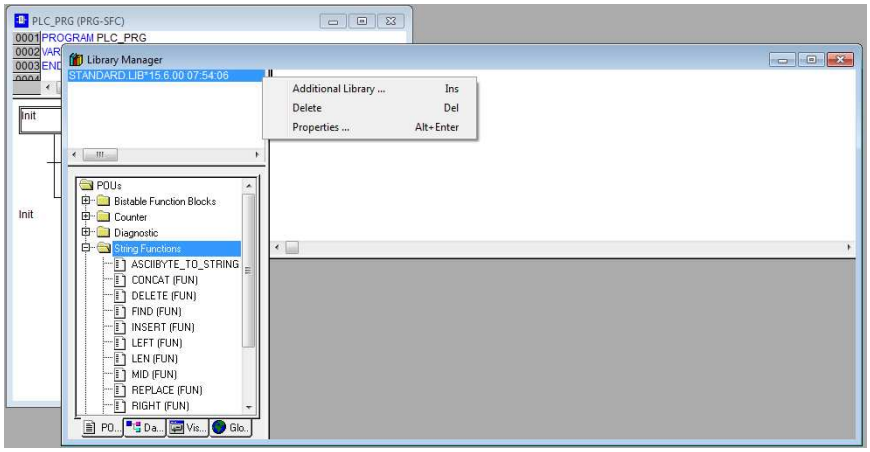
Para guardar el proyecto ir a FILE – Save As…. y escribir el nombre del proyecto deseado. Todo archivo se guarda por defecto en la siguiente carpeta y con la extensión (*.pro)

C:/Archivos de Programa/WAGO-IO-PRO 32/Project

Inserción de bibliotecas

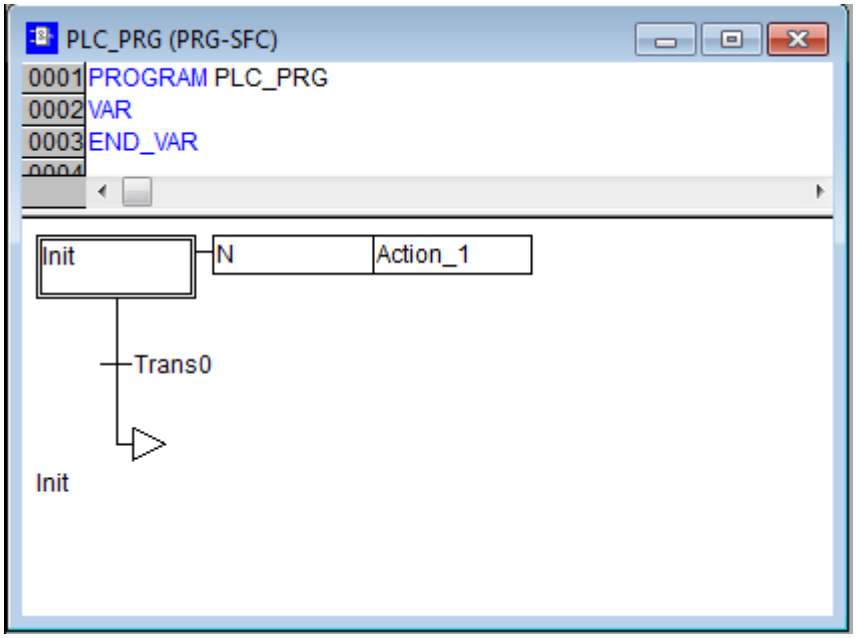
Para la creación del proyecto es necesario contar con las herramientas, en la programación con grafcet se requieren dos bibliotecas, la ***Standard*** y la de la norma IEC.

Para anexarlas, ir al menú Window y escoger la opción “Library Manager”, aparece un recuadro en el cual está por defecto la librería Standard.lib, añadiremos la librería Iecsfc.lib de la siguiente manera, hacer click derecho sobre la librería predeterminada y escoger la opción “Additional Library….”, aparece un recuadro con algunas librerías, escogemos la librería **Iecsfc.lib**, y tenemos listas las herramientas para programar.

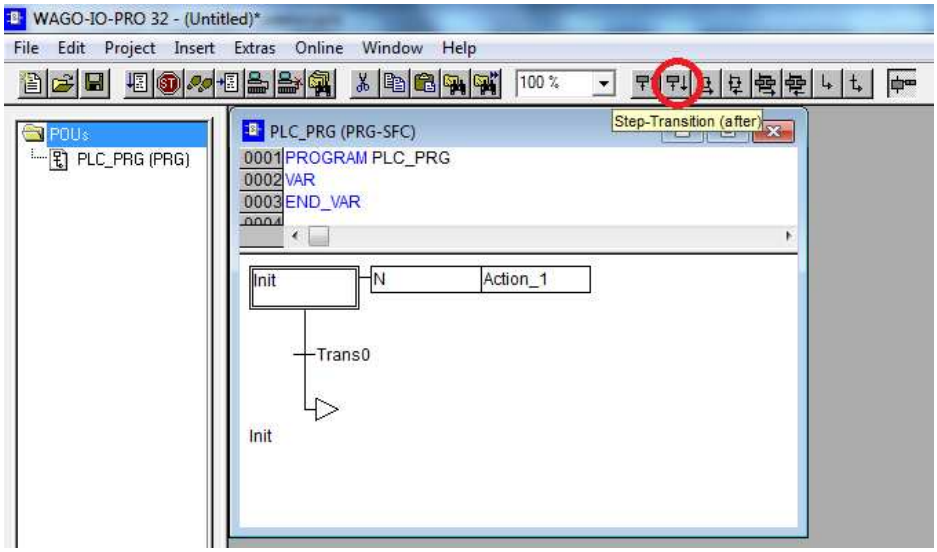


Programación en Grafcet

El grafcet que aparece en el editor, tiene la etapa principal, se reconoce por ser la primera y porque es un cuadro con doble línea, la acción asociada a esta etapa que es el cuadro a la derecha con su cualificador y el actuador (Action_1), la transición (Trans0) y el retorno.



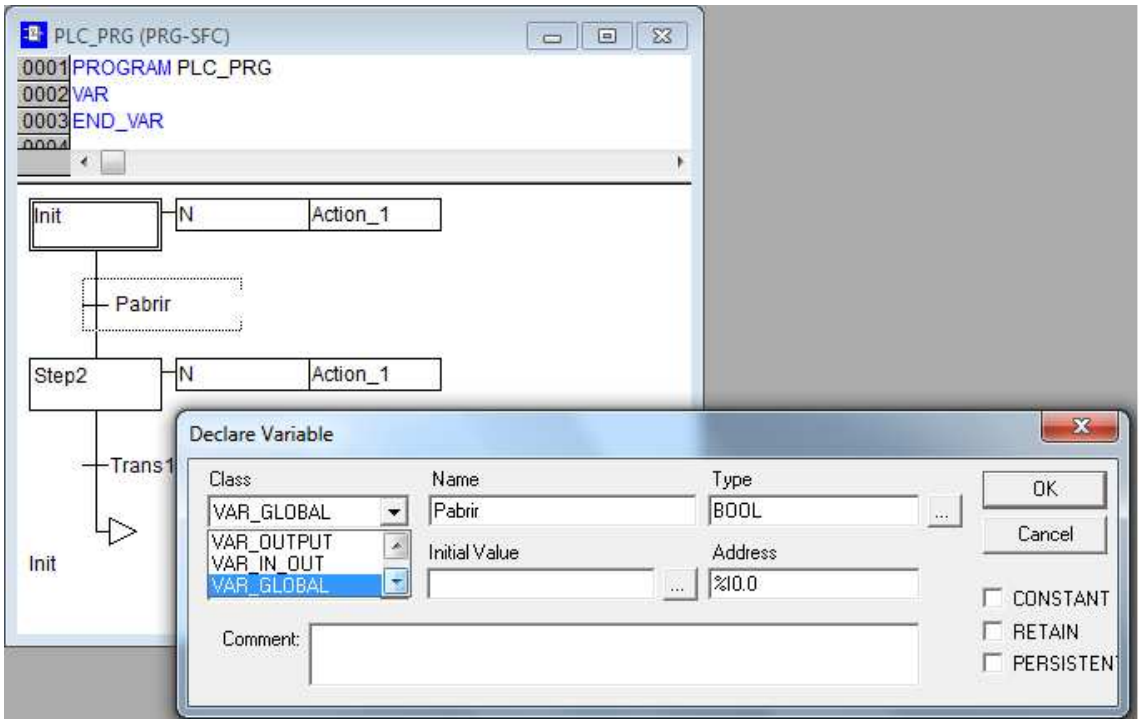
Para anexar más etapas se debe ubicar el cursor sobre la transición cero (Trans0) e ir a la barra de herramientas y dar click sobre la opción “Step-Transition (after)” o “Transición de paso después”.



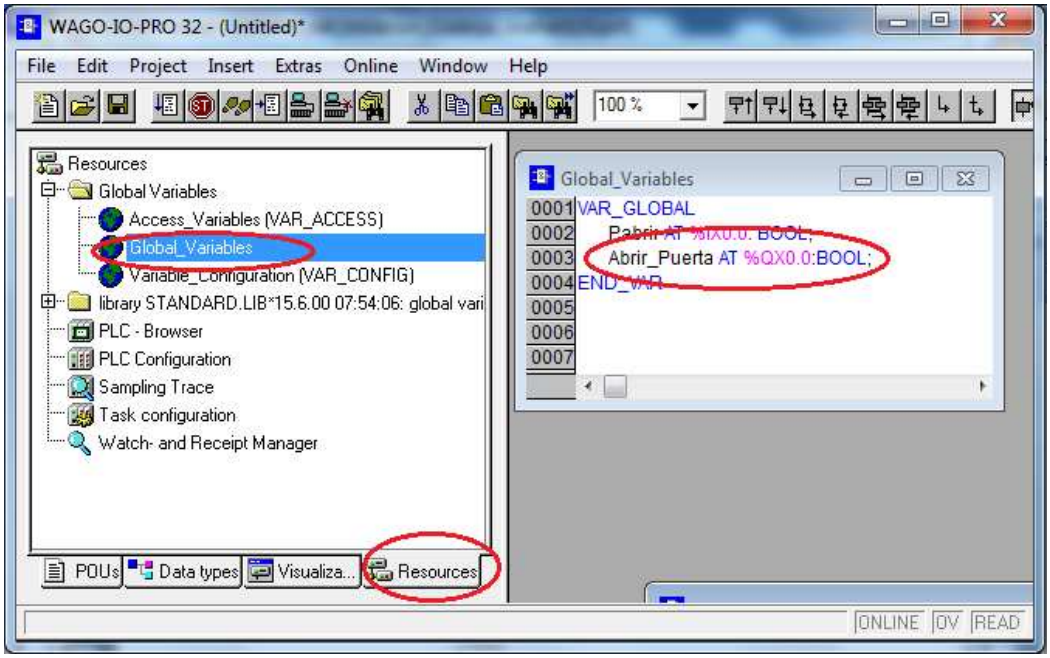
La nueva etapa se agregó después de la transición, si se quiere anexar la nueva etapa antes de la transición cero se da click sobre el icono al lado del usado.

Creación de variables

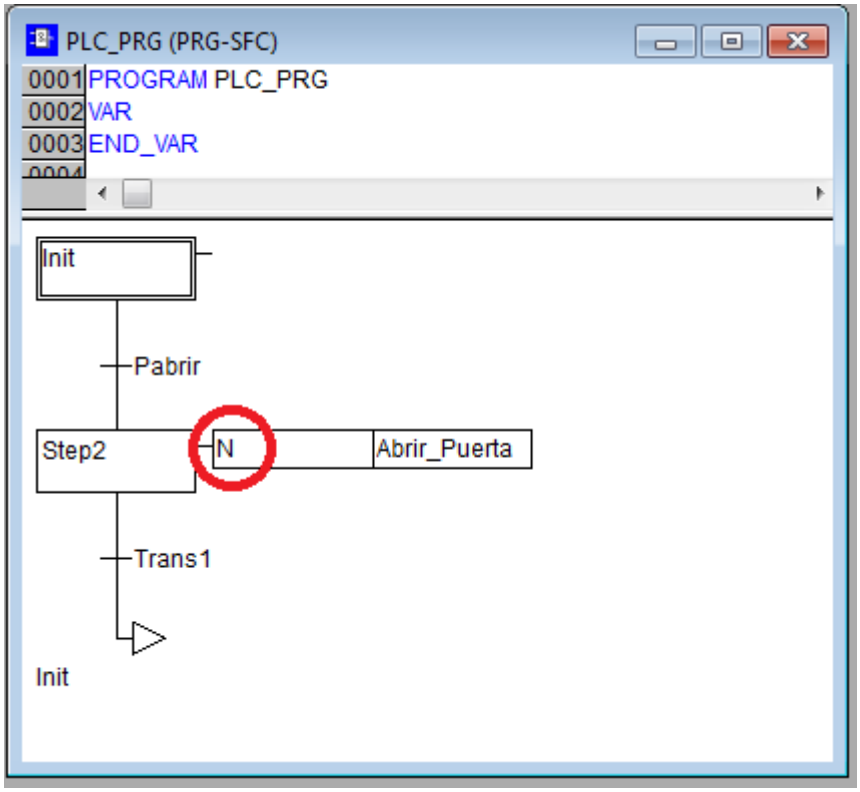
Posicionar el cursor sobre la transición cero, en este caso se escribe el nombre del sensor o el mando, por ejm: Pulsador abrir: Pabrir, al terminar de asignar nombre aparece la ventana para la declaración de variables, en él se selecciona el tipo de variable en “Class”, se selecciona Variable global, y se le asigna una dirección de entrada, como lo presenta la noma IEC 61131-3. Para entradas se usa I y para salida se usa en el direccionamiento Q. Ejm. %I0.0.



La asignación de variables de salida, o las acciones, debe configurarse en la pestaña de “Resources” en la parte inferior izquierda, llamado organizador de objetos, escoger la sección “Global Variable” y dentro de las sentencias VAR_GLOBAL … END_VAR se añade lo siguiente:
“Abrir_Puerta AT %QX0.0:BOOL”



Finalmente se tiene un grafcet sencillo.



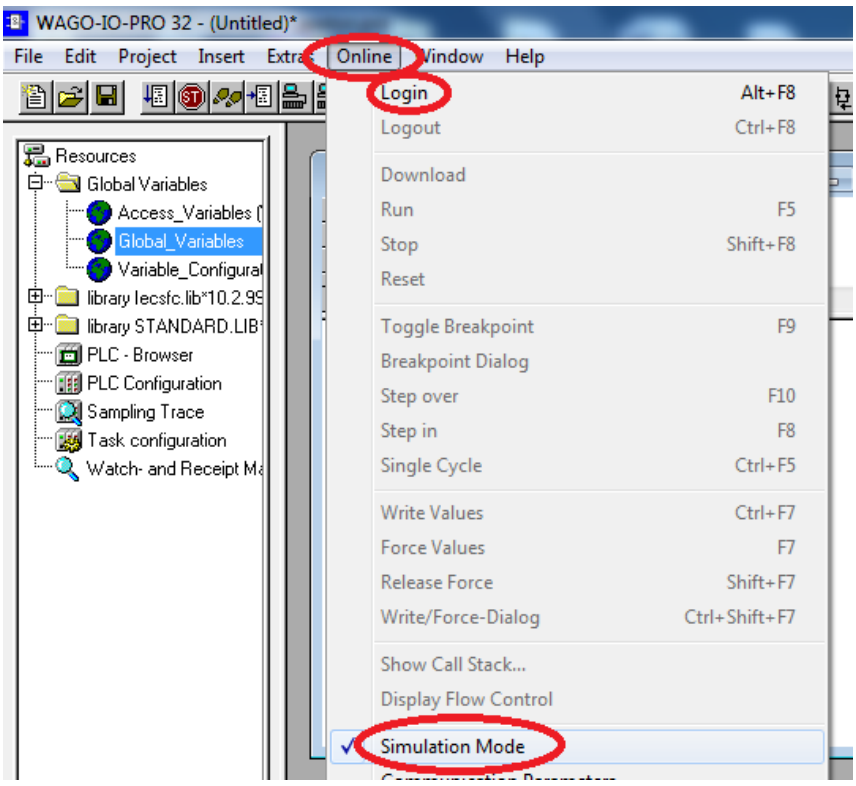
Las acciones tienen determinadas características como la N, a continuación una tabla de su significado.

N	Acción Non Stored	La acción se ejecuta solo mientras la etapa este activa
R	Acción Memorizada, RESET	Cuando la etapa se activa el actuador permanecerá en estado bajo hasta que se produzca un RESET
D	Acción retardada	La acción se ejecuta después de un tiempo
L	Acción limitada en el tiempo	La acción se ejecuta solo en una lapso
P	Impulso	La acción se realiza solo por un instante de tiempo
S	Acción memorizada, SET	Mantiene la acción así la etapa ya no esté activa

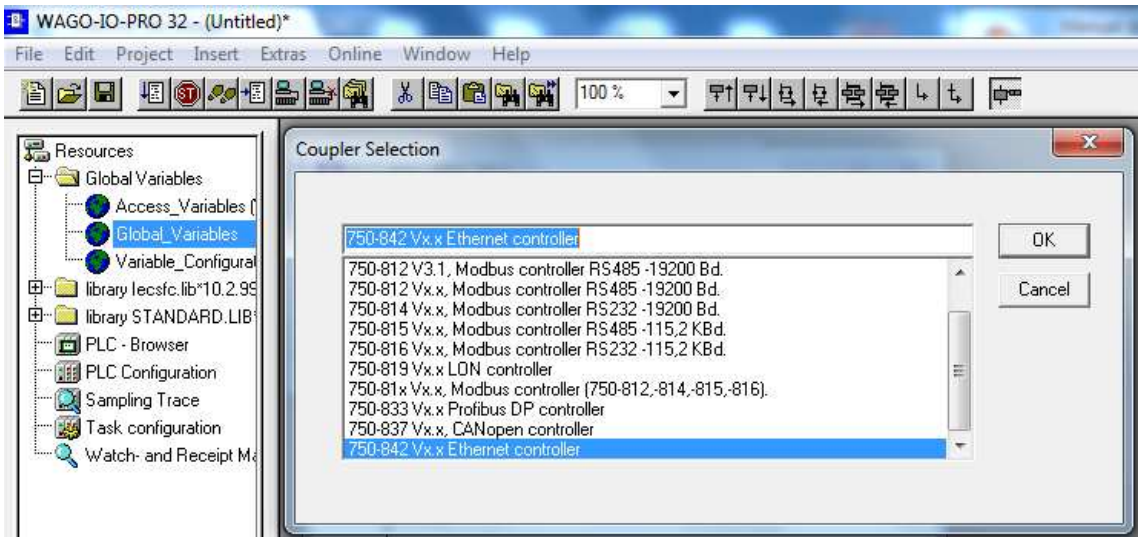
Proceso de simulación sin el PLC

Para completar el ejercicio, a la transición 1 le declaramos como una variable de entrada Pcerrar. Ahora para ejecutar el programa seguimos los siguientes pasos:

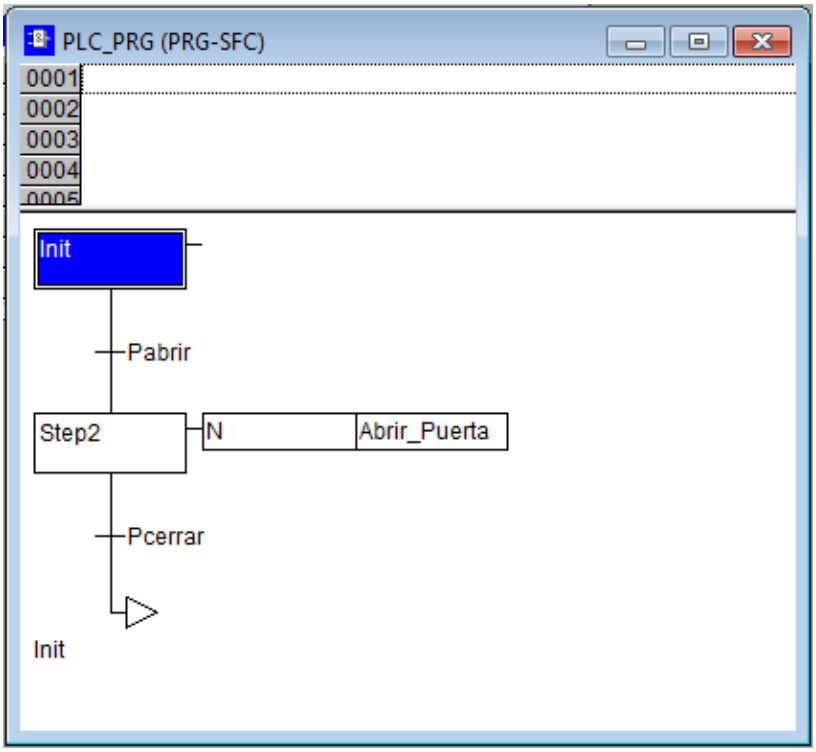
1. Ir al menú Online y habilitar la opción “Simulation Mode”.
2. Ir al menú Login y a continuación aparece un recuadro denominado acoplador de selección.



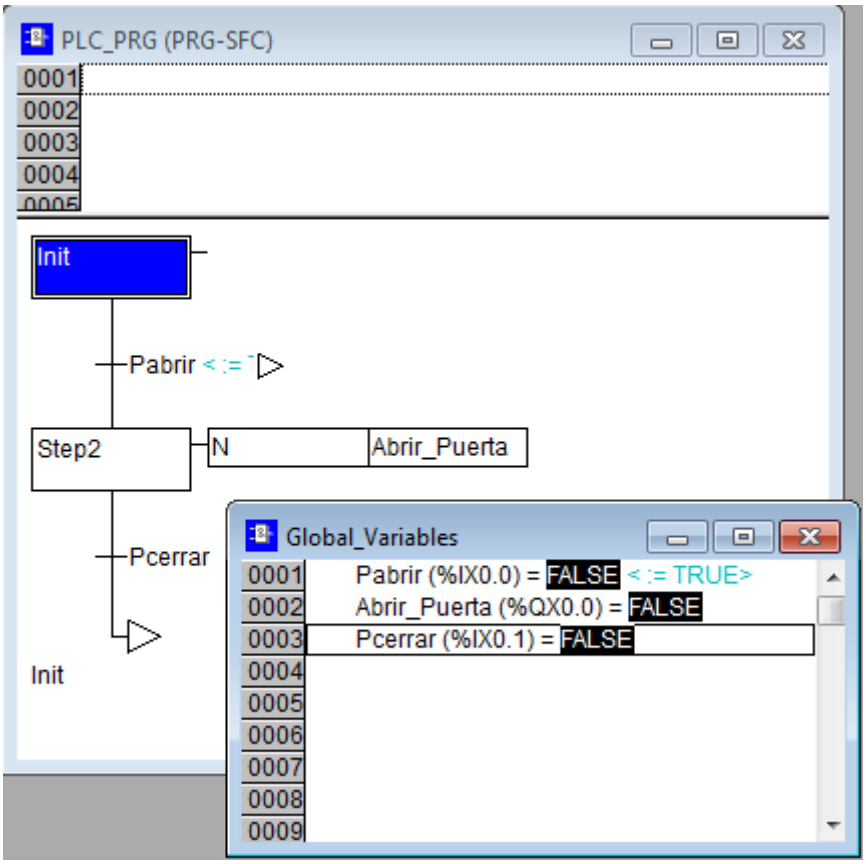
3. Por lo general se escoge la última opción para no tener problemas de configuración. Cuando tengamos conectado el PLC al equipo y estamos usando el programa WAGO, al deshabilitar el simulador se cargará automáticamente el programa para las respectivas pruebas.



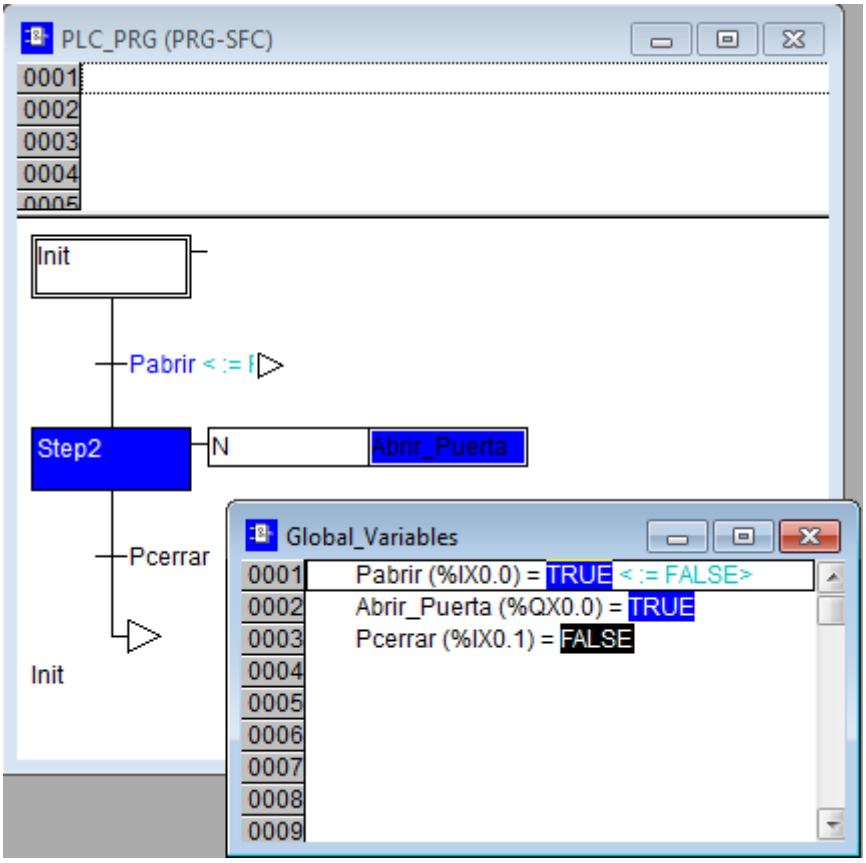
4. En el momento de pulsar OK tendremos habilitado el programa. En el menú Online pulsamos la opción RUN y tenemos el programa listo para ser probado.



Para que el programa siga ejecutando paso por paso, abrimos en “Recursos”, y acceder a las variables globales,



Hacemos doble click para cambiar el valor booleano TRUE en la variable de entrada Pabrir y después pulsar CTRL+F7, esto permite que el programa siga a la segunda etapa, y podemos volver a pulsar doble click en la variable de entrada Pabrir para que tengamos el valor FALSE y se deshabilite la entrada. A continuación pulsamos doble click en la entrada Pcerrar para tener el valor TRUE y después CTRL+F7 y volvemos a la posición Init, comprobando que el programa sí funciona.



Habrán programas que se requerirán secuencias simultaneas, saltos, etc. se puede acceder a la barra de herramientas y manipular las diferentes opciones. Requerirá mucha práctica para poder usar hábilmente este programa. Este es solo un manual básico para programar con WAGO IO-PRO 32.