PONTIFICIA UNIVERSIDAD CATOLICA DEL ECUADOR SEDE AMBATO

Facultad de Ingeniería
Escuela de Sistemas

DISERTACION DE GRADO PREVIA LA OBTENCION DEL TITULO DE INGENIERO EN SISTEMAS

"Estudio de los Autómatas Programables y su simulación a través del computador"

Guillermo Efraín Almeida Garzón
Mauricio Xavier Rodríguez Mora

DIRECTORA DE TESIS: Ms.C. ROXANA MERIÑO M.

Ambato 1999

PONTIFICIA UNIVERSIDAD CATOLICA DEL ECUADOR SEDE AMBATO

Facultad de Ingeniería
Escuela de Sistemas

DISERTACION DE GRADO PREVIA LA OBTENCION DEL TITULO DE INGENIERO EN SISTEMAS

FACULTAD DE INGENIERIA
ESCUELA DE SISTEMAS

"Estudio de los Autómatas Programables y su simulación a través del computador"

Director: .....
MsC. Roxana Mriño M.

Revisores: .....
MsC. Wigberto Sánchez

.....
Ing. Patricio Chambers

Guillermo Efraín Almeida Garzón
Mauricio Xavier Rodríguez Mora

DEDICATORIA

A Dios que me concedió la salud, voluntad y perseverancia para conseguir mis objetivos.

A mi esposa y a mi pequeña hija quienes estuvieron junto a mi en todo momento y se constituyeron en el apoyo firme y la motivación principal para el desarrollo y culminación de mis estudios.

A mis padres quienes inculcaron en mí el amor por los estudios y la superación constante.

Guillermo

DEDICATORIA

Este Trabajo lo dedico en primer lugar a Diós, por haberme brindado la posibilidad de llegar a concluir los estudios; a mis padres y hermanos por estar siempre pendientes y dispuestos a extenderme su mano en mis triunfos y caídas; a mi familia en general por el apoyo incondicional que me brindaron a cada instante y en cada momento del tiempo de estudios; a mis amigos y compañeros de trabajo que directa o indirectamente me han brindado su colaboración para la culminación de está carrera .

Mauricio

AGRADECIMIENTO

Un agradecimiento a todo el Personal docente de la Pontificia Universidad Católica del Ecuador Sede Ambato, quienes compartieron con nosotros sus conocimientos durante el tiempo de estudios, a nuestra directora de Tesis por su colaboración incondicional en la realización de éste trabajo, de la misma manera a nuestros revisores por su colaboración en el desarrollo y culminación del mismo.

Guillermo

Mauricio

INDICE

INDICE GENERAL

TEMA **Pág.**

CAPITULO I

CAPITULO II

CAPITULO III

CAPITULO IV

INTRODUCCION

La industria en nuestro país, dentro del proceso de integración andina, se ha visto abocada a introducirse en un proceso sistemático y acelerado de desarrollo, para ser capaz de competir en condiciones parejas con las industrias de los países vecinos.

El procedimiento de control de los procesos industriales para lograr una mayor productividad y un control de calidad mucho más efectivo, constituye una difícil tarea, así como también la intervención de personal especializado.

El control automático de procesos industriales permite una mayor capacidad de crecimiento en productividad, eficiencia, calidad y seguridad en la industria. En los momentos actuales, es necesario que se hagan todos los esfuerzos para que la producción industrial en nuestro país ingrese al mercado internacional a precios competitivos. En nuestro medio se están logrando buenos resultados con la implementación de los Autómatas Programables Industriales, y es necesario ampliar el campo de aplicación de los mismos hacia diferentes áreas, dentro del proceso de producción.

Los Autómatas Programables llamados también Controladores Lógicos Programables (PLCs) constituyen una de las herramientas de automatización industrial utilizadas para el control de procesos. Los PLCs no son otra cosa que procesadores de propósito específico dedicados a controlar procesos en tiempo real.

Estos dispositivos electrónicos no solamente son capaces de controlar los procesos industriales a través de los datos que le llegan desde los puntos de control, sino que también pueden transmitir estos datos hacia otros dispositivos que también los utilicen.

Desde esta perspectiva, es posible generar redes de autómatas que controlen varios procesos y en conjunto controlen todo el proceso de producción de una industria. Estos autómatas así enlazados constituyen una red de bajo nivel.

Esta idea sugiere la integración de las redes de autómatas programables que controlen los procesos industriales o las redes de computadoras de más alto nivel que manejan la información de una industria.

Como primera aproximación surge el hecho de hacer uso de un simulador informático que pueda ser útil en el aprendizaje y comprensión de la programación y funcionamiento de un autómata programable, sin la necesidad de recurrir al dispositivo mismo. Es necesario destacar que el simulador va orientado a todos los que deseen conocer la programación y funcionamiento básico de estos dispositivos electrónicos.

Para la consecución del objetivo final, el Estudio de los Autómatas Programables y su Simulación a través del Computador, el desarrollo del presente trabajo ha sido estructurado en dos etapas:

- La primera etapa, desarrollada en los Capítulos I y II, constituye un estudio teórico de los parámetros de referencia, que permiten conceptualizar la herramienta, dentro del medio ambiente de operación y sus características funcionales.

Esto comprende una ambientación dentro de los procesos industriales y un estudio de la información interna y externa que maneja un Autómata Programable Industrial, para discriminar aquello que es útil para el diseño del Simulador Informático.

Para el desarrollo de estos capítulos se ha utilizado como fuente bibliográfica principal el texto de Ingeniería de Control Moderna cuyo autor es OGATTA, en el mismo que se realiza un exhaustivo tratamiento acerca de los fundamentos del control automático.

También se ha utilizado en forma complementaria el texto de Electrónica Industrial de HUMPHRIES, el tratado de Autómatas Programables Nivel I de Urteaga, así como muchos otros textos auxiliares, de los cuales se ha obtenido extractos de información bibliográfica, los mismos que se hace referencia en la bibliografía, al finalizar el presente trabajo.

- La segunda etapa, desarrollada en los Capítulos III y IV, consiste en el estudio específico del autómata programable tomado como base, así como también el desarrollo de la Metodología de Análisis y Diseño del Software aplicado a este caso en particular.

Para el desarrollo de éstos últimos capítulos se emplean principalmente el Manual de SIEMENS LOGO! para el estudio del autómata programable específico, así como también el texto ANALISIS Y DISEÑO ORIENTADO A OBJETOS Método de Grady Booch y la guía oficial de Borland DELPHI de Michelle Manning en el aspecto de programación orientada a objetos.

El resultado final, constituirá un Simulador Informático cuyas características permitirán utilizar un lenguaje simple de programación de Autómata para el diseño y aplicación de circuitos de Control Automático.

Para el desarrollo de este proyecto, se ha creído conveniente utilizar como base los microcontroladores lógicos programables o autómatas de marca SIEMENS serie LOGO!.

El primer trabajo significativo en control automático fue el regulador centrífugo de James Watt para el control de la velocidad de una máquina de vapor, en el siglo dieciocho. Otros avances relevantes en la primeras etapas del desarrollo de la teoría de control se deben a Minorsky, Hazen y Nyquist, entre muchos otros. EN 1922 Minorsky trabajó en controladores automáticos de dirección en barcos y mostró cómo se podría determinar la estabilidad a partir de las ecuaciones diferenciales que describen el sistema. En 1932, Nyquist desarrolló un procesamiento relativamente simple para determinar la estabilidad de los sistemas de lazo cerrado sobre la base de la respuesta a lazo abierto con excitación sinusoidal en régimen permanente. En 1934 Hazen, quién introdujo el término servomecanismos para los sistemas de control de posición, desarrolló el diseño de servomecanismos repetidores capaces de seguir con exactitud una entrada cambiante.

Durante la década de los años cuarenta, los métodos de respuesta en frecuencia posibilitaron a los ingenieros el diseño de sistemas lineales de control de lazo cerrado que satisfacían los comportamientos requeridos. De fines de los cuarenta a principios de los cincuenta, Evans desarrolló el método por completo del lugar de las raíces. Desde fines de la década de los cincuenta, el énfasis en problemas de diseño de control se desplazó del diseño de uno de los muchos sistemas que funcionan, al diseño de un sistema óptimo en algún sentido determinado..

En los años 60, en muchas industrias se consideraba a los ordenadores como la mejor forma de conseguir aumentar la eficiencia, confiabilidad, productividad, y automatización de los procesos industriales. Los ordenadores poseen la capacidad de tomar y analizar datos a velocidades extremadamente altas, tomar decisiones, y luego entregar la información al proceso de control. Sin embargo, existían desventajas asociadas al control por ordenador tales como alto coste, complejidad de los programas, reticencias del personal de la industria para confiar en una

máquina, y falta de personal entrenado en la tecnología de ordenadores. Por lo que las aplicaciones de los ordenadores en esta época fueron principalmente en las áreas de recopilación de datos, monitorización on-line de la producción y sistemas en bucle abierto de ayuda a la toma de decisiones .

Sin embargo, a mediados de los 60, surgió un nuevo concepto en los controladores electrónicos: los controladores programables (programmable controller, PC). El concepto de PC se desarrolló a partir de una mezcla de la tecnología de ordenadores de estado sólido y de los controladores secuenciales tradicionales, tal como el cilindro de avance gradual (dispositivo de conmutación mecánico rotativo) y el programador de estado sólido con módulos enchufables. El primer PC surgió para enfrentarse a los problemas planteados en la industria del automóvil, la cual tenía que efectuar costosos cambios en los controles de la línea de montaje cada vez que se introducía un nuevo modelo en producción. Los primeros PCs se instalaron en 1969 como sustitutos electrónicos de los controles por relés electrónicos. El PC presentaba el mejor compromiso entre la técnica de los diagramas lógicos en escala con relés, y la tecnología de estado sólido existente. Esto aumentó la eficiencia del sistema de fabricación de la industria del automóvil eliminando el costoso trabajo de tener que conexionar de nuevo los controles de relés utilizados en el proceso de la línea de montaje. El PC redujo las pérdidas de tiempo debidas a las modificaciones, aumentó la flexibilidad, y redujo considerablemente los requerimientos de espacio de los antiguos controles mediante relés.

Después de la introducción del PC en la industria de fabricación en 1969, el uso del PC se ha extendido también a la industria de procesos. Generalmente, una industria de procesos realiza las funciones y operaciones necesarias para modificar un material física o químicamente.

1.2. DEFINICION DE AUTOMATIZACION

Automatizar consiste en proceder al tratamiento automático de informaciones, transmitir las decisiones resultantes de este tratamiento y ejecutar tales decisiones mediante una acción o actividad concreta realizada por un autómata.

Partiendo de la progresiva mecanización, es decir, la sustitución de la mano de obra por el trabajo mecánico, ha ido evolucionando forzosamente la automatización con desarrollo espontáneo de los procesos de trabajo en una máquina o grupo de máquinas.

La nota característica es aquí la liberación de la mano del hombre de la ejecución de operaciones siempre iguales y reiteradas. El desarrollo se lleva a cabo en tres etapas bien definidas:

- En primer lugar la máquina ayuda al hombre y complementa sus fuerzas (máquinas auxiliares, servomecanismos), él dirige y manda.

- Luego la máquina sustituye al hombre y se hace cargo del trabajo, que se repite continuamente (máquinas-herramientas, líneas de transferencia) mientras el hombre inspecciona.

- Por último la máquina supera al trabajo manual; la máquina totalmente automática, trabaja más de prisa, mejor, sin pausas y con mayor economía, pero requiere también vigilancia y mantenimiento.

Cuando se realiza la automatización, los procesos parciales de un procedimiento son combinados en un ciclo automático. Las tareas de muchas máquinas se acoplan entre sí. Por esta razón, las máquinas individuales tienen que construirse en escala creciente de forma que sean controlables y regulables.

Finalmente podemos decir que la AUTOMATIZACION DE UN PROCESO (una máquina, grupo de máquinas o en general un sistema industrial) consiste en asegurar su conducción mediante un dispositivo tecnológico (automatismo o autómata).

1.3. FACTORES PARA AUTOMATIZAR UN PROCESO

Entre las múltiples razones que existen para automatizar un proceso industrial, se pueden citar los siguientes factores:

1. Seguridad.

La complejidad de ciertas operaciones, multiplica la posibilidad de errores

2. Calidad.

La constancia de un proceso depende de la posibilidad de mantener dentro de límites establecidos factores tales como la fatiga, la monotonía derivada de la repetición sistemática.

3. Rapidez.

Se puede preveer la realización de numerosas operaciones a partir de una ordenación constante y estricta, de forma que se evite reflexionar al final de cada operación para prevenir lo que sigue a continuación.

4. Precisión.

Los límites de la habilidad manual humana para procesos industriales pueden ser superados mediante aparatos y herramientas de gran complejidad.

5. Optimización de Recursos Industriales

Al eliminar toda posible interrupción del proceso debido a factores humanos, la demanda de esfuerzo a la máquinas no presenta, como en el caso del hombre, inconvenientes de tipo ético o moral, y sí únicamente económico, de manutención y puesta a punto.

6. Reducción de los Costes

Al reducir el factor mano de obra y permitir la competitividad industrial, es posible mantener la propia continuidad empresarial y el subsiguiente mantenimiento de los puestos de trabajo adecuados. En este punto conviene insistir, en el factor social que la automatización conlleva. Suele imputarse a un autómata la responsabilidad de la desaparición de muchos puestos de trabajo al sustituir al hombre en tareas normalmente repetitivas pesadas y peligrosas pero nada más lejos de la realidad.

La competitividad empresarial que aporta la automatización en el terreno industrial permite, como se ha dicho, el mantenimiento de muchos puestos de trabajo que no podrían conservarse si esta competitividad no tuviera lugar.

1.4. VENTAJAS Y DESVENTAJAS DE LA AUTOMATIZACION

VENTAJAS

De acuerdo a esto se puede decir que la automatización de un proceso responde a dos criterios perfectamente definidos:

- **UN CRITERIO ECONOMICO.**

  Ya que su implantación conduce a:

1. Aumento de la Productividad
2. Mejora de la calidad de un Producto
3. Disminución de Costes de Producción

- **UN CRITERIO DE MEJORA DE CONDICIONES DE TRABAJO**

  Referido específicamente a:

1. Incremento de la Seguridad
2. Supresión de Tareas Repetitivas

DESVENTAJAS

Los efectos de automatización son de dos tipos, industriales y sociales. Se han expuesto las razones de ambos a pesar de lo cual, conviene reflexionar sobre estos puntos desde ópticas distintas.

Las consecuencias del automatismo conllevan en principio la reducción de la mano de obra, lo que permite rapidez en las amortizaciones de los equipos, pudiéndose realizar nuevas inversiones que permitan avanzar en el progreso técnico.

Un equipamiento no tiene porqué durar muchas décadas como antaño sino que se espera de él sólo algunos años de eficiencia, ya que será desbordado prontamente por las modernas tecnologías . La robotización, al ser considerada como una inversión de valor conocido y ya establecido, permite una evaluación exacta de las amortizaciones produciéndose el efecto anteriormente citado. Aunque la importancia de estos efectos es evidente, es la componente social por razones obvias la que más sensibilizada mantiene a la opción pública.

A primera vista, por la reducción de los tiempos de fabricación y subsiguientemente de la mano de obra que la automatización entraña, parecería que la robótica, tuviera que ser la causa del paro creciente actual, y más preocupante en el futuro. No obstante tal como se ha dicho antes, muchas empresas, y con ello sus colaboradores, deben su puesto de trabajo a la aplicación de los robots en sus industrias, independientemente de los efectos indirectos del desarrollo en la aplicación de tecnologías de punta, etc.

1.5. METODOS DE AUTOMATIZACION

Para conseguir automatizar un sistema existen dos posibles soluciones según sea el método empleado: el analógico y el digital.

En el método analógico las informaciones presentan el valor de las magnitudes físicas que varían de forma continua como son una tensión eléctrica, la presión de un embolo hidráulico, la temperatura de una caldera, etc. Estas magnitudes son enviadas al instrumental analógico clásico, como termómetro, manómetros, voltímetros, amperímetros, etc.

En el método numérico o digital, las informaciones señalan si la magnitud que se pretende automatizar existe o no, sin importar en principio su valor. Para poder transmitir una información digital, a un autómata, es preciso adjudicar un valor arbitrario a cada una de las posibilidades o estados que puede presentar tal magnitud: un 1 si tal magnitud existe, y un 0 si no existe

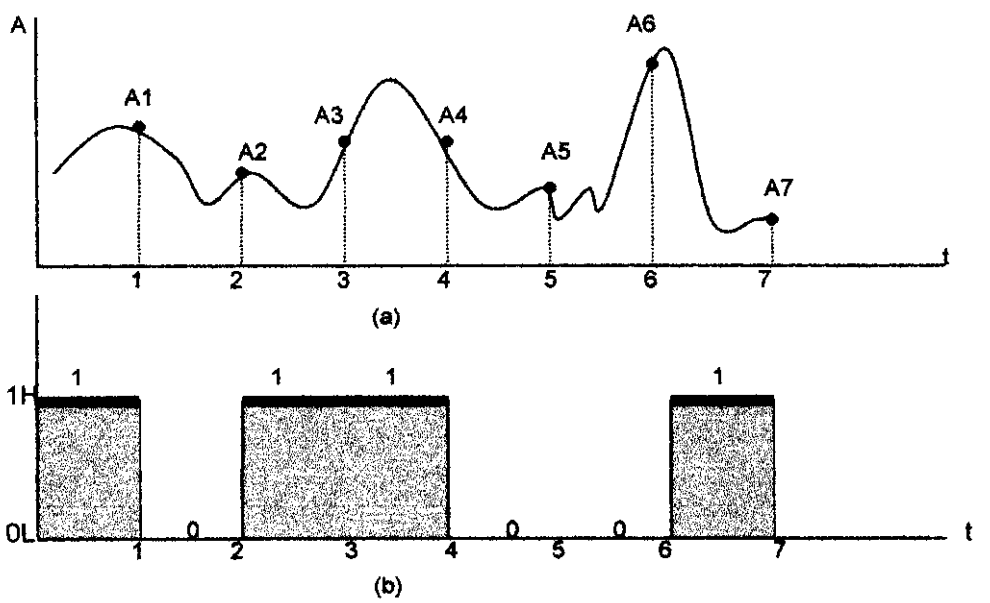


Figura 1-1.
(a) Gráfico de una señal analógica (b) Gráfico de una señal digital

1.6. COMPONENTES DE UN SISTEMA AUTOMATIZADO

Un sistema automatizado esta compuesto de dos partes claramente diferenciadas:

- **PARTE OPERATIVA** que constituye la propia máquina, y efectúa las operaciones (transformación de materia prima, transporte, soldadura, ensamblaje).

- **PARTE COMANDO** que suministra órdenes a la parte operativa, la cual suministra a la parte comando las informaciones relativas al estado del proceso.

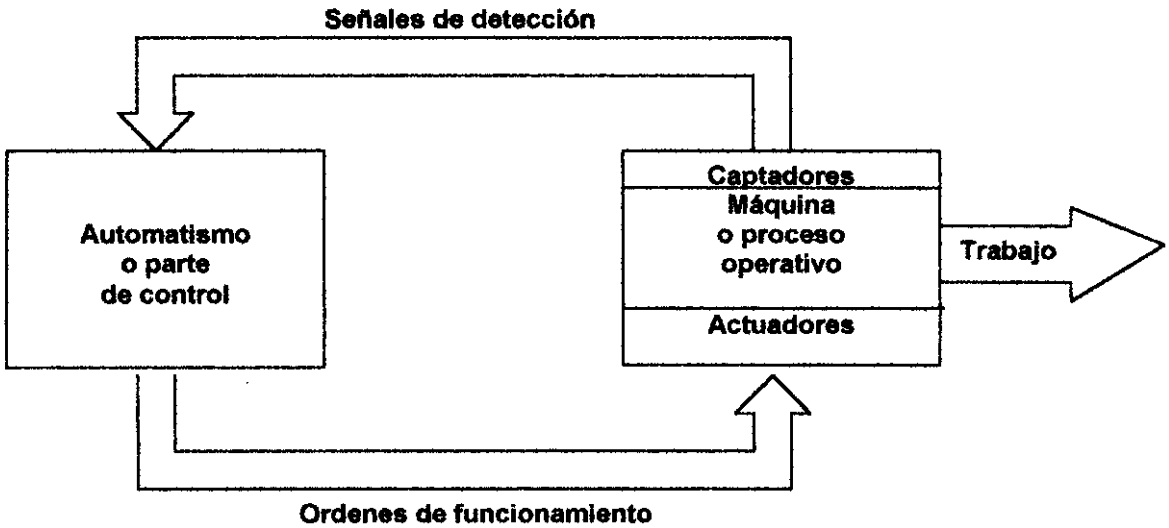


Figura 1-2
Componentes de un Sistema Automatizado

A todo lo expuesto anteriormente se añade el diálogo, órdenes/información entre la parte operativa y la parte comando. Esta última a la vez intercambia la información con el operador del cual recibe consignas y al cual suministra informaciones.

En la figura 1-3 se representa el proceso automatizado, donde la parte operativa informa del estado del proceso a la parte comando a través de los captadores, y esta a su vez, en función de la información recibida por el proceso con los captadores, y de las consignas del operador o del programa, manda órdenes a la parte operativa, cambiando el estado de los comandos.

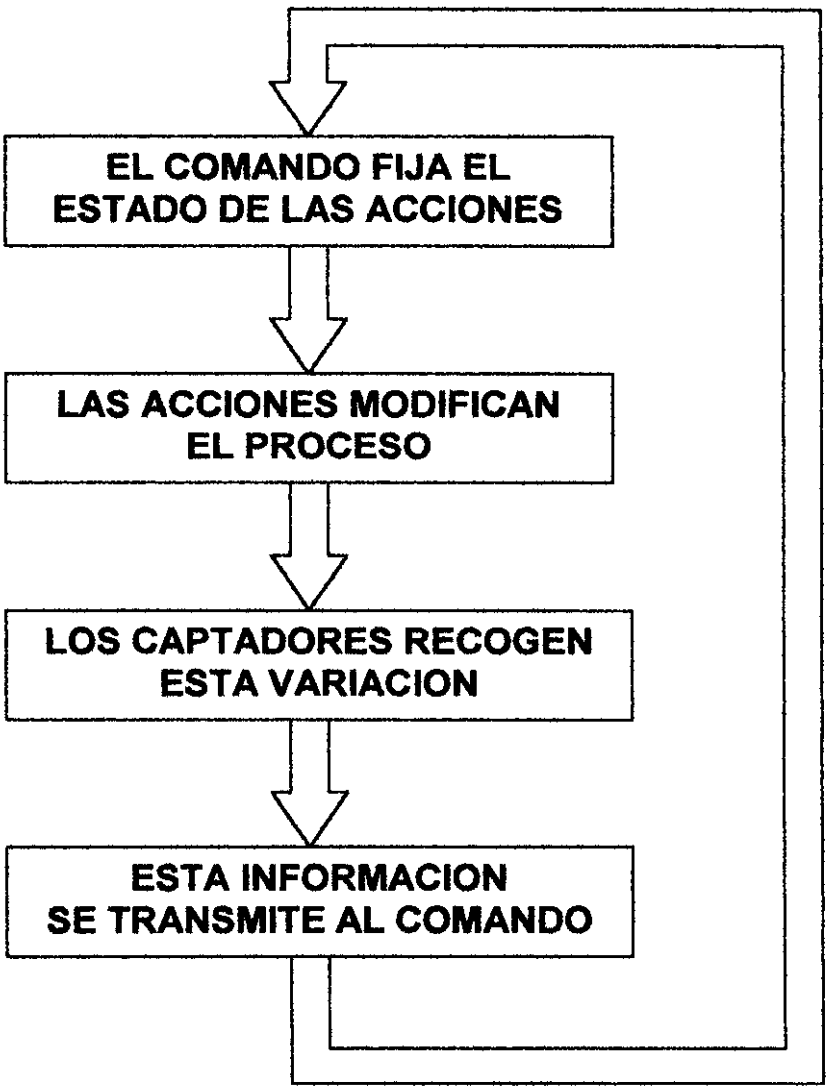


Figura 1-3
Bucle de Automatización

Un sistema automatizado es según se ha visto un sistema en bucle. En efecto en referencia a la figura 1-3 se destaca un bucle evidente.

Habiendo precisado la noción de bucle, es difícil definir donde están las salidas y las entradas. Generalmente se opta por definirlas desde el punto de vista del comando, así los captadores son las entradas para la parte comando y los actuadores son sus salidas.

1.7. SISTEMAS DE CONTROL AUTOMATICO

Los desarrollos más recientes en la teoría de control moderna están en el campo de control óptimo de sistemas, tanto determinísticos como estocásticos, así como un sistema de control complejo con adaptación y aprendizaje. Ahora que las computadoras digitales se han abaratado y reducido en tamaño, estas pueden utilizarse como parte integral de estos sistemas de control. Las aplicaciones recientes de la teoría de control moderna incluyen sistemas no ingenieriles como los de biología, biomedicina, economía y socioeconomía.

1.7.1. DEFINICIONES

Variable Controlada

La variable controlada es la cantidad o condición que se mide y controla. La variable manipuladora es la cantidad o condición modificada por el controlador, al fin de afectar la variable controlada. Normalmente la variable controlada es la salida del sistema. Control significa medir el valor de la variable controlada del sistema, y aplicar al sistema la variable manipulada para corregir o limitar la desviación del valor medido, respecto al valor deseado.

Ejemplos de variables manipulables o controladas son presión, temperatura, caudal de un fluido, y nivel de líquido.

Plantas.

Una planta es cualquier objeto físico que deba controlarse (como un reactor químico, vehículo espacial, etc.), funcionando conjuntamente, cuyo objetivo es realizar una operación determinada.

Procesos.

Se define un proceso como una operación o desarrollo natural, caracterizado por una serie de cambios graduales, progresivamente continuos que se suceden uno a otro de un modo relativamente fijo, y que tienden a un determinado resultado o fin.

Sistemas

Un sistema es una combinación de componentes que actúan conjuntamente y cumplen un determinado objetivo. Un sistema no se encuentra limitado a objetivos físicos. El concepto de sistema puede aplicarse a fenómenos dinámicos abstractos, como los que se encuentran en economía.

Perturbaciones

Una perturbación es una señal que tiende a afectar inversamente el valor de una salida de un sistema. Si la perturbación se genera dentro del sistema se la denomina interna, mientras que una perturbación externa se genera fuera del sistema y constituye una entrada. Cada sistema de control de procesos tiene una o más perturbaciones. Las perturbaciones tienden a variar la variable controlada. Ejemplo de perturbación puede considerarse la temperatura que rodea un tanque en un sistema en el que se calienta la leche a la temperatura de pasteurización.

Control Retroalimentado

Es una operación que, en presencia de perturbaciones, tiende a reducir la diferencia entre la salida de un sistema y alguna entrada de referencia, realizándolo sobre la base de esta diferencia. Aquí únicamente se especifican las perturbaciones no previsibles ya que las previsibles o conocidas, siempre pueden compensarse dentro del sistema

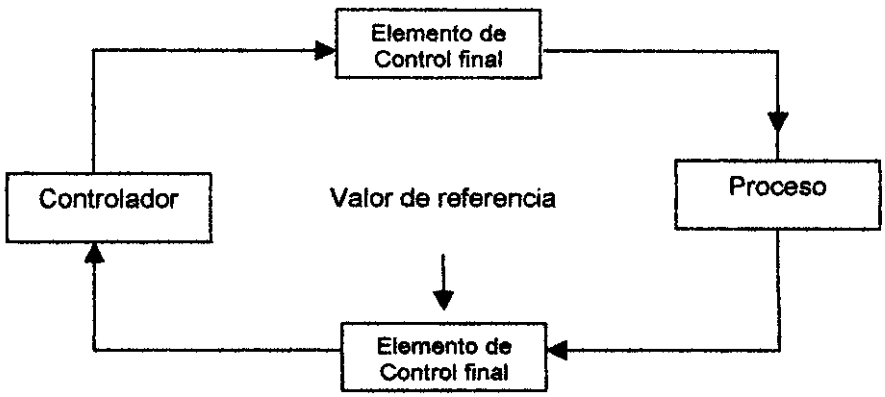


Figura 1-4
Diagrama de bloque de un sistema de control con retroalimentación en bucle cerrado

Sistemas de Control Retroalimentado

Se denomina a aquel que tiende a mantener una relación preestablecida entre la salida y alguna entrada de referencia, comparándolas y utilizando la diferencia como medio de control. Los sistemas de control retroalimentado no están limitados al campo de la Ingeniería, sino que se les puede encontrar en áreas ajenas a la misma.

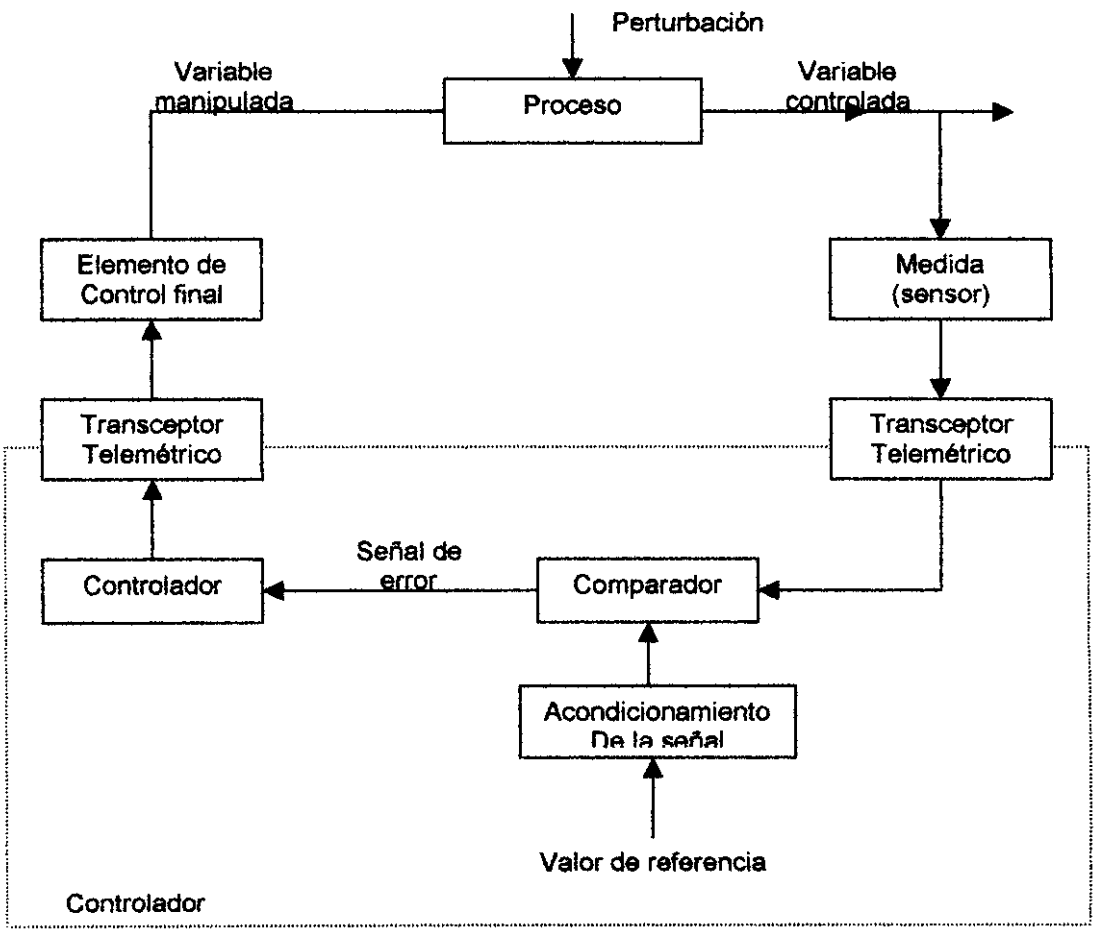


Figura 1-5
Diagrama de bloques ampliado de un sistema de control con retroalimentación en bucle cerrado

Servosistemas

Servosistema o servomecanismo es un sistema de control retroalimentado en el que la salida es algún elemento mecánico, sea posición, velocidad o aceleración. Por lo tanto, los términos servosistema o sistema de control de posición, velocidad o aceleración son sinónimos. Estos servosistemas se utilizan ampliamente en la industria moderna. En determinadas ocasiones servosistema se denomina también a un sistema de control cuya salida debe seguir con exactitud una trayectoria determinada en el espacio (como la posición de una aeronave en el espacio en un sistema de aterrizaje automático).

Sistemas de regulación automática

Es un sistema de control retroalimentado en el que la entrada de referencia o la salida deseada son, o bien constantes o bien varían lentamente en el tiempo, y donde la tarea fundamental consiste en mantener la salida en el valor deseado a pesar de las perturbaciones presentes. Hay muchos ejemplos de sistema de regulación automática, como el regulador centrífugo de Watt, la regulación automática de tensión de una planta generadora eléctrica ante variaciones de carga eléctrica, y los controles automáticos de presión y temperatura en un proceso químico.

Sistemas de control de procesos

A un sistema de regulación automático en el que la salida es una variable como temperatura, presión, flujo, etc., se le denomina sistema de control de proceso, este tiene amplia aplicación en la industria. En estos sistemas se usan controles programados.

Sistemas de control de lazo cerrado

Se les denomina así a los sistemas de control retroalimentado. En la práctica, se utiliza indistintamente la denominación control retroalimentado o control de lazo cerrado. La señal de error actuante, que es la diferencia entre la señal de entrada y la retroalimentación (que puede ser la señal de salida o una función de la señal de salida y sus derivadas), entra al controlador para reducir el error y llevar la salida del sistema a un valor deseado. El termino lazo cerrado implica siempre el uso de la acción de control retroalimentado para reducir el error del sistema.

Sistemas de control de lazo abierto

El control de lazo abierto, implica una predicción de cuánta acción es necesaria para llevar a cabo un proceso, es decir en un sistema de lazo abierto no se realiza ninguna comprobación durante el proceso para ver si es necesaria una acción correctiva para conseguir el resultado final. Son los que la salida no tienen efecto sobre la acción de control. En otras palabras, en un sistema de control de lazo abierto la salida ni se mide ni se retroalimenta para compararla con la entrada. En cualquier sistema de control de lazo abierto, no se compara la salida con la entrada de referencia. Por lo tanto, para cada entrada de referencia corresponde una operación fija. Así la precisión del sistema depende de la calibración. En presencia de perturbaciones, un sistema de control de lazo abierto no cumple su función asignada. Por ejemplo, el control de tráfico con señales accionadas en función del tiempo. Un ejemplo práctico lo constituye una lavadora de ropa doméstica. El remojo, lavado y enjuague en la lavadora se cumplen por tiempos. La máquina no mide la señal de salida, es decir, la limpieza de la ropa.

Sistemas de control de lazo cerrado versus de lazo abierto

Una ventaja del sistema de control de lazo cerrado es que el uso de la retroalimentación hace que la respuesta del sistema sea relativamente insensible a perturbaciones externas y a variaciones internas de parámetros del sistema. De este modo, es posible utilizar componentes relativamente imprecisos y económicos, y lograr la exactitud de control requerida en determinada planta, cosa que sería imposible en un control de lazo abierto.

Desde el punto de vista de la estabilidad, en el sistema de control de lazo abierto, esta es más fácil de lograr, ya que en él la estabilidad no constituye un problema importante. En cambio, en los sistemas de lazo cerrado, la estabilidad sí es un problema importante, por su tendencia a sobrecorregir errores que pueden producir oscilaciones de amplitud constante o variable.

Sistemas de Control Adaptable

Las características dinámicas de la mayoría de sistemas de control no son constantes por diversas razones, como el deterioro de los componentes al paso del tiempo o las modificaciones en los parámetros o en el medio ambiente. Aunque en un sistema de control retroalimentado, se atenúan los efectos de pequeños cambios en las características dinámicas, si las modificaciones en los parámetros del sistema y en el medio son significativas, un sistema, para ser satisfactorio ha de tener capacidad de adaptación. Adaptación implica la capacidad de autoajustarse o automodificarse de acuerdo con las modificaciones imprevisibles del medio o estructura. Los sistemas de control que tienen algún grado de capacidad de adaptación se denominan sistemas de control adaptable.

En un sistema de control adaptable, las características dinámicas deben estar identificadas en todo momento, de manera que los parámetros de control puedan ajustarse para mantener un comportamiento óptimo, de este modo un sistema de control adaptable es un sistema no estacionario.

Sistema de Control de Aprendizaje

Muchos sistemas de control que aparentemente son de lazo abierto, pueden convertirse en sistemas de lazo cerrado si un operador humano se considera como un controlador, que compara la entrada y la salida y realiza las acciones correctivas basadas en la diferencia resultante o error.

1.7.2. CLASIFICACION DE SISTEMAS DE CONTROL

Los sistemas de control pueden clasificarse de diversos modos:

- Sistemas de Control Lineales versus no lineales
- Sistemas de Control Invariante en el Tiempo versus Control Variable en el Tiempo
- Sistemas de Control de Tiempo Continuo versus de Tiempo Discreto
- Sistemas de Control con una Entrada y una Salida versus con Múltiples Entradas y Múltiples Salidas.
- Sistemas de Control con Parámetros Concentrados versus con Parámetros Distribuidos.
- Sistemas de Control Determinísticos versus Estocásticos.

Sistemas de Control Lineales versus No Lineales

La mayoría de los sistemas físicos no son lineales en varios sentidos, sin embargo, si la extensión de variaciones de la variable de sistema no es amplia, el sistema puede linealizarse dentro de un rango relativamente estrecho de valores de las variables.

Para sistemas lineales se aplica el principio de superposición. Aquellos sistemas a los que no es aplicable este principio son los sistemas no lineales.

Sistemas de Control Invariante en el Tiempo versus Control Variable en el Tiempo

Un sistema de control invariante en el tiempo (Sistema de Control con coeficientes constantes) es aquel en que los parámetros no varían en el tiempo. La respuesta de tal sistema es independiente del tiempo en el que se aplica la entrada.

En cambio, un sistema de control variable en el tiempo es aquel en el cual los parámetros varían con el tiempo; su respuesta depende del tiempo en el que se aplica una entrada.

Sistemas de Control de Tiempo Continuo versus de Tiempo Discreto

En un sistema de control de tiempo continuo, todas las variables son función de un tiempo continuo t. Un sistema de control de tiempo discreto abarca una o más variables que son conocidas solo en instantes discretos de tiempo.

Sistemas de Control con una Entrada y una Salida versus con Múltiples Entradas y Múltiples Salidas.

Un sistema puede tener una entrada y una salida por ejemplo un sistema de control de posición, donde hay un comando de entrada y una salida controlada. Se designa a un sistema así como un sistema de control con una entrada y una salida. Algunos sistemas pueden tener múltiples entradas y múltiples salidas por ejemplo un sistema de control de proceso con dos entradas (presión y temperatura) y dos salidas (presión de salida y temperatura de salida).

Sistemas de Control con Parámetros Concentrados versus con Parámetros Distribuidos.

Los sistemas de control que pueden describirse mediante ecuaciones diferenciales ordinarias son sistemas de control con parámetros concentrados, mientras que los sistemas de control con parámetros distribuidos son aquellos que pueden describirse mediante ecuaciones diferenciales parciales.

Sistemas de Control Determinísticos versus Estocásticos.

Un sistema de control es determinístico si la respuesta a la entrada es predecible y repetible. De no serlo, el sistema de control es estocástico.

1.7.3. EJEMPLOS DE SISTEMAS DE CONTROL

A continuación se detallan varios ejemplos de sistemas de control de lazo cerrado.

Sistema de Control de Velocidad

En el diagrama esquemático de la figura 1-6 aparece el principio básico del regulador de Watt para una máquina. De acuerdo con la diferencia entre la velocidad deseada y la real, se ajusta la cantidad de combustible que ingresa al motor. La secuencia de pasos se puede describir de la siguiente forma: la velocidad del controlador se ajusta de modo que, a la velocidad deseada, no fluya aceite a presión por ninguno de ambos accesos al cilindro de potencia. Si la velocidad efectiva cae por debajo del valor deseado, debido a alguna perturbación, la disminución de fuerza centrífuga de la velocidad del regulador hace que la válvula de control se desplace hacia arriba. Esto disminuye la provisión de combustible, y la velocidad de la máquina se reduce hasta alcanzar la velocidad deseada.

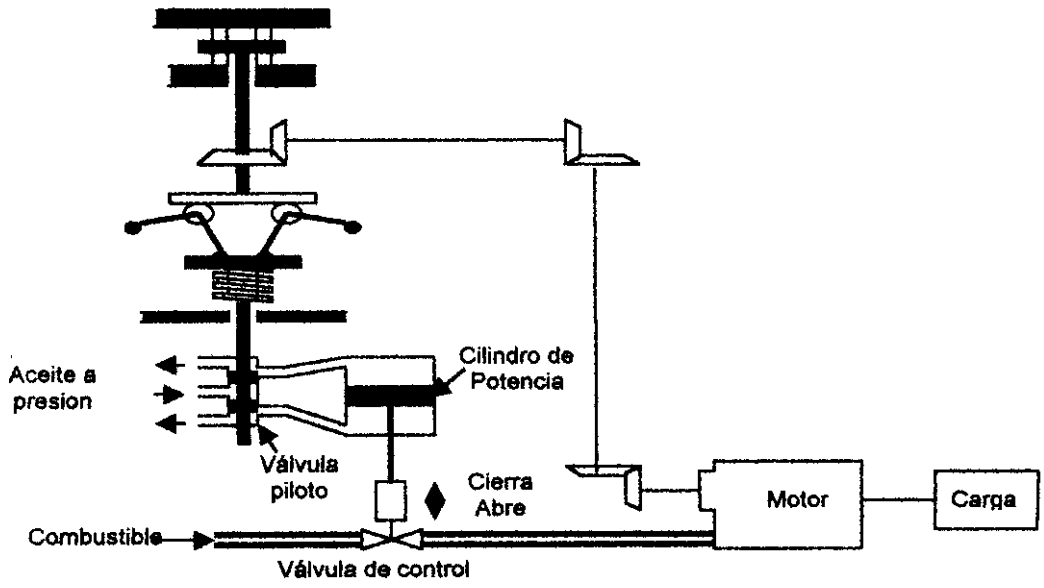


Figura 1-6
Sistema de control de velocidad

Sistemas de control de robot.

En la industria se utilizan frecuentemente robots industriales para mejorar la productividad. El robot puede realizar tareas monótonas y complejas sin errores en su operación. El robot trabaja en un ambiente intolerable para operadores humanos. Por ejemplo, puede funcionar a temperaturas extremas (tanto altas como bajas) o en un medio de alta o baja presión, bajo el agua o en el espacio. Hay robots especiales para combatir el fuego, para exploración en inmersión o para desplazamiento en el espacio, entre otros.

El robot industrial debe manejar partes mecánicas que tienen formas y pesos particulares. Por tanto, deben poseer al menos un brazo, una articulación y una mano. Deben tener suficiente potencia para realizar la tarea y la capacidad al menos para una movilidad mínima. De hecho, los robots modernos pueden desplazarse libremente dentro del espacio limitado de una fábrica.

El robot industrial debe poseer algunos sensores. En robots de bajo nivel, se instalan interruptores del tipo denominado microswitch a modo de elementos sensores en los brazos. El robot tomo contacto primeramente con un objeto, y luego, a través de sus microinterruptores, confirma la existencia de un objeto en el espacio y procede al siguiente paso de tomarlo o asirlo.

En un robot de alto nivel se utiliza medios ópticos (como un sistema de televisión) para explorar el ambiente que rodea a un objeto. Reconoce las imágenes y determina la presencia y orientación del objeto. Se requiere un computador para el procesamiento de señales en el reconocimiento de imágenes.

En algunas aplicaciones el robot computarizado reconoce la presencia y orientación de cada parte mecánica mediante el proceso de reconocimiento de imágenes, que consiste en leer los códigos numéricos asociados con las imágenes. El robot recoge la pieza y la desplaza hacia el lugar de montaje, donde ensambla las diversas partes para formar un componente. La función del controlador la realiza un computador digital programado.

Sistema de control de brazo de robot.

El movimiento en línea recta es un movimiento en un solo grado de libertad. El brazo de un robot tiene en realidad tres grados de libertad (movimiento hacia arriba-abajo, movimiento hacia adelante-atrás y movimiento hacia izquierda-derecha). La articulación o muñeca en el extremo del brazo, tiene también tres grados de libertad, y la mano uno, que es el agarre o aprehensión (movimiento de asir). En total, el sistema de brazo de robot tiene siete grados de libertad. Si el cuerpo del robot debe moverse en un plano se añaden grados de libertad. En general, las manos de robot pueden tener partes intercambiables; se le pueden colocar distintos tipos de dispositivos de agarre a la articulación, para servir como mano en la aprehensión de distintos tipos de objetos mecánicos.

Para posicionar la articulación y la mano se utiliza un servosistema. Como el movimiento de un brazo de robot requiere frecuentemente velocidad y potencia, como fuente de potencia se usa presión hidráulica o neumática. Para necesidades de potencia medianas, se pueden utilizar motores de red, y para casos en que se requiere bajo potencia, se puede recurrir a motores de pasos.

Para el control de movimiento secuenciales, se almacenan las señales de comando en discos magnéticos. En sistemas robóticos de alto nivel, a menudo se utiliza el control por repetición. En esta modalidad, primeramente un operador humano "enseña" al robot la secuencia de movimiento deseada, actuando sobre algún mecanismo asociado al brazo; el computador en el robot memoriza la secuencia de movimiento deseada. Entonces el robot reproduce fielmente la secuencia de movimientos.

Sistema de control de la fuerza agarre de la mano del robot.

Utiliza un dispositivo sensor de fuerza y otro de deslizamiento. Si la fuerza de aprehensión es demasiado pequeña, la mano del robot dejará caer el objeto mecánico, y si es demasiado grande, lo puede dañar o aplastar. La mano recoge y levanta el objeto con la fuerza de agarre preajustada. Si hay algún deslizamiento durante el ascenso, será detectado por el dispositivo sensor de deslizamiento, él que enviará una señal de retorno al controlador, el cual a su vez aumentará la fuerza de agarre. De este modo, se puede lograr una fuerza razonable que evite el deslizamiento, pero que no produzca ningún daño al objeto.

Sistema de control numérico.

El control numérico es un método de control del movimiento de los componentes de máquinas utilizando números. El control numérico puede controlar el movimiento de una cabeza cortante por medio de información binaria contenida en un disco. Para poner en marcha el sistema, se alimenta el disco a un lector. La señal de pulso de entrada modulada en frecuencia, se compara con la señal del pulso de retroalimentación. El controlador realiza las operaciones matemáticas sobre la diferencia entre ambas señales de pulsos. El convertidor digital a analógico convierte el

pulso de salida del controlador en una señal analógica que representa cierta magnitud de voltaje, la que, a su vez, hace rotar al servomotor. La posición de la cabeza es controlada de acuerdo a la entrada al servomotor. El transductor acoplado a la cabeza cortante convierte el movimiento en una señal eléctrica, que a su vez es convertida a una señal de pulso por el convertidor analógico a digital; luego esta señal se compara con la señal de pulso de entrada. Si hay alguna diferencia entre ambas, el controlador envía una señal servomotor para reducir esa diferencia, como se indicó anteriormente. El control numérico tiene la ventaja de que permite producir piezas complejas con tolerancias uniformes, a la máxima velocidad de tallado.

Sistema de control de temperatura.

La temperatura en el interior del horno se mide con un termómetro, que es un dispositivo analógico. La temperatura se convierte a un valor de temperatura digital, por un convertidor A/D y con está se alimenta un controlador a través de una interfaz. La temperatura digital se compara con la temperatura de entrada programada, y ante cualquier discrepancia (error), el controlador envía una señal al calefactor, a través de un amplificador y relevador, para llevar la temperatura del horno al valor deseado.

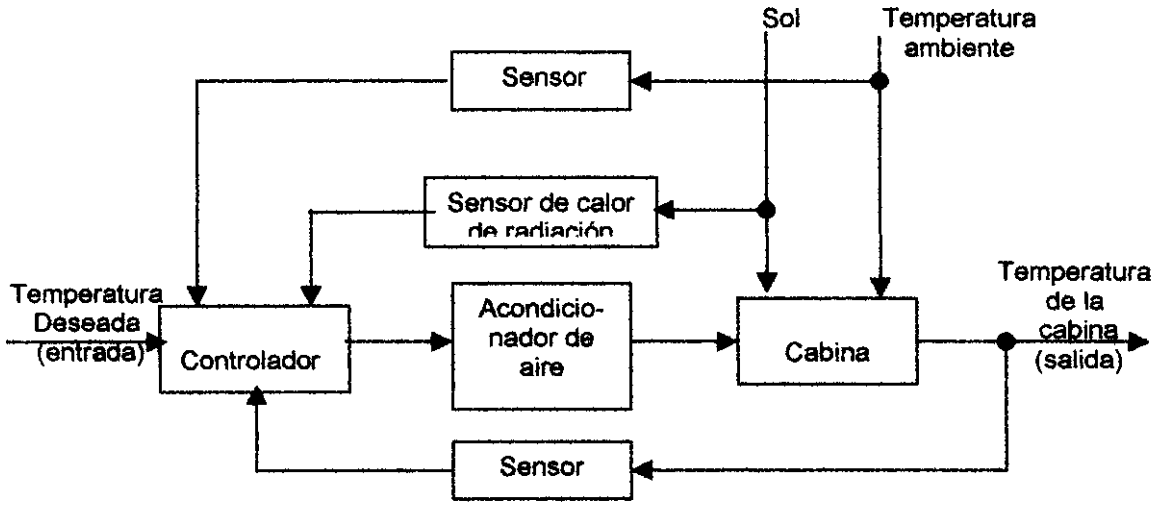


Figura 1-7
Control de la temperatura de la cabina de un automóvil

La figura 1-7 muestra un diagrama funcional del control de temperatura de una cabina de un coche. La entrada al controlador es la temperatura deseada, convertida a un voltaje. La temperatura efectiva de la cabina se convierte a un voltaje por medio de un sensor, y se le retroalimenta al controlador para comparación con la entrada. La temperatura ambiente y el calor del sol transferido por radiación actúan como perturbaciones, debido a que no son constantes durante la marcha del vehículo. Este sistema utiliza tanto control retroalimentado como control de prealimentado. (El control brinda acción correctiva antes que las perturbaciones afecten la salida).

La temperatura en la cabina del vehículo varía considerablemente, según el lugar en que se mida. En vez de instalar múltiples sensores para medir la temperatura, y obtener un promedio de los mismos, es más económico colocar un ventilador de succión en el sitio donde normalmente los pasajeros sienten la temperatura. Entonces la temperatura del aire del exterior brinda una indicación de la temperatura de la cabina y se le considera como salida del sistema.

El controlador recibe la señal de entrada, la señal de salida y las señales de sensores desde las fuentes de perturbación. A su vez, el controlador envía óptima de control al equipo acondicionador de aire para controlar la cantidad de aire refrigerado, de modo que la temperatura de la cabina sea igual a la temperatura deseada.

Sistema de control de tráfico.

El control de tráfico por medio de señales activadas sobre una base de tiempos, constituye un sistema de control de lazo abierto.

Sin embargo, si la cantidad de automotores esperando en cada señal de tráfico en un área congestionada se mide continuamente, y esa información se lleva a una computadora central que controla tales señales, el sistema se convierte en sistema de lazo cerrado.

El movimiento de tráfico en redes es muy complejo porque la variación en el volumen de tráfico depende mucho de la hora y el día de la semana, así como de muchos otros factores. En algunos casos se puede suponer una distribución de Poisson de llegadas a las intersecciones, pero esto no es necesariamente válido para todos los problemas de tráfico. De hecho, minimizar el tiempo medio de espera es un problema de control muy complejo.

Sistemas biológicos.

Sea el caso de dos especies de bacterias competidoras, cuyas respectivas poblaciones son x_1 y x_2. Son competidoras en el sentido que consumen alimentos de la misma fuente. Bajo condiciones, las poblaciones x_1 y x_2 varían en el tiempo de acuerdo con:

$$x_1 = a_{11}x_1 - a_{12}x_1x_2$$

$$x_2 = a_{21}x_2 - a_{22}x_1x_2$$

donde a_{11}, a_{12}, a_{21}, a_{22} son constantes positivas y x_1 y x_2 son no negativas.

Estas ecuaciones se denominan ecuaciones de competencia de Volterra.

Si se suministra cierto componente químico a las especies, las poblaciones varían de acuerdo con las siguientes expresiones:

$$x_1 = a_{11}x_1 - a_{12}x_1x_2 - b_1u$$

$$x_2 = a_{21}x_2 - a_{22}x_1x_2 - b_2u$$

donde b_1 y b_2 son constantes positivas y u es la entrada de control (en este ejemplo, la cantidad de componentes químico). Un problema interesante se produce cuando se requiere minimizar la población x_2 en el máximo posible. Este es un ejemplo de sistema biológico al que se puede aplicar la teoría de control.

Sistema de control de inventario.

Otro sistema de control cerrado, lo constituye la programación industrial del ritmo de producción y nivel de inventario. El nivel del inventario real, que es la salida del sistema, se compara con el nivel de inventario deseado, que puede variar ocasionalmente según el mercado. Si aparece cualquier diferencia entre el nivel de inventario real y el deseado, el ritmo de producción se ajusta de manera que la salida siempre iguale o esté cercana al valor deseado, que se elige para maximizar las utilidades.

Sistemas empresariales.

Un sistema empresarial puede consistir de varios grupos. Cada tarea asignada a un grupo, representa un elemento dinámico del sistema. Para el funcionamiento correcto de tales sistemas, hay que establecer métodos retroalimentados para el control de los logros de cada grupo.

El acoplamiento mutuo entre grupos funcionales debe ser mínimo para reducir atrasos inútiles en el sistema. Cuanto menor sea ese acoplamiento, más suave será el flujo de señales y de materiales.

El sistema empresarial constituye un elemento de sistema de lazo cerrado. Un buen diseño de trabajo de dirección futuro. Nótese que en este sistema las perturbaciones son la falta de personal, la interrupción de las comunicaciones, los errores humanos, etc.

Para una buena dirección es obligatorio establecer un sistema de estimación bien fundamentado, para los datos estadísticos. (Es bien conocido el hecho de que el rendimiento de un sistema como éste se puede mejorar, utilizando el tiempo de adelanto p "anticipación").

Para aplicar la teoría de control de mejorar el rendimiento de esos sistemas, hay que representar las características dinámicas de los grupos de componentes en el sistema, mediante un conjunto de ecuaciones relativamente simple.

Aunque es difícil deducir los modelos matemáticos para los diversos grupos, la aplicación de técnicas de optimización a sistemas empresariales mejora significativamente su rendimiento.

CAPITULO II

AUTOMATAS PROGRAMABLES INDUSTRIALES

2.1. GENERALIDADES

Los Autómatas Programables Industriales llamados también PLCs (Controladores Lógicos Programables) aparecieron en los Estados Unidos en el año de 1969, respondiendo a los deseos de los industriales del automóvil de desarrollar cadenas de automatización, que pudieran seguir la evolución de las técnicas y de los modelos fabricados. El PLC sustituyo así a los armarios de relés a causa de su flexibilidad (puesta en acción y evolución), pero también porque los automatismos de mando complejo cuyos costes de cableado y puesta a punto eran elevados.

El pliego de condiciones de estas máquinas comprendía también especificaciones de utilización en un medio industrial perturbado, sobre la simplicidad de su manejo por el personal y naturalmente sobre el coste del desarrollo de los automatismos. Se descartaba así las otras soluciones programadas tradicionales; miniordenador, etc.

De todas estas especificaciones nace la definición de lo que tiene que ser un autómata programable de tipo industrial:

Un autómata programable industrial es una máquina electrónica programada por personal no necesariamente informático y destinada a gobernar automatismos secuenciales y combinatorios en ambiente industrial y en tiempo real.

APORTACION

Como ventajas de los autómatas programables con respecto a la lógica cableada se podrían enunciar las siguientes:

- Posibilidad de simulación del proceso para la puesta en marcha del mismo sin necesidad de que sea a pie de máquina.

- Posibilidad de monitorización del proceso, lo que facilita la localización de averías y por lo tanto el mantenimiento del proceso.

- Flexibilidad a la hora de cambios en el proceso (Evolución del mismo, cambios en la producción etc.).

A su vez, con relación a otras soluciones programadas como miniordenadores, se podrían citar otras ventajas:

- Protección contra las agresiones del medio industrial

- Facilidad de manejo por utilizar lenguajes de alto nivel y de filosofía parecida al lenguaje de relés.

- Posibilidad de adaptar directamente los elementos del proceso a las entradas-salidas del autómata.

Todas estas aportaciones de los Autómatas programables, hacen que sean muy utilizados en la mayoría de los sectores industriales, entre los que se podrían citar:

- Metalurgia y Siderurgia
- Mecánica y Automotriz
- Industrias químicas y petrolíferas
- Industrias agrícolas y alimentarias
- Transportes y manutención
- Etc.

2.2. TECNOLOGIAS DE AUTÓMATAS

Para la realización del órgano de mando (parte comando) de un automatismo se dispone de numerosas herramientas que se pueden clasificar en dos grupos diferentes:

- Soluciones Cableadas
- Soluciones Programadas

A continuación se detalla una Tabla sobre las diferentes opciones tecnológicas generales:

TIPO	FAMILIA TECNOLÓGICA	SUBFAMILIAS ESPECIFICAS	
Lógica Cableada	Eléctrica	Relés electromagnéticos	
		Electroneumática	
		Electrohidráulica	
	Electrónica	Electrónica estática	
Lógica Programada	Electrónica	Sistemas Informáticos	Microordenadores Miniordenadores
		Microsistemas(universales específicos)	
		Autómatas Programables	

Tabla 1
Tecnologías de Autómatas

En la lógica cableada, se dispone de unas uniones materiales(cableado) que forman el mando, pudiendo tratarse de un circuito neumático, hidráulico, eléctrico o electrónico. En estos sistemas, a medida que el automatismo se complica, comienza a presentar los siguientes inconvenientes:

- Ocupan mayor espacio
- Sistemas poco flexibles para la evolución de los mandos (mejoras, nuevas funciones, etc.)
- Dificultad de dominar problemas complejos
- Complejidad en la búsqueda de averías y en las reparaciones
- Coste elevado para sistemas complejos en comparación con las soluciones programadas.

La lógica programada aparece para contrarrestar todos estos inconvenientes y en ella el circuito cableado es sustituido por un programa informático (SOFTWARE) contenido en una máquina que puede ser un autómata programable, un microcontrolador o incluso un miniordenador (figura 2-1).

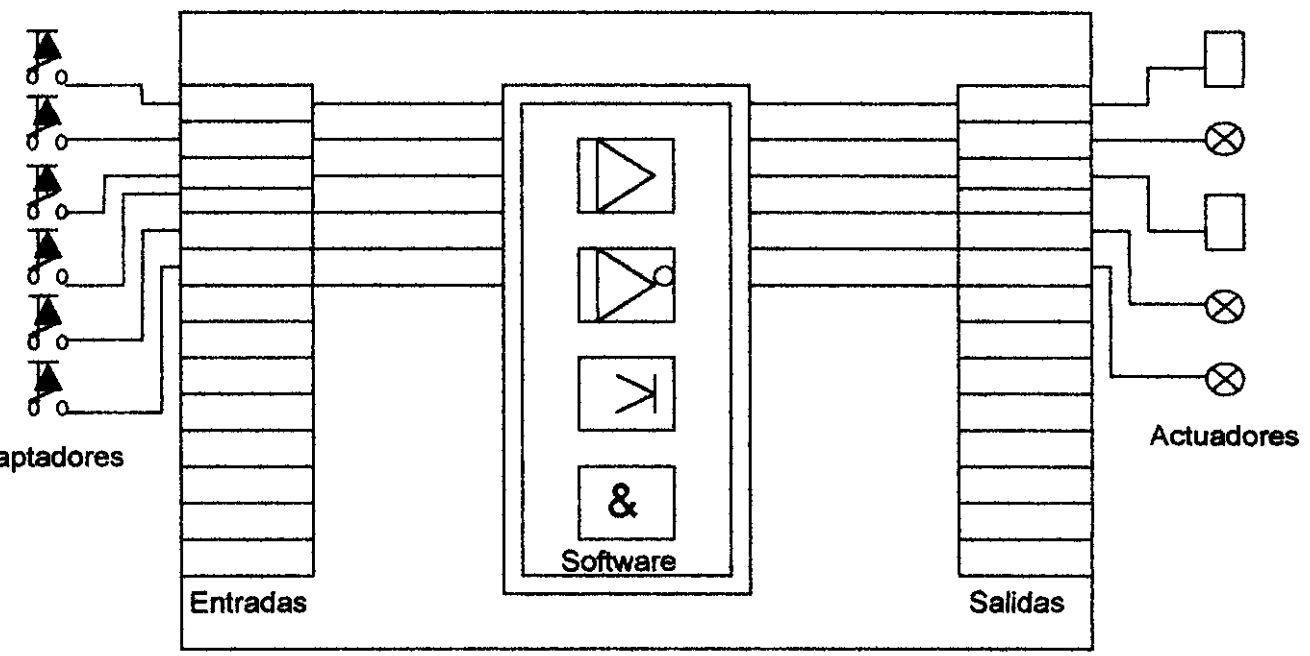


Figura 2-1
Lógica Programada

De todas estas herramientas, la más utilizada y adecuada para el mando de los automatismos es el Autómata Programable Industrial (PLC), una máquina especialmente diseñada para estos trabajos.

Debido al hecho de que la unidad lógica de un sistema programado no comporta más que una muestra de cada función de base, la lógica programable reemplaza el tratamiento instantáneo global de un automatismo cableado por un tratamiento escalonado en el tiempo y selectivo.

Es decir que las funciones se realizan las unas después de las otras. Esto puede parecer un método muy lento, pero se debe recordar que la velocidad de tratamiento es tal que permite realizar miles de instrucciones en unos pocos milisegundos.

2.3. CATEGORIAS DE LOS AUTOMATAS

La clasificación de los automatismos se hace en dos grandes familias, según los tipos de entradas salidas:

- Automatismos con entradas salidas analógicas
- Automatismos con entradas salidas digitales

Los automatismos con entradas salidas analógicas son aquellos cuyas entradas-salidas pueden tomar un número infinito de valores entre dos límites.

Así por ejemplo, un motor eléctrico alimentado con una tensión variable que puede tomar un número infinito de velocidades dentro del rango de sus valores nominales.

Los automatismos digitales por el contrario son aquellos cuyas entradas salidas pueden tomar un número finito de valores. Por ejemplo un display alfanumérico, que puede tomar distintos valores correspondientes a otros tantos caracteres, pero que forman un conjunto finito de elementos.

Un caso particular de los sistemas digitales son aquellos que únicamente pueden tomar dos valores, los llamados sistemas binarios o todo-nada. En este caso las entradas-salidas solo pueden tomar el valor activado o no activado:

- Una bombilla(encendida o apagada)
- Un interruptor(activado o desactivado)

Un gran porcentaje de los sistemas controlados por autómatas programables son del tipo binario y dentro de éstos se puede hacer otra división en dos grandes grupos:

- Sistemas combinatorios
- Sistemas secuenciales

Un automatismo combinatorio es un sistema en el cual a una combinación de entradas corresponde una combinación y una sola de salidas.

Por lo tanto el conocimiento del estado de las entradas es suficiente para determinar el estado que han de tener las salidas (Figura 2-2).

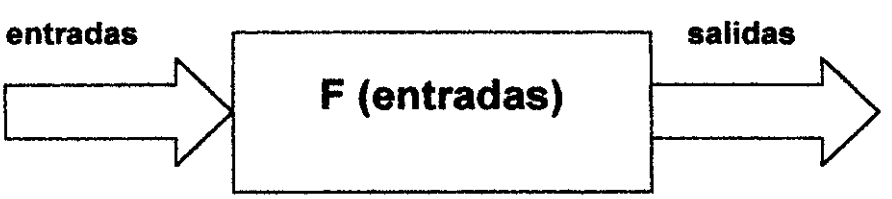


Figura 2-2

Un caso claro de un sistema combinatorio, es una lámpara gobernada por un interruptor.

ENTRADA	SALIDA
Interruptor activado	Lámpara encendida
Interruptor desactivado	Lámpara apagada

Un automatismo, en cambio es secuencial cuando a una determinada combinación de variables de entrada, pueden corresponder diferentes combinaciones de variables de salida.

Es decir, para conocer el estado de las salidas, no es suficiente con conocer el estado de las entradas, es necesario saber también la situación anterior del sistema.

Un ejemplo típico de un sistema secuencial, sería una lámpara controlada por un pulsador; la misma acción de activar el pulsador puede suponer el encendido o el apagado de la lámpara, dependiendo del estado en el que se encuentra esta ultima antes de accionar el pulsador.

El hecho de tener que conocer el estado anterior del sistema, introduce en los automatismos secuenciales la noción de memoria por medio de la realimentación como se indica en la siguiente figura (figura 2-3).

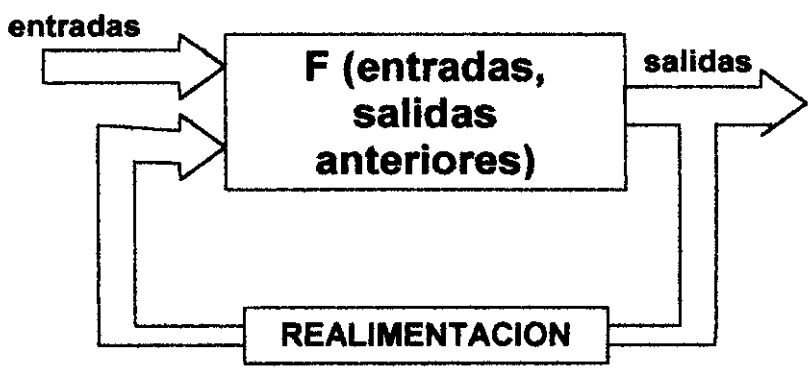


Figura 2-3

2.4. ESTRUCTURA Y FUNCIONAMIENTO

Un autómata programable se encuentra constituido generalmente en dos partes bien diferenciadas: el material llamado Hardware y la lógica programada denominada Software.

2.4.1. HARDWARE

Un autómata programable es un conjunto electrónico compuesto por:

- Las entradas, donde se conectan los captadores
- Las salidas, donde se conectan los actuadores
- La memoria que sirve de soporte al programa(lista de instrucciones a ejecutar)
- La CPU(Unidad Central de Proceso) que es el cerebro del autómata
- La alimentación y circuitos anexos diversos
- Los equipos o unidades de programación

El detalle de todos estos componentes se ilustra en la figura 2-4

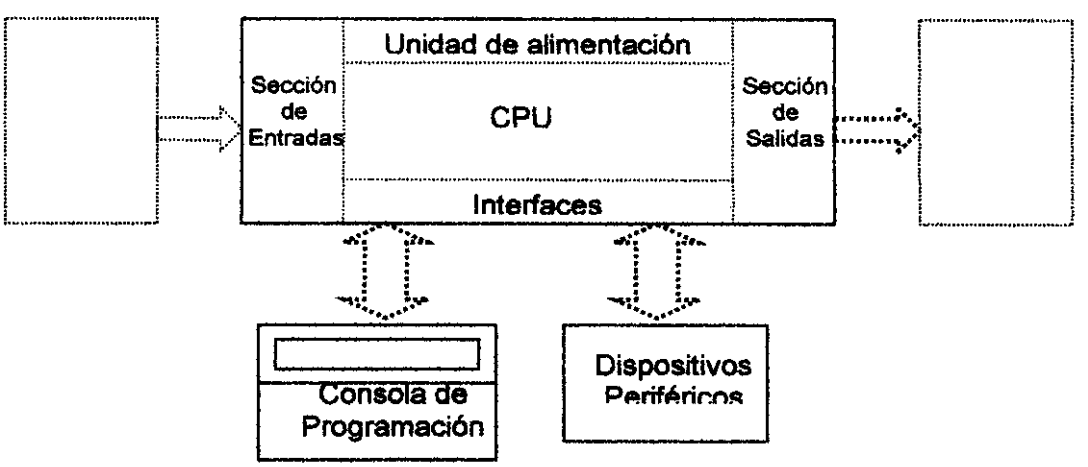


Figura 2-4
Autómata Programable con sus periféricos y unidad de alimentación

2.4.1.1. Entradas-Salidas

Los módulos de entradas-salidas son los módulos de conexión del autómata con el proceso a controlar a través del modulo de entradas, él modulo recibe información del proceso y por otra parte, a través del modulo de salidas manda información a este.

Ambos módulos se han de encargar además de captar la información, de adaptarla a los niveles utilizados por autómata (Normalmente señales en forma de tensión DC de 15 o 12 V). Además deben de tener los acopladores necesarios (optoacopladores) para proteger la circuitería interna de posibles incidentes eléctricos en el proceso a controlar.

Los primeros autómatas, debido a su función de controlar automatismos lógicos, disponían únicamente de entradas-salidas digitales. Hoy en día las nuevas funciones asignadas a los autómatas, hacen que aparezcan nuevos módulos de Entradas/Salidas como: E/S analógicas, módulos de controladores PID, contadores rápidos, módulos ASCII.

2.4.1.2. Memoria

Llamamos memoria a cualquier dispositivo que nos permite almacenar información en forma de bits (ceros y unos)

Tipos de Memorias

No todas las memorias son iguales; se distinguen dos tipos fundamentales:

A. Memoria RAM (Random Access Memory), memoria de acceso aleatorio o memoria de lectura y escritura. En este tipo de memorias se pueden realizar los procesos de lectura y escritura por procedimiento eléctrico, pero su información desaparece al faltarle la corriente.

B. Memoria ROM (Read Only Memory), o memoria de solo lectura. En estas memorias se puede leer su contenido, pero no se puede escribir en ellas; los datos e instrucciones los graba el fabricante y el usuario no puede alterar su contenido. Aquí la información se mantiene ante la falta de corriente.

Pero estas no son todas las memorias disponibles; la siguiente tabla muestra otros tipos en las que los sistemas de programación, de borrado, volatilidad o permanencia marcan sus diferencias:

TIPO DE MEMORIA	SISTEMA DE PROGRAMACION	SISTEMA DE BORRADO	ANTE EL CORTE DE TENSION LA MEMORIA
RAM O memoria de lectura-escritura	Eléctrica	Eléctrico	Se pierde Es volátil
ROM O memoria de solo lectura	Durante Su proceso De fabricación	Es imposible Su Borrado	Se mantiene
PROM O memoria programable	Eléctrica	Es imposible Su Borrado	Se mantiene
EPROM O memoria modificable	Eléctrica	Por rayos UV	Se mantiene
EEPROM O memoria modificable	Eléctrica	Eléctrico	Se mantiene

Tabla 2.
Tipos de Memorias semiconductoras

Utilización de las memorias

Dependiendo de la función asignada, un autómata programable, utilizará un tipo de memoria u otro:

A. Memoria de Usuario: El programa de usuario, normalmente se graba en memoria RAM ya que no solo ha de ser leído por el microprocesador, sino que ha de poder ser variado cuando el usuario lo desee, utilizando la unidad de programación. La desconexión de la alimentación o el fallo de la misma borraría esta memoria, ya que al ser la RAM una memoria volátil necesita estar constantemente alimentada y es por ello que los autómatas que la utilizan, llevan incorporadas una batería tampón que impide su borrado.

B. Memoria de Tabla de Datos: La memoria de esta área también es del tipo RAM y en ella se encuentra por un lado la imagen de los estados de las entradas-salidas y por otro los datos numéricos y variables internas como contadores, temporizadores, marcas, etc.

C. Memoria y Programa del Sistema: Esta memoria se encuentra dividida en 2 áreas: la llamada memoria del sistema que utiliza memoria RAM y la que corresponde al programa del sistema, que lógicamente es un programa fijo, grabado por el fabricante y por lo tanto el tipo de memoria utilizado es ROM.

D. Memoria EPROM y EEPROM: Independientemente de otro tipo de aplicaciones, este tipo de memorias se utiliza como memorias copia para grabación y archivo de programas de usuario, función que en algunos autómatas es realizada también mediante discos magnéticos (diskettes).

En la siguiente tabla se representan todas las memorias presentes en un autómata programable con indicación de las funciones asignadas a cada una de ellas.

Programas ejecutivos Firmware del sistema (ROM o EPROM)
Memorias Temporales (RAM o EPROM)
Memoria imagen O tabla de estados De E/S (RAM)
Memoria de datos Numéricos y variables Internas (RAM)
Memoria del programa De usuario (RAM)

Tabla 3.
Memorias de un Autómata Programable

2.4.1.3. Unidad Central de Procesamiento

La Unidad Central de Procesamiento(CPU) se halla conformada por tres elementos fundamentales para su funcionamiento, estos son el microprocesador, memoria y circuitos auxiliares asociados.

Esta unidad es la encargada de realizar el control de la máquina y ejecutar los mandatos de las instrucciones del programa.

Es un circuito integrado a gran escala de integración que realiza un gran volúmen de operaciones las mismas que las podemos agrupar en:

- Operaciones de tipo lógico
- Operaciones de tipo aritmético
- Operaciones de control de la transferencia de información dentro del autómata.

Los circuitos internos del microprocesador son de tres tipos:

A. Circuitos de la Unidad Aritmético Lógica (ALU).

Es la parte del microprocesador donde se efectúan todos las operaciones aritméticas y las decisiones lógicas para controlar el autómata.

B. Circuitos de la Unidad de Control(UC).

Organiza todas las tareas del microprocesador. Así por ejemplo, cuando una instrucción del programa codificada en un código máquina (ceros y unos) llega al microprocesador, la Unidad de Control sabe, mediante una pequeña memoria ROM incluida, cual es la secuencia de señales que tiene que emitir para que se ejecute la instrucción.

C. Registros.

Los registros del microprocesador son memorias en las que se almacenan temporalmente datos, instrucciones o direcciones mientras necesitan ser utilizados por el microprocesador. Los registros más importantes de un microprocesador son los de instrucciones, direcciones, acumulador, contador de programa y el de bandera o de estado.

Contador de Programa

Denominado también puntero o pointer, contiene permanentemente la dirección de la instrucción en curso de ejecución. Describe entonces la zona de memoria donde esta almacenado el programa actualmente activo. Su evolución es automática: incremento de +1 al final de una instrucción, salvo en el caso de instrucción de salto en que se le impone la dirección de la nueva instrucción.

Registro de Instrucciones

Recibe de la memoria central el código de operación de la instrucción a ejecutar y realiza las validaciones correspondientes para dicha operación.

Registro de Direcciones

Recibe al mismo tiempo que el registro de instrucciones la dirección de operando de instrucción a ejecutar e indica el camino hacia esa dirección.

El Acumulador

Es el registro donde se realizan las operaciones de las instrucciones a ejecutar. Algunos autómatas disponen de dos acumuladores (A y B) para la realización de operaciones complejas, pero otros utilizan otra alternativa que es la de las pilas.

Las Pilas

Las pilas son un almacén ordenado de información con unas determinadas formas de adquisición y de evolución de información. Existen dos tipos de pilas, las pilas LIFO y las FIFO:

Pilas LIFO:

Al almacenar la información, las ya almacenadas se desplazan una posición hacia debajo de forma que la nueva información queda en primer lugar y es la primera en salir al realizar una llamada a la pila.

Pilas FIFO:

El almacenamiento es igual que las anteriores pero a la hora de recoger una información de la pila, se hace por la parte inferior del almacén, es decir, la primera almacenada es la primera en salir. Los autómatas utilizan actualmente el primero de los modelos de pila, la pila LIFO.

D. Buses.

No son circuitos en sí, sino zonas conductoras en paralelo que transmiten datos, direcciones, instrucciones y señales de control entre las diferentes partes del microprocesador. Se puede hacer la diferencia entre los buses internos y externos. Los primeros unen entre sí las diferentes partes del microprocesador, mientras que los segundos son pistas de circuito impreso que unen chips independientes.

Funciones de la Unidad Central de Procesamiento

En la memoria ROM del sistema, el fabricante ha grabado una serie de programas fijos (software del sistema) y es a éstos programas a los que accederá el microprocesador para realizar las funciones que correspondan en función del tiempo que trabajen. El software de sistema de cualquier autómata consta de una serie de funciones básicas que realizan en determinados tiempos de cada ciclo: en el inicio o conexión, durante el ciclo o ejecución del programa y a la desconexión.

Este software del sistema es ligeramente variable para cada autómata pero en general contiene las siguientes funciones:

- Supervisión y control del tiempo de ciclo, tabla de datos, alimentación, batería, etc.
- Autotest en la conexión y durante la ejecución del programa.
- Inicio del ciclo de exploración del programa y de la configuración del conjunto.
- Generación del ciclo base de tiempo.
- Comunicación con periféricos y la unidad de programación

Hasta que el software del sistema no haya ejecutado todas las acciones necesarias que le corresponden no se inicia el ciclo de programa de usuario.

2.4.1.4. Equipos y Unidades de Programación

La unidad de programación es el medio material del que se auxilia al programador para grabar o introducir en la memoria de usuario las instrucciones del programa.

Funciones Principales

La gama de funciones que son capaces de ejecutar los equipos de programación son múltiples y variados siendo las principales las que se describen a continuación:

- Programación
- Grabación de Programas
- Visualización y verificación dinámica del programa
- Cambio de los modos de servicio

Desde el punto de vista constructivo, podemos distinguir tres tipos principales:

A. Unidades de tipo Calculadora
B. Consolas de Programación
C. Unidad con PC

2.4.1.5. Periféricos

Como elementos auxiliares y físicamente independientes del autómata, los equipos periféricos realizan funciones concretas de gran importancia. El número de periféricos acoplables a los autómatas aumenta día a día pero en general los más comunes podrían ser:

- Impresora
- Unidades de memoria
- Lectores de memoria
- Display y teclados alfanuméricos

2.4.1.6. Tamaño de los Autómatas Programables

La clasificación de los autómatas programables en cuanto tiene relación a su tamaño se realiza en función del número de Entradas/Salidas: son admitidos los tres grupos siguientes:

1. **Gama Baja.** Hasta un máximo de 128 E/S. La memoria del usuario de que disponen suele alcanzar un valor máximo de 4K instrucciones.

2. **Gama Media.** De 128 a 512 E/S. La memoria de usuario de que disponen suele alcanzar un valor máximo de hasta 16 K instrucciones.

3. **Gama Alta.** Más de 512 E/S. Su memoria de usuario supera en algunos de ellos los 100 K instrucciones.

2.4.1.7. Ciclos de Trabajo de los Autómatas Programables

Los programas que un autómata debe de ejecutar están formados por instrucciones, las mismas que se van ejecutando en un orden determinado, excepto cuando se dan instrucciones de salto.

La ejecución es síncrona de forma que un reloj va ordenando la ejecución de las instrucciones unas detrás de otras y así al llegar a la última instrucción se comienza de nuevo por la primera.

Se trata entonces de una ejecución cíclica en la que el programa se lee y ejecuta una y otra vez.

Se llama período de un autómata (tiempo de SCAN) al tiempo que tarda en leer 1K (1024) instrucciones básicas (una instrucción puede estar compuesta de varias instrucciones básicas).

Cuando se utilizan instrucciones de salto, puede ocurrir que el programa ingrese en un ciclo cerrado, permaneciendo continuamente en una parte del programa. Para evitar esto, los autómatas incorporan lo que se llama el WATCHDOG, que paraliza la ejecución de un programa en caso de que transcurrido un determinado tiempo no se llegue al final del mismo.

La Unidad Central de Procesamiento (CPU) de un autómata debe compartir el tiempo de lectura y ejecución del programa con la adquisición de los datos de Entrada/Salida (E/S). Para agilitar este proceso existe una zona de memoria, espejo de los valores de E/S que es al cual accede el autómata cuando existe una instrucción de llamada a E/S.

Existen diferentes tipos de ciclos, en función de cómo se haga el refresco de los valores del espejo, con los nuevos valores de E/S, estos son:

A. Todas las entradas son adquiridas en el principio del ciclo y las salidas son controladas después de que todas las instrucciones hayan sido leídas y ejecutadas.

B. Las entradas son adquiridas como en el caso anterior, pero las salidas son controladas después de la resolución de un determinado número de instrucciones (ecuaciones lógicas).

C. Tanto la adquisición de entradas como el control de salidas se efectúan después de la resolución de un determinado numero de instrucciones (ecuaciones lógicas).

D. Las entradas son adquiridas cada n milisegundos, independientemente de la duración del ciclo, y las salidas son controladas cada vez que una instrucción del programa lo demanda

2.4.1.8. Funcionamiento de un Autómata Programable

Cada autómata tiene una estructura diferente. No obstante, se puede resumir un principio de funcionamiento común en todos los autómatas.

En la siguiente figura se representa el modo de funcionamiento de un autómata con los elementos anteriormente mencionados.

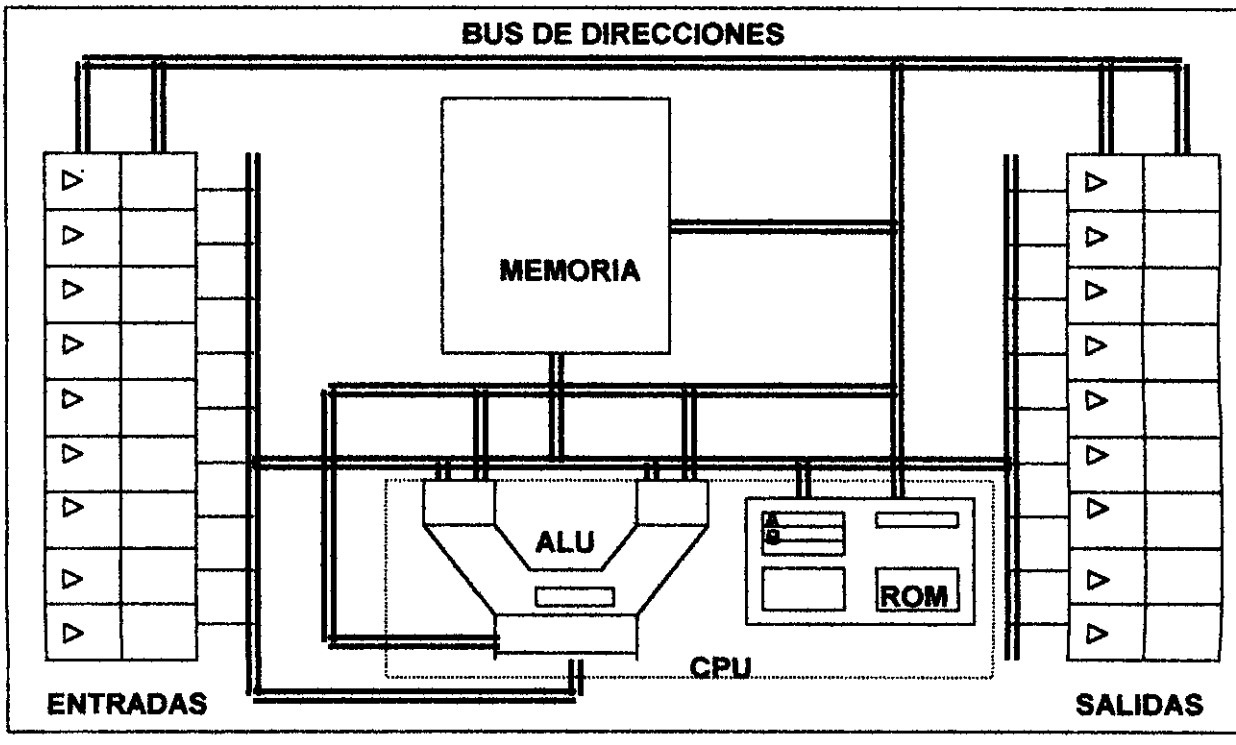


Figura 2-5

Funcionamiento de un Autómata Programable

Supongamos que vayamos a realizar con este autómata el control de un punto de luz que se puede encender desde dos puntos indistintamente. Para ello los dos interruptores de encendido se conectaran a la entrada e_1 y e_2 del autómata y de la misma forma la luz quedará conectada a la salida S_1.

A la hora de realizar el programa para el control de este ejemplo se ha de tener en cuenta que se han de realizar las siguientes operaciones, realizar la función "O" entre las entradas e_1 y e_2 y escribir el resultado de dicha operación en la salida S_1.

El algoritmo correspondiente a este proceso podría ser el siguiente:

Leer e_1
Leer e_2
Función "O"
Resultado a S_1
Fin.

Figura 2-6

El programa correspondiente a ese algoritmo estará almacenado en la memoria de programa del autómata (traducido a lenguaje propio de cada autómata), de forma que a cada instrucción le corresponderá una dirección de memoria.

Para la ejecución del programa la CPU irá asignado al controlador del programa la dirección correspondiente a la instrucción a ejecutar en cada instante, de forma que los códigos

correspondientes a dichas instrucciones se guardarán en los registros de instrucciones y de direcciones, para así ser ejecutadas por la ALU de forma que el resultado de dichas operaciones será asignado al registro acumulador por la CPU.

Todo el intercambio de información mencionado anteriormente circula por el bus de direcciones o el bus de datos indicados en la figura. De forma que las direcciones de los elementos a tratar y los códigos de las instrucciones a ejecutar circulan por el bus de direcciones, quedando el bus de datos reservado a la información del resultado de las operaciones y valor de entrada/salida y elementos internos.

2.4.1.9. Software. Lenguaje de los Autómatas Programables

El programa Software básico de un sistema informático tiene por objeto permitir la utilización del hardware según las especificaciones indicadas por el usuario por medio de un lenguaje de programación. Los lenguajes de los autómatas se pueden distinguir entre:

A. Lenguajes Literales

Lenguajes cuyas instrucciones se escriben en forma de expresiones literales utilizando partes textuales, palabras reservadas y símbolos:

Ej: IF, ELSE, /, +, *

A.1.- Lenguaje Booleano:

Permite transcribir directamente una ecuación booleana elemento tras elemento, tanto si son operandos como operadores.

(A * B + C)D = Y

El valor calculado del primer miembro es asignado al segundo.

A.2.- Lenguaje Nemónico:

Emplea el formulismo de los lenguajes de ensamblador. Dicho lenguaje es una transcripción elemental e inmediata de las instrucciones del lenguaje máquina. Este lenguaje pone al alcance del programador el conjunto de posibilidades de la máquina, pero su escritura es menos concisa y más pesada.

```
LD          3200
AND         0000
LD          3201
AND NOT     3202
OR LD
OUT         3200
```

B. Lenguajes Gráficos

Los lenguajes gráficos han sido los primeros que se han utilizado, en los autómatas programables industriales, puesto que el lenguaje de relés es el más antiguo de los lenguajes.

B.1.- Lenguaje de relés:

Es la transcripción directa del esquema eléctrico a relé. Es un lenguaje gráfico en el que se concatenan componentes lógicos y símbolos de asignación. Un programa se materializa en un esquema del tipo abajo representado.

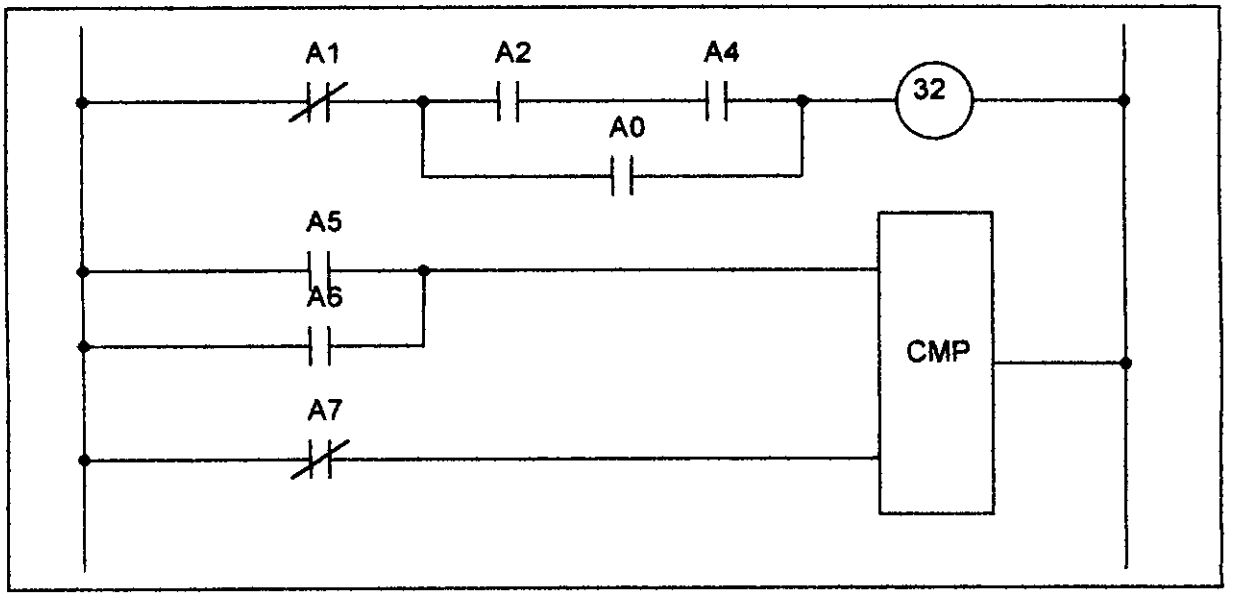


Figura 2-7

Ejemplo de lenguaje de relés

B.2.- Lenguaje GRAFCET:

El GRAFCET fue originalmente una metodología destinada a concebir y representar gráficamente el algoritmo, o sea, a efectuar su análisis funcional.

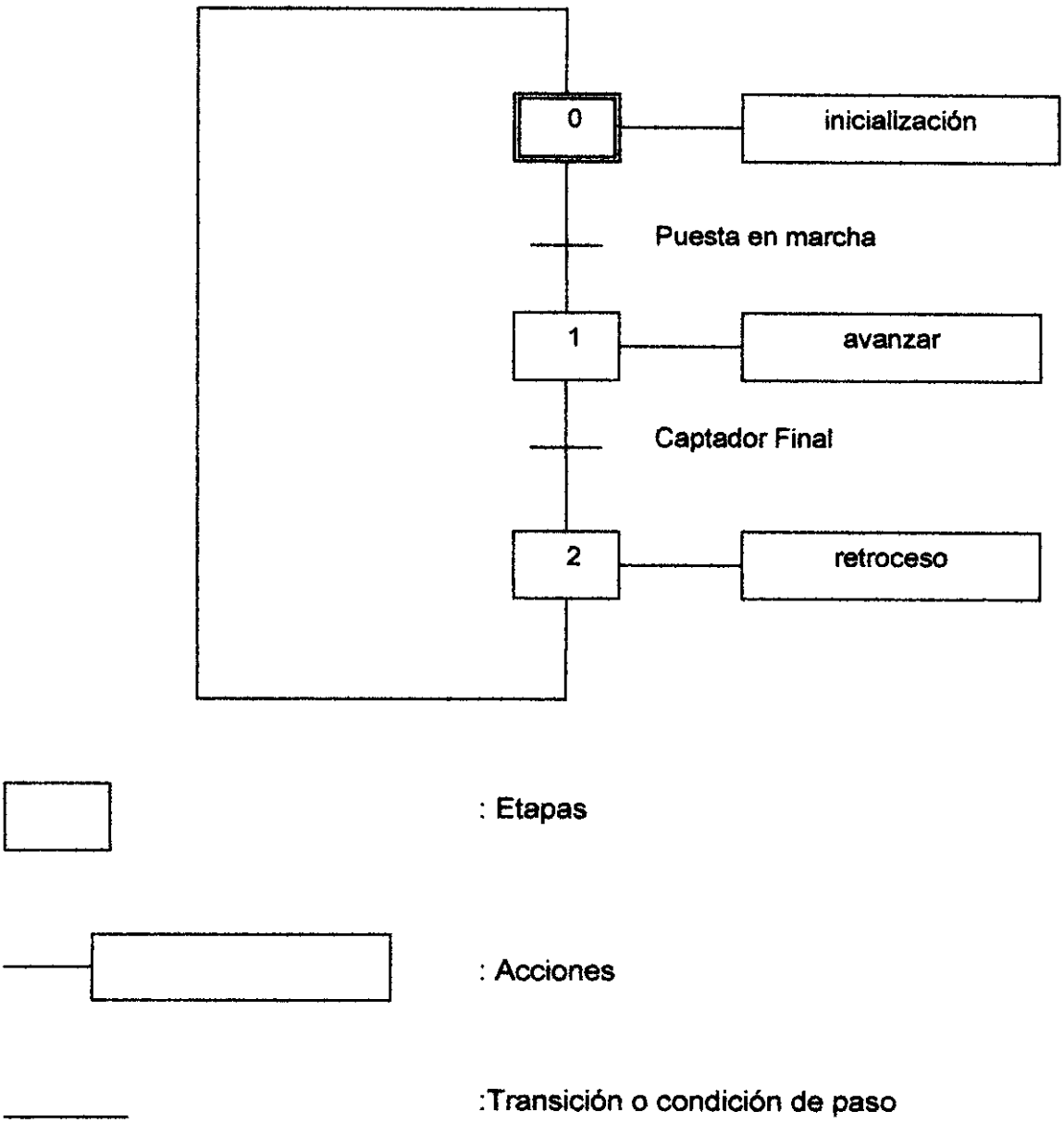


Figura 2-8

Ejemplo de Lenguaje Grafcet

2.4.1.10. CUADRO COMPARATIVO DE LAS PRINCIPALES MARCAS Y TECNOLOGÍAS DE AUTOMATAS PROGRAMABLES

FABRICANTE	MODELO	I/o total del sistema	I/o discretas máximo	I/o analógicas máximo	Diagrama lógico en escalera	Lenguaje de alto nivel	Capacidad PID	Documentación	Diagnostico	Tipo de Interfaz	Velocidad de Exploración	Tipo de memoria	País de Origen
TELEMECANIQUE	TSX 27	80	80		SI	SI		SI	SI	F	2ms	RAM EPROM	FRA
	MPC007	256	256	32	SI			SI	SI		32ms	RAM EPROM	JAP
	TSX 17	120	120	12	SI	SI	SI	SI	SI	J	10ms	RAM EPROM	FRA
	TSX 47 jr	80	80	22	SI	SI	SI	SI	SI	B	2ms	RAM EPROM	FRA
	TSX 47	256	256	44	SI	SI	SI	SI	SI	B	2ms	RAM EPROM	FRA
	TSX 47-30	256	256	54	SI	SI	SI	SI	SI	B	0.5ms	RAM EPROM	FRA
	TSX 67-30	512	512	64	SI	SI	SI	SI	SI	B	0.5ms	RAM EPROM	FRA
	TSX 87-10	1024	1024	128	SI	SI	SI	SI	SI	B	0.5ms	RAM EPROM	FRA
	TSX 87-30	2048	2048	256	SI	SI	SI	SI	SI	B	0.5ms	CMOS RAM EPROM	FRA
TEXAS INSTRUMENTS	5TI	512	512		SI			SI	SI	L	5.2ms	RAM EPROM	USA
INDUSTRIAL SYST.	510	40	40		SI			SI	SI	A,L		RAM EPROM	USA
	TI100	128	128		SI			SI	SI	L	5ms	RAM EPROM	JAP
	TI160	24	18	6	SI			SI	SI	B,I		RAM NOVRAM	USA
	520C-1102	512	512	512	SI	SI	SI	SI	SI	B,I	4ms	RAM EPROM	USA
	530C-1104	1023	1023	1023	SI	SI	SI	SI	SI	B,I	4ms	RAM EPROM	USA
	530C-1108	1023	1023	1023	SI	SI	SI	SI	SI	B,I	4ms	RAM EPROM	USA
	530C-1112	1023	1023	1023	SI	SI	SI	SI	SI	B,I	4ms	RAM,EPROM	USA
	525-1102	512	512	64	SI	SI	SI	SI	SI	A,I	3.7ms	R EPROM EEP	USA
	525-1104	1023	1023	1023	SI	SI	SI	SI	SI	B,I	3.7ms	R EPROM EEP	USA
	525-1108	1023	1023	1023	SI	SI	SI	SI	SI	B,I	3.7ms	R EPROM EEP	USA
	525-1212	1023	1023	1023	SI	SI	SI	SI	SI	B,I	3.7ms	R EPROM EEP	USA
	535-1204	1023	1023	1023	SI	SI	SI	SI	SI	B,I	0.83ms	R EPROM EEP	USA
	535-1212	1023	1023	1023	SI	SI	SI	SI	SI	B,I	0.83ms	RAM EPROM	USA
TOSHIBA	EX200	240	224	16	SI		SI	SI	SI	B	9ms	CMOS RAM	JAP
	EX250	240	256	16	SI		SI	SI	SI	B	7ms	CMOS RAM	JAP
	EX500	544	512	32	SI		SI	SI	SI	B	5ms	CMOS RAM	JAP
	EX14B	34	34		SI			SI	SI	B	60ms	CMOS RAM	JAP
	EX20-PLUS	40	40	2	SI			SI	SI	B	60ms	CMOS RAM	JAP
	EX28B	28	28		SI			SI	SI	B	60ms	CMOS RAM	JAP
	EX40-PLUS	80	80	2	SI			SI	SI	B	60ms	CMOS RAM	JAP
SQUARE D CO.	SY/MAX 50	256	256	32	SI				SI		7ms	R EPROM EEP	JAP
	SY/MAX 100	40	40		SI			SI	SI	J	10ms	RAMUVPROM	UK
	SY/MAX 300	256	256	112	SI	SI	SI	SI	SI	J	30ms	RAM UVPROM	USA
	SY/MAX 500	2000	2000	1792	SI	SI	SI	SI	SI	J	2.6ms	RAM UVPROM	USA
	SY/MAX 700	14K	14K	3584	SI	SI	SI	SI	SI	J	1.3ms	RAM BUBLE	USA
	SY/MAX LC	80		28	SI	SI	SI	SI	SI	J	200ms	RAM	USA
WESTINGHOUSE	PC-100	30	30							I	8ms	CMOS RAM	JAP
ELECTRIC CO.	PC-110	112	112							I	8ms	CMOS RAM	JAP
	PC1100	144	128	16	SI		SI	SI	SI	A	8ms	CMOS RAM	USA
	PC-900	288	256	32	SI	SI	SI	SI	SI	A	20ms	CMOS RAM	USA
	PC-700	576	512	64	SI	SI	SI	SI	SI	A	8ms	CMOS RAM	USA
	HPPC	8192	8192	8192	SI	SI	SI	SI	SI	A	0.8ms	CMOS RAM	USA
TURNBULL CONTROLS	6433	32	32	32		SI	SI	SI	SI	J		RAM	UK
UTICOR TECHNOLOGY	DIR. ONE	128	128		SI			SI	SI	A	20ms	RAM	JAP
VEEDER-ROOT CO.	V-12	120	120	15	SI			SI	SI		40ms	CMOS RAM	JAP
	V-12 EXP	80	80	8	SI			SI	SI		45ms	CMOS RAM	JAP
TEMPATRON, LTD	TPC 9000	252	252	60		SI	SI	SI	SI	BI	10ms	RAM EPROM	UK
SIEMENS	LOGO! 230RL	24	30		SI		SI	SI	SI	B	6ms	RAM	GER
	LOGO! LB11	24	30	5	SI		SI	SI	SI	B	6ms	RAM	GER
	SISMATIC S5 100U	120	120	15	SI	SI	SI	SI	SI	J	20ms	RAM EPROM	GER

Considerando que la elección de un autómata programable para su estudio particular, no necesariamente debe orientarse al más grande o con mayor capacidad de memoria si no más bien al que brinde las características funcionales de manejo, programación y simulación básicas requeridas para su estudio específico.

Según este criterio, se ha elegido como base del estudio al microcontrolador lógico programable SIEMENS de la serie LOGO!, el mismo que presenta las siguientes características de funcionamiento:

- El hardware es de manejo simple y fácil de comprender utilizando como novedad un display de programación integrado al dispositivo.
- Permite la creación y edición de programas Offline para aplicaciones de control automático.
- Dispone de un lenguaje gráfico de programación orientada a funciones lógicas.
- Realiza la transferencia de programas de autómata a computador o viceversa.

Estas características sumadas a la velocidad de procesamiento, y su económico costo, hacen que éste autómata posea las cualidades necesarias para su estudio en el siguiente capitulo y la implementación del sistema informático en el computador.

CAPITULO III

AUTOMATAS PROGRAMABLES SIEMENS

SERIE LOGO!

3.1. Características del Autómata Programable SIEMENS serie LOGO!

LOGO! es el nuevo modulo lógico universal de SIEMENS. Este autómata programable lleva integrados los siguientes elementos:

- Control
- Unidad de Operación Visualización
- Fuente de Alimentación
- Interfaz para módulos de programa y cable de PC
- Ciertas funciones usuales en la práctica por ejemplo para activación/desactivación temporizada y relé de impulsos.
- Reloj (LOGO! 230RC, LOGO! 230 RCL, LOGO! 24RC)
- Determinadas entradas y salidas según el tipo del equipo.

El microcontrolador LOGO! ofrece soluciones que abarcan desde la pequeña instalación doméstica, pasando por cometidos de automatización pequeños, hasta aplicaciones de gran magnitud.

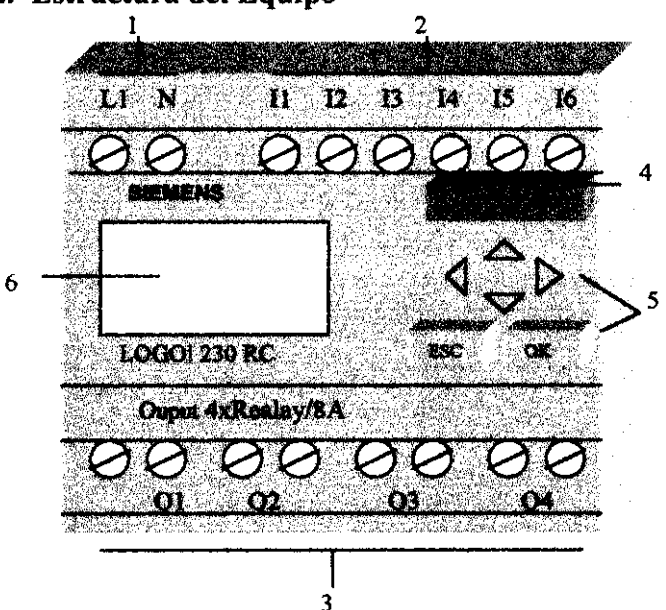
El campo de aplicación de este modelo de autómata, permite dar solución a circuitos de instalaciones técnicas en edificios (alumbrado de escaleras, luz exterior) y en la construcción de máquinas y aparatos (controles de puertas, instalaciones de ventilación, bombas de aguas residuales, etc.).

3.1.1. Tipos de Equipo

El equipo se ha previsto tanto para 24 Voltios como para 230 Voltios:

- Variante estándar con 6 entradas y 4 salidas, integrada en 72 x 90 x 55 mm
- Variante ". L" con 12 entradas y 8 salidas, así como funciones ampliadas, integrada en 126 x 90 x 55 mm.
- Variante ". LB11" con 12 entradas y 8 salidas, así como funciones ampliadas y conexión del bus adicional de la interfaz AS, a través del que hay disponibles en el sistema bus otras 4 entradas y otras 4 salidas. Todo ello integrado en 126 x 90 x 55 mm.

3.1.2. Estructura del Equipo



1. Alimentación
2. Entradas
3. Salidas
4. Receptáculo de módulo con revestimiento
5. Panel de manejo (teclas)
6. Display LCD

Figura 3-1

3.1.3. Identificación del Equipo LOGO!

En la identificación de LOGO! pueden reconocerse diferentes propiedades:

- 24: Versión de 24 V c.c.
- 230: Versión de 115/230 V a.c.
- R: Salidas de relé
- C: Reloj de temporización semanal integrado
- L: Cantidad doble de entradas y salidas, así como funciones ampliadas
- B11: Esclavo con conexión de bus interfaz AS

En el transcurso restante de esta descripción se utilizan pequeños pictogramas para identificar los tipos con funciones diferentes. Estos pictogramas aparecen cuando las informaciones sólo se refieren a una parte de las variantes de LOGO!

Variante estándar con 6 entradas y 4 salidas, integrada en 72 x 90 x 55 mm.

Variante ". L" con 12 entradas y 8 salidas, integrada en 126 x 90 x 55 mm.

Variante ". LB11" con 12 entradas y 8 salidas, así como conexión de bus adicional de interfaz AS con 4 entradas virtuales y 4 salidas virtuales, integrada en 126 x 90 x 55 mm.

Las indicaciones que conciernen solo a las versiones ". C" o solo a las versiones ". R" se identifican correspondientemente en el texto.

3.2. Montaje y cableado de LOGO!

LOGO! debe montarse en una caja de distribución o un armario de conexiones. Tras el montaje, los bornes deben quedar cubiertos para impedir con certeza que se toquen por descuido piezas de LOGO! Bajo tensión.

LOGO! solo podrá ser montado y cableado por un especialista calificado que conozca y observe las reglas generales de la técnica, así como las prescripciones y normas vigentes en cada caso.

Dimensiones

LOGO! tiene las dimensiones para equipos de instalación estipuladas en DIN 43880.

LOGO! debe montarse sobre un perfil normalizado de 35mm de ancho según DIN EN 50022.

Anchura de LOGO!

- LOGO!: 72 mm de ancho, equivalente a 4 unidades de división (versión estándar)
- LOGO!...L: 126 mm de ancho, equivalente a 7 unidades de división
- LOGO!...LB11: 126mm de ancho, equivalente a 7 unidades de división

3.2.1. Conexión de las entradas de LOGO!

Condiciones

A las entradas se conectan sensores, los cuales pueden ser pulsadores, conmutadores, barreras luminosas, sensores de luminosidad, etc.

Cambio de estado de conmutación 0 – 1 / 1 – 0

Al cambiar del estado de conmutación 0 al 1 y del estado 1 al 0 debe estar aplicado por menos 50 metros el estado de conmutación 1 ó el estado de conmutación 0, respectivamente, para que LOGO! reconozca el nuevo estado de conmutación.

Conexión

Para la conexión de los sensores a LOGO! se describe gráficamente a continuación el procedimiento según el modelo a emplear:

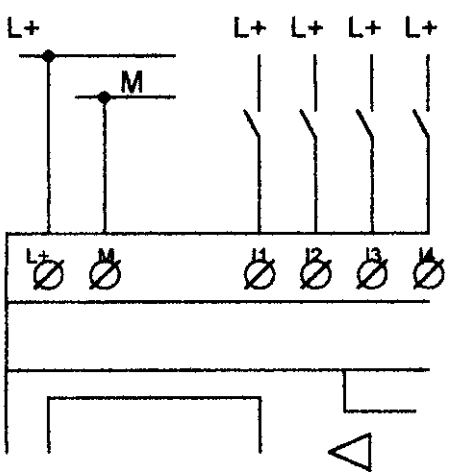


Figura 3-2
LOGO! 24

LOGO! 24...
Las entradas de LOGO! 24... no poseen separación galvánica, por lo que requieren el mismo potencial de referencia (masa) que la tensión de alimentación.

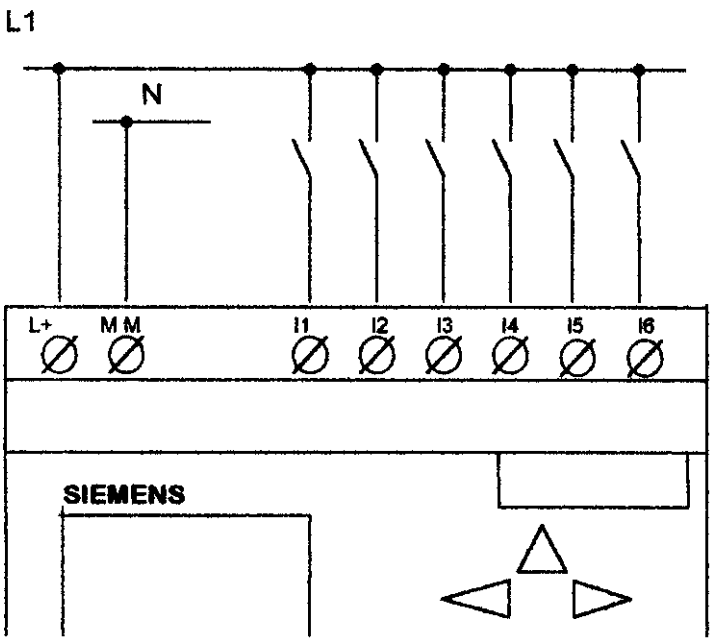
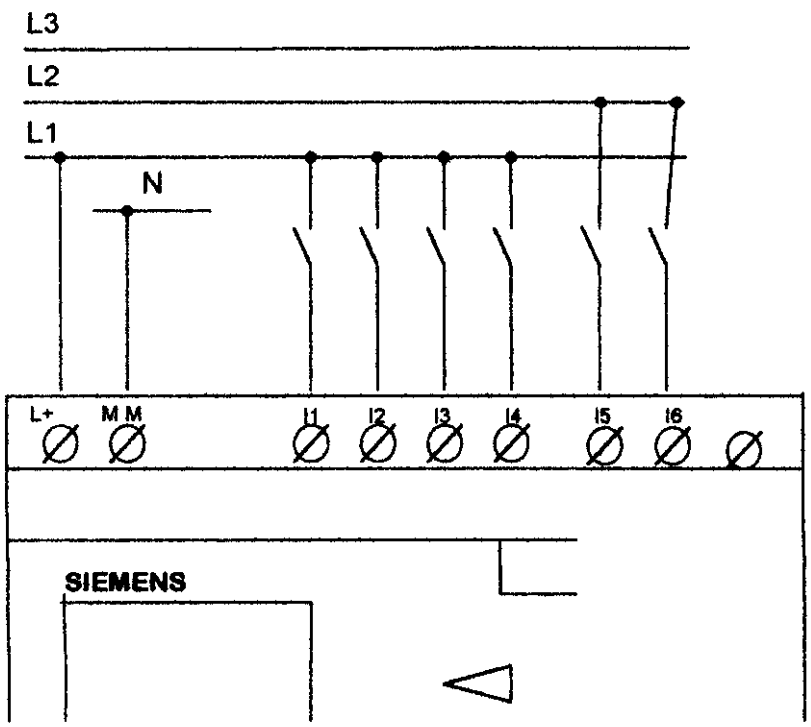


Figura 3-3
LOGO! 230 ...(versión estándar)



LOGO! 230...L...
Las entradas de LOGO! L... están combinadas en grupos de 4 entradas. Para dichos grupos rige lo mismo que para las entradas individuales de un LOGO! estándar. Sólo son posibles fases diferentes entre los distintos bloques.

Figura 3-4
LOGO! 230 .. L...

3.2.2. Conexión de las salidas

LOGO! 230R... y LOGO! 24R...

Las salidas de LOGO! 230R... y LOFO! 24R... son relés. En los contactos de los relés se encuentra separado el potencial de la tensión de alimentación y de las entradas.

Condiciones para las salidas de relé

A las salidas pueden conectarse distintas cargas, por ejemplo, lámparas, tubos fluorescentes, motores, contactores, etc. La carga conectada a LOGO! ...R... debe atenerse a las propiedades siguientes:

- La máxima corriente de conmutación depende de la carga y de la cantidad de maniobras deseadas.

- En el estado conectado (Q=1) puede circular como máximo una corriente de 8 amperios (10 A para LOGO!...RL..) en caso de carga óhmica y una de 2 amperios (3ª para LOGO!...RL...) en caso de carga inductiva.

Conexión

Para conectar la carga a las variantes de LOGO! ...R...

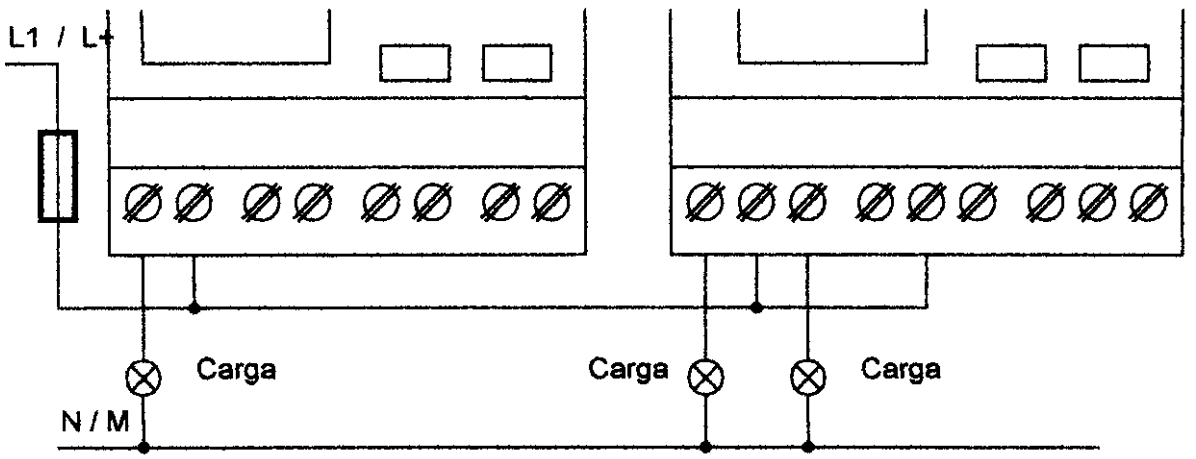


Figura 3-5

Protección mediante fusible automático máximo 16 A, características B16, por ejemplo: interruptor de potencia 5SX2 116-6 (sí se desea)

LOGO! 24... con salidas de transistor

Las variantes de LOGO! 24... con salidas de transistor se reconocen por faltar la letra R en su designación de tipo. Las salidas están protegidas contra cortocircuitos y sobrecargas. No es necesario aplicar por separado la tensión de la carga, ya que LOGO! 24 asume la alimentación de esta tensión.

Condiciones para las salidas de transistor

La carga conectada a LOGO! 24... debe atenerse a las propiedades siguientes:

- La máxima carga de conmutación es de 0,3 amperios por cada salida.
- En el estado conectado (Q=1) puede circular como máximo una corriente de 0,3 amperios.

Conexión

Para conectar la carga a LOGO! 24:

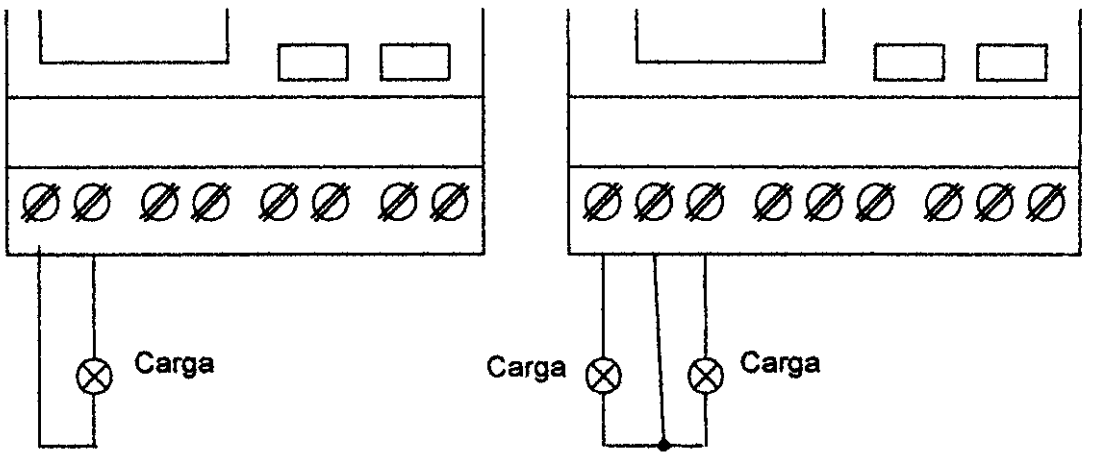


Figura 3-6

3.2.3. Conexión del bus ASi (sólo LOGO! ...LB11)

LOGO!...LB11

Es posible implementar LOGO! ...LB11 en una red como esclavo ASi. A través de un conductor bifilar se puede entonces:

- Introducir y procesar 4 entradas adicionales a través del bus ASi.
- Operar 4 salidas adicionales hacia un maestro superpuesto del bus ASi.

La configuración de LOGO! ...LB11 en el bus ASi se lleva a cabo mediante el maestro ASi utilizado por usted.

Condiciones para la operación con un maestro ASi

Téngase en cuenta que LOGO! ...LB11 debe estar dado de alta en el sistema ASi, es decir, que LOGO! recibe una dirección del maestro del bus.

Conexión

Enlace el cable de conexión de bus con un enchufe homologado para el sistema. Cerciórese de que la polaridad es correcta.

A continuación, enchufe el conector cableado en la interfaz identificado mediante AS-Interface.

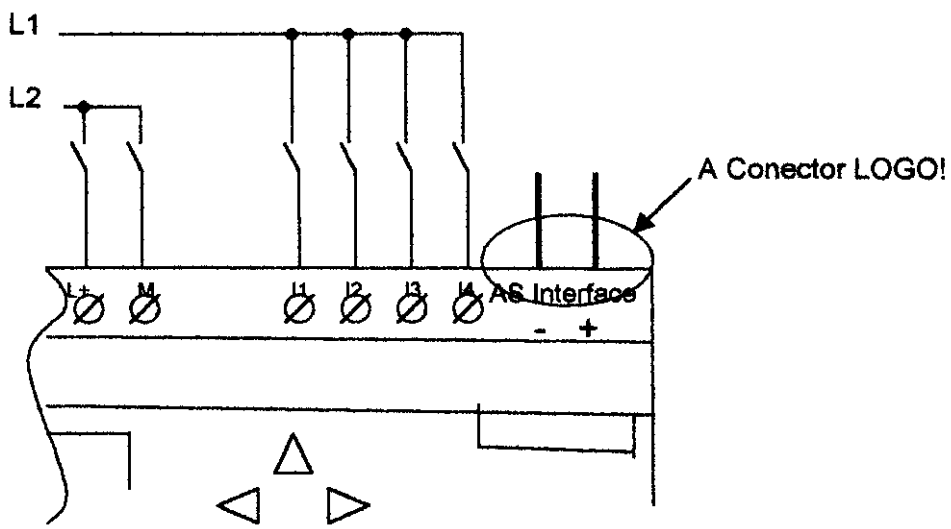


Figura 3-7
LOGO! ...LB11

3.2.4. LOGO! ...LB11 en el bus ASi

A fin de poder aprovechar las funciones ASi, el maestro del bus debe conocer a LOGO! ...LB11. Ello sucede al enlazar LOGO! ...LB11 con el conductor del bus. El maestro reconoce la dirección del esclavo.

LOGO! ...LB11lleva ajustada la fábrica la dirección = 0. El maestro adjudica una nueva dirección diferente a 0.

Si en el sistema no existen conflictos de direcciones o bien sólo hay enchufado un esclavo con la dirección 0, no se requiere otras operaciones de su parte.

3.2.5. Conectar LOGO!/reposición de la red

LOGO! no cuenta con interruptor de red. La reacción de LOGO! a la conexión varía según:

- Si hay almacenado un programa en LOGO!.
- Si hay insertado un módulo de programa
- El estado en que se hallaba LOGO! antes de desconectarse la red.

A continuación se describe la reacción de LOGO! durante las situaciones posibles:

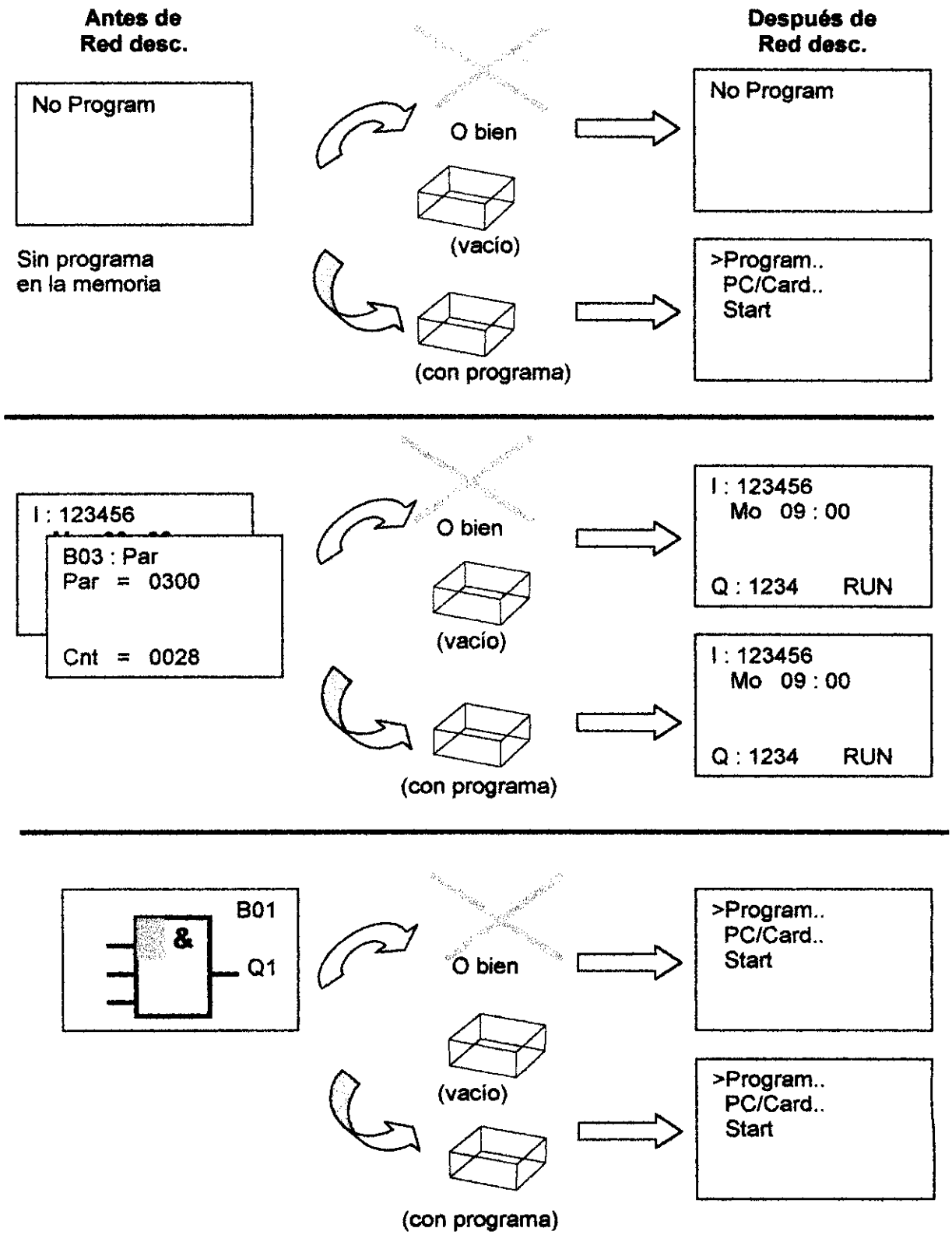


Figura 3-8
Conexión LOGO / reposición red

He aquí 5 reglas sencillas para comprender el arranque de LOGO!:

1. Si no hay ningún programa en LOGO! ni en el módulo de programa insertado, LOGO! visualiza : No Program.
2. Si el módulo de programa contiene un programa, es copiado éste automáticamente en LOGO!, sobrescribiéndose el programa que hubiera en LOGO!.
3. Si existe un programa en LOGO! o en el módulo de programa, LOGO! pasa al estado de servicio que ocupa antes de desconectarse la red.
4. Si se utiliza una variante de LOGO!...L con módulo rojo o amarillo y se ha activada la remanencia para una función por lo menos o bien se prevé una función con remanencia activada continuamente, se conservan sus valores actuales tras desconectarse la red.
5. En las demás variantes, son repuestos los tiempos y los valores de cómputo al desconectarse la red, pero el programa se conserva almacenado a prueba de cortes de la red.

Estados de servicio de LOGO!

En LOGO! se prevén dos estados de servicio: STOP y RUN

LOGO! en estado STOP	LOGO! en estado RUN
Si se visualiza en el display "No program" o bien al conectarse LOGO! a la clase de servicio "Programación".	Si se visualiza en el display la máscara para observar las entradas y salidas (tras START en el menú principal) o bien al conectarse LOGO! a la clase de servicio "Parametrización"
Acciones de LOGO!: • No son leídas las entradas, • No es procesado el programa y • Están siempre abiertos los contactos de relé o desconectadas las salidas de transistor.	Acciones de LOGO!: • LOGO! lee el estado de las entradas, LOGO! calcula mediante el programa el estado de la salidas y LOGO! Activa o desactiva los relés salidas de transistor.

3.3.2. Bornes

LOGO! cuenta con entradas y salidas:

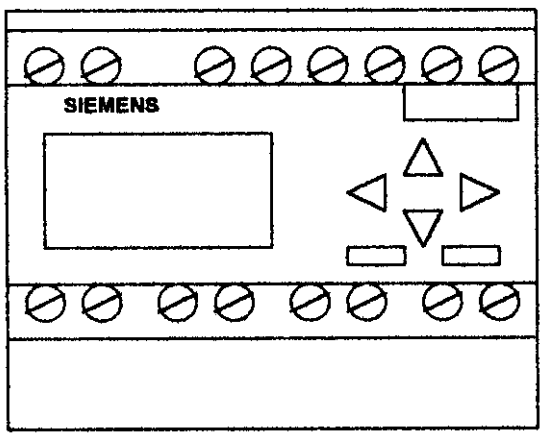


Figura 3-9
Bornes de LOGO!

Las entradas se designan con la letra I y una cifra. Visto LOGO! por delante, los bornes para las entradas aparecen arriba.

Las salidas se designan con la letra Q y una cifra. Los bornes de las salidas se hallan en la parte inferior.

Bornes de LOGO!

Se entiende por borne a todas las conexiones y estados que encuentran aplicación en LOGO!. Las entradas y las salidas pueden tener el estado '0' o el estado '1'. Como es sabio, el estado '0' significa que la entrada no lleva aplicada tensión y el estado '1' que hay aplicada tensión.

Hemos previsto los bornes hi, lo y x para facilitar la entrada de la programación. 'hi' (high) lleva asignado fijamente el estado '1' y 'lo' (low) el estado '0'.

Si no se desea cablear la entrada de un bloque, debe utilizarse el borne 'x'.

LOGO! conoce los bornes siguientes:

Bornes			
Entradas	I1 a I6	I1 a I12	I1 a I12, así como Ia1 a Ia4(Interfaz AS)
Salidas	Q1 a Q4	Q1 a Q8	Q1 a Q8, así como Qa1 a Qa4(interfaz AS)
Io	Señal con nivel '0' (desc.)		
Hi	Señal con nivel '1' (con.)		
X	Terminal existente no utilizado		

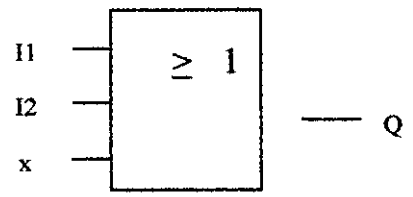
3.3.3. Bloques y números de bloques

Un bloque en LOGO! es una función que convierte informaciones de entrada en informaciones de salida. Antes tenía usted que cablear los distintos elementos en el armario de conexiones o en la caja de distribución.

En la programación se enlazan bornes con bloques. A tal efecto, basta con elegir la conexión deseada en el menú **Co**. Este menú se denomina Co basándose en el término inglés Connector (borne).

Los bloques más sencillos son combinaciones lógicas:

- Y
- O
- ...



Aquí las entradas I1 e I2 están conectadas al bloque O

La última entrada del bloque no se utiliza, identificándos

por ello mediante x.

Bastante más eficiente son las funciones especiales:

- Relé de impulsos
- Contador
- Retardo de activación
- ...

Representación de un bloque en el display de LOGO!

Tenemos muestra de una visualización típica en el display de LOGO!. Se ve aquí que sólo puede representarse cada vez un bloque. Debido a ello, hemos previsto los números de bloque para ayudarle a usted a controlar un circuito en conjunto.

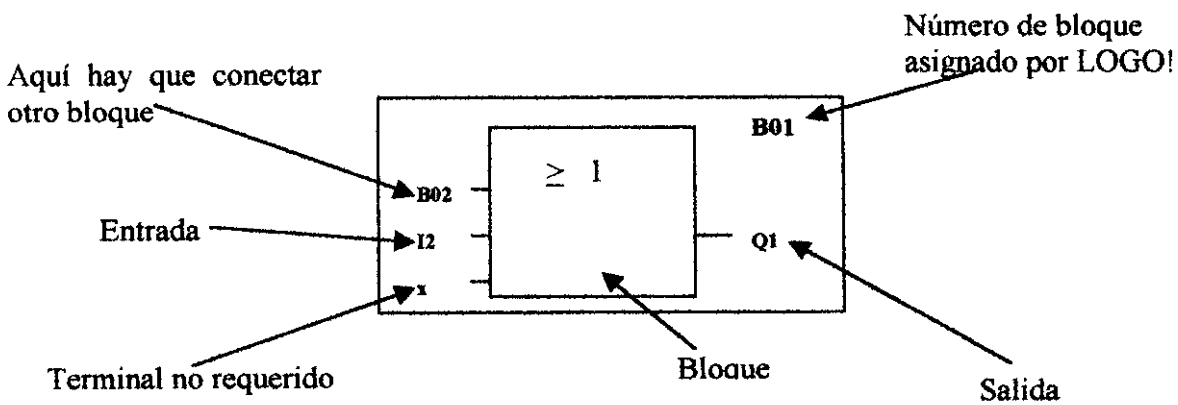


Figura 3-10
Visualización en el display de LOGO!

Asignación de un número de bloque

Cada vez que se intercala un bloque en un programa LOGO! adjudica un número a ese bloque.

A través del número de bloque, LOGO! muestra la relación existente entre los bloques. Es decir, que los números de bloque sirven primero únicamente para su orientación en el programa.

Número de bloque

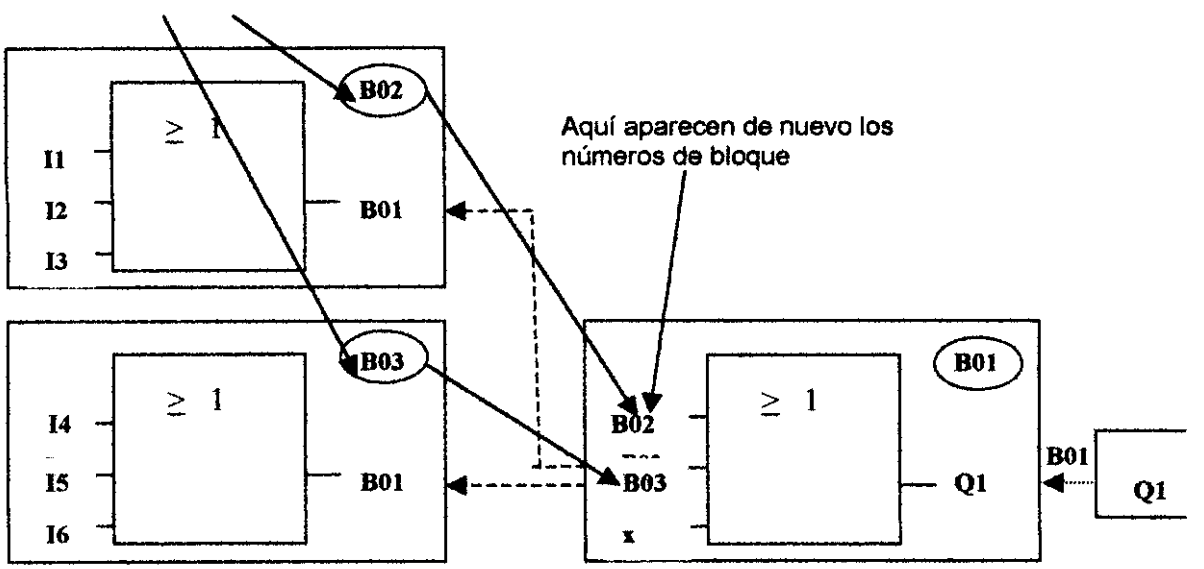


Figura 3-11
Visualización en el display de LOGO!

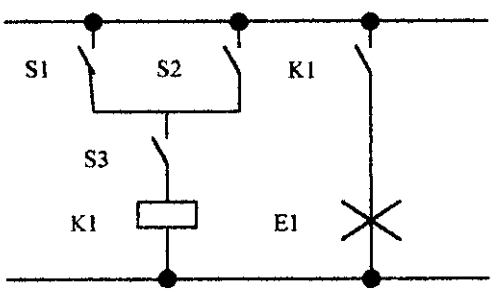
En el diagrama general se ven tres representaciones en el display de LOGO!, que constituyen un conjunto el programa. Puede verse cómo LOGO! relaciona los bloques entre sí a través de sus números.

Pero los números de bloque tienen aún otra ventaja, que puede aprovechar: A través de su número de bloque, es posible añadir casi cualquier bloque a una entrada del bloque actual. De esta manera, pueden utilizarse repetidas veces los resultados intermedios de relaciones lógicas u otras operaciones. Con ello se ahorra trabajo y capacidad de memoria, a la vez que su circuito resulta más transparente. En dicho caso tiene que saber cómo se denominaron los bloques de LOGO!.

3.3.4. Del esquema de circuitos a LOGO!

Representación de un circuito en el esquema de circuitos

Seguramente es conocido la representación de un circuito en el esquema de circuitos. Aquí se ilustra un ejemplo:



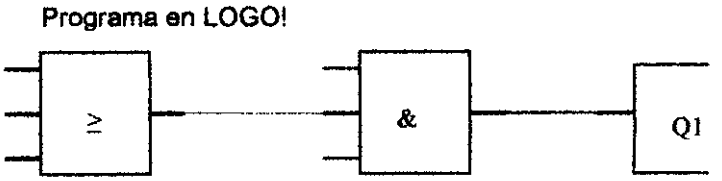
El consumidor E1 es activado y desactivado a través del interruptor (S1 O S2) Y S3(O=OR; Y=AND).

Se excita el relé K1 al cerrarse S1 o S2 y además S3

Figura 3-12

Realización del circuito mediante LOGO!

En LOGO! se realiza un circuito interconectado bloques y bornes:

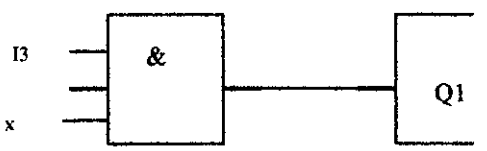


Para convertir un circuito en LOGO!, debe comenzar por la salida del circuito.

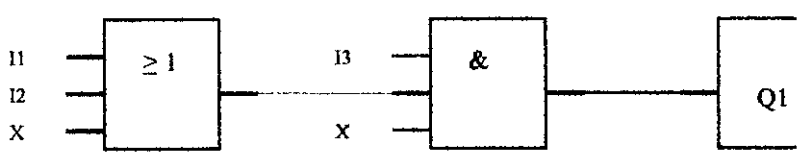
La salida es la carga o el relé que debe efectuar la conmutación.

El circuito es convertido es bloques. A tal efecto, debe procesar el circuito desde la salida hasta la entrada.

Paso 1: La Salida Q1 va seguida de una conexión en serie del contacto de cierre S# con otro elemento del circuito. Esta conexión en serie equivale a un bloque Y:



Paso 2: S1 y S2 están conectados en paralelo. Esta conexión es paralelo equivale a un bloque O:



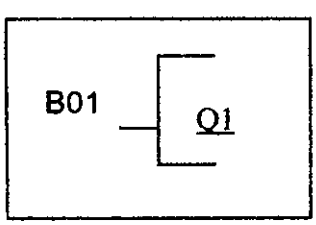
Con ello se ha descrito íntegramente el circuito para LOGO!. Por último, cierre las entradas y las salidas en LOGO!

En este programa no se requiere la última entrada del bloque O. En los programas de LOGO! se identifica con una "x" cada entrada no utilizada. Introducir ahora la 'x' (según el principio ya conocido):

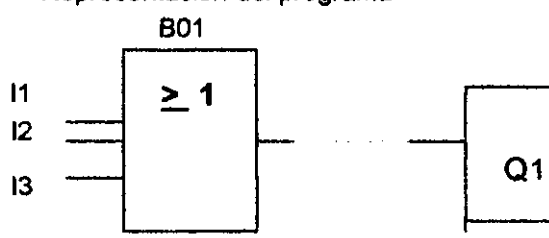
1. Pasara al modo de introducción — Tecla **OK**
2. Elegir la lista Co: — Teclas ▲ o ▼
3. Aceptar la lista Co: — Tecla **OK**
4. Elegir x: — Teclas ▲ o ▼
5. Aceptar x: — Tecla **OK**

Así quedan cableadas todas las entradas del bloque y el programa está completo para LOGO!. LOGO! retrocede a la salida Q1.

En el display aparece



Representación del programa



Si se desea ver de nuevo el primer programa, es posible desplazar el cursor a través del programa mediante las teclas ◀ o ▶.

Acto seguido, se termina la introducción del programa. Para ello, proceder de la siguiente manera:

1. Regreso al menú de programación: Tecla **ESC**

Si no se regresa ahora al menú de programación, significa que se ha olvidado cablear íntegramente un bloque. LOGO! muestra el punto del programa donde se ha olvidado hacer algo (por razones de seguridad, LOGO! sólo acepta programas completos).

2. Regreso al menú principal: Tecla **ESC**

Conmutación de LOGO! a RUN

3. Posicionar '>' en 'Start': Teclas ▼ o ▲
4. Confirmar Start: Tecla **OK**

LOGO! se conmuta a RUN, apareciendo el display siguiente:

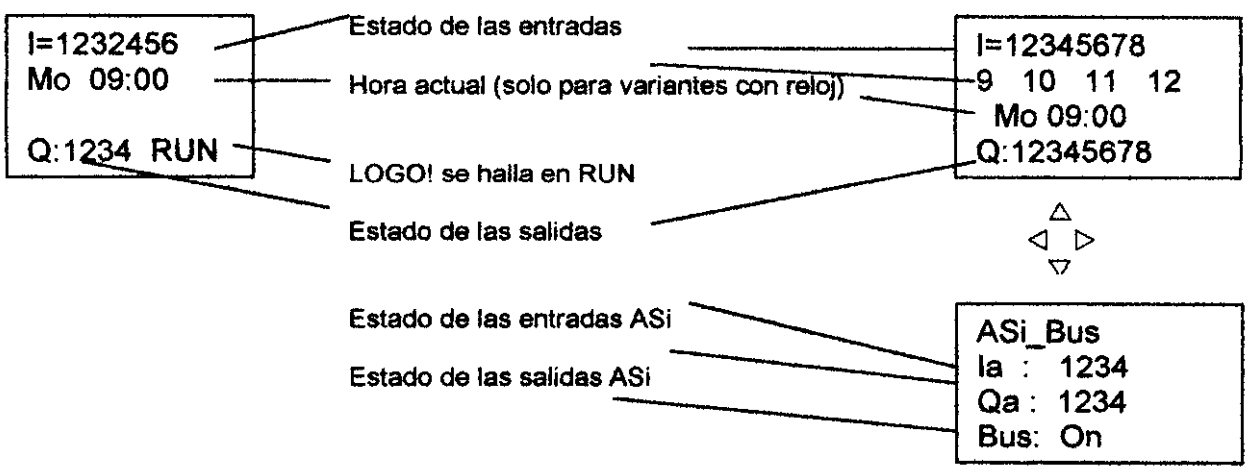


Figura 3-16
Visualización de LOGO! en RUN

Significado de "LOGO! se halla en RUN?"

En RUN, LOGO! procesa el programa. A tal efecto, LOGO! lee primero los estados de las entradas, determina los estados de las salidas a base del programa recién indicando y activa o desactiva los relés en las salidas.

Representación del estado de una entrada o salida en LOGO!:

I : 123456 Mo 09:00 Q: 1234	Entrada/salida con estado '1': inverso Entrada/salida con estado '0': no inverso	I : 12345678 9 10 11 12 Mo 09:00 Q : 12345678

Consideremos esto en nuestro ejemplo:

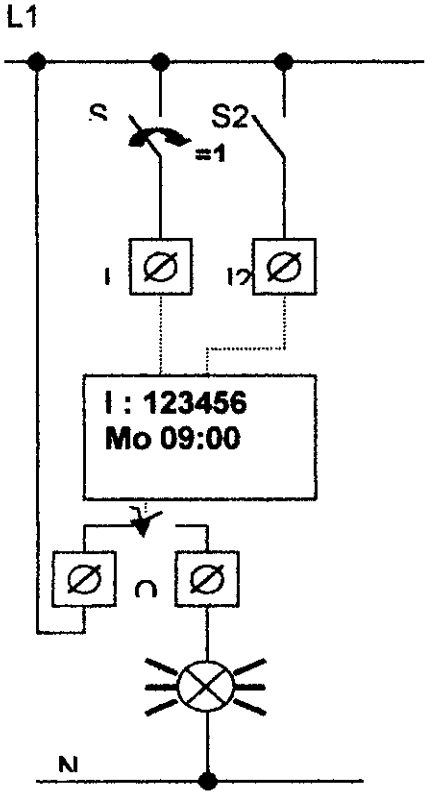


Figura 3-17

Si está cerrado el interruptor S1, hay aplicada tensión a la entrada I1 y ésta presenta el estado '1'.

LOGO! calcula mediante el programa el estado para las salidas.

La salida Q1 tiene aquí el estado '1'.

Si Q1 tiene el estado '1', LOGO! activa el relé Q1 y se aplica tensión al consumidor conectado a Q1.

Próximo Paso

Ahora ha introducido usted con éxito el primer circuito.

En el apartado siguiente se explica cómo puede usted modificar los programas existentes y utilizar en los mismos funciones especiales.

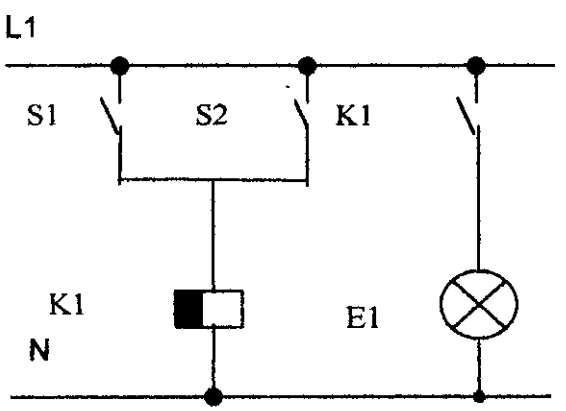
3.3.7.4 Segundo Programa

Mediante el segundo programa se muestra los puntos siguientes:

- Cómo se intercala un bloque en un programa existente.
- Cómo se elige un bloque para una función especial.
- Cómo se introducen los parámetros.

Para el segundo programa se modifica algo el primero.

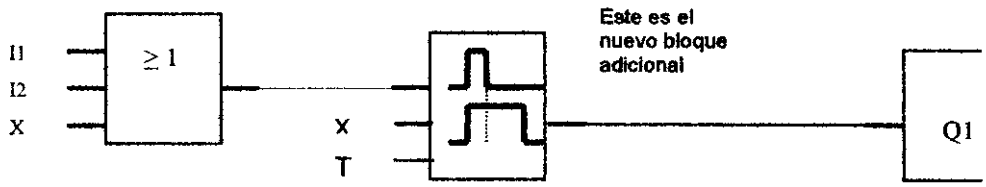
Vea primeramente el esquema de circuitos para el segundo programa:



La primera parte del circuito ya es conocida. Los dos interruptores S1 y S2 conectan un relé. Este relé debe activar el consumidor E1 y desactivarlo con 12 minutos de retardo

Figura 3-18

Representación del programa correspondiente en LOGO!:



Del primer programa son conocidos el bloque O y el relé de salida Q1. Sólo es nuevo el retardo de desactivación.

Forma de modificar el primer programa:

Conmutar LOGO! al modo de edición.

Como se dijo anteriormente, se efectúan de la siguiente manera:

1. Conmutar LOGO! al modo de servicio "Programación" (pulsando la teclas◀ , ▶ y **OK** simultáneamente).

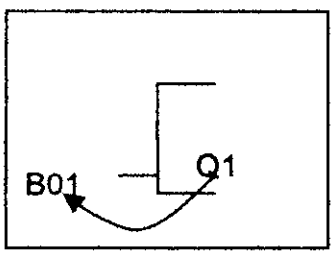
2. Elegir en el menú principal "Program..." (desplazar '>' hacia "Program..." y pulsar **OK**).

3. Elegir en el menú de programación "Edit Prg" (desplazar '>' hacia "Edit Prg" y pulsar **OK**).

Ahora es posible modificar el programa existente.

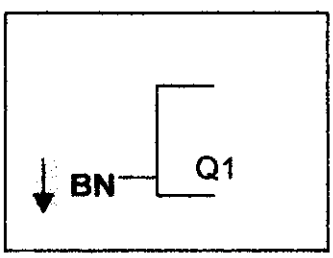
Intercalar un bloque adicional en un programa

Posicionar el cursor en la letra B de BO1 (BO1 es el número de bloque O).



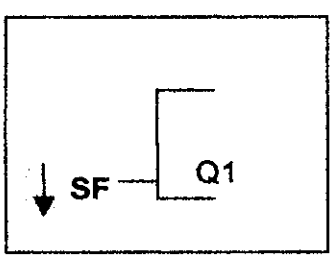
Desplazar el cursor pulsando la tecla ◀

Aquí se intercala el nuevo bloque. Pulsar la tecla **OK**:



LOGO! visualiza la lista BN.

Elegir la lista SF (tecla ▼).



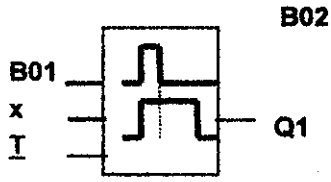
La lista SF incluye los bloques para funciones especiales

Pulsar la tecla **OK**

Se visualiza el bloque de la primera función especial:

1. Posicionar el cursor e R: Teclas ▲ o ▼
2. Conmutar al modo de introducción: Tecla **OK**
3. Elegir la lista Co: Teclas ▲ o ▼
4. Asumir la lista Co: Tecla **OK**
5. Elegir 'x': Teclas ▲ o ▼
6. Asumir 'x': Tecla **OK**

En el display debería aparecer:



Introducir ahora el tiempo T para el retardo de desactivación:

1. Si el cursor no se halla aún bajo T, posicionarlo allí: Teclas ▲ o ▼
2. Conmutar al modo de introducción: Tecla **OK**

Si se prevén parámetros LOGO! visualiza la ventana de parámetros:

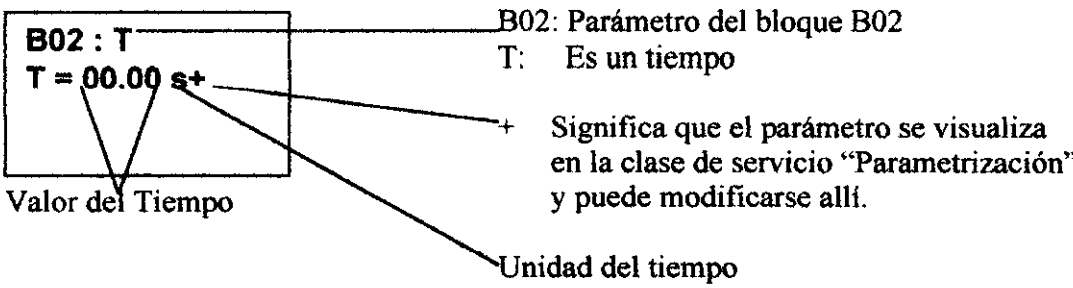


Figura 3-19
Ventana de Parámetros de LOGO!

El cursor se halla en el primer dígito del valor del tiempo.

Para modificar el valor del tiempo:

Mediante las teclas ◀ y ▶ se desplaza el cursor.

Mediante la teclas ▲ y ▼ se modifica el valor en ese dígito

Una vez introducido el valor del tiempo, pulsar la tecla **OK**.

Ajustar el tiempo T = 12.00 minutos:

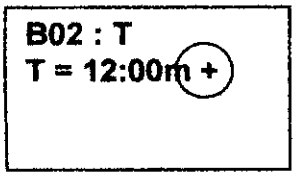
1. Posicionar el cursor en el primer dígito: Teclas ◀ o ▶
2. Elegir la cifra '1': Teclas ▲ o▼
3. Posicionar el curso en el segundo dígito: Teclas ◀ o ▶
4. Elegir la cifra '2': Teclas ▲ o▼
5. Posicionar el cursor en las unidades: Teclas ◀ o ▶
6. Elegir la unidad m para minutos: Teclas ▲ o▼

Visualización/enmascaramiento de parámetros - Clase de protección

Si se desea que en el modo de parametrización no se visualiza el parámetro:

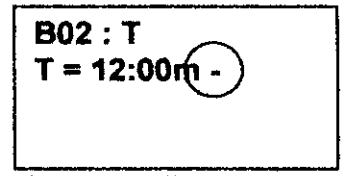
7. Posicionar el cursor en la clase de protección: Teclas ◀ o ▶
8. Elegir la clase de protección '-': Teclas ▲ o▼

En el display debería aparecer ahora:



Clase de protección +:
Tiempo t modificable en la clase de servicio "Parametrización"

O bien



Clase de protección -:
Tiempo T no modificable en la clase de servicio "Parametrización"

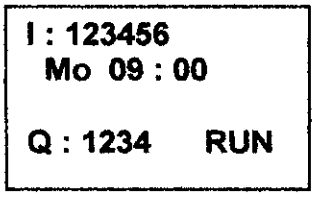
9. Concluir la introducción: Tecla **OK**

Ahora está completa la rama del circuito para Q1. LOGO! muestra la salida Q1. Es posible observar el programa nuevamente en el display, desplazándose dentro del programa por medio de las teclas. Mediante ◀ o ▶ de un bloque a otro, y mediante ▲ o▼ y hacia las distintas entradas en un bloque.

Como ya se expuso para el primer programa, la introducción del circuito puede concluirse de la siguiente forma:

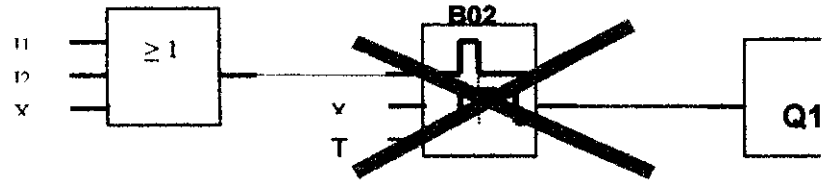
1. Regreso al menú de programación — Tecla **ESC**
2. Regreso al menú principal: — Tecla **ESC**
3. Llevar '>' a 'Start': — Teclas ▲o▼
4. Confirmar Start: — Tecla **OK**

LOGO! se halla ahora nuevamente en RUN:



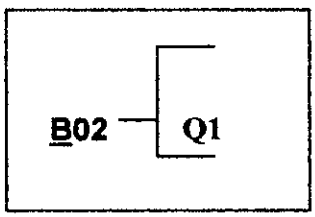
3.3.7.5 Segundo Programa

Supongamos que en el programa siguiente se desea borrar el bloque BO2 y enlazar BO1 directamente con Q1.



Procédase para ello de la forma siguiente:

1. Conmutar LOGO! a la clase de servicio 'Programación' (pulsación triple).
2. Elegir 'Edit Prg' pulsando la tecla **OK**.
3. Posicionar el cursor en la entrada de Q1, es decir, bajo BO2. Utilice para ello la tecla:



4. Pulsar la tecla **OK.**
5. Ahora se aplica directamente el bloque B01 a la salida Q1 en vez del bloque B02:

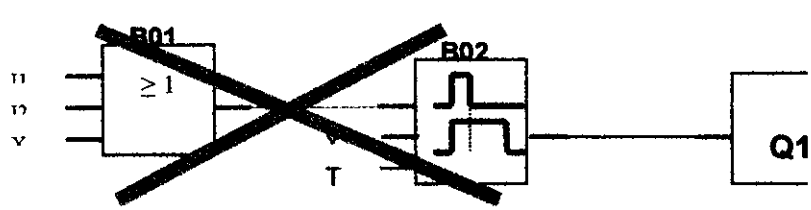
   Elegir la lista BN y pulsar OK.

   Elegir B01 y pulsar OK.

Resultado: El bloque B02 se ha borrado, porque ya no se utiliza en todo el circuito. En vez del mismo, la salida lleva aplicado directamente B01.

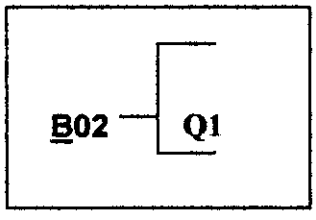
3.3.7.6. Borrar varios bloques consecutivos

Supongamos que en el programa siguiente se desean borrar los bloques BO1 y BO2.



Procédase par ello de la forma siguiente:

1. Conmutar LOGO! a la clase de servicio 'Programación' (pulsación triple).
2. Elegir 'Edit Prg' pulsando la tecla **OK**.
3. Posicionar el cursor en la entrada de Q1, es decir, bajo BO2:



4. Pulsar la tecla **OK**.
5. Ahora se aplicar el conector x a la salida Q1 en vez del bloque BO2:

   Elegir la lista Co y pulsar OK

   Elegir x y pulsar OK.

Resultado: El bloque BO2 se ha borrado, porque ya no se utiliza en todo el circuito. Con el bloque BO2 se borraron todos los bloques conectados al mismo (en el ejemplo también el bloque BO1).

3.3.7.7. Corrección de introducciones erróneas

LOGO! permite corregir muy fácilmente las introducciones erróneas:

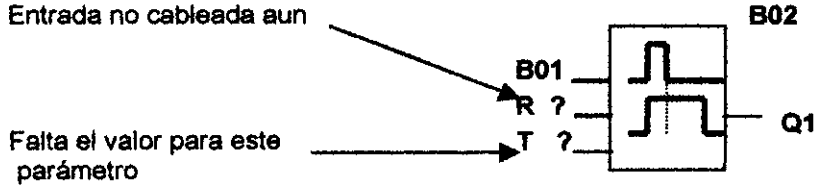
- Mientras no haya acabado la introducción, se puede retroceder un paso mediante ESC
- Si ya ha acabado la introducción, repetir sencillamente ésta:

1. Posicionar el cursor al punto que deba corregirse.
2. Conmutar al modo de introducción: tecla **OK**.
3. Introducir el cableado correcto para la entrada.

Para poder sustituir un bloque por otro es condición indispensable que el bloque nuevo cuente con la misma cantidad de entradas que el antiguo. Sin embargo, también es posible borrar el bloque antiguo e intercalar uno nuevo elegible discrecionalmente.

3.3.7.8. "?" en el display

Si se ha introducido un programa y desea abandonar "Edit Prg" mediante ESC, LOGO! comprueban si están cableadas todas las entradas de todos los bloques. Si se hubiera olvidado alguna entrada o parámetro, LOGO! visualiza el primer punto donde se olvidó algo y marca con un signo de interrogación todas las entradas y parámetros no cableadas y los parámetros que faltan.



Cablear ahora correctamente la entrada e introducir un valor adecuado para el parámetro. Entonces puede abandonarse "Edit Prg" pulsando la tecla **ESC**.

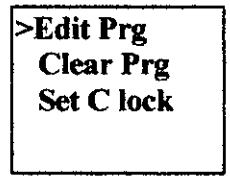
3.3.7.9. Borrar programas

Manera de borrar un programa:

1. Conmutar LOGO! a la clase de servicio "Programación": Teclas ▲ , ▼ y OK simultáneamente.

2. Desplazar el '>' mediante las teclas ▲ o ▼ hacia "Program.." y pulsar la tecla **OK**

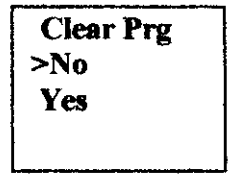
LOGO! pasa al menú de programación:



3. Desplazar el '>' hacia 'Clear Prg': Teclas ▲ o ▼

4. Aceptar 'Clear Prg': Tecla **OK**

Para evitar que se borre por descuido el programa, hemos previsto además una consulta:



Si no se desea borrar el programa, dejar '>' en 'No' y pulsar la tecla **OK**.

Si se tiene la certeza de que debe borrar el programa almacenado en LOGO!, entonces

5. Desplazar '>' hacia 'Yes': Teclas ▲ o ▼
6. Pulsar **OK**

LOGO! borra el programa y regresa a continuación al menú de programación:

3.3.8. Funciones

LOGO! pone a disposición diferentes elementos en el modo de programación. Para su orientación, hemos distribuido dichos elementos en distintas 'listas', que especificaremos a continuación:

- ↓**Co:** Lista de bornes (Connector) para
  - Entradas: I1, ...
  - Salidas: Q1, ...
  - Niveles: lo, hi
  - no conectados: x
- ↓**GF:** Lista de funciones básicas AND, OR, ...
- ↓**SF:** Lista de funciones especiales
- ↓**BN:** Lista de bloques ya listo en el circuito y utilizable posteriormente

Contenido de las listas

Todas las listas incluyen elementos disponibles en LOGO!. Normalmente se trata de todos los bornes, todas las funciones básicas y todas las funciones especiales que conoce el respectivo LOGO!. También van incluidos todos los bloques que usted ya ha creado en LOGO! antes de haber solicitado la respectiva lista **↓BN**.

No visualización de algunos elementos

LOGO! ya no visualiza todos los elementos en los casos siguientes:

- si no puede insertarse ningún otro bloque

  En este caso no hay disponible capacidad de memoria o se alcanzó la cantidad máxima de bloques posibles (30).

- si un bloque requiere más capacidad de memoria que la disponible aún en LOGO!

- si resultaran entonces más de 7 bloques conectados en serie

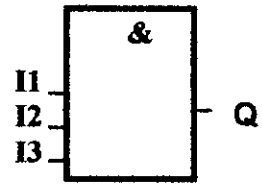
3.3.9. Funciones básicas

En la lista GF se especifican los bloques de funciones básicas para la introducción de un circuito. Se prevén las siguientes funciones básicas:

3.3.9.1. Y (AND)

La conexión en serie de varios contactos de cierre se representa así en el esquema de circuitos:

Símbolo en LOGO!



Este bloque se denomina Y (AND) porque la salida Q de Y sólo ocupa el estado 1 cuando I1 e I2 e I3 tienen el estado 1, es decir cuando están cerrados.

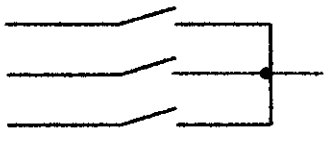
Tabla lógica para la función Y

I1	I2	I3	Q
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

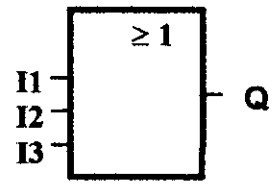
Para Y rige x = 1
(x significa que la entrada no se utiliza)

3.3.9.2. O (OR)

La conexión en paralelo de varios contactos de cierre se representa así en el esquema de circuitos:



Símbolo en LOGO!



Este bloque se denomina O porque la salida Q de O siempre ocupa el estado 1 cuando I1 **o** I2 **o** I3 tienen el estado 1, es decir, cuando están cerrados (o sea, que por lo menos una entrada debe tener el estado 1).

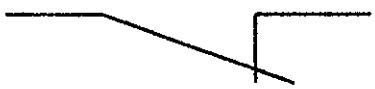
Tabla lógica para la función O

I1	Y2	I3	Q
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

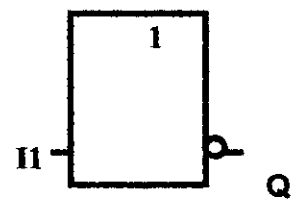
Para O rige x = 0
(x significa que la entrada no se utiliza)

3.3.9.3. INVERSOR (NOT)

Un inversor se representa así en el esquema de circuitos:



Representación del INVERSOR en LOGO!:



Este bloque se denomina INVERSOR porque la salida Q1 ocupa el estado 1 cuando la entrada tiene el estado 0 y viceversa, es decir, que INVERSOR invierte el estado en la entrada.

Ejemplo de la ventaja que supone INVERSOR: Para LOGO! ya no se requiere ningún contacto de apertura, pues basta con utilizar un contacto de cierre y convertirlo en uno de apertura mediante el bloque INVERSOR.

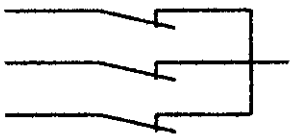
Tabla lógica para el bloque INVERSOR

I1	Q
0	1
1	0

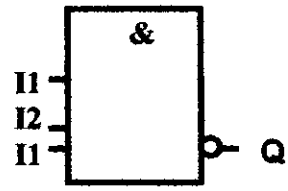
Para INVERSOR rige x = 1
(x significa que la entrada no se utiliza)

3.3.9.4. Y-NEGADA (NAND)

La conexión en paralelo de varios contactos de apertura se representa así en el esquema de circuitos:



Símbolo en LOGO!



Este bloque se denomina Y-NEGADA porque la salida Q de Y-NEGADA sólo ocupa el estado 0 cuando I1 e I2 e I3 tienen le estado 1, es decir, cuando están cerrados.

Tabla lógica para la función Y-NEGADA

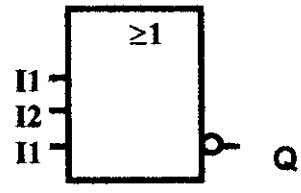
I1	I2	I3	Q
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Para Y-NEGADA rige x = 1
(x significa que la entrada
no se utiliza)

3.3.9.5. O-NEGADO (NOR)

La conexión en serie de varios contactos de apertura se representa así en el esquema de circuitos:

Símbolo en LOGO!



La salida O-NEGADO sólo está activada (estado 1) cuando están desactivadas las entradas(estado 0). Tan pronto como se active alguna de las entradas (estado 1), es desactivada la salida.

Este bloque se denomina O-NEGADO porque la salida Q de O-NEGADO sólo ocupa el estado 1 cuando todas las entradas tienen el estado 0. Tan pronto como alguna de las entradas ocupe el estado 1, la salida de O-NEGADO tiene el estado 0.

Tabla lógica para la función Y-NEGADA

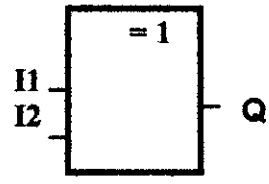
I1	I2	I3	Q
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Para O-NEGADO rige x = 0
(x significa que la entrada
no se utiliza)

3.3.9.6. O-EXCLUSIVO (XOR)

En el esquema de circuitos, un O EXCLUSIVO es una conexión en serie de 2 alternadores:

Símbolo en LOGO!:



La salida de O-EXCLUSIVO ocupa el estado 1 cuando las entradas tienen estados diferentes.

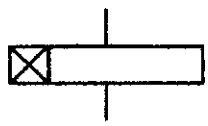
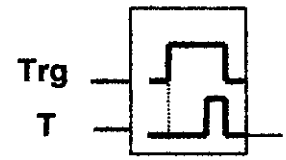
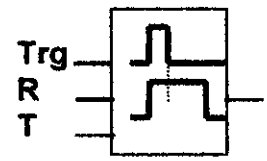
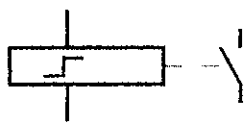
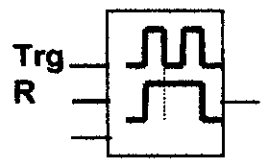
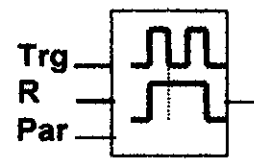
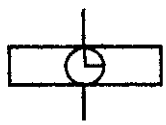
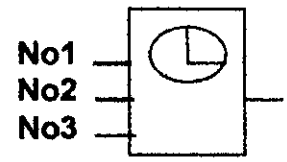
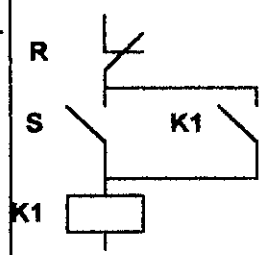
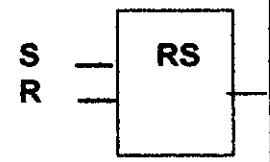
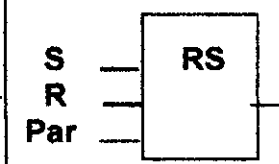
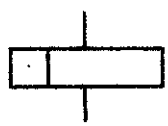
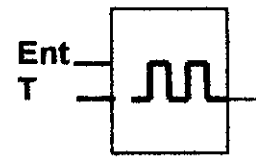
Tabla lógica para la función O-EXCLUSIVO

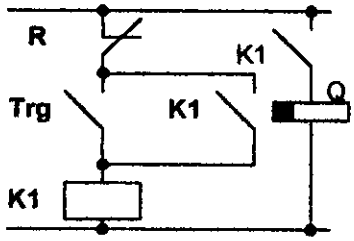
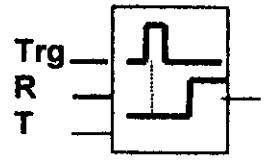
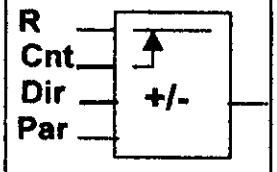
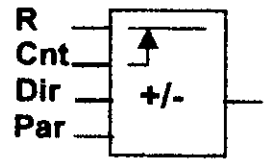
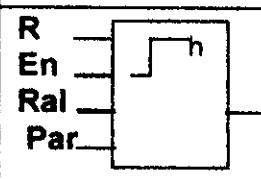
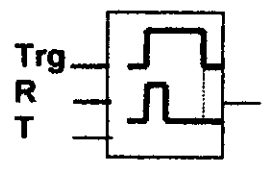
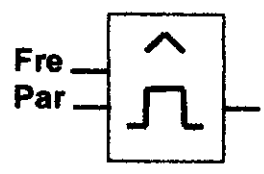
I1	I2	Q
0	0	0
0	1	1
1	0	1
1	1	0

Para O-EXCLUSIVO rige x = 0
(x significa que la entrada no se
utiliza)

3.3.10 Funciones especiales

En la lista SF se especifican los bloques de funciones especiales apra la introducción de un programa en LOGO!. Se prevén las siguientes funciones especiales:

Función	Representación en el esquema de circuitos	Representación en LOGO!...	Representación en LOGO!...L...	Re
Retardo De activación				
Retardo de desactivación				
Relé de Impulsos				Re
Reloj de temporización				
Relé disipador				Re
Salida de Impulsos				

Función	**Representación en el esquema de circuitos**	**Representación en LOGO!...**	**Representación en LOGO!...L...**	**Re**
Retardo de activación memorizable				
Contador adelante/ atrás				Re
Contador de horas de servicio				Re
Relé disipador/ Salida de Impulsos				
Interruptor de valor de umbral				Re

Re Este estado está almacenado de forma remanente a prueba de cortes de la red si hay enchufado un módulo para remanencia (sòlo para LOGO!...L...) y se ha definido la función como remanente.

R tiene prioridad ante las demás entradas para las funciones.

Remanencia

Para LOGO! en la versión estándar rige lo siguiente:

En **LOGO!...-L...** existe la posibilidad de mantener remanentes en algunas funciones los estados de conmutación, tiempos y valores de cómputo. A tal efecto.

- deben haberse definido como remanentes los valores correspondientes; y
- tiene que haber enchufado un módulo amarillo o rojo que permita la conservación de datos remanente.

Tras un corte de la red, el programa sigue funcionando con los valores actuales de la interrupción.

Tras un corte/reposición de la red, se repone el tiempo ya transcurrido en las funciones de temporización o el valor acumulado en el contador.

Si se opera tanto en LOGO! como sus entradas con las misma tensión de alimentación, podría almacenarse un valor erróneo para las funciones remanentes debido al puenteo del corte de la tensión. En las funciones especiales controladas por flancos, se podría generar entonces eventualmente un flanco adicional tras reponerse la red.

Por consiguiente, hay que cerciorarse de que LOGO! y sus entradas sean alimentados por separado.

Borne X en las entradas de las funciones especiales

Si se cablean con el borne x las entradas de funciones especiales, se prevé para las mismas el valor 0. Es decir, que dichas entradas llevan aplicada una señal low.

3.3.10.1. Precisión de los tiempos (todas las variantes) y del reloj de tempozización

Precisión de T

Todos los componentes electrónicos tienen ciertos márgenes de error. Por tal razón, podrían presentarse ligeras divergencias respecto al tiempo T ajustado. Para LOGO! la discrepancia es del 1% como máximo.

Ejemplo:

En 1 hora (3600 segundos) la discrepancia es de 1%, es decir, ± 36 segundos.

Por consiguiente, en 1 minuto la discrepancia es de sólo ± 0,6 segundos.

Precisión del reloj de temporización

A fin de que esta divergencia no afecte a la exactitud de marcha del reloj de temporización en las variantes C, es comparado éste regularmente con una base de tiempo muy exacta y reajustado. De esta forma, resulta una máxima discrepancia de marcha de ± 5 segundos/días.

3.3.10.2. Parámetro T

En algunas de las funciones especiales indicadas a continuación es posible parametrizar un valor de tiempo T. Para el preajuse de este tiempo téngase en cuenta:

Indicar siempre un tiempo T ≥ 1,10 s. Para T = 0,05 s y T = 0,00 s no está definido el tiempo T.

3.3.10.3. Retardo de activación

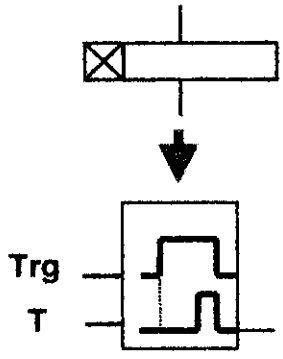
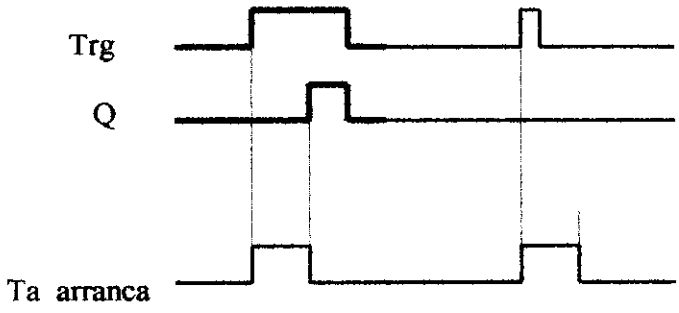
Esquema de circuitos/Símbolo en LOGO!	Cableado	Descripción
	Entrada Trg	A través de la entrada Trg (abreviatura de trigger) se indica el tiempo para el retardo de activación.
	Parámetro T	T es el tiempo tras el que debe activarse la salida (la señal de salida pasa de 0 a 1).
	Salida Q	Q se activa al transcurrir el tiempo T parametrizado, si está activada aún Trg.

Diagrama de temporización



El sector del diagrama de temporización representado en negrita aparece también en el símbolo para el retardo de activación.

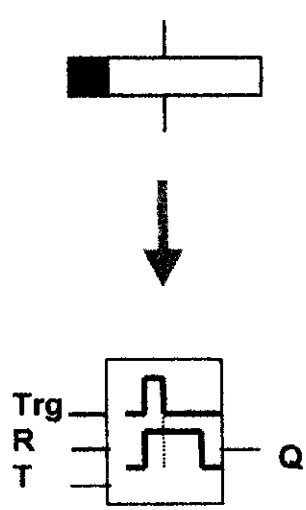
Al pasar de 0 a 1 el estado en la entrada Trg se indica el tiempo Ta (Ta es la hora actual en LOGO!). Si el estado de la entrada Trg permanece en 1 por lo menos mientras dure el tiempo parametrizado T, la salida es conmutada a 1 al terminar el tiempo T (la salida es activada posteriormente a la entrada). Si el estado en la entrada Trg pasa nuevamente a 0 antes de terminar el tiempo T, vuelve a reponerse el tiempo.

La salida se repone nuevamente a 0 si la entrada Trg se halla en el estado 0.

Aplicación

Supresión de rebotes en los interruptores

3.3.10.4. Retardo de desactivación

Esquema de ciruitos/Símbolo en LOGO!	Cableado	Descripción
	Entrada Trg	A través de la entrada Trg abreviatura de trigger) se inicia el tiempo para el retardo de desactivación
	Entrada R	A través de la entrada R (reset) se repone el tiempo para el retardo de desactivación y se conmuta la salida 0 (la reposición tiene prioridad ante Trg)
	Parámetro T	T es el tiempo tras el que debe desactivarse la salida (la señal de salida pasa de 1 a 0)
	Salida Q	Q se activa al trascurrir el tiempo T parametrizado, si está activada aún Trg.

3.3.10.5. Relé de impulsos

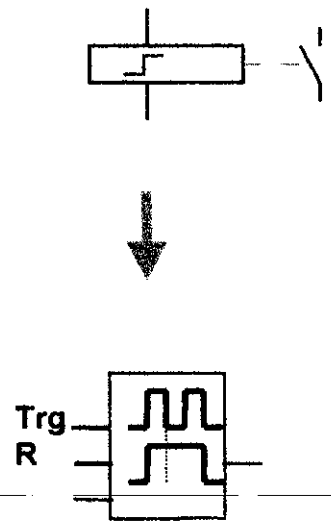
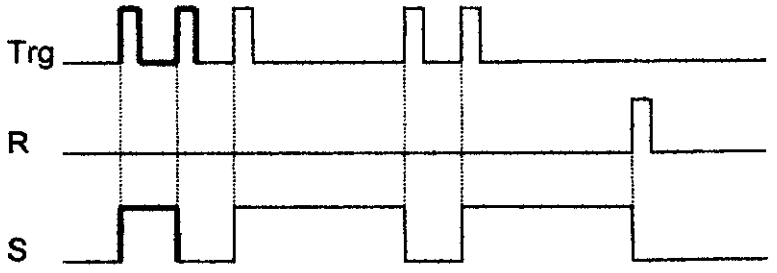
Esquema de Circuitos/ Simbolo en LOGO!	Cableado	Descripción
	Entrada Trg	A través de la entrada Trg (abreviatura de trigger) se activa y desactiva la salida Q
	Entrada R	A través de la entrada R (reset) se repone el tiempo y se conmuta la salida a 0 (la reposición tiene prioridad ante Trg)
	Parámetro Par	Par se prevé sólo en las variantes de LOGO!...-L... Este parámetro permite activar y desactivar la remanencia. Rem: off=sin remanencia on=estado almacenable con remanencia
	Salida Q	Q se activa con Trg y se desactiva con la próxima Trg.

Diagrama de temporización



El sector del diagrama de temporización representado en negrita aparece también en el símbolo para el relé de impulsos

Cada vez que se conmuta de 0 a 1 el estado en la entrada Trg, la salida Q cambia su estado, es decir, que es activada o desactivada.

A través de la entrada R se repone el relé de impulsos a su estado inicial, es decir, que la salida se conmuta a 0.

Comportamiento tras conexión de red

El comportamiento tras conexión de red depende del LOGO! utilizado:

LOGO! 230 RC, LOGO! LB11	**LOGO! LB11**
Tras la conexión de la red queda repuesto siempre el relé de impulsos y la salida Q siempre a 0.	Si no se ha para metrizado la remanencia, tras la conexión de red queda repuesto el relé de impulsos y la salida Q conmutada a 0.
	Si se ha parametrizado la remanencia, tras la conexión de la red queda ajustado el estado actual antes de la desactivación.

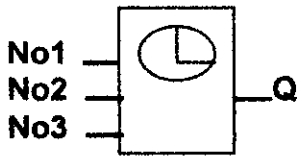
Aplicación

Alumbrado de pasillos

3.3.10.6. Reloj de temporización

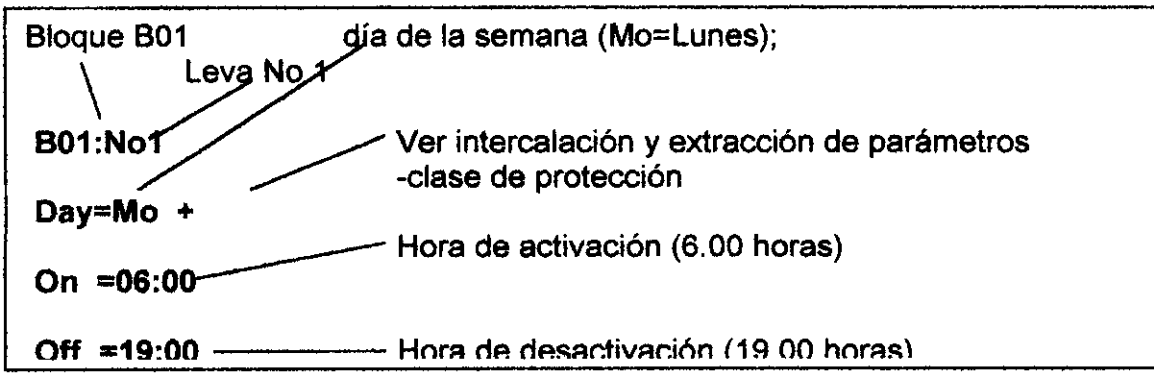
El reloj de temporización se prevé sólo en las variantes de LOGO! con la designación C (clock = reloj), por ejemplo LOGO! 230 RC.

Cada reloj de temporización cuenta con 3 levas.

Símbolo en LOGO!	Cableado	Descripción
	Parámetro No 1, No 2, No 3	A través de los parámetros 'No' se ajustan los tiempos de activación y desactivación para cada una de las levas del reloj de temporización.
	Salida Q	Q se activa si está activada una de las levas parametrizadas.

Parámetros "No1", "No2", "No3"

He aquí la ventana de parámetros para la leva No1:



Hora de desactivación

Cualquier hora entre 00:00 y 23:59

--:-- significa sin desactivación

Acumulación de la hora

En LOGO!...C sigue funcionando el reloj interno incluso si falla la tensión de red, es decir, que el reloj cuenta con una reserva de marcha. La duración de esta reseva en LOGO!...C depende de la temperatura ambiente. Para una temperatura ambiente de 25 grados, la reserva de marcha típica es de 80 horas.

Superposición de levas

A través de las levas se determinan las horas de activación y desactivación. A la hora de activación, el reloj conecta la salida si está no estuviera aún conectada. En una hora de desactivación, el reloj desconecta la salida si ésta no estuviera aún desconectada.

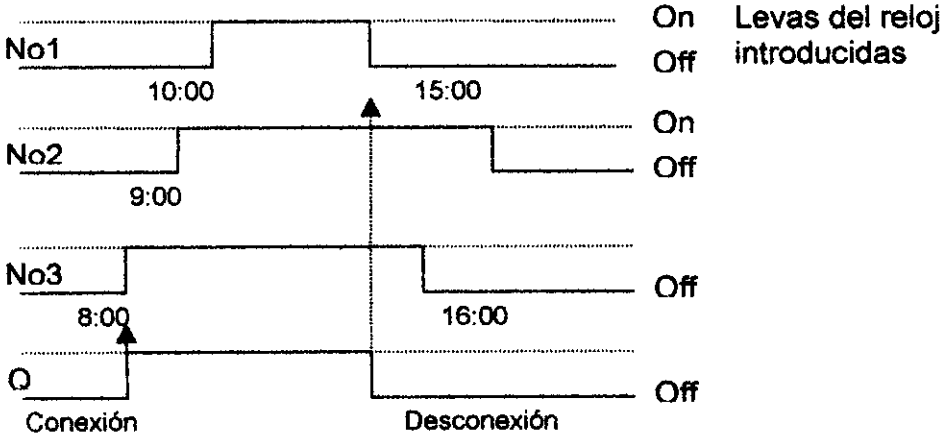


Figura 3-20

Prioridad en caso de horas de activación y desactivación idénticas

Si se indican para un reloj la activación y la desactivación a la misma hora, pero en levas diferentes, se produciría una contradicción. En tal caso, la leva "No3" tiene prioridad sobre la leva "No2" y está -a su vez- sobre la leva "No1".

3.3.10.7. Ajuste del reloj de temporización

Las horas de activación/desactivación se introducen de la siguiente manera:

1. Posicionar el cursor en uno de los parámetros **"No"** del reloj.
2. Pulsar la tecla OK. LOGO! abre la ventana de parámetros para esa leva. El cursor se halla sobre el día de la semana.
3. Mediante las teclas ▲ y ▼, elegir uno o varios días de la semana.
4. Mediante la tecla □ , llevar el cursor al primer dígito de la hora de activación.
5. Ajustar la hora de activación.

   Modificar el valor en la posición correctamente mediante las teclas ▲ y ▼. Desplazar el cursor entre las distintas posiciones mediante las teclas .□ y □

   El valor --:-- sólo puede ajustarse en la primera posición (--:-- significa sin activación/desactivación).
6. Mediante la tecla □ , llevar el cursor al primer dígito de la hora de desactivación.
7. Ajustar la hora de desactivación.
8. Terminar la introducción pulsando la tecla OK.

El cursor se halla en el parámetro No2 (leva 2). Ahora se puede parametrizar otra leva de la misma manera que los puntos anteriores.

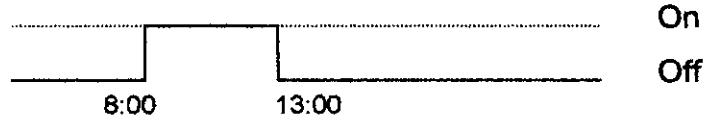
3.3.10.8. Reloj de temporización: Ejemplos

El reloj permite combinar discrecionalmente varias horas de activación/desactivación

Ejemplo 1

La salida de reloj debe estar activada cada día (es decir, desde el lunes al domingo) entre las 8:00 y las 13:00 horas:

B01: No1
Day=Mo..Su
On=08:00
Off=13:00

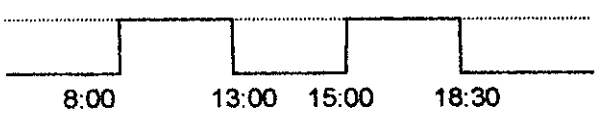


Ejemplo 2

La salida de reloj debe estar activada cada día de las 8:00 a las 13:00 horas y de las 15:00 a las 18:30 horas. A tal efecto se requieren 2 levas:

B01: No1
Day=Mo..Su
On=08:00
Off=13:00

B01: No2
Day=Mo..Su
On=15:00
Off=18:00

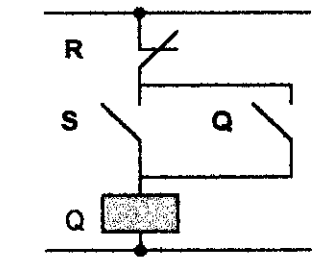
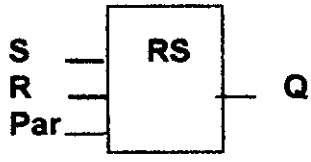


On=Con='1'=Tensión aplicada

Off=Desc='0'= Tensión no aplicada

3.3.10.9. Relé con autorretención

A menudo se necesita un circuito donde se mantenga un estado activado, a lo cual se denomina autorretención.

Esquema de Circuitos/Símbolo en LOGO!	Cableado	Descripción
	Entrada S	A través de la entrada S (set) se conmuta la salida Q a 1
	Entrada R	A través de la entrada R (reset) se repone la salida Q a 0. Si tanto S como R son 1, es repuesta la salida (la reposición tiene prioridad ante la activación).
	Parámetro Par	Par se prevé sólo en las variantes de LOGO!...-L... Este parámetro permite activar y desactivar la remanencia. Rem: off=sin remanencia on=estado almacenable con remanencia
	Salida Q	Q se activa y desactiva cíclicamente según el tiempo de cadencia T.

Función de conmutación

Un relé de autorretención es un sencillo elemento de memorización binario. El valor a la salida de los estados en las entradas y del estado anterior en la salida. En la tabla siguiente se expone su lógica:

Sn	Rn	Q	Explicación
0	0		Estado inalterado
0	1	0	Reposición
1	0	1	Activación
1	1	0	Reposición (la reposición tiene prioridad ante la activación).

3.3.10.10. Generador de impulsos simétrico

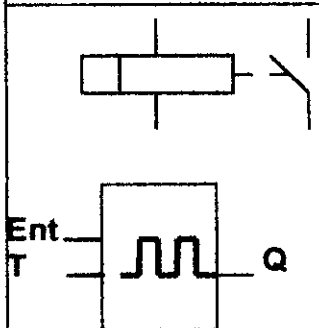
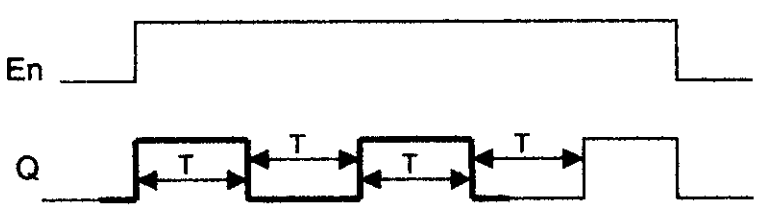
Esquema de circuitos/Símbolo en LOGO!	Cableado	Descripción
	Entrada En	A través de la entrada En (enable) es activado y desactivado el generador de impulsos
	Parámetro T	T es el tiempo durante el que está activada o desactivada la salida.
	Salida Q	Q se activa y desactiva cíclicamente según el tiempo de cadencia T.

Diagrama de temporización



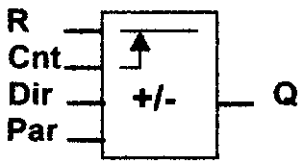
El sector del diagrama de temporización, representado en negrita, aparece también en el símbolo para el generador de impulsos simétrico

A través del parámetro T se indica la duración del tiempo de activación y de desactivación. A través de la entrada En (enable = liberación) es activado el generador de impulsos, es decir, que éste conmuta la salida a 1 durante el tiempo T, a continuación la salida a 0 durante el tiempo T, y así sucesivamente, hasta que la entrada En lleva 0.

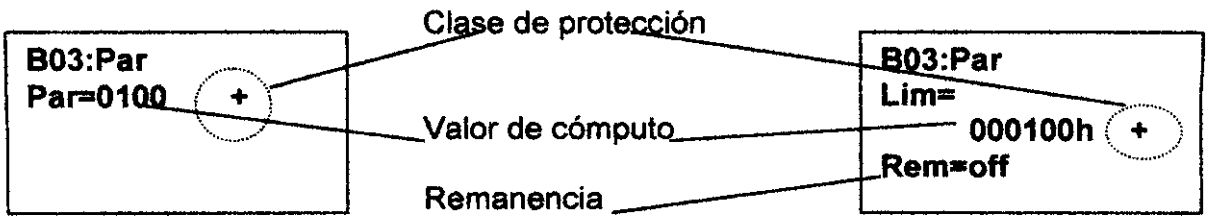
Cuando el estado de la entrada Trg pasa de) a 1, se inicia el tiempo actual Ta. Al alcanzar Ta el tiempo T, se conmuta a 1 la salida Q. Si se conmuta de nuevo la entrada Trg, no se altera Ta.

La salida y el tiempo Ta no se repone nuevamente a 0 hasta que la entrada R presente el estado 1.

3.3.10.12. Contador adelante/atrás

Símbolo en LOGO!	Cableado	Descripción
	Entrada R	A través de la entrada R (reset) se reponen a cero el valor de cómputo interno y la salida (la reposición tiene prioridad ante Cnt).
	Entrada Cnt	El contador cuenta los cambios del estado 0 al estado 1 registrados en la entrada Cnt (count = cómputo). So se cuentan los cambios de estado 1 al 0. Máxima frecuencia de cómputo en los bornes de entrada: 5 Hz.
	Entrada Dir	A través de la entrada Dir (dirección) se incia el sentido decómputo: Dir = 0: Cómputo progresivo Dir = 1: Cómputo regresivo
	Parámetro Par	Téngase en cuenta lo indicado sobre el parámetro preajustado Par a continuación de esta tabla.
	Salida Q	Q se activa al alcanzarse el valor de cómputo (parámetro Par o Lim).

Parámetro preajustado Par



Si el valor de cómputo interno es igual o mayor que Par (parámetro) o Lim, es activada la salida. Si se rebasa este valor por defecto o por exceso, es detenido el contador.

Par puede estar comprendido ente 0 y 9.999.

Lim puede estar comprendido entre 0 y 999.999.

Rem: En caso de LOGO!...L..., este parámetro permite activar y desactivar la remanencia para el valor de cómputo interno Cnt.

off = sin remanencia

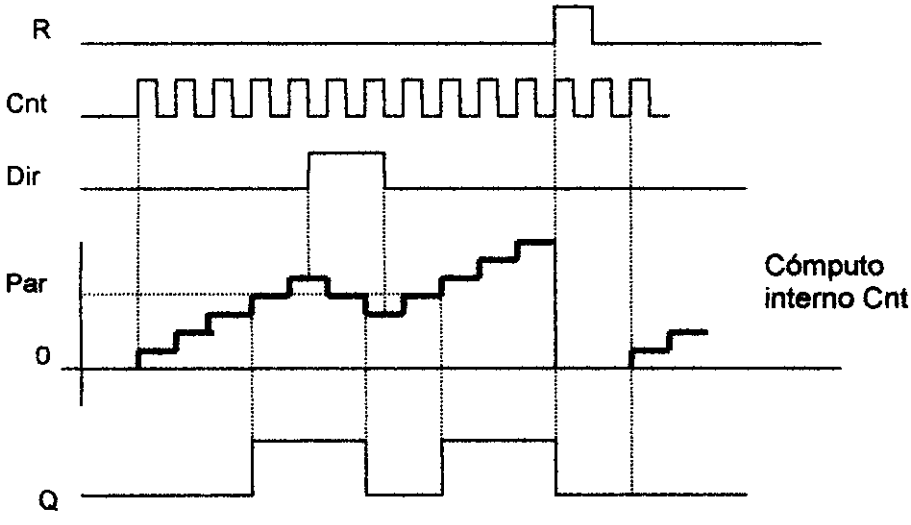
On = valor de cómputo Cnt almacenable con remanencia

Clase de protección:

+: Parámetro Par o Lim modificable durante el servicio

-: Parámetro Par o Lim modificable sólo en este punto durante la programación; no es posible modificarlo durante el servicio.

Diagrama de temporización



Durante cada flanco positivo en la entrada Cnt. el contador interno es incrementado en uni (Dir=0) o decrementado en un (Dir=1). Si el valor de cómputo interno es igual o mayor que el valor determinado mediante Par, se conmuta la salida Q a 1. A través de la entrada de reposición R es posible reponer a '0000' ó '000000' el valor de cómputo interno. Mientras R=1, la salida se halla en 0.

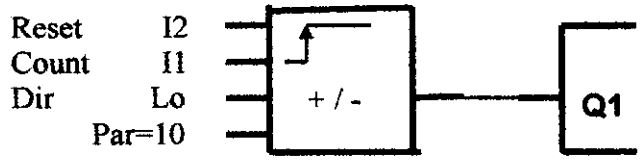
Remanencia

En LOGO!...L... existe la posibilidad de mantener remanentes en algunas funciones los estados de conmutación, tiempos y valores de cómputo. A tal efecto,

- deben haberse definido como remanentes los valores correspondientes; y,
- tienen que haber enchufado un módulo amarillo o rojo que permita la conservación de datos remanente.

Tras un corte de la red, el programa sigue funcionando con los valores actuales antes de la interrupción.

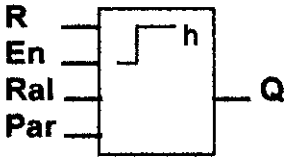
Ejemplo:



Cada vez que I1 ocupa el estado 1, es incrementado en 1 el valor de cómputo interno. Tan pronto como el valor de cómputo interno Cnt alcanza el valor 10 ajustado mediante Par, se conmuta a 1 la salida del contador.

3.3.10.13. Contador de horas de servicio

Esta función sólo está disponible en las variantes de LOGO!...L...

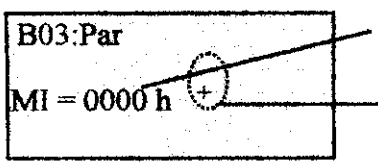
Símbolo en LOGO!	Cableado	Descripción
	Entrada R	**R = 0:** cómputo posible si Ral no = 1 **R = 1:** contador detenido A través de la entrada R (reset) se repone la salida. El tiempo restante MN del intervalo de mantenimiento es ajustado al valor MN = MI..
	Entrada En	En es la entrada de supervisión. LOGO! Mide el tiempo que está activada dicha entrada.
	Entrada Ral	**Ral = 0:** cómputo posible si R no=1 **Ral = 1:** contador detenido. A través de la entrada Ral (reset all) se reponen el contador y la salida. es decir, que sucede lo siguiente: • salida Q = 0, • horas de servicio medidas OT = 0 y tiempo restante del intervalo de mantenimiento MN = MI.
	Parámetro Par: MI	**MI:** intervalo de mantenimiento preajustable en la unida horas.
	Salida Q	Si el tiempo restante MN = 0 (ver el diagrama de temporización), es activada la salida.

MI = Valor de cómputo parametrizable

MN = Tiempo restante

OT = Tiempo total trancurrido desde la última señal 1 en la entrada Ral.

Parámetro preajustado Par

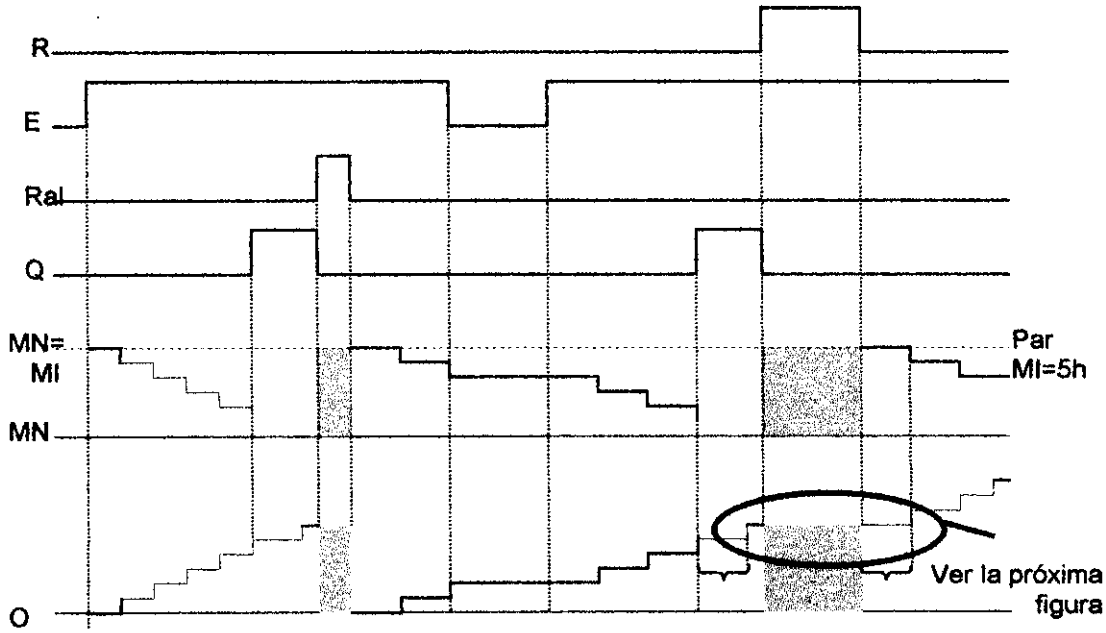


MI es el intervalo de tiempo parametrizable, que puede estar comprendido entre 0 y 9.999.

Clase de protección:

+:	Tiempo de supervisión preajustable modificable durante el servicio
-:	Tiempo de supervisión preajustable modificable sólo en este punto durante la programación; no es posible modificarlo durante el servicio..

Diagrama de temporización



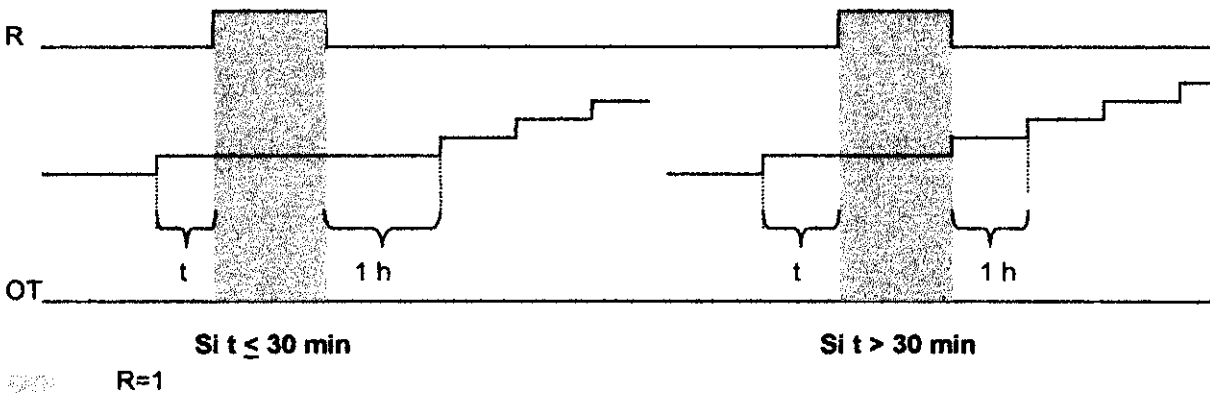
El contador no sigue contando mientras este activada R o Ral

MI = Intervalo de tiempo parametrizable

MN = Tiempo restante

OT = Tiempo total transcurrido desde la última señal 1 en la entrada Ral

Comportamiento tras la restitución de R



El contador de horas de servicio supervisa la entrada En. Mientras dicha entrada lleve aplicado el valor 1, LOGO! determina el tiempo transcurrido y el tiempo restante. LOGO! visualiza estos tiempos en la clase de servicio "Parametrización". Si el tiempo restante es igual a cero, se conmuta a 1 la salida Q.

Acciones mediante la entrada de reposición R	**Acciones mediante la entrada de reposición Ral**
Se repone la salida Q Se ajusta el contador del tiempo restante al valor preajustado MI	Se repone la salida Q Se ajusta el contador del tiempo restante al valor preajustado MI Se repone a 0 el contador interno OT
El contador interno OT permanece inalterado	

Valor límite par OT

Si se repone el contador de horas de servicio mediante la señal R, se conservan en OT las horas de servicio acumuladas. El valor límite para el contador OT es de 99.999 horas.

Cuando el contador de horas de servicio alcanza este valor, no sigue contando las horas.

Remanencia

En LOGO!...L... está ajustado siempre a remanencia el valor de cómputo interno. Para poder aprovechar esta remanencia, debe haber enchufado un módulo amarillo o rojo.

En caso de un corte de red, el programa sigue funcionando con los valores actuales de la interrupción.

Aplicación

Supervisión de intervalos de mantenimiento

3.3.10.14. Relé disipador - Salida de impulsos

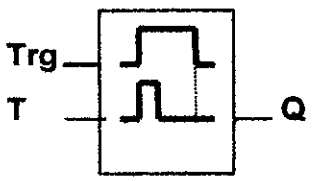
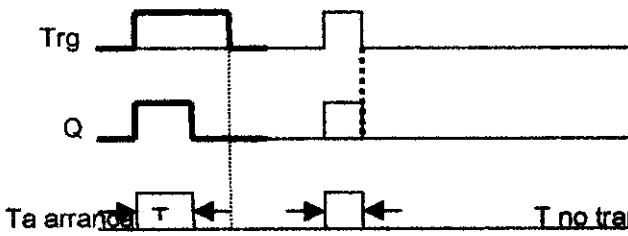
Símbolo en LOGO!	Cableado	Descripción
	Entrada Trg	A través de la entrada Trg (abreviatura de trigger) se inicia el tiempo para el relé disipador.
	Parámetro T	T es el tiempo tras el que se desactiva la salida (la señal de salida pasa de 0 a 1).
	Salida Q	Q se activa con Trg y permanece así hasta que transcurre el tiempo T

Diagrama de temporización



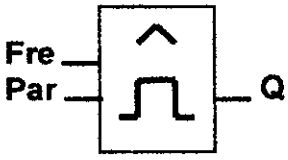
El sector del diagrama de temporización representado en negrita, aparece también en el símbolo del relé disipador.

Cuando la entrada Trg ocupa el estado 1, la salida Q se conmuta inmediatamente a 1. A la vez se inicia en LOGO! el tiempo actual Ta y la salida permanece activada. Cuando Ta alcanza el valor ajustado a través de T (Ta=T), es repuesta la salida Q al estado 0 (salida de impulsos).

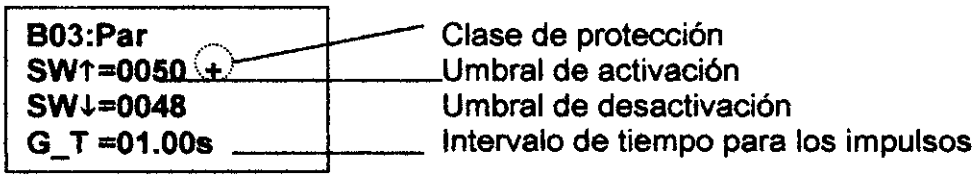
Si la entrada Trg se conmuta de 1 a 0 antes de transcurrir el tiempo preajustado, la salida pasa también inmediatamente de 1 a 0.

3.3.10.15. Conmutador de valor de umbral para frecuencias

Esta función sólo está disponible en las variantes de LOGO!...L...

Símbolo en LOGO!	Cableado	Descripción
	Entrada Fre	En Fre se aplica la entrada que suministra los impulsos a contar. Utilizar para ello • La entrada I12 para procesos de computo rápidos (entradas de 24 V): max. 150 Hz • Otra entrada o elemento de circuito cualquiera para frecuencias de computo más lentas.
	Parámetro Par: SW↑ SW↓ G_T	SW↑:Umbral de activación SW↓:Umbral de desactivación G_T:Intervalo de tiempo durante el que se miden los impulsos aplicados
	Salida Q	Q se activa o se desactiva en función de SW↑ y de SW↓ (ver descripción más abajo).

Parámetro preajustado Par



SW↑ es el umbral de activación; puede estar comprendido entre 0000 y 9999.

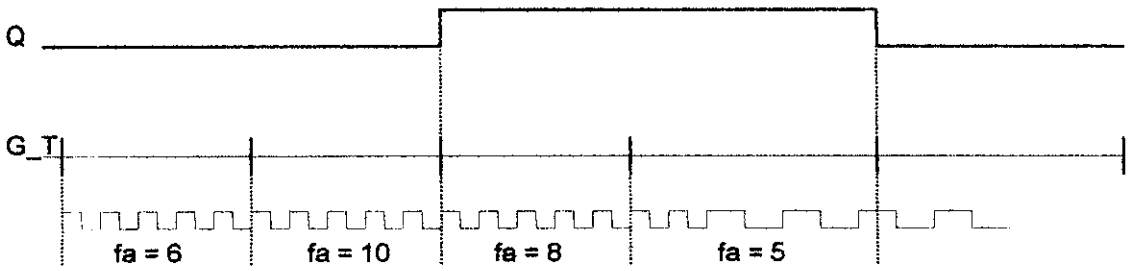
SW↓ es el umbral de desactivación; puede estar comprendido entre 0000 y 9999.

G_T es el intervalo de tiempo durante el que se miden los impulsos aplicados a Fre; puede estar comprendido entre 00.05 s y 99.95 s.

Clase de protección:

+:	Umbrales de conmutación preajustables modificables durante el servicio
-:	Umbrales de conmutación preajustables modificables solo en este punto durante la programación; no es posible modificarlos durante el servicio..

Diagrama de Temporización



Umbral de activación: SW↑ = 9 Umbral de desactivación: SW↓ = 5

El conmutador de valor de umbral mide las señales en la entrada Fre. Los impulsos son registrados a través de un intervalo de tiempo G_T parametrizable. Si durante el tiempo G_T los valores medidos son **mayores** que el umbral de activación y de desactivación, se activa la salida Q.

Q se desactiva nuevamente cuando la cantidad de impulsos medidos es **igual o menor** que el valor del umbral de desactivación.

Nota.

Si se ajusta previamente el tiempo G_T con 1 s, LOGO! Envía de vuelta en el parámetro fa la frecuencia actual en Hz.

fa es siempre la suma de los impulsos medidos por cada unidad de tiempo G_T.

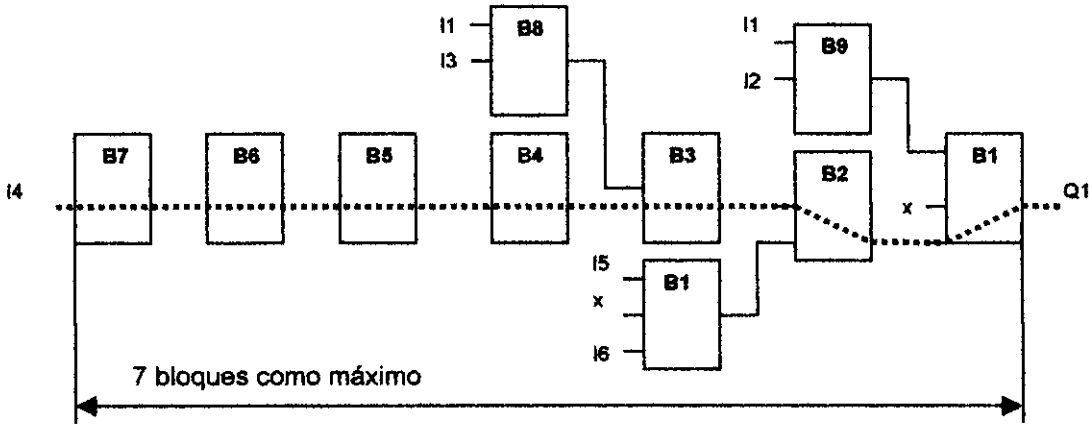
3.3.11. Capacidad de almacenamiento y magnitud de un circuito

Para un programa (o bien un esquema de circuitos) rigen determinadas limitaciones:

- Cantidad de bloques conectados en serie
- Capacidad de almacenamiento

Cantidad de bloques conectados en serie

Entre una salida y una entrada es posible prever hasta **7 bloques** en serie:



Capacidad de Almacenamiento

LOGO! supervisa el aprovechamiento de la memoria, ofreciendo en las listas de funciones solo aquellas funciones para las que aún baste efectivamente la capacidad de memoria.

A continuación se exponen a este respecto únicamente dos condiciones marginales que deberían tenerse presentes:

- En un programa son posibles **30 bloques** como máximo.

  Recordemos que un bloque puede ser tanto una sencilla función AND como una función especial compleja (p. Ej. Un contador de horas de servicio).
- Se emplean varias funciones especiales, se reduce correspondientemente la cantidad de bloques posibles.

Para determinar exactamente la capacidad de memoria requerida, debe referirse específicamente el tipo de dispositivo a emplear.

3.4. Parametrización de LOGO!

Se entiende por parametrización, al ajuste de los parámetros para bloques. Es posible ajustar tiempos de retardo en funciones cronológicas, tiempos de activación para relojes de temporización, el valor de umbral para un contador, el intervalo de supervisión para un contador de horas de servicio y los umbrales de activación y desactivación para un conmutador de valor de umbral.

Los parámetros pueden ajustarse

- En el modo de servicio "Programación" o
- En el modo de servicio "Parametrización".

En el modo de servicio "Parametrización", el programador ajusta un valor para un parámetro. Se ha previsto la clase de servicio "Parametrización" para poder modificar parámetros sin tener que alterar el programa protegiendo de esta manera el programa y el circuito.

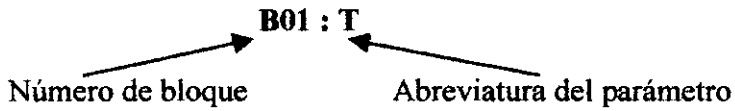
3.4.1. Parámetros

He aquí algunos parámetros:

- Tiempos de retardo de un relé temporizador
- Tiempos de activación (levas) de un reloj
- Valor de umbral para un contador
- Intervalo de supervisión para un contador de horas de servicio
- Umbrales de conmutación para un conmutador de valor de umbral

Cada parámetro se identifica mediante el numero de bloque y la abreviatura del parámetro.

Ejemplos:



B01:T	Tiempo de retardo ajustable en el bloque B01
B02:No1	Bloque B02 de un reloj de temporización. No1 es la primera leva de este reloj de temporización.
B03:Par	El bloque B03 es un contador. Par es el valor de computo del contador
B04:Par	El bloque B04 es un contador en LOGO!...-L... Par caracteriza a varios parámetros que pueden ser supervisados.
B05:Par	El bloque B05 es un contador de horas de servicio. Par caracteriza a varios parámetros que pueden ser supervisados.
B06:Par	El bloque B06 es un conmutador de valor de umbral. Par caracteriza a varios parámetros que pueden ser supervisados.

CAPITULO IV

ANALISIS DEL SISTEMA

4.1. ESTUDIO DE LA HERRAMIENTA DE PROGRAMACION

4.1.1. Introducción.

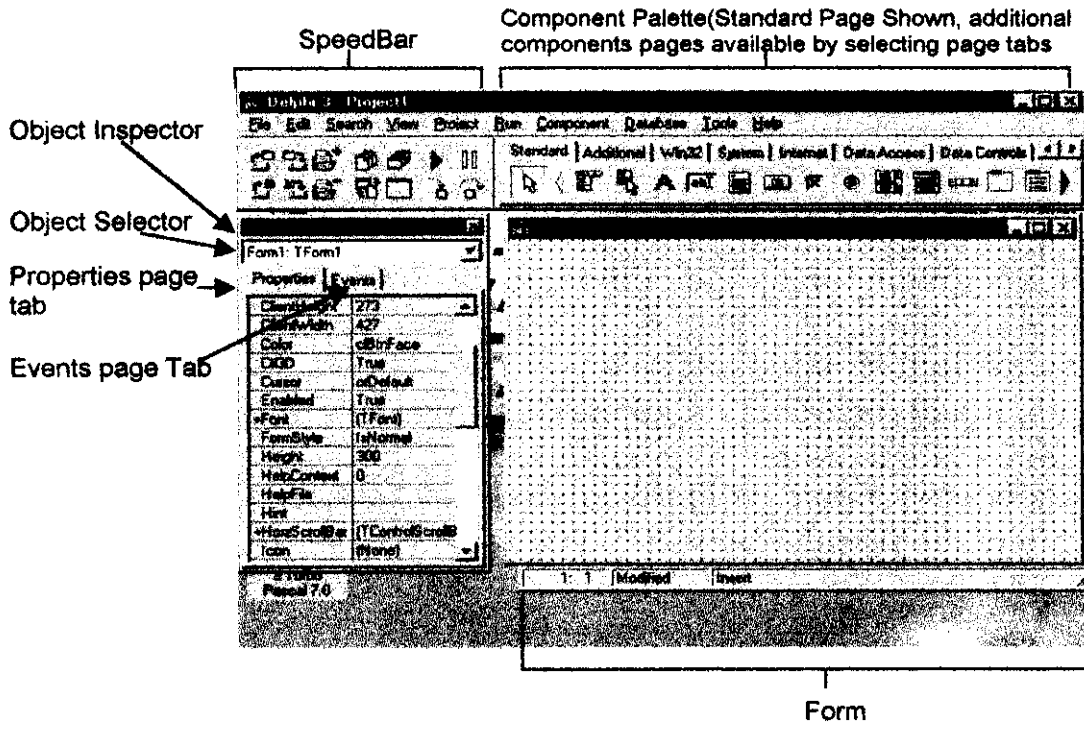
Para el desarrollo del Sistema Informático, se utilizará como herramienta de programación el ambiente de desarrollo Delphi versión 3.0. Esta herramienta fue seleccionada en base a las siguientes consideraciones:

- Constituye un poderoso ambiente de desarrollo de aplicaciones que combina un sistema de desarrollo visual y un lenguaje orientado a objetos, basado en componentes que permiten la elaboración rápida y eficiente de aplicaciones bajo Windows utilizando todas las bondades de este Sistema Operativo.

- Está desarrollado en base a su poderoso lenguaje Object Pascal, sucesor del tradicional Pascal, el mismo que constituye un lenguaje orientado a objetos capaz de soportar la herencia, encapsulamiento y polimorfismo.

- Además se puede destacar que el ambiente de programación DELPHI, contiene un conjunto de herramientas de diseño, y de base de datos con las que se puede diseñar y probar prototipos de aplicaciones así como también crear poderosas aplicaciones cliente/servidor y reportes.

4.1.2. Ambiente de trabajo de Delphi



Speed Buttons (botones de velocidad): Permite seleccionar opciones de menú directamente. Esto da mayor velocidad en el diseño de la aplicación.

Menu Bar (barra de menú): Es un menú estándar al estilo Windows con las opciones de ambiente Delphi

Component Palette (paleta de componentes): Contiene íconos que representan los componentes de la Biblioteca de componentes visuales que se utilizan en las aplicaciones.

Palette page tabs (tabuladores de la página de la paleta de componentes): Permiten accesar las diferentes categorías de componentes en la paleta de componentes.

Form (forma): En muchas aplicaciones, una forma es la representación visual de la ventana principal del programa. En Delphi una forma puede representar otras ventanas, como por ejemplo MDI (Multiple Ducument Interface). Una aplicación puede tener una o varias formas.

Object Inspector (Inspector de objetos): Muestra todas las propiedades y eventos de uno o más componentes seleccionados en la forma. Es una de las herramientas más importantes.

Unit windows (ventana de unidades): Esta ventana muestra el código en Pascal asociado a una unidad (unit). Cada forma utilizada tiene una unit asociada donde se programan los eventos para los cuales una aplicación dará una respuesta, para cada objeto o componente de la forma en cuestión.

Properties and events page tabs (tabuladores de páginas Propiedades y eventos): Permite escoger entes las páginas Propiedades y Eventos del Inspector de Objetos.

4.1.3. Ficheros de las aplicaciones en Delphi

Cuando se crea un aplicación en Delphi este crea un conjunto de ficheros, con determinadas extensiones:

Ficheros de código fuente

Ficheros de Unidades o Módulos

Ficheros de Proyecto.

Ficheros de código fuente:

Cuando se insertan componentes en una forma y cuando se modifican las propiedades y eventos de la forma y sus componentes Delphi escribe el texto de su programa en Object Pascal. Este texto es llamado Código Fuente. Hay que seguir los siguientes pasos para examinar el código fuente para un programa.

1. Seleccione File/Open Project para abrir un programa con extensión .dpr.

2. Seleccione en el menú View/Unit o hacer Click en el botón de velocidad Select Unit from list y escoja su programa, se mostrará el texto correspondiente en la ventana de unidades. Este texto está almacenado en un fichero de extensión .Pas.

3. Seleccione nuevamente View/Units y escoja el módulo (nombre programa), la ventana de unidades ahora muestra dos ficheros que pueden seleccionarse haciendo click en el tabulador de páginas de la ventana. El texto mostrado se corresponde con el fichero de extensión .dpr.

Ficheros de Unidades:

Una Unit es una unidad autocontenida de código que pude ser incorporada al programa principal a través de la cláusula USES.

Las Units son creadas automáticamente por Delphi cuando se crea una nueva forma y su estructura es la misma que la de las units tradicionales.

En la sección TYPE de la Unit Delphi incorpora el tipo TFrom1 que es un objeto de la clase TForm. La clase TForm1 hereda los miembros (datos y métodos) de su ancestro Tfrom. La clase TFrom1 adiciona dos miembros, uno es un objeto TButton llamado Cerrar y el otro es un procedimiento que manipula el evento OnClick del botón.

La directiva de compilación ($R *.DFM) es para abrir y leer ficheros con extensión .DFM que significa Delphi Form. El * (asterisco) significa que Delphi debe buscar el mismo nombre del proyecto pero con extensión .DFM. Este fichero contiene las propiedades de la forma.

Ficheros de Proyecto:

El programa con extensión .dpr, al igual que las units, es creado automáticamente por Delphi. El proyecto tiene dos propósitos:

- Declarar las units de la aplicación
- Ejecutar el programa

La estructura del fichero .DPR (Delphi Proyect) es la misma que la de un programa Pascal.

En la sección uses, se ponen en uso dos units: Forms, que provee capacidades para las formas y Principal, indicando que está en el fichero de extensión .Pas y se corresponde con la forma Form1.

La directiva {$R *.RES} permite ofrecer un fichero de recursos (binario) para el fichero ejecutable.exe.

La instrucción Application.CreateForm(Tform1, Form1); crea el objeto forma Form1 en memoria y la instrucción Application.Run; ejecuta la aplicación.

4.1.4 Extensiones de los ficheros de Delphi

- .~* (Por ejemplo Hola.~dp) son ficheros copias de resguardo. Este fichero puede ser borrado

- **.Exe** Fichero ejecutable de la aplicación. Es el fichero final para la distribución de la aplicación y puede ser borrado porque Delphi lo crea cada vez que se compila.

- **.Opt** Almacena opciones seleccionadas en Options/Proyect. Puede borrarse si no se están utilizando opciones no estándares.

- **.Pas** Contienen código en Pascal. Normalmente hay un .pas para cada forma en la aplicación, pero pueden crearse otros ficheros .pas por parte de programadores experimentados. No borrar estos ficheros mientras no se ha terminado con la aplicación.

- **.Res** Contiene recursos binarios como iconos, bitmaps, etc. Con el editor de imágenes en la opción Tools, se puede modificar este fichero. Nunca modifique el fichero de recursos del proyecto (nombre _del _proyecto.Res).

Nota: No borrar los ficheros .Dfm, .Dpr y .Pas a menos que quieran eliminarse del proyecto con pleno conocimiento de lo que se hace.

4.1.5. Categorías de los Componentes Visuales

La paleta de componentes de Delphi 3.0, se divide en las siguientes categorías:

- Estándar (Standard)
- Adicional (Additional)
- Windows 32 bits (Win32)
- Sistema (System)
- Internet (Internet)
- Acceso de datos (Data Access)
- Control de Datos (Data Controls)
- Reporte Rápido (Qreport)
- Diálogos (Dialogs)
- Windows3.1 (Win3.1)
- Ejemplos (Samples)
- ActivarX (ActiveX)

Cada categoría se corresponde con una página de la paleta de componentes.

Estándar (Standard)

Main Menu Diseña una barra de menú acompañada de menú Drop Down

PopupMenu Diseña pop-up local cuando se hace click con el botón derecho del mouse

Label — Crea una etiqueta para mostrar texto en forma de título que
usuario no puede accesar.

Edit — Muestra una área donde el usuario puede entrar o modificar ur
línea de texto simple.

Memo — Muestra un área donde el usuario puede entrar o modific
múltiples líneas de texto.

Button — Botón que puede ser utilizado para disparar una acción.

CheckBox — Caja de chequeo para opciones de tipo si/no.

RadioButton — Para presentar opciones mutuamente excluyentes. Se usa
generalmente con cajas de grupo.

ListBox — Lista donde pueden ser seleccionadas una o varias.

ComboBox — Combinación del Edit y ListBox para mostrar una lista
opciones.

ScrollBar — Para cambiar la porción visible de una lista o forma o pa
moverse a través de un rango por incremento.

GroupBox Para agrupar componentes relacionados en una forma.

RadioGroup Agrupa RadioButtons en una forma.

Panel Agrupa componentes como botones de velocidad (SpeedButtons) en una barra de herramientas.

Adicional (Additional)

BitBtn Botón que puede mostrar un bitmap, botón con acciones definidos.

SpeedButton Botón que puede ser adicionado a un panel para hacer una barra de velocidad.

MaskEdit Da un formato a los datos o limita la entrada de datos para validar.

StringGrid Forma de manipular cadenas en filas y columnas.

DrawGrid Para mostrar información no textual en columnas y filas.

Image Muestra bitmaps, iconos o metafiles. control para poner

Shape Para dibujar formas geométricas, objetos vectorizados.

Bevel Para dibujar bordes o líneas reisadas o incrustadas.

ScrollBox Área 'escroleable' más pequeña que la forma

CheckListBox Despliega una lista 'escroleable' con check boxes junto a cada ítem.

Splitter Divide el área de trabajo de una forma, dentro de zonas que pueden ser cambiadas de tamaño.

Static Text Es un control basado en ventanas que despliega texto sobre una forma

Chart Es el más importante componente de TeeChart se deriva de Tpanel y hereda todas su funcionalidad tiene nuevas capacidades gráficas y de dibujo.

Windows 32 bits (Win32)

TabControl Conjunto de tabulaciones ascendentes, empleadas para iniciar acciones; las tabulaciones se pueden ordenar en filas sencillas o múltiples.

PageControl — Conjunto de páginas usadas para incluir otros componentes; las tabulaciones se pueden ordenar en filas sencillas o múltiples.

ImageList — Retiene múltiples imágenes para emplearlas como iconos con los componentes de TreeView y ListView.

RichEdit — Acepta y presenta múltiples líneas de texto, y soporta propiedades de fuentes, formateo, impresión e inserción de objetos OLE.

TrackBar — Establece e indica los valores actuales en una escala de valores en una barra deslizable, horizontal o vertical.

ProgressBar — Indica el avance de operaciones prolongadas mediante la exhibición del porcentaje que se ha completado.

UpDown — Control de botón giratorio que se acciona en ambos extremos de valores permitidos; tiene una orientación vertical u horizontal y se puede relacionar con otro control

Timer Permite medir el tiempo de intervalos.

PaintBox Area rectángular para dibujar dentro de la forma.

FileListBox Para listar los ficheros del directorio actual.

DirectoryListBox Muestra los directorios en la torre actual.

DriveComboBox Muestra un combo box para mostrar el drive actual y permite cambiar de drive.

FilterComboBox Combo box para mostrar el fichero de filtro (*.txt) y se puede escoger en una lista de filtros.

MediaPlayer Muestra un panel de control estilo VCR para ejecutar multimedia (video y sonido).

OLEContainer Crea un área para cliente OLE (Object Linked and Embedding) en la forma, aplicación que trabaja cliente servidor.

DDEClientConv Establece una conexión cliente a una aplicación servidora DDE (Dynamic Data Exchange) intercambio dinámico de datos.

DDEClienItem Especifica el dato del cliente que será transferido durante una conversación DDE.

DDEServerConv Establece una conexión al servidor desde una aplicación cliente DDE.

DDEServerItem Especifica el dato del servidor que será transferido durante una conversación DDE.

Acceso de datos (Data Access)

DataSource Establece y mantiene una conexión entre Delphi y un servidor de Base de Datos remoto.

Table Enlaza una tabla de base de datos con una aplicación Delphi.

Query Construye y ejecuta una pregunta SQL a un servidor de base de datos local o remoto.

StoredProc Almacena localmente procedimientos SQL.

DataBase Enlace entre los componentes Table o Query y los componentes para los datos en la aplicación.

Controles de datos (Data Controls)

	DBGrid	Malla para mostrar los datos de una tabla.
	DBNavigator	Para navegar a través de los registros, mover adelante o atrás.
	DBText	Versión del Label para base de datos.
	DBEdit	Versión del Edit para base de datos. muestra información de un campo.
	DDBMemo	Versión del Memo para base de datos
	DBImage	Versión del Image para base de datos, almacena imagen una base de datos.
	BDDBListBox	Versión del ListBox para base de datos. desde una lista accesa a los elementos de una tabla.

Diálogos (Dialogs)

	OpenDialog	Muestra una caja de diálogo de Abrir típica de Windows.
	SaveDialog	Muestra una caja de diálogo de Salvar típica de Windows.
	FontDialog	Muestra una caja de diálogo de Letras típica de Windows.
	ColorDialog	Muestra una caja de diálogo de Color típica de Windows.
	PrintDialog	Muestra una caja de diálogo de Imprimir típica de Windows.
	PrinterSetupDialog	Muestra una caja de diálogo de Abrir típica de Windows.
	FindDialog	Muestra una caja de diálogo de Buscar típica de Windows.
	ReplaceDialog	Muestra una caja de diálogo de Reemplazar típica de Windows

Windows 3.1(Win3.1)

	DBLookupList	Presenta una lista desplazable de elementos desde una base de datos secundaria, basada en la relación con el campo en la base de datos principal.
	DBLookupCombo	Combina las capacidades de DBEdit y DBLookupList
	TabSet	Conjunto de tabulaciones descendentes empleadas para iniciar acciones (como la creación de índice de múltiples páginas), las tabulaciones se pueden ordenar en una sola fila.
	Outline	Presenta datos relacionados en una jerarquía.
	Header	Fila de secciones de tamaño variable para ajustar la presentación de datos en columnas.
	TabbedNotebook	Combinación de componentes de TabSet y Notebook.
	Notebook	Conjunto de páginas empleado para incluir otros componentes.

FileListBox Una list box especializada que lista todos los archivos en un directorio específico.

DriveComboBox Una combo box especializada que muestra todos los drives disponibles cuando se ejecuta una aplicación.

FilterCombobox Es una combo box especializada que presenta el usuario con una elección de filtros de archivo.

Ejemplos (Samples)

Gauge Muestra barra, texto o parte de formas geométricas como indicador progresivo.

ColorGrid Provee un selector de colores, paleta de colores.

SpinButton Permite incrementar o decrementar rápidamente un valor en un Edit.

SpinEdit Combinación de spin con edit.

DirectoryOuline Listado jerárquico de la estructura de directorios de drive actual.

Calendar Calendario.

ActivarX (ActiveX)

Chartfx — Herramienta de gráficos para crear y presentar gráficos de área, barras, acercamiento alto-bajo, líneas, Pareto, circulares, de puntos, de difusión y de líneas curvas, en segunda y tercera dimensiones.

VSSpell — Herramienta de verificación ortográfica. Incluye un diccionario con más de cien mil palabras, soporte para diccionarios personales y la capacidad de usar varios diccionarios al mismo tiempo.

F1Book — Herramienta de hoja de cálculo con 125 funciones integradas; las hojas pueden tener otros controles de Visual Components.

VtChart — Herramienta de gráficos que crea y presenta gráficos fotorrealistas de área, barras, combinadas, Gantt, acercamiento alto-bajo, líneas, circulares, polares, de difusión y escalonadas, en segunda y tercera dimensiones utilizando datos multidimensionales.

Graph — Herramienta de gráficos que crea y presenta gráficos de área, barras en segunda y tercera dimensiones, Gantt, acercamiento alto-bajo, línea, Log/Lin, circulares y polares

4.2. ANALISIS DEL SISTEMA ORIENTADO A OBJETOS

4.2.1. ESTUDIO PRELIMINAR

La automatización de sistemas en los últimos tiempos ha conocido un nuevo dispositivo de control: el autómata programable.

Sus excelentes prestaciones (flexibilidad, fácil programación, larga vida y reducido volúmen), han provocado la masiva implantación del mismo dentro de los procesos industriales.

Las autoridades educativas y formativas conscientes de este hecho, han introducido el autómata programable en los pénsums de estudio de múltiples profesiones técnicas, con el fin de estudiar el funcionamiento y simulación de estos importantes dispositivos electrónicos.

De esta manera y concibiendo el tema como una herramienta informática de apoyo para el diseño de automatísmos de control, es necesario destacar inicialmente, que el arte de diseñar se va complicando a medida que el nivel tecnológico crece; así mismo, las exigencias también son crecientes, como la competitividad del mercado, si se tiene en cuenta que el sector electrónico, específicamente el relacionado con la automatización industrial, es uno de los cuales actualmente está más presionado, se hace imprescindible la utilización de herramientas informáticas para el diseño de circuitos de control automático.

Intereses del Usuario:

- Sistema Informático flexible y Sencillo
- Sistema Informático de Fácil Adaptación
- Sistema con Interfaz Gráfica
- Sistema con entorno amigable
- Sistema Orientado a Objetos
- Sistema con programa de ayuda en línea
- Simulación gráfica de programas

Responsabilidades del Sistema:

- Programación gráfica de Autómata
- Depuración y compilación de Programas
- Edición de programas
- Grabación de Programas
- Simulación de Programas
- Modificar el estado de entradas y salidas de acuerdo a programación

4.2.2. ANALISIS

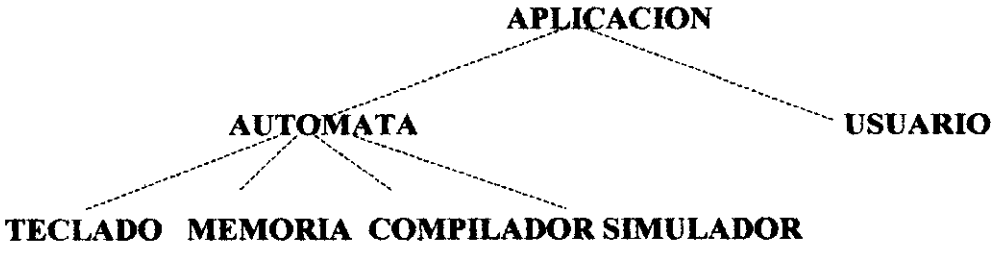
4.2.2.1. Definición Preliminar de las Clases:

- Autómata
- Simulador
- Display
- Impresora
- Disco
- Funciones Lógicas
- Funciones Especiales
- Entradas
- Salidas
- Teclado de Programación
- Usuario
- Programa
- Memoria

4.2.2.2. Definición de los Guiones

AGENTE	ACCION	RECEPTOR	RESULTADO
Aplicación	Inicializar Programa	Autómata	Inicializar Datos
Autómata	Cambiar a Listo	Consola de Programas	Estado Listo
Aplicación	Editar Programa	Usuario	Programa editado
Usuario	Digitar teclado	Teclado	Despliegue de datos
Teclado	Almacenar programa	Memoria	Programa en memoria
Memoria	Verificar Programa	Compilador	Programa Compilado
Usuario	Grabar Programa	Autómata	Programa Grabado
Autómata	Grabar Programa	Disco	Programa en Disco
Usuario	Simulación	Autómata	Simular Programa
Autómata	Simular Programa	Simulador	Programa Simulado
Simulador	Display resultados	Display	Programa en Display

4.2.2.3. Grafo de Control



4.2.2.4. Definición de Responsabilidades y Atributos de cada Clase.

CLASE	RESPONSABILIDAD	PROPIEDADES VISIBLES
Controlador de Clave	Controlar Claves de Acceso a Aplicación	Clave válida o no válida
Controlador de Datos	Validar ingreso de datos en Aplicación	Datos válidos o no válidos
Aplicación Autómata	Acceso al Sistema. Controlador de Datos Ingreso de datos por teclado. Grabar información. Simular programa de Autómata.	Datos correctos de acceso
Grabar.	Obtener respaldo de la información.	Disco, Diskette.
Simular	Ejecutar instrucciones de Programa de Autómata	Simulación paso a paso Simulación total
Teclado.	Ingresar Datos	Ayuda manejo de Aplicación. Ayuda ingreso de información.

CLASE	RESPONSABILIDAD	PROPIEDADES VISIBLES
Compilador	Programa en memoria Compilar posibles errores de lenguaje. Ejecutar líneas de instrucción.	Corrección de posibles errores
Memoria.	Almacenar instrucciones. Lenguaje Programación.	Ayuda para trabajo del Compilador
Impresora	Imprimir programas de Autómata	Imprimir Comentarios s/n

4.2.2.5. Documentación del Análisis.

DEFINICION DE LA CLASE:

Nombre: Aplicación Autómata

Responsabilidades:

- Acceso al sistema.
- Controlador de Datos.
- Ingreso por Teclado.
- Grabar Información.
- Simular Autómata.
- Imprimir programa

DEFINICION DE RELACIONES

CLASE	TIPO RELACION	RELACIONADO
Aplicación	Tener conocimiento	Lenguaje Programación
Autómata.	sobre programación	Aplicación.
	y funcionamiento de	
	un Autómata Programable.	

DEFINICION DE LA CLASE:

Nombre: Selector de Lenguaje

Responsabilidades:
- Seleccionar Lenguaje de Autómata.

DEFINICION DE LA CLASE:

Nombre: Teclado.

Responsabilidades:
- Control e ingreso de Datos de programación
- Elección de funciones lógicas o especiales
- Selección de nivel de edición o plano general

ATRIBUTOS:

CLASE	RESPONSABILIDAD	PROPIEDADES VISIBLES
Teclado.	Control y edición de Datos	Ayuda de Programación
	Control y edición de Funciones	
	Ayuda manejo Sistema.	
	Ayuda ingreso de	
	información.	

COLABORACIONES:

SERVICIOS SOLICITADOS	COLABORACION
Acceder datos del Sistema.	Equipo Computo.
Manejable.	

DEFINICION DE RELACIONES

CLASE	TIPO RELACION	RELACIONADO
Teclado	Depende de el equipo de cómputo en el que se va a desarrollar la aplicación.	Equipo Cómputo.

DEFINICION DE LA CLASE:

Nombre: Compilador

Responsabilidades:
- Compilar posibles errores de edición del programa.
- Ejecutar las líneas de instrucción.

ATRIBUTOS:

CLASE	RESPONSABILIDAD	PROPIEDADES VISIBLES
Compilador.	Compila posibles errores Ejecutar las líneas de instrucción.	Programa sin errores o programa con errores

COLABORACIONES:

SERVICIOS SOLICITADOS	COLABORACION
Ejecutar Programa.	Equipo Computo.
Compilar Líneas de instrucción.	Memoria Disponible.
Establecer listado de posibles errores	Lenguaje Programación.

DEFINICION DE RELACIONES

CLASE	TIPO RELACION	RELACIONADO
Compilador	Depende de la memoria que exista en el equipo de cómputo y las líneas de instrucción que se deba ejecutar.	Equipo Cómputo. Aplicación. Lenguaje Programación

DEFINICION DE LA CLASE:

Nombre: Memoria.

Responsabilidades:
- Almacenar programa.
- Interactuar con compilador

ATRIBUTOS:

CLASE	RESPONSABILIDAD	PROPIEDADES VISIBLES
Memoria.	Almacenar programa. Interactuar con compilador.	Ayuda Tipo Compilador.

COLABORACIONES:

SERVICIOS SOLICITADOS	COLABORACION
Espacio libre en disco.	Equipo Computo.
Disco en buen estado.	Aplicación.
Programa ya ejecutado.	

DEFINICION DE RELACIONES

CLASE	TIPO RELACION	RELACIONADO
Grabar.	Depende del tipo de información	Equipo Cómputo.
	que desee obtenerse	Aplicación.

4.2.3. DISEÑO DEL SISTEMA

4.2.3.1. Refinamiento del Análisis

COMPILADOR
TIPOS DE LENGUAJE ARCHIVOS DE INSTRUCCIONES Y COMANDOS
LEER PROGRAMA, VERIFICAR CODIGO DE CIRCUITO VISUALIZAR RESULTADOS

PROGRAMAR
NOMBRE DE LENGUAJE NUMERO DE ENTRADAS NUMERO DE SALIDAS FINALIZAR
ELEGIR PROGRAMAS INGRESAR PROGRAMA CONTROLAR INSTRUCCIONES FINALIZAR PROGRAMA

EDITAR
CAMPO TIPO DE DATO OPCION
LEER CAMPO ACTUALIZAR DATO BORRAR DATO INSERTAR DATO

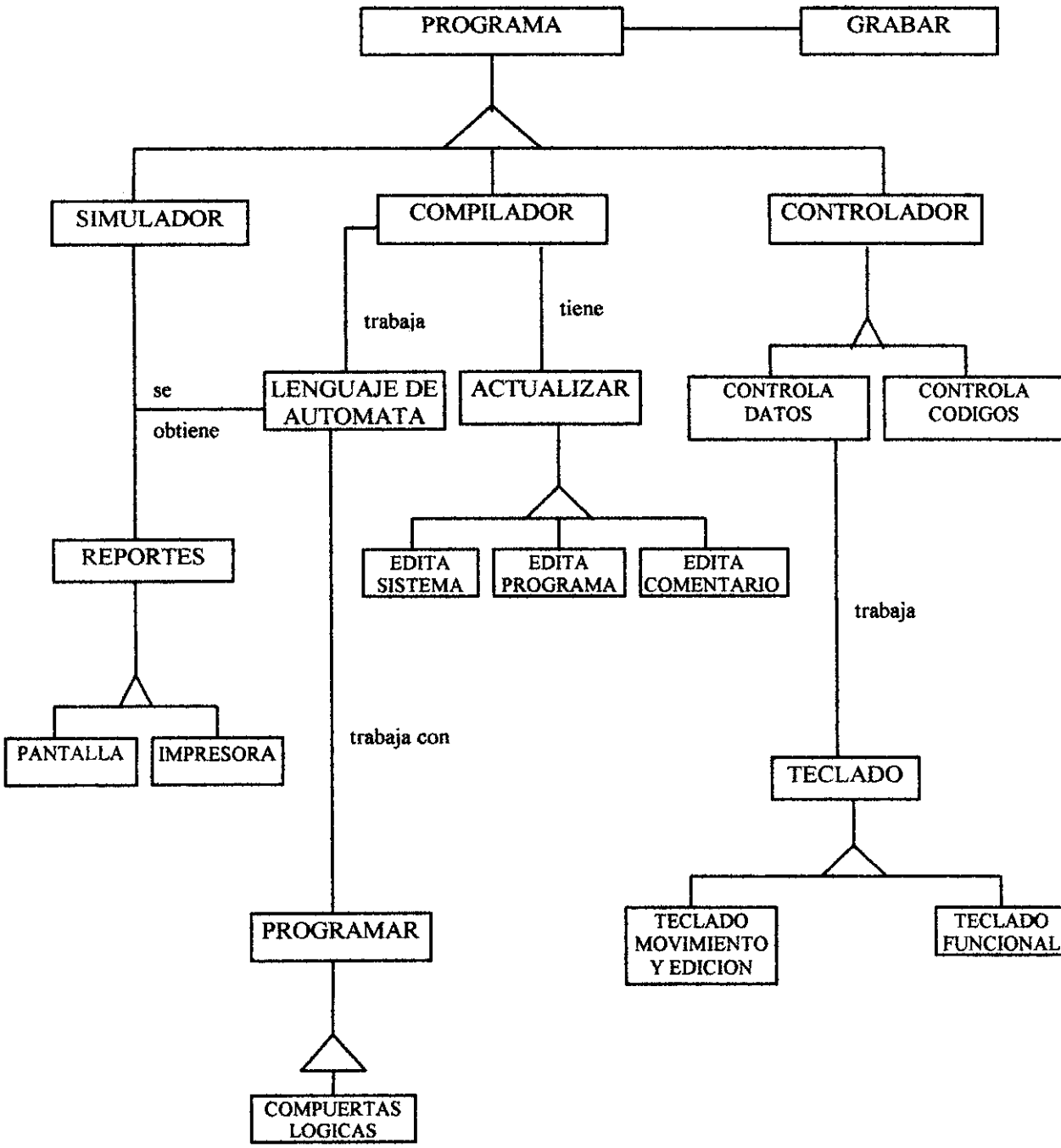
GRABAR
OPCION NOMBRE DEL ARCHIVO DESTINO
INGRESAR OPCION LEER NOMBRE ARCHIVO GRABAR ARCHIVO

ABRIR
OPCION NOMBRE DEL ARCHIVO RUTA
INGRESAR OPCION LEER NOMBRE ARCHIVO ABRIR ARCHIVO

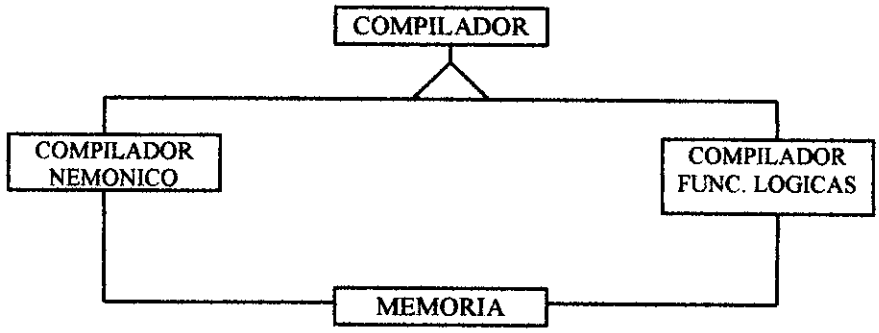
SIMULADOR
OPCION SIMULAR NOMBRE ARCHIVO
LEER ARCHIVO PROGRAMA EJECUTAR PROGRAMA VISUALIZAR EJECUCION GRAFICA

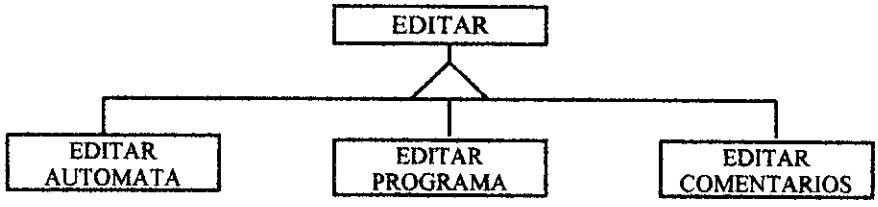
TECLADO
NOMBRE TECLA TIPO TECLA FUNCION
LEER TECLADO EJECUTAR FUNCION VISUALIZAR FUNCION

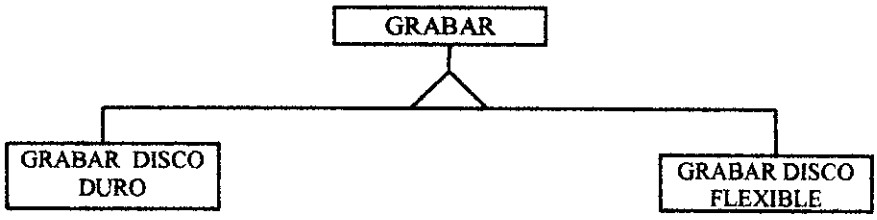
4.2.3.2. Diagrama de Clases

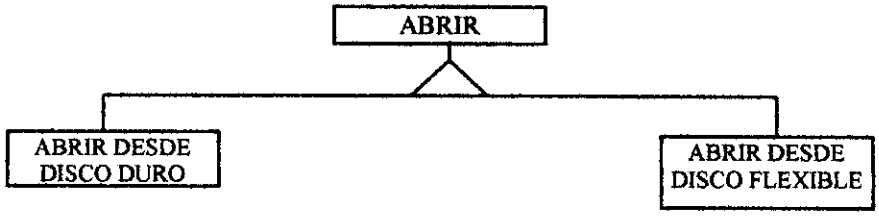


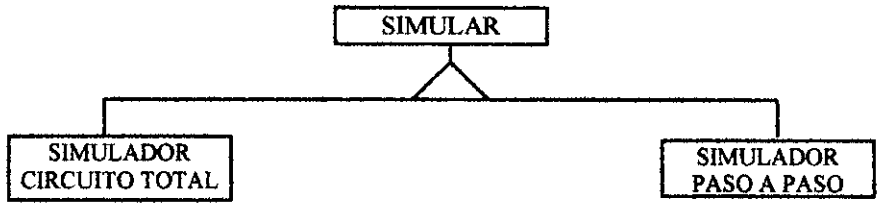
4.2.3.3. Refinamiento de las Jerarquías

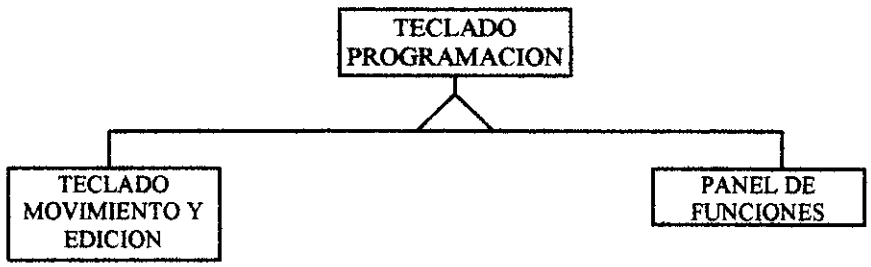


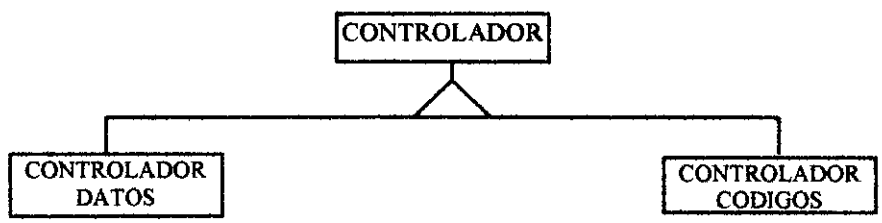






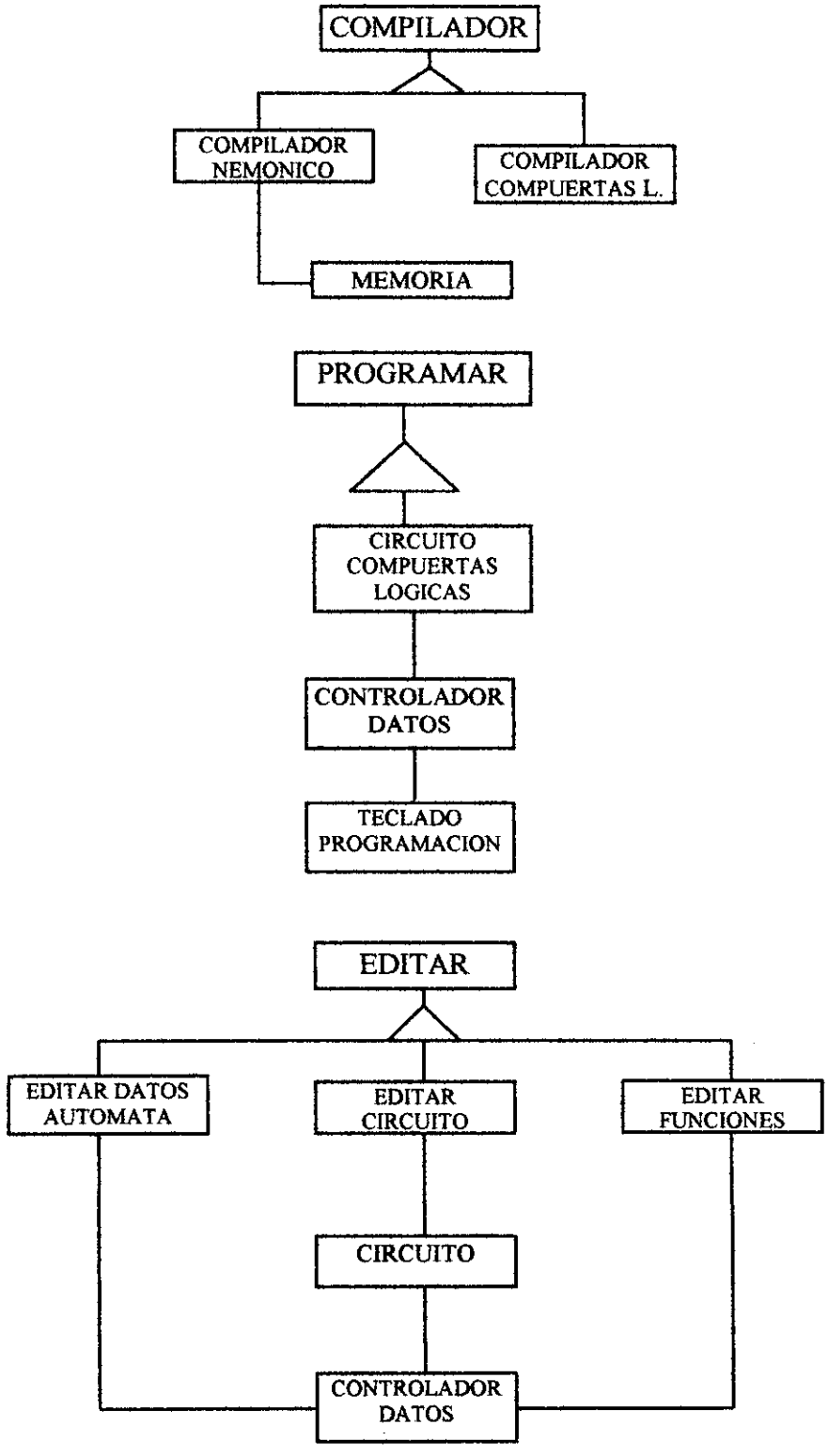

SIMULAR
SIMULADOR CIRCUITO TOTAL
SIMULADOR PASO A PASO

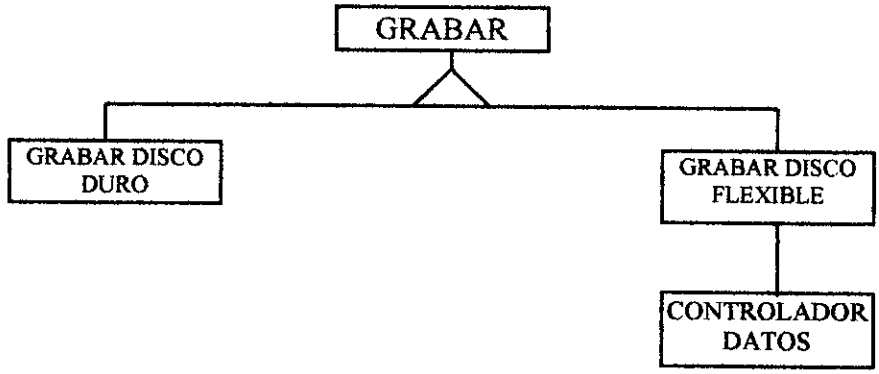

TECLADO PROGRAMACION
TECLADO MOVIMIENTO Y EDICION
PANEL DE FUNCIONES

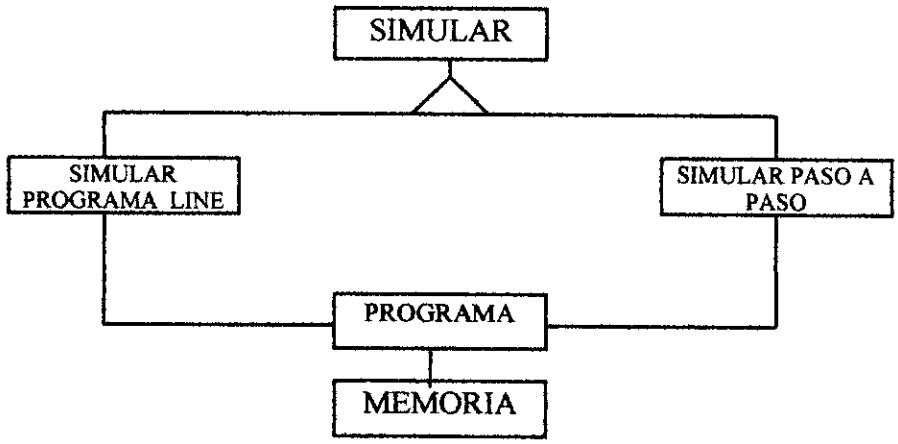

CONTROLADOR
CONTROLADOR DATOS
CONTROLADOR CODIGOS

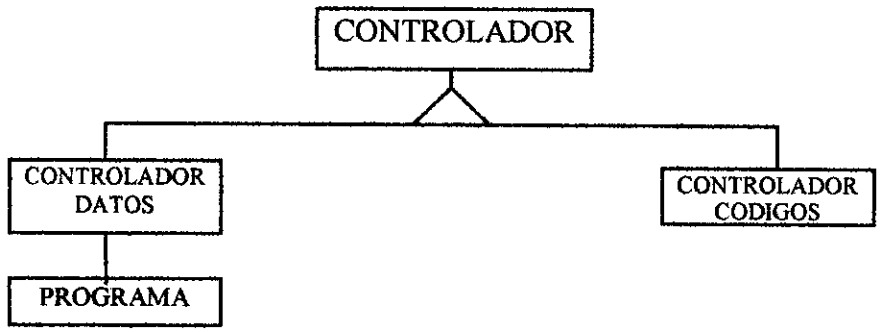

REPORTES
REPORTES PANTALLA
REPORTES IMPRESORA

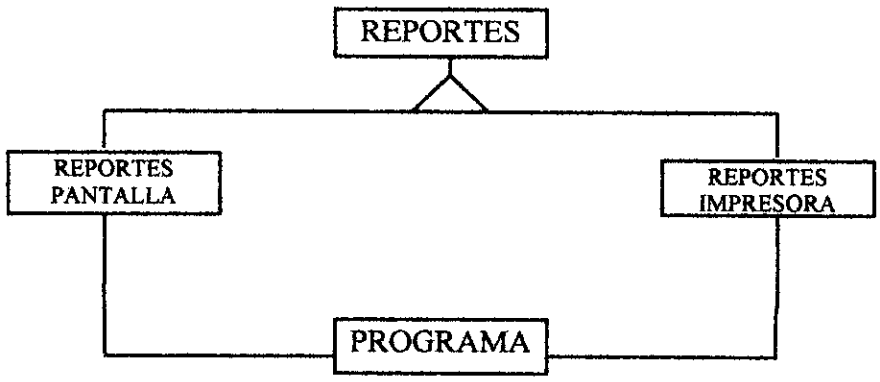
4.2.3.4. Refinamiento de las Colaboraciones



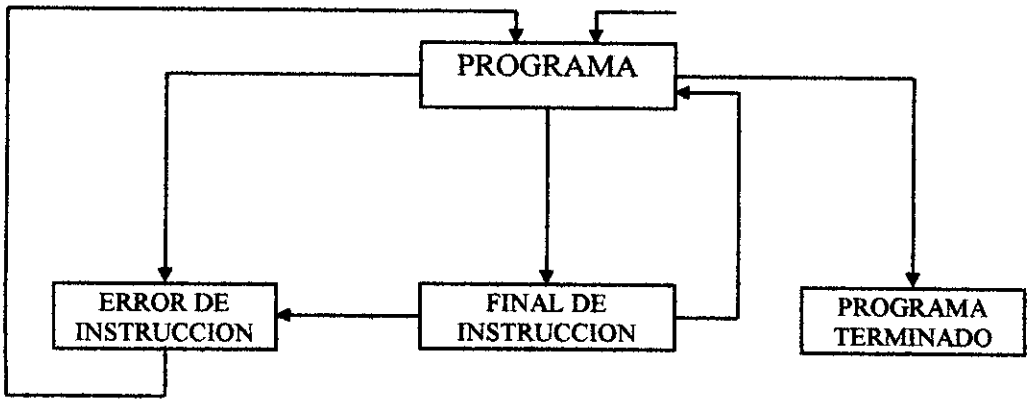

GRABAR
GRABAR DISCO DURO
GRABAR DISCO FLEXIBLE
CONTROLADOR DATOS


SIMULAR
SIMULAR PROGRAMA LINE
SIMULAR PASO A PASO
PROGRAMA
MEMORIA


CONTROLADOR
CONTROLADOR DATOS
CONTROLADOR CODIGOS
PROGRAMA


REPORTES
REPORTES PANTALLA
REPORTES IMPRESORA
PROGRAMA

4.2.3.5. Diagrama de Transición de Estados



NODO: PROGRAMAR

Alcanzado desde: Inicio para programar

Acción: mantiene el tema programar

Salida: tiene 3 salidas

1. error de alguna instrucción de programación
2. línea de instrucción o programación terminada
3. programación terminada, alcanza posición del tema instrucción.

NODO: ERROR EN INSTRUCCION

Alcanzado desde: final de instrucción o programar por ocurrir error.

Acción: emite tipo de error que se ocasiona en la instrucción.

Salida: pasa a programa por error en la instrucción.

NODO: FINAL DE INSTRUCCION.

Alcanzado desde: programa

Acción: programar líneas de instrucciones

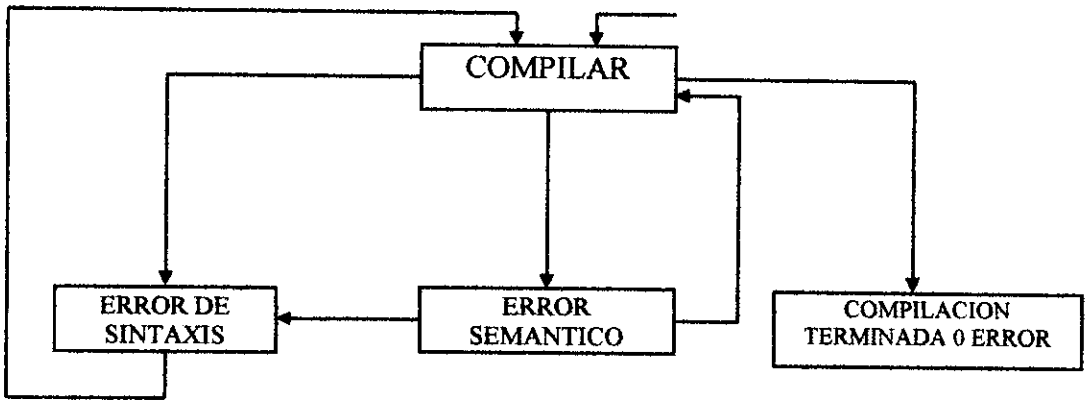
Salida: paso a programar al termino de finalizar líneas de instrucción.

NODO: PROGRAMA TERMINADO

Alcanzado desde: programa al alcanzar ultima línea de instrucción.

Acción: programa terminado libera un programa

Salida: es el estado final



NODO: COMPILAR

Alcanzado desde: inicio compilación

Acción: mantiene el tema compilar

Salida: tiene 3 salidas

1. error de sintaxis
2. línea semántico
3. compilación terminada.

NODO: ERROR DE SINTAXIS

Alcanzado desde: error semántico o compilación por ocurrir error.

Acción: emite tipo de error de sintaxis.

Salida: pasa a compilar por error de sintaxis.

NODO: ERROR SEMANTICO.

Alcanzado desde: compilar

Acción: compilar programa

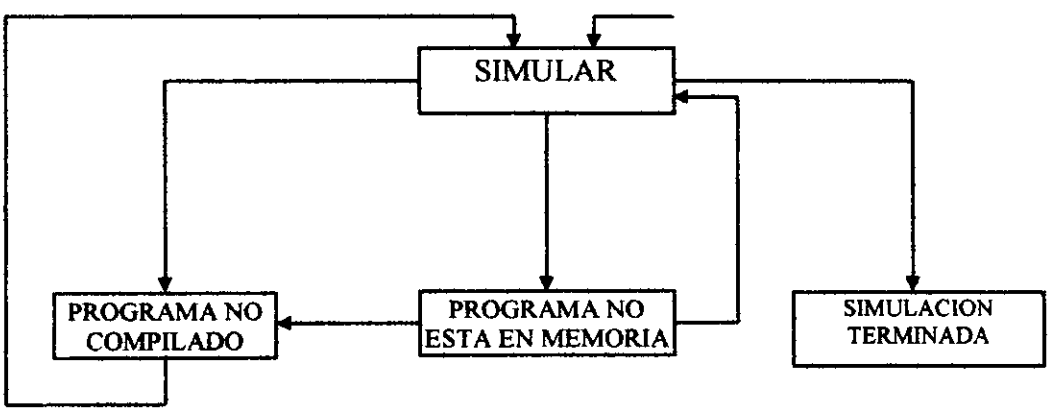
Salida: paso a compilar al termino de finalizar compilación programa.

NODO: PROGRAMA COMPILADO

Alcanzado desde: programa al alcanzar ultima línea de compilación.

Acción: programa compilado leyera un programa para ejecución

Salida: es el estado final



NODO: SIMULAR

Alcanzado desde: inicio simulación

Acción: mantiene el tema simular

Salida: tiene 3 salidas

1. programa no compilado
2. programa no esta en memoria
3. simulación terminada.

NODO: PROGRAMA NO COMPILADO

Alcanzado desde: programa no esta en memoria.

Acción: emite tipo de error por no existir programa

Salida: pasa a simular por no haber programa.

NODO: PROGRAMA NO ESTA EN MEMORIA

Alcanzado desde: simular

Acción: simular programa

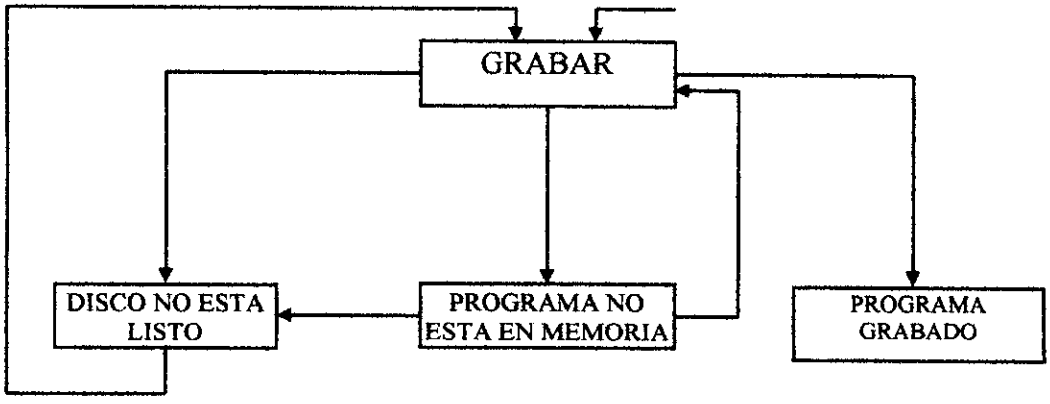
Salida: paso a simular al estar programa en memoria.

NODO: SIMULACION TERMINADA

Alcanzado desde: programa en memoria.

Acción: programa simulado

Salida: es el estado final



NODO: GRABAR

Alcanzado desde: inicio grabado

Acción: mantiene el tema grabar

Salida: tiene 3 salidas

1. disco no esta listo
2. programa no esta en memoria
3. programa grabado.

NODO: DISCO NO ESTA LISTO

Alcanzado desde: programa no esta en memoria.

Acción: emite tipo de error por no existir disco

Salida: pasa a grabar por no haber disco.

NODO: PROGRAMA NO ESTA EN MEMORIA

Alcanzado desde: grabar

Acción: grabar programa

Salida: paso a grabar al estar programa en memoria.

NODO: PROGRAMA GRABADO

Alcanzado desde: programa en memoria.

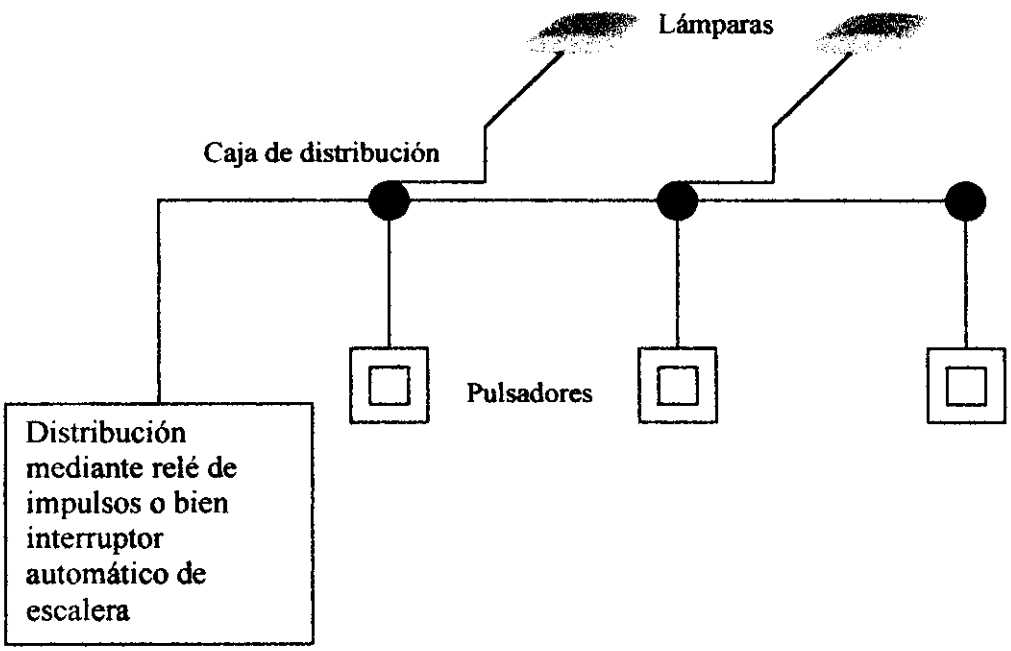
Acción: programa grabado

Salida: es el estado final

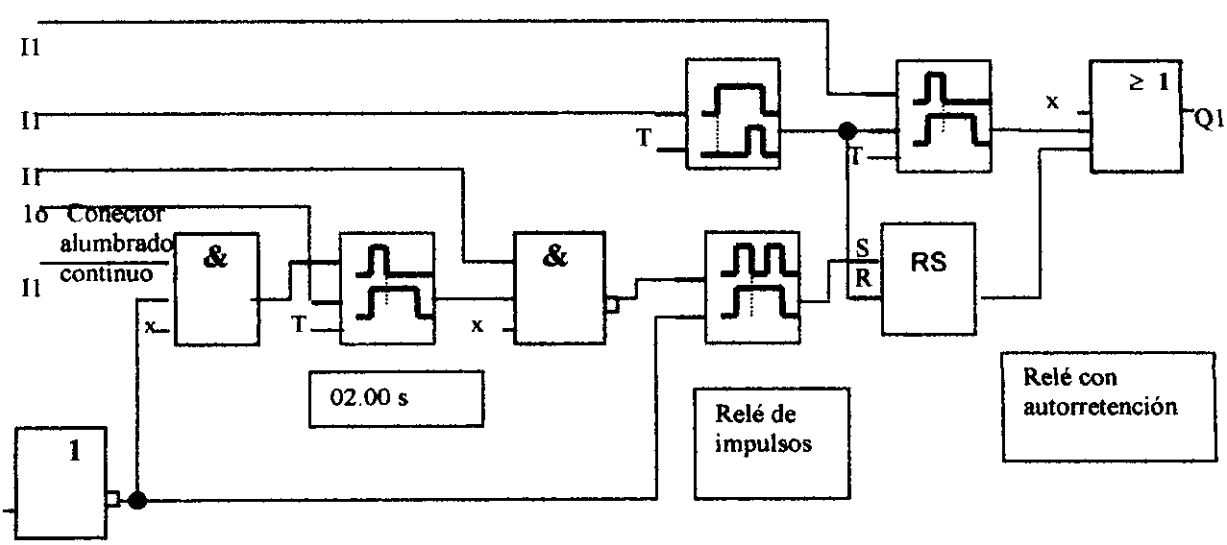
ANEXOS

ANEXOS

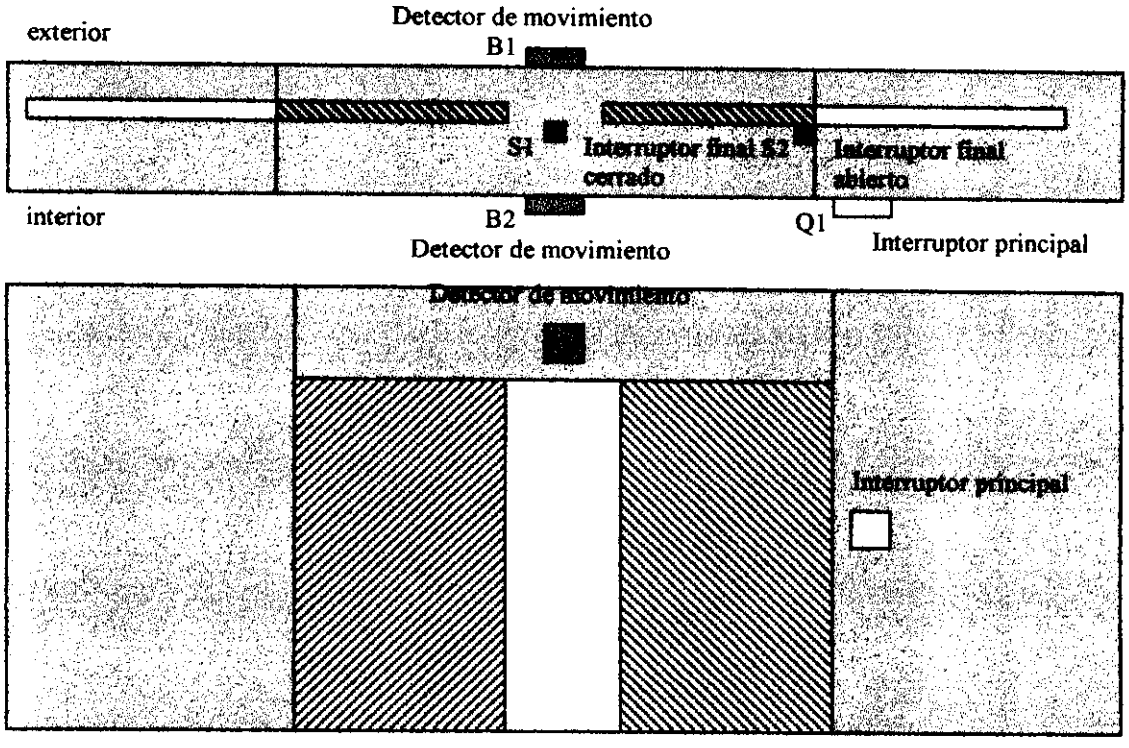
ANEXO 1. ALUMBRADO DE ESCALERAS O DE PASILLO



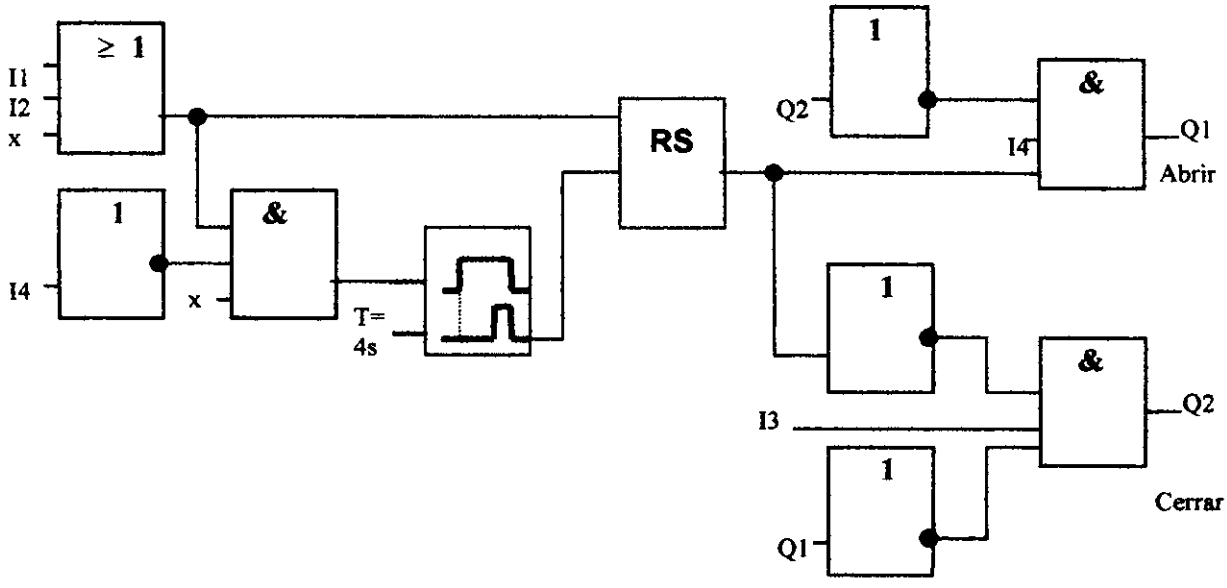
Pulsador de confort mediante LOGO!



ANEXO 2. PUERTA AUTOMATICA



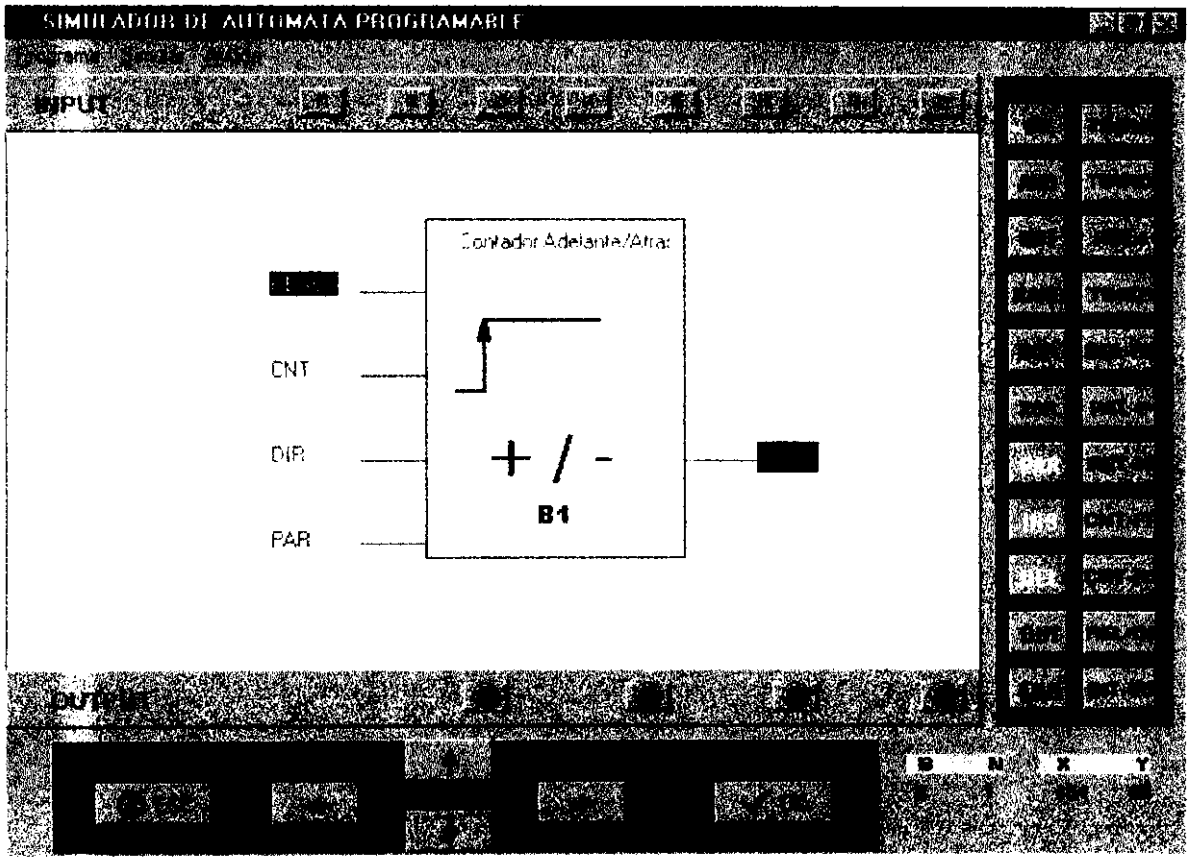
Esquema de circuito del control de puerta mediante LOGO!



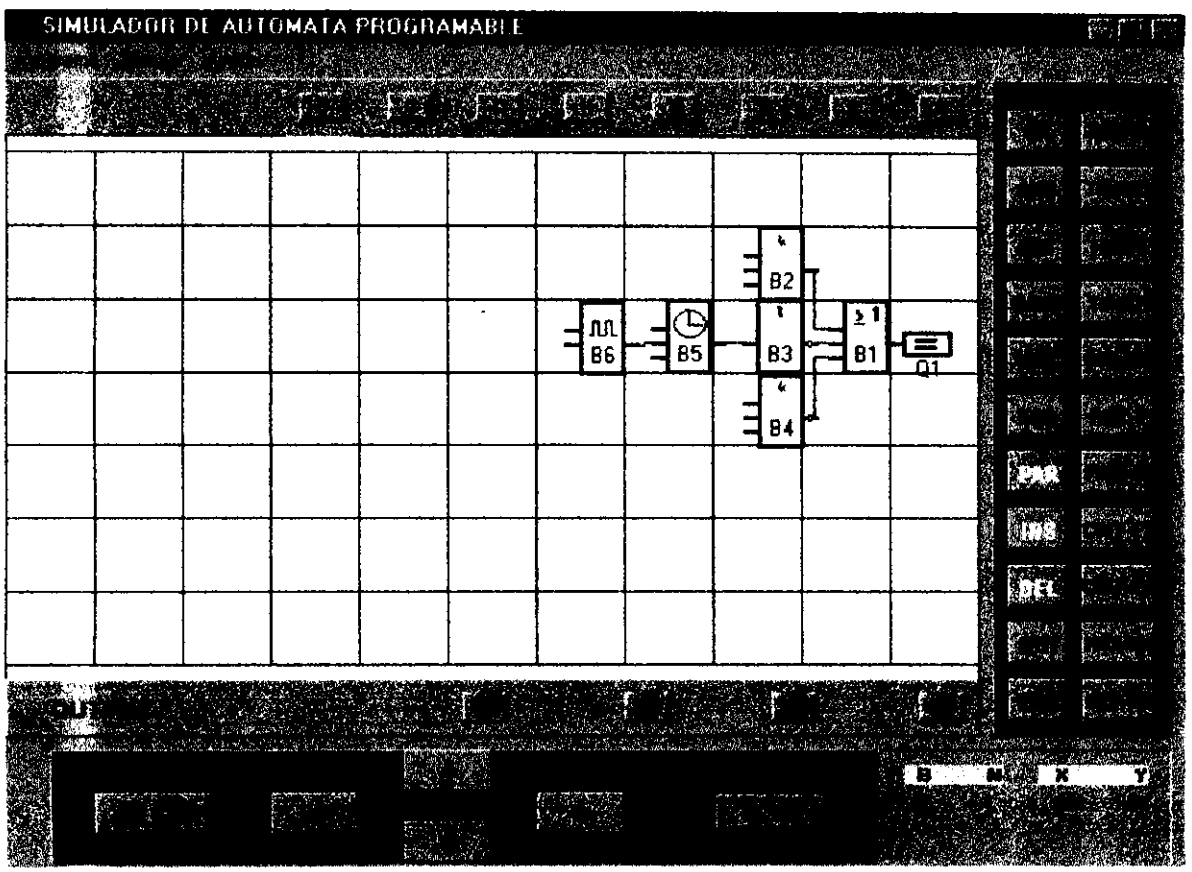
ANEXO 3.


PONTIFICIA UNIVERSIDAD CATOLICA
SEDE AMBATO
SIMULADOR DE AUTOMATAS PROGRAMABLES
GUILLERMO ALMEIDA
MAURICIO RODRIGUEZ
1999

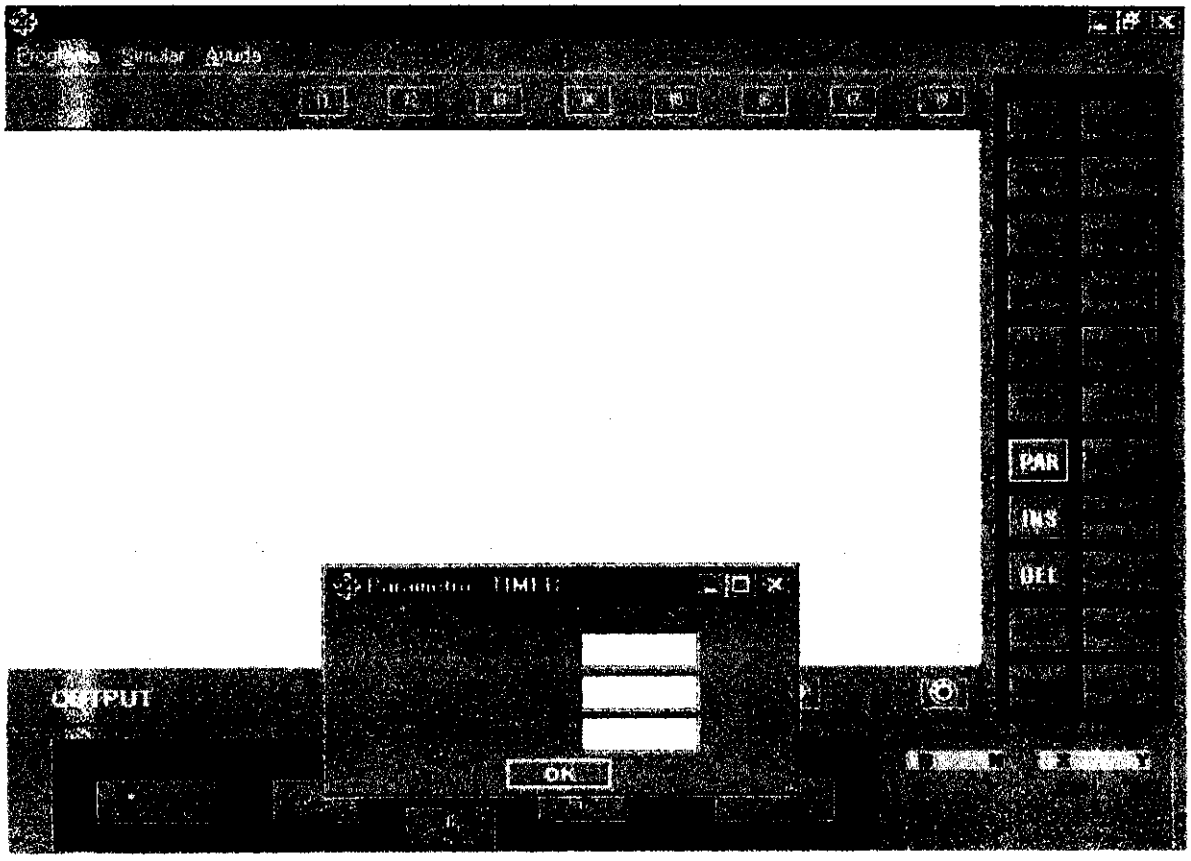
ANEXO 4.


SIMULADOR DE AUTOMATA PROGRAMABLE
INPUT
Contador Adelante/Atras
CNT
DIR
PAR
+ / -
B1
OUTPUT

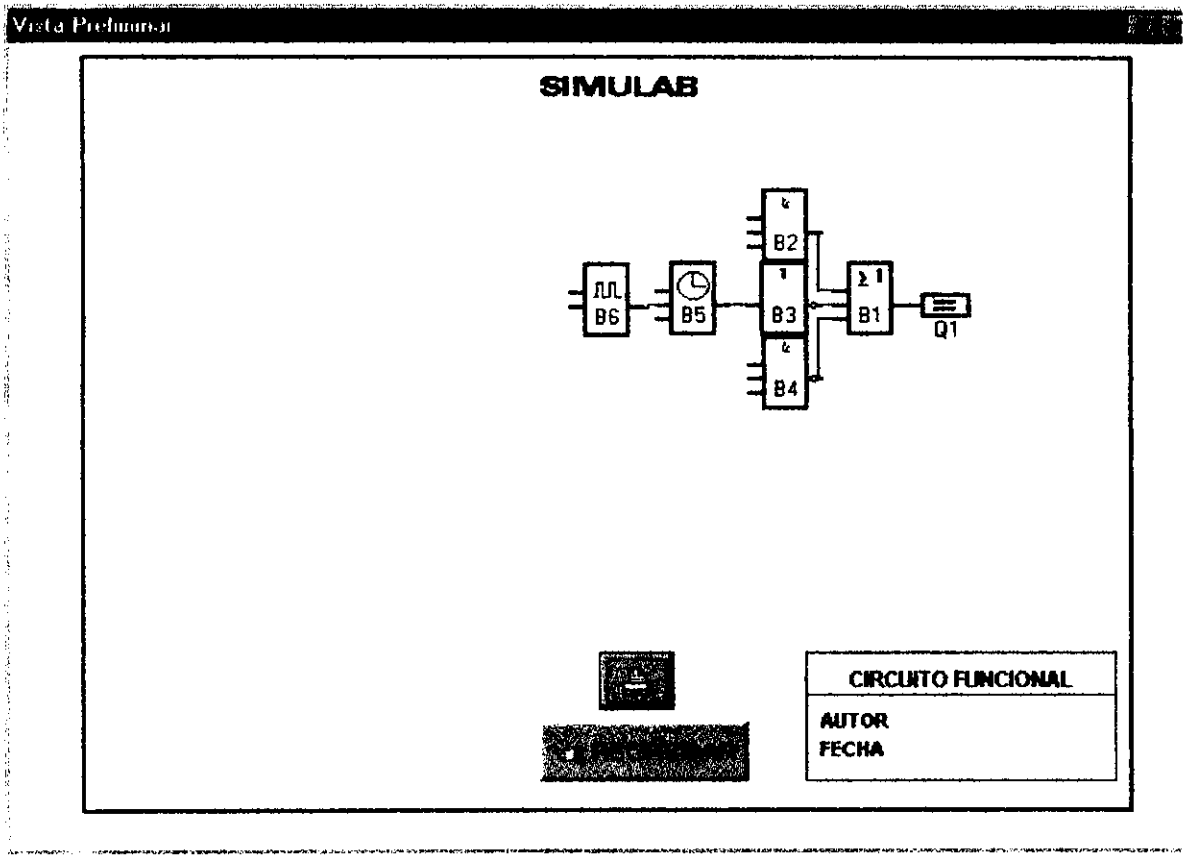
ANEXO 5.


SIMULADOR DE AUTOMATA PROGRAMABLE
B2
B6
B5
B3
B1
Q1
B4
PRG
INS
DEL
B
M
X
Y

ANEXO 6.



ANEXO 7.


Vista Preliminar
SIMULAB
B2
B6
B5
B3
B1
Q1
B4
CIRCUITO FUNCIONAL
AUTOR
FECHA

CONCLUSIONES Y RECOMENDACIONES

- El desarrollo del presente trabajo constituye un estudio e investigación de carácter técnico bibliográfico dirigido a los autómatas programables, que son los dispositivos electrónicos de control, que en la mayoría de países a nivel mundial, han alcanzado su apogeo en el área de la automatización industrial.

- Es por esta razón que en el presente estudio no se ha pretendido descubrir o inventar, sino más bien conocer, observar y analizar el tema con una mayor profundidad y detalle, para de ésta forma diseñar un trabajo, a medida y criterio adecuados al tema estudiado, dando un especial énfasis al área de sistemas.

- Es importante destacar el hecho que la mayor parte de procesos de prueba hoy en día, utilizan simuladores informáticos que permiten bajo ciertas condiciones iniciales programadas, observar resultados, que tienen altos niveles de eficacia y exactitud, factores que justifican plenamente su implementación.

- El objetivo propuesto se cumplió en su totalidad, ya que se ha desarrollado un Software de Simulación de Autómata Programable, de acuerdo a lo estudiado durante la realización del presente trabajo.

- Se recomienda que el tratamiento y desarrollo de este tipo de trabajos se lo efectúe con mayor frecuencia, puesto que el futuro de la ciencia y la tecnología, se encuentra en los simuladores informáticos, que bajo condiciones de seguridad humana, ahorro de energía y tiempo permiten comprobar la funcionalidad de procesos que bajo condiciones normales, presentarían alto riesgo o tendrían una larga duración, previo a su desarrollo final.

- Finalmente se recomienda que se desarrollen simuladores informáticos de procesos industriales, ya que ahorran dinero, permiten evaluar su funcionamiento y además sirven como material de estudio.

- La tecnología informática actual propende a la preparación de ingenieros en sistemas informáticos con vastos conocimientos de electrónica puesto que es la combinación hardware-software humana adecuada, por lo tanto, se recomienda la incorporación de mayor cantidad de horas clase en esta área para las generaciones futuras.

BIBLIOGRAFIA

1. Ingeniería de Control Moderna, Ogatta Katsuhiko, Segunda Edición, 1994.

2. Electrónica Industrial, Dispositivos, Sistemas para procesos y Comunicaciones Industriales, Editorial Paraninfo, James T. Humphries Leslie P. Sheets, Madrid España, 1996.

3. Autómatas Programables Nivel I. SWISSCONTACT, Urteaga , 1995.

4. Aparato de Programación PG-710. SWWISSCONTACT. Urteaga, 1995.

5. Aplicaciones a la Electrónica, Enciclopedia Teórico-Práctico, Autómatas y Robots Industriales, Copyring Boixareu Editores S.A., 1984

6. Siemens. LOGO!, Manual Edición 3, Siemens AG 1996 All rights reserved.

7. Análisis y Diseño Orientado a Objetos, Métodos de Grady Booch.

8. Análisis y Diseño Orientado a Objetos, Método de COAD & YOURDON.

9. Delphi, Guía oficial de Borland. Michelle M. Manning. 1996.

10. Control Systems: Analysis, Desing and Simulation, Brewer J. W, Englewood Cliffs, N.S. Prentice Hall 1974.

11. Fundamentals of Automatic Control N.Y.; Wegrick R.C., McGraw Hill 1975 Reston, Va. Reston, 1978.

12. D.E. Digital and Analog Controls Reston; Va.: Needles, M.A. y Baker, Reston 1985.

13. Programables Controllers – Principles and Applications, Webb, J.W.,Columbus Ohio Merrill, 1988.

14. Circuitos Integrados Digitales y Computadores, Barry G. Woollard, Cuarta Edición. Madrid 1985.