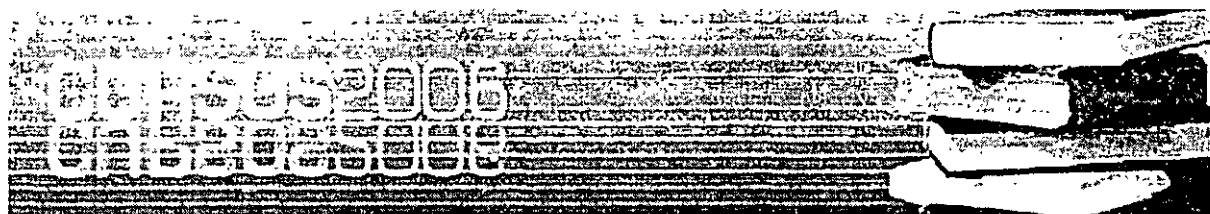




FACULTAD DE INGENIERÍA UNAM
DIVISIÓN DE EDUCACIÓN CONTINUA



CURSOS ABIERTOS

CIRCUITOS LÓGICOS PROGRAMABLES

CA 522

TEMA
APUNTES GENERALES

EXPOSITOR: Ing. José Luis Ramírez Gutiérrez y
Francisco Rodríguez Ramírez
Jueves 10 al viernes 25 de agosto de 2006
PALACIO DE MINERÍA



Pontificia Universidad Católica de Chile
Facultad de Ingeniería

CONTROLADORES LÓGICOS PROGRAMABLES

Domingo Mery

Noviembre 1994



Pontificia Universidad Católica de Chile
Facultad de Ingeniería

CONTROLADORES LÓGICOS PROGRAMABLES

1. INTRODUCCIÓN

- 1.1 ¿Qué es un PLC?
- 1.2 Desarrollo histórico
- 1.3 Aplicaciones de los PLC

2. FUNDAMENTOS DEL CONTROL LÓGICO

- 2.1 Sistemas numéricos
- 2.2 Aritmética binaria
- 2.3 Números binarios con y sin signo
- 2.4 Circuitos lógicos
- 2.5 Compuertas lógicas

3. ESTRUCTURA DE UN PLC

- 3.1 Unidades funcionales
- 3.2 Interfaces de estado sólido
- 3.3 Administración de entradas y salidas de un PLC

4. LENGUAJES DE PROGRAMACIÓN ORIENTADOS A PLC

- 4.1 Lenguajes de programación
- 4.2 Programación con diagrama escalera
- 4.3 Programación con bloques funcionales
- 4.4 Programación con lógica booleana

5. PROGRAMACIÓN DE UN PLC

- 5.1 Contactos
- 5.2 Bobinas (solenoides)
- 5.3 Relés de control
- 5.4 Cajas lógicas RELAY LADDER
- 5.5 Diseño y documentación de programas
- 5.6 Funciones RLL
- 5.7 Instrucciones discretas
- 5.8 Instrucciones condicionales
- 5.9 Instrucciones de bits

- 5.10 Instrucciones COUNTER/TIMER
- 5.11 Instrucciones DRUM
- 5.12 Instrucciones MATRIX
- 5.13 Instrucciones matemáticas
- 5.14 Instrucciones para movimiento de información en memoria
- 5.15 Instrucciones para operación con palabras
- 5.16 ONE-SHOT

6. ESPECIFICACIONES DE UN PLC INDUSTRIAL

7. CONSIDERACIONES DE INSTALACIÓN Y MONTAJE

- 7.1 Preparación del lugar de instalación
- 7.2 Consideraciones de seguridad
- 7.3 Encapsulado (Enclosure)
- 7.4 Consideraciones de temperatura
- 7.5 Consideraciones eléctricas

8. APLICACIONES DE PLC

- 8.1 Sistema de control de un túnel
- 8.2 Supervisión de oleoductos

9. EXPERIENCIAS DE LABORATORIO

- 9.1 Reconocedor de productos en una cinta transportadora
- 9.2 Control de un proceso de mezclado en la fabricación de galletas
- 9.3 Control de tráfico
- 9.4 Control de demanda

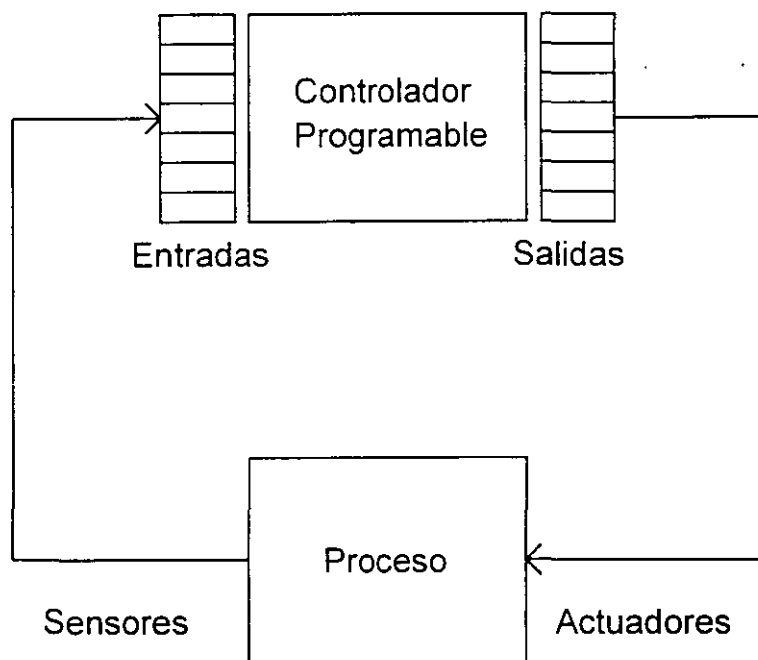
10. TENDENCIAS Y PERSPECTIVAS

- 10.1 Introducción
- 10.2 Hardware
- 10.3 Software
- 10.4 Comunicaciones

1. INTRODUCCIÓN

1.1 ¿Qué es un PLC?

Un PLC (*Programmable Logic Controller* - controlador lógico programable) es un dispositivo de estado sólido, diseñado para controlar secuencialmente procesos en tiempo real en un ámbito industrial.



Ejemplo del empleo de un PLC en un control de procesos.

Dentro de las funciones del PLC se puede mencionar:

- Adquirir datos del proceso por medio de las entradas digitales y analógicas.
- Tomar decisiones en base a reglas programadas.
- Almacenar datos en memoria.
- Generar ciclos de tiempo.
- Realizar cálculos matemáticos.
- Actuar sobre dispositivos externos mediante las salidas digitales y analógicas.
- Comunicarse con otros sistemas externos.

1.2 Desarrollo histórico

Los antecesores del PLC fueron los sistemas de control basados en relés (1960). Una aplicación típica de estos sistemas utilizaba un panel de 300 a 500 relés y miles de conexiones por medio de alambres, lo que implicaba un costo muy elevado en la instalación y el mantenimiento del sistema, estimado en US \$30 a \$50 por relé.

Luego surgieron los sistemas lógicos digitales construidos mediante circuitos integrados (1970), sin embargo eran productos diseñados para una aplicación específica y no eran controladores de propósitos generales. Muchos de ellos empleaban microprocesadores, pero su programación en un lenguaje poco familiar para los ingenieros de control (*Assembler*), hacía que el mantenimiento fuese inapropiado.

Los primeros controladores completamente programables fueron desarrollados en 1968 por la empresa de consultores en ingeniería Bedford y Asociados, que posteriormente pasó a llamarse MODICOM.

El primer Controlador Lógico Programable fue construido especialmente para la General Motors Hydramatic Division y se diseñó como un sistema de control con un computador dedicado.

Este primer modelo MODICOM, el 084, tuvo una gran cantidad de modificaciones, obteniéndose como resultado los modelos 184 y 384 desarrollados a principios de la década de los '70.

Con estos controladores de primera generación era posible:

- Realizar aplicaciones en ambientes industriales.
- Cambiar la lógica de control sin tener que cambiar la conexión de cables.
- Diagnosticar y reparar fácilmente los problemas ocurridos.

Los primeros PLC, que sólo incorporaban un procesador para programas sencillos y dispositivos de entrada/salida, evolucionaron hasta los equipos actuales, que integran:

- Módulos multiprocesadores.
- Entradas y salidas digitales de contacto seco, de relé o TTL.
- Entradas y salidas analógicas para corriente o voltaje.
- Puertas de comunicación serial o de red.
- Multiplexores análogos,
- Controladores PID.
- Interfaces con CTR, impresoras, teclados, medios de almacenamiento magnético.

1.3 Aplicaciones de los PLC

El PLC es usado en la actualidad en una amplia gama de aplicaciones de control, muchas de las cuales no eran económicamente posibles hace algunos años. Esto debido a:

- El costo efectivo por punto de entrada/salida ha disminuido con la caída del precio de los microprocesadores y los componentes relacionados.
- La capacidad de los controladores para resolver tareas complejas de computación y comunicación ha hecho posible el uso de PLC en aplicaciones donde antes era necesario dedicar un computador.

Existen 5 áreas generales de aplicación de PLC:

- Control secuencial.
- Control de movimiento.
- Control de procesos.
- Monitoreo y supervisión de procesos.
- Administración de datos.
- Comunicaciones.

Ejemplos de aplicaciones de PLC

- Annunciators
- Auto Insertion
- Bagging
- Baking
- Blending
- Boring
- Brewing
- Calendaring
- Casting
- Chemical Drilling
- Color Mixing
- Compressors
- Conveyors
- Cranes
- Crushing
- Cutting
- Digestors
- Drilling
- Electronic Testing
- Elevators
- Engine Test Stands
- Extrusion
- Forging
- Generators
- Gluing
- Grinding
- Heat Treating
- Injection Molding Assembly
- Motor Winding
- Oil Fields
- Painting
- Palltizers
 - Pipelines
- Polishing
- Reactors
- Robots
- Rolling
- Security Systems
- Stretch Wrap
- Slitting
- Sorting
- Stackers
- Stitching
- Stack Precipitators
- Threading
- Tire Building
- Traffic Control
- Textile Machine
- Turbines
- Turning
- Weaving
- Web Handling
- Welding

2. FUNDAMENTOS DEL CONTROL LÓGICO

2.1 Sistemas numéricos

Los sistemas numéricos son utilizados para la representación de números. Un sistema numérico de base n tiene n numerales, dígitos o símbolos distintos.

Mediante una combinación de los n dígitos es posible la representación de cualquier número. El sistema empleado por las personas es el decimal, debido al uso original de los diez dedos para contar. Sin embargo los sistemas digitales utilizan el sistema binario y sus derivados (octal y hexadecimal) ya que usan los *bits*: dígitos que sólo toman dos valores.

Sistema decimal

Está basado en 10 numerales o dígitos:

0, 1, 2, 3, 4, 5, 6, 7, 8 y 9.

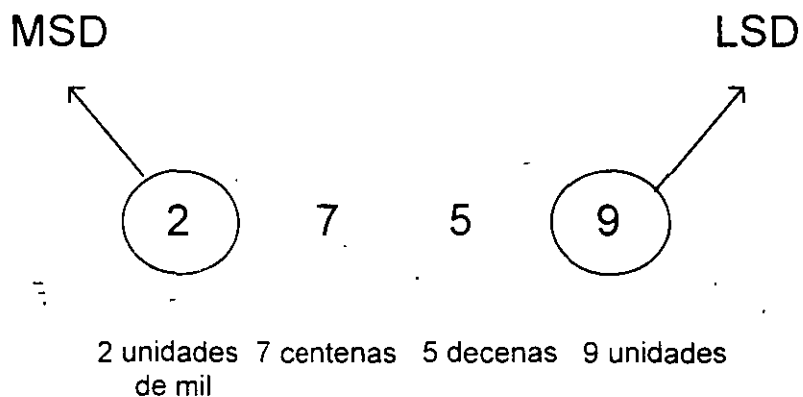
Mediante estos dígitos es posible representar cualquier número. Por ejemplo la representación de 2759_{10} es:

$$\begin{array}{cccc} & 10^3 & 10^2 & 10^1 & 10^0 \\ & \downarrow & \downarrow & \downarrow & \downarrow \\ & 1000 & 100 & 10 & 1 \\ & \downarrow & \downarrow & \downarrow & \downarrow \\ & 2 & 7 & 5 & 9 \\ 2759_{10} = & 2 \times 1000 + & 7 \times 100 + & 5 \times 10 + & 9 \times 1 \end{array}$$

Los sistemas numéricos se basan en un sistema posicional ponderado. El valor del dígito depende de su posición.

El dígito de mayor ponderación es denominado MSD (*Most Significant Digit*), y se ubica en la primera posición de izquierda a derecha.

El dígito de menor ponderación se denomina LSD (*Least Significant Digit*), y se ubica en la posición del extremo derecho.



Sistema binario

Está basado en los dígitos 0 y 1, de modo que cualquier cifra entera puede ser representada por medio de estos 2 numerales. Por ejemplo la representación de 11011_2 es:

$$\begin{array}{rcccccc}
 & 2^4 & & 2^3 & & 2^2 & & 2^1 & & 2^0 \\
 & \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow \\
 & 16 & & 8 & & 4 & & 2 & & 1 \\
 & \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow \\
 11011_2 = & 1 & & 1 & & 0 & & 1 & & 1 \\
 & 1 \times 16 + & & 1 \times 8 + & & 0 \times 4 + & & 1 \times 2 + & & 1 \times 1 \\
 & = & & 27_{10}
 \end{array}$$

El número 11011 en base 2 es el número 27 en base 10.

Números típicos en sistema binario:

b7	b6	b5	b4	b3	b2	b1	b0	Bit
128	64	32	16	8	4	2	1	← Ponderación
0	0	0	0	0	0	0	1	= 17_{10}
0	0	0	0	0	0	1	0	= 27_{10}
0	0	0	0	0	1	0	0	= 47_{10}
0	0	0	0	1	1	1	1	= 15_{10}
1	0	0	0	0	0	0	0	= 128_{10}
1	1	1	1	1	1	1	1	= 255_{10}

En los computadores digitales se utilizan niveles de voltajes para las representaciones. Normalmente se adoptan los siguientes valores (niveles TTL).

Desde	Hasta		Representa
0.0 Volt	0.4 Volt	→	0 lógico
0.4 Volt	2.4 Volt	→	Incertidumbre
2.4 Volt	5.0 Volt	→	1 lógico

En general, 0 lógico = nivel bajo y 1 lógico = nivel alto.

Sistema BCD (*Binary - Coded - Decimal*)

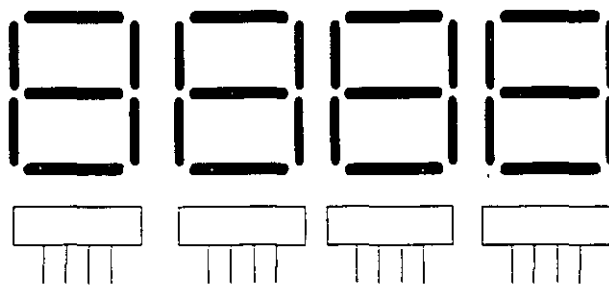
El sistema numérico BCD se basa en ponderaciones 8-4-2-1 empleando esta tabla.

Número	BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Así por ejemplo el número 831 es representado en BCD como:

$$831_{10} = 1000\ 0011\ 0001_{\text{BCD}}$$

El sistema BCD es ampliamente utilizado en visualizadores digitales (*Displays*), donde cada dígito es codificado por separado.



Display de 4 dígitos.

Sistema Octal

Se basa en ocho dígitos: 0, 1, 2, 3, 4, 5, 6 y 7. Por ejemplo la representación de 375_8 es:

$$\begin{array}{rcccc}
 & 8^2 & 8^1 & 8^0 & \\
 & \downarrow & \downarrow & \downarrow & \\
 & 64 & 8 & 1 & \\
 & \downarrow & \downarrow & \downarrow & \\
 & 3 & 7 & 5 & \\
 375_8 = & 3 \times 64 + & 7 \times 8 + & 5 \times 1 & \\
 = & 253_{10} & & &
 \end{array}$$

La transformación de octal a binario se obtiene como:

421	421	421	Ponderación binaria
3=	7=	5=	
0+2+1	4+2+1	4+0+1	Representación octal
011	111	101	Representación binaria

Es decir,

$$375_8 = 253_{10} = 011111101_2$$

Sistema Hexadecimal

Se basa en los 16 dígitos:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E y F.

Cuya equivalencia en el sistema decimal es:

Hexadecimal	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Por ejemplo la representación de $B13_{16}$ es:

$$\begin{array}{r} 16^2 \quad 16^1 \quad 16^0 \\ \downarrow \quad \downarrow \quad \downarrow \\ 256 \quad 16 \quad 1 \\ \downarrow \quad \downarrow \quad \downarrow \\ B_{16} \quad 1_{16} \quad 3_{16} \\ \downarrow \quad \downarrow \quad \downarrow \\ 11_{10} \quad 1_{10} \quad 3_{10} \\ B13_{16} = 11 \times 256 + 1 \times 16 + 3 \times 1 \\ = 2832_{10} \end{array}$$

La transformación de un número hexadecimal a uno binario se obtiene de la siguiente manera:

8421	8421	8421	Ponderación binaria
B=11=	1=	3=	Representación
8+0+2+1	0+0+0+1	0+0+2+1	hexadecimal
1011	0001	0011	Representación binaria

Es decir,

$$B13_{16} = 2832_{10} = 101100010011_2$$

Conversión decimal

Para convertir un número de sistema decimal a sistema binario se divide el número por 2, el resto representa el dígito binario de menor ponderación, el resultado se divide nuevamente por 2 hasta que el resultado sea cero. Por ejemplo, se desea convertir 29_{10} a código binario:

Operación	Resultado	Resto
$29/2=$	14	1
$14/2=$	7	0
$7/2=$	3	1
$3/2=$	1	1
$1/2=$	0	1

Representación binaria: 11101_2

Si el microprocesador utiliza 8 ó 16 bits, se debe anteponer tantos ceros como sea necesario.

Para convertir de decimal a octal se utiliza el mismo método dividiendo por 8. Por ejemplo 229_{10} es:

Operación	Resultado	Resto
$229/8=$	28	5
$28/8=$	3	4
$3/8=$	0	3

Representación octal: 345_8

Para convertir de decimal a hexadecimal se divide por 16. Por ejemplo 227_{10} es:

Operación	Resultado	Resto
$227/16=$	14	3
$14/16=$	0	$14=E_{16}$

Representación hexadecimal: $E3_{16}$

2.2 Aritmética binaria

Las operaciones binarias básicas son la adición o suma y la sustracción o resta.

Suma binaria

La suma de dos números binarios es:

Operación	Arrastre	Resultado
0 + 0	0	0
0 + 1	0	1
1 + 0	0	1
1 + 1	1	0

Por ejemplo:

$$\begin{array}{r} 1010 \\ + 0111 \\ \hline 10001 \end{array} \quad \begin{array}{r} 10_{10} \\ + 7_{10} \\ \hline 17_{10} \end{array}$$

Resta binaria

Para restar números binarios se utiliza el método conocido como complemento dos:

El sustraendo se convierte en su equivalente negativo y luego se suma al minuendo. Es decir:

$$A - B = A + (-B)$$

Para formar el número negativo equivalente:

- *Complemento 1: Se cambia cada bit por su complemento. Por ejemplo:*

$$010110 \rightarrow 101001$$

- *Se adiciona 1 al resultado anterior.*

Por ejemplo:

Realizar la resta: $11_{10} - 7_{10}$

- | | |
|---|-------------------------|
| 1) Representación de 7_{10} en binario: | 00000111 |
| 2) Complemento 1 de 7_{10} : | 11111000 |
| 3) Complemento 2 de 7_{10} : Adición de 1 | 11111001 |
| 4) Representación de 11_{10} en binario: | 00001011 |
| 5) Adición de 11_{10} y (-7_{10}) : | [1] 00000100 = 4_{10} |

(El arrastre 1 no se considera.)

2.3 Números binarios con y sin signo

Los números binarios pueden ser representados con y sin signo.

Números binarios sin signo

Un número binario de 8 bits sin signo se expresa:

b7	b6	b5	b4	b3	b2	b1	b0	Posición
2 ⁷	2 ⁶	2 ⁵	2 ⁴	2 ³	2 ²	2 ¹	2 ⁰	Ponderación
128	64	32	16	8	4	2	1	

Por lo tanto el rango de variación del número binario sin signo, en su equivalente decimal es:

$$00000000_2 = 0_{10}$$

$$11111111_2 = 255_{10}$$

Números binarios con signo

En números binarios con signo se utiliza comúnmente la notación complemento 2. En esta notación para números de 8 bits, el bit b7 indica el signo. El rango es -128_{10} a 127_{10} .

Si $b7 = 1$, el número es negativo.

Si $b7 = 0$, el número es positivo.

Algunos valores típicos en complemento 2 son:

b7	b6	b5	b4	b3	b2	b1	b0	Número decimal
1	0	0	0	0	0	0	0	-128
1	0	0	0	0	0	0	1	-127
1	1	1	1	1	1	1	0	-2
1	1	1	1	1	1	1	1	-1
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1
0	0	0	0	0	0	1	0	2
0	1	1	1	1	1	1	1	127

2.4 Circuitos lógicos

El diseño de circuitos lógicos se basa en la operación de variables digitales que sólo pueden tomar dos estados posibles:

ABIERTO o CERRADO

APAGADO o ENCENDIDO

BLANCO o NEGRO

OFF o ON

La expresión matemática de estos conceptos requiere de los números binarios:

$A = 0 \rightarrow$ FALSO, OFF, CONTACTO ABIERTO, RELÉ DESENERGIZADO, LÁMPARA APAGADA.

$A = 1 \rightarrow$ VERDADERO, ON, CONTACTO CERRADO, RELÉ ENERGIZADO, LÁMPARA ENCENDIDA.

El estado de un relé o contacto se identifica según su condición normal:

NO = *Normally open* - normalmente abierto

NC = *Normally close* - normalmente cerrado



2.4.1 Operadores NOT, AND y OR

Las operaciones matemáticas binarias se realizan con los operadores NOT, AND y OR.

NOT

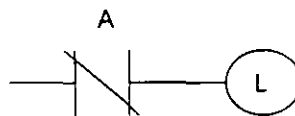
El operador NOT denota una salida verdadera si la entrada es falsa, y una salida falsa si la entrada es verdadera. Las distintas nomenclaturas son:

$$L = \text{NOT } A \qquad L = \bar{A}$$

Tabla de verdad para el operador NOT:

A	L
0	1
1	0

El circuito con un contacto NC representa el concepto lógico AND:



La lámpara L se encenderá sólo cuando A no esté conectado.

AND

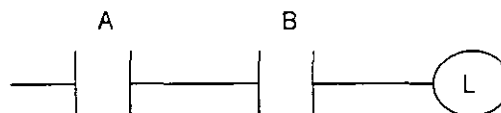
El operador AND denota una salida verdadera si y sólo si sus entradas son verdaderas. Las distintas nomenclaturas son:

$$L = A \text{ AND } B \quad L = AB \quad L = A * B$$

Tabla de verdad para el operador AND:

A	B	L
0	0	0
0	1	0
1	0	0
1	1	1

El circuito serie representa el concepto lógico AND:



La lámpara L se encenderá sólo si los contactos A y B están cerrados.

OR

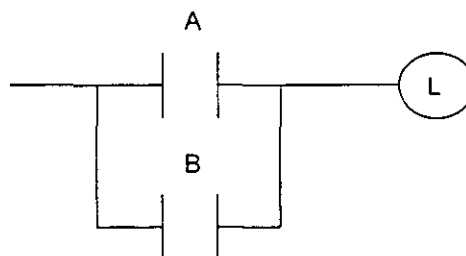
El operador OR denota una salida verdadera si hay alguna de las entradas (o ambas) verdaderas. Las distintas nomenclaturas son:

$$L = A \text{ OR } B = A + B$$

Tabla de verdad para el operador OR:

A	B	L
0	0	0
0	1	1
1	0	1
1	1	1

El circuito paralelo representa el concepto lógico OR:



La lámpara L se encenderá si alguno de los contactos A ó B (o ambos) está(n) cerrado(s).

Ejemplos

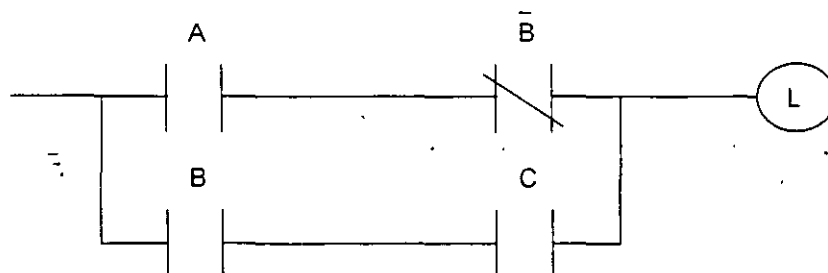
Existen operaciones binarias más complicadas, tales como:

1) $L = \bar{A}B + BC$

Esta función se basa en una combinación de operadores AND, OR y NOT. La tabla de verdad se representa a continuación:

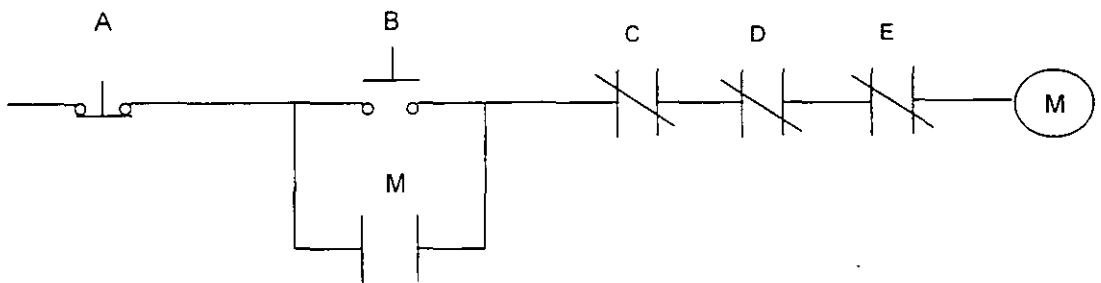
A	B	C	$\bar{A}B$	BC	L
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	0	1	1
1	0	0	1	0	1
1	0	1	1	0	1
1	1	0	0	0	0
1	1	1	0	1	1

El diagrama circuital es:



$$2) \quad M = \bar{A} (B + M) \bar{C} \bar{D} \bar{E}$$

Esta función corresponde al sistema de partida/parada del motor M. El diagrama circuital se representa en el siguiente esquema:

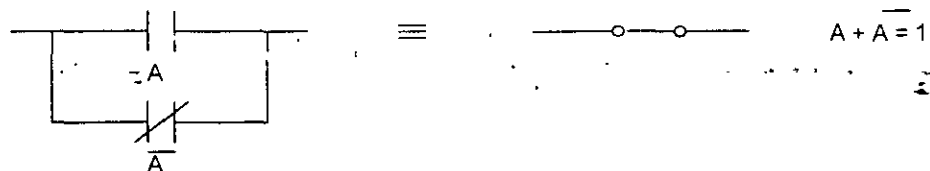
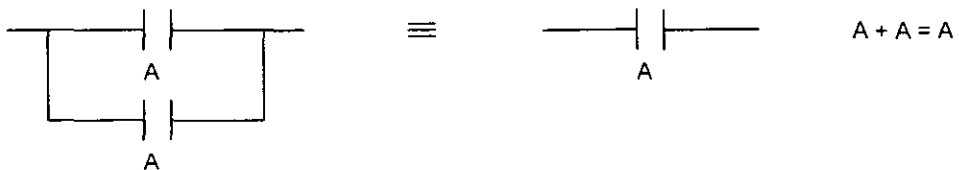
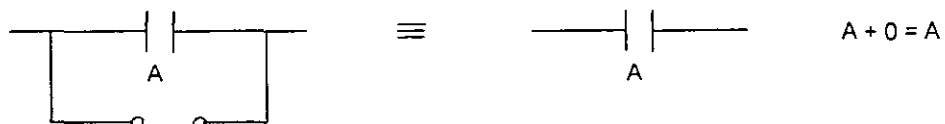
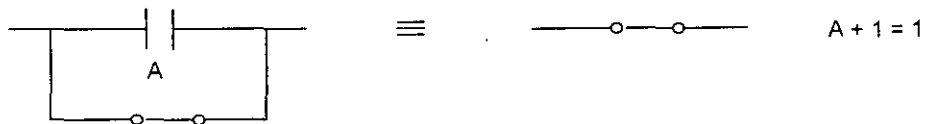


El motor (M) partirá sólo si el botón de parada (A) no está presionado y las sobrecargas (C, D, E) están cerradas y el botón de partida está presionado (B) o el contacto (M) está cerrado.

Se observa que en este sistema existe una realimentación de la variable M, que es la forma de realizar un elemento de memoria. Al encender el motor M presionando el pulsador B, se cerrará el contacto M. De esta forma al levantar B, el motor sigue energizado por el lazo B OR M.

2.4.2 Postulados lógicos Booleanos

Existen diversos postulados utilizados para simplificar las operaciones algebraicas. Entre ellos se puede mencionar.



2.4.3 Teoremas Booleanos

Los teoremas Booleanos permiten reducir la complejidad y extensión de las expresiones lógicas.

Teoremas de Morgan

Los teoremas de Morgan son:

$$\text{Para la adición: } \overline{A + B} = \bar{A} * \bar{B}$$

$$\text{Para la multiplicación: } \overline{A * B} = \bar{A} + \bar{B}$$

Se pueden utilizar los teoremas de Morgan en operaciones más complejas, como por ejemplo:

$$\begin{aligned}\overline{(A + \bar{B}C)DE} &= \overline{(A + \bar{B}C) + DE} \\ &= \bar{A}\bar{B}C + \bar{D} + \bar{E} \\ &= \bar{A}(B + \bar{C}) + \bar{D} + \bar{E}\end{aligned}$$

Teorema del término incluido

El teorema del término incluido establece que si un término de una expresión algebraica binaria está totalmente incluido en otro término, entonces el término mayor es redundante. Es decir,

$$L = A + AB = A$$

Ejemplo:

$$L = AB + ABCD = AB$$

Teorema de los productos opcionales

Se dice que un producto es opcional si su presencia o ausencia no afecta al resultado.

Ejemplo:

$$L = AB + A\bar{B} = A(B + \bar{B}) = A$$

Teorema de las sumas opcionales

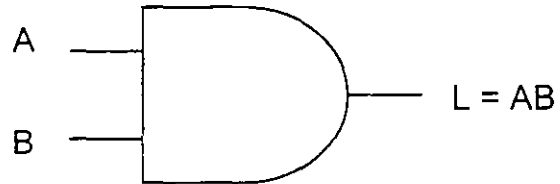
Una suma es opcional si su presencia o ausencia en la expresión Booleana no altera el resultado.

Ejemplo:

$$L = A(A + B) = A + AB = A(1 + B) = A$$

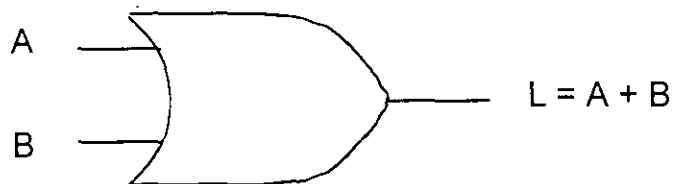
2.5 Compuertas lógicas

Puerta AND



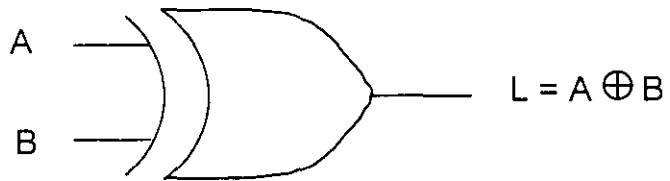
A	B	L
0	0	0
0	1	0
1	0	0
1	1	1

Puerta OR (inclusivo)



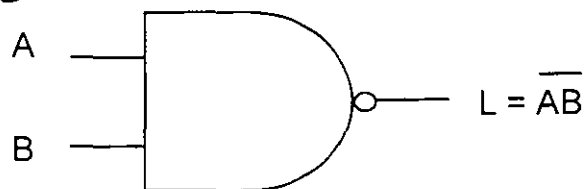
A	B	L
0	0	0
0	1	1
1	0	1
1	1	1

Puerta OR exclusivo



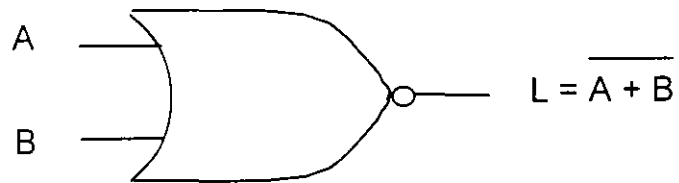
A	B	L
0	0	0
0	1	1
1	0	1
1	1	0

Puerta NAND



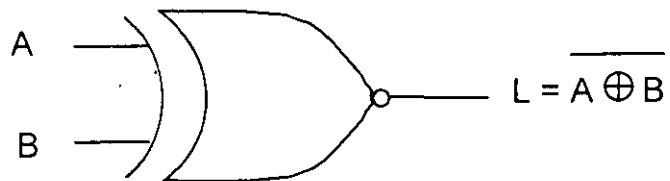
A	B	L
0	0	1
0	1	1
1	0	1
1	1	0

Puerta NOR



A	B	L
0	0	1
0	1	0
1	0	0
1	1	0

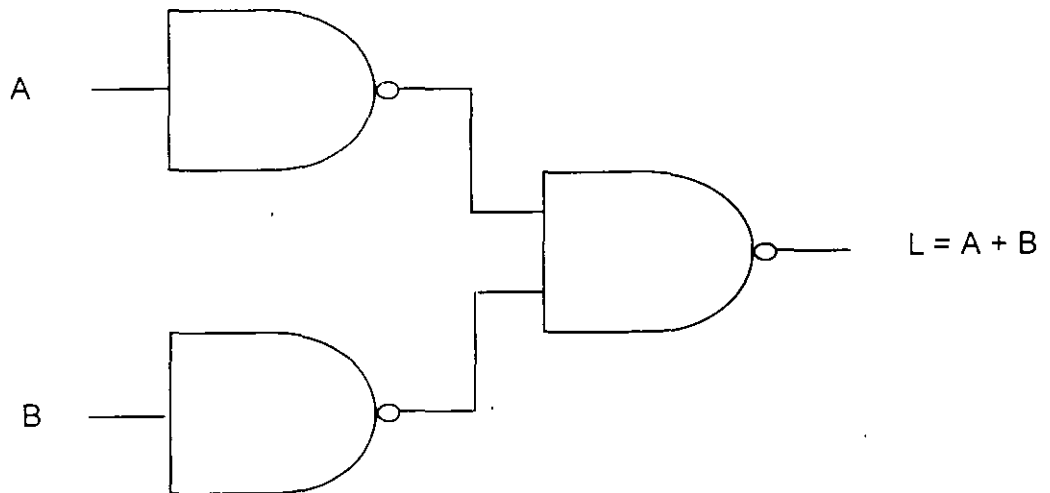
Puerta NOR exclusivo



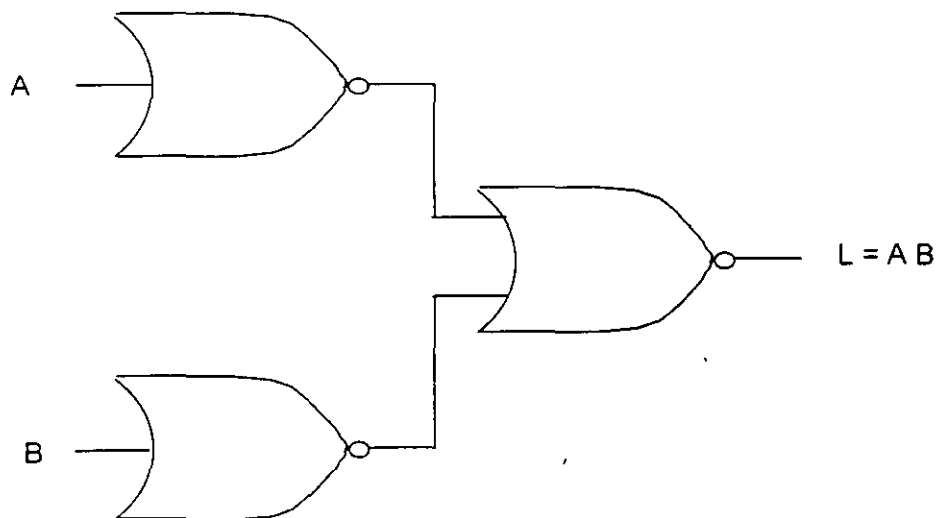
A	B	L
0	0	1
0	1	0
1	0	0
1	1	1

Ejemplos

Construcción de una puerta OR mediante puertas NAND:

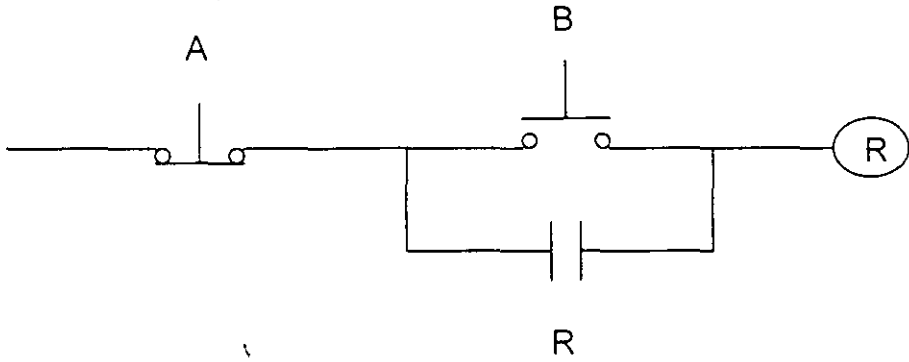


Construcción de una puerta AND mediante puertas NOR:



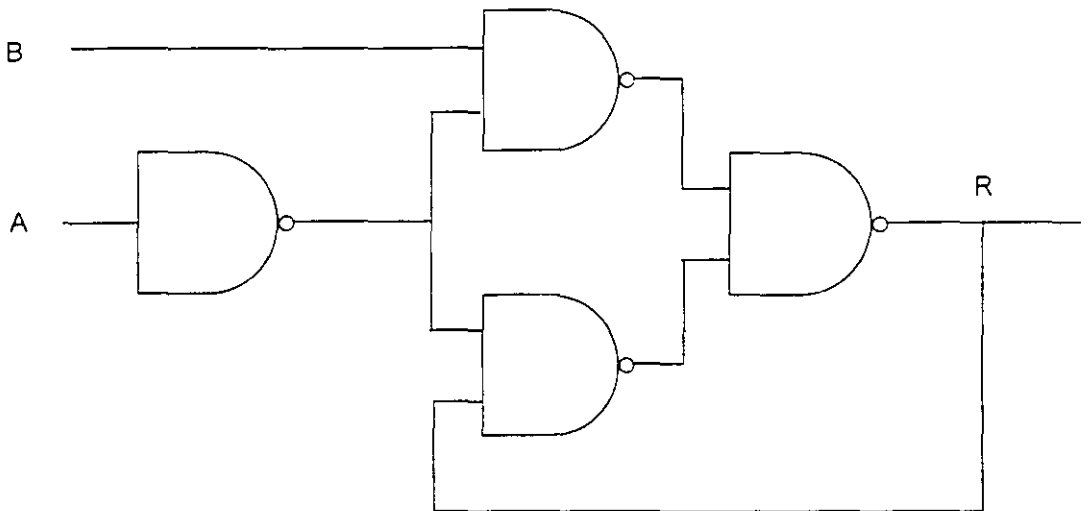
Ejemplo

Diseñar el circuito lógico correspondiente a la figura empenado sólo puertas NAND.



Expresión lógica:

$$R = \bar{A} (B + R) = \bar{A}B + \bar{A}R$$

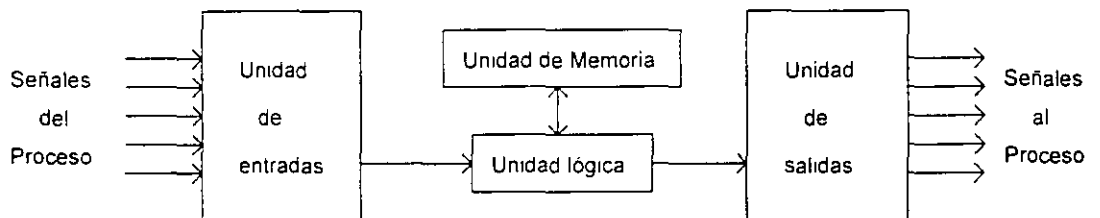


3. ESTRUCTURA BÁSICA DE UN PLC

3.1 Unidades funcionales

Un controlador lógico programable se compone de cuatro unidades funcionales:

- Unidad de entradas
- Unidad de salidas
- Unidad lógica
- Unidad de memoria



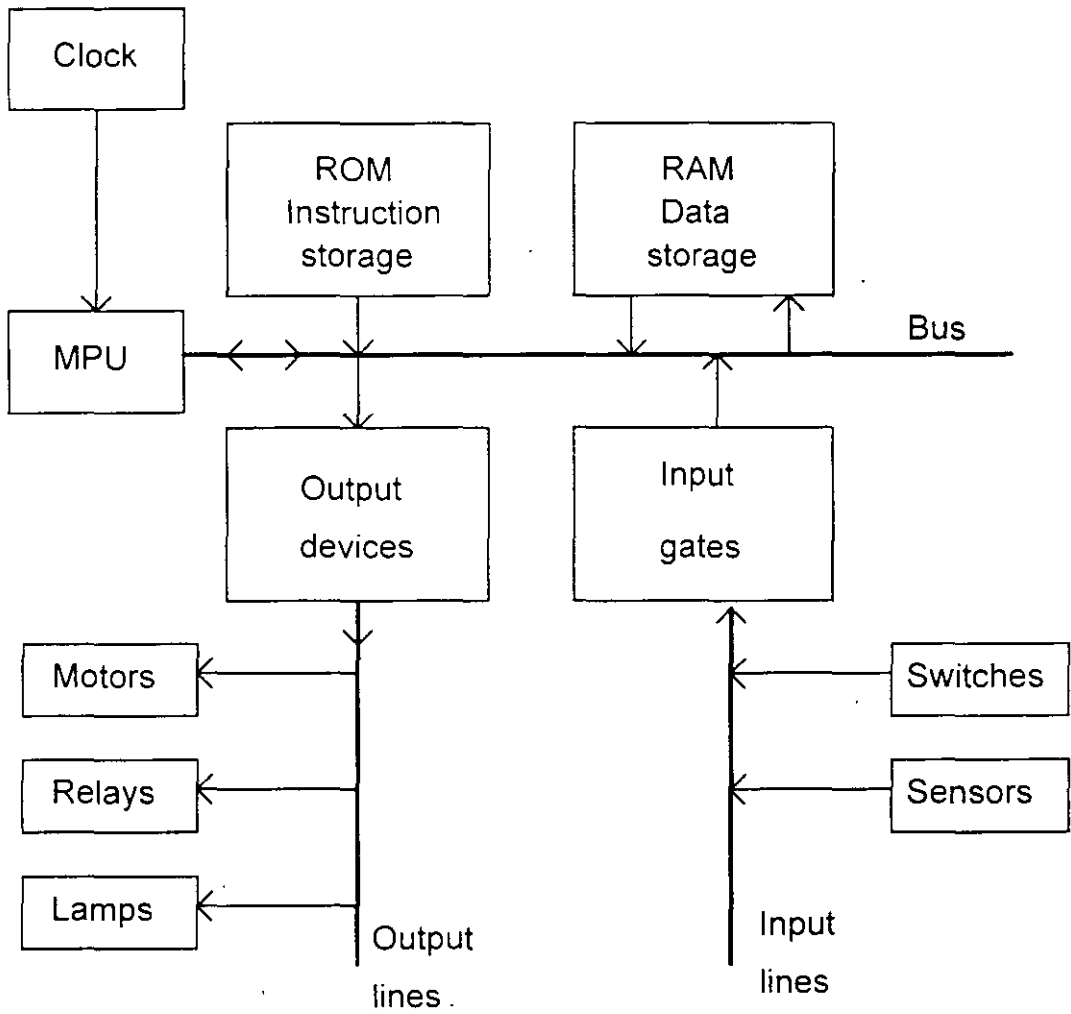


Diagrama de un PLC con dispositivos de entrada y salida.

Unidad de Entradas

Proporciona el aislamiento eléctrico necesario y realiza el acondicionamiento de las señales eléctricas de voltaje, provenientes de los *switches* de contactos ON-OFF de terreno. Las señales se adecúan a los niveles lógicos de voltaje de la Unidad Lógica.

Unidad de Salidas

Acepta las señales lógicas provenientes de la Unidad Lógica, en los rangos de voltaje que le son propios y proporciona la aislación eléctrica a los *switches* de contactos que se comandan hacia terreno.

Las unidades de entrada/salida del PLC, son funcionalmente iguales a los bancos de relés, que se empleaban en los antiguos controladores lógicos de tipo tambor. La diferencia radica en que las unidades de entrada/salida de los PLC son de estado sólido.

La eliminación de contactos mecánicos se traduce en una mayor velocidad de operación y mayor tiempo entre fallas (MTBF).

Unidad Lógica

El *corazón* de un PLC es la Unidad Lógica, basada en un microprocesador. Ejecuta las instrucciones programadas en memoria, para desarrollar los esquemas de control lógico que se especifican.

Algunos equipos antiguos implementan la unidad lógica en base a elementos discretos: compuertas NAND, NOR, FLIP-FLOP, CONTADORES como máquinas de estado. Este tipo de controladores son *HARDWIRE*, versus aquellos que utilizan memorias, denominados *SOFTWARE*.

Memoria

Almacena el código de mensajes o instrucciones que ejecuta la Unidad Lógica. La memoria se divide en PROM o ROM y RAM.

ROM: Memoria de sólo lectura (*Read Only Memory*). Memoria no volátil que puede ser leída pero no escrita. Es utilizada para almacenar programas y datos necesarios para la operación de un sistema basado en microprocesadores.

RAM: Memoria de acceso aleatorio (*Random Access Memory*). Memoria volátil que puede ser leída y escrita según sea la aplicación. Cualquier posición de memoria puede ser accesada en cualquier momento.

Por medio de ellas, se puede utilizar un PLC en procesos diferentes sin necesidad de readecuar o transformar el equipo; sólo se debe modificar el programa. Para el control de un proceso *BATCH*, se pueden almacenar varias recetas en la memoria y acceder aquélla que interesa.

Las PROM o ROM almacenan los programas permanentes que coordinan y administran los recursos del equipo.

La RAM guarda los programas de aplicación que pueden sufrir modificaciones. Esta memoria es respaldada con baterías, con el propósito de no perder la información al existir cortes de fluido eléctrico.

El sistema opera a través de la interacción con el procesador (Unidad Lógica) y la Memoria.

Cuando se enciende el equipo, el procesador lee la primera palabra de código (instrucción) almacenada en memoria y la ejecuta.

Una vez que termina de ejecutar la instrucción leída, busca en memoria la siguiente instrucción y así sucesivamente hasta que se completa la tarea.

Esta operación se llama ciclo de búsqueda-ejecución (FETCH-EXECUTE CYCLE).

3.2 Interfaces de Estado Sólido

La función de los módulos de entrada y salida es conectar el PLC con el mundo exterior de los motores, *switches* límites, alumbrados, y dispositivos de medición. Estos módulos se realizan a través de elementos de estado sólido.

Las aplicaciones iniciales de dispositivos de estado sólido en el control de partida de equipos de potencia, se remontan a la década de 1950 con la utilización de diodos y transistores.

Sin embargo, en la práctica las aplicaciones comenzaron en 1957 con la aparición del primer rectificador controlado de silicio (SCR).

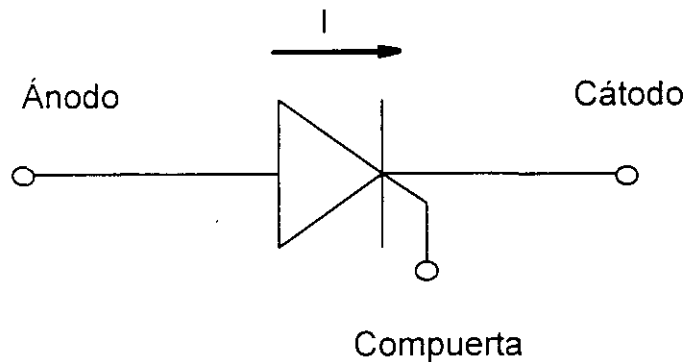
Los componentes de estado sólido empleados en las aplicaciones de control industrial, han reemplazando a los relés mecánicos en muchas de las funciones que llevaban a cabo.

Los dispositivos de estado sólido presentan muchas ventajas respecto a los relés, tales como alta velocidad de operación, pequeño tamaño y bajo consumo de potencia.

Sin embargo, son eléctricamente menos robustos y más sensibles a temperaturas elevadas y a la interferencia electromagnética (EMI).

Rectificador controlado de silicio SCR

El SCR, o denominado también tiristor, es utilizado como un interruptor electrónico que deja pasar corriente en un solo sentido.



El SCR, al recibir un pulso por la compuerta, deja pasar corriente sólo en el sentido ánodo → cátodo, en este caso su comportamiento es similar al del diodo.

Condiciones para el inicio de la conducción de un SCR:

- 1) Ánodo positivo respecto al cátodo.
- 2) Pulso positivo entre la compuerta y el cátodo.

El SCR permanecerá en el modo de conducción mientras el valor de la corriente esté por encima del valor crítico mínimo y se mantenga la diferencia de potencia positivo del ánodo con respecto al cátodo.

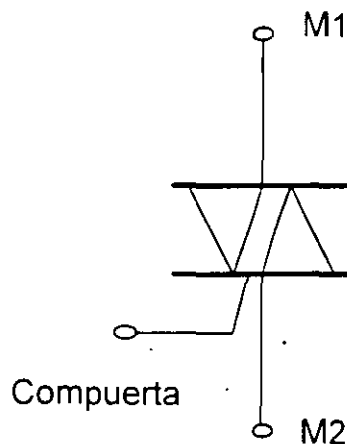
El SCR también entrará en conducción si la tensión ánodo-cátodo sobrepasa los límites específicos del SCR (conducción por avalancha).

Generalmente, se emplea el SCR en circuitos de corriente alterna (AC). Mediante un pulso de control en la compuerta, que debe aplicarse durante el medio ciclo positivo, el SCR entra en conducción.

Existen diversos circuitos electrónicos utilizados para enviar los pulsos correspondientes a la compuerta del SCR. Algunos de ellos emplean microprocesadores, circuitos temporizadores, sensores de fase, UJT, etc.

EL TRIAC

El TRIAC se utiliza como un interruptor electrónico que deja pasar corriente en ambos sentidos. Su construcción es la de dos SCR conectados en anti-paralelo.



El TRIAC tiene un amplio campo de uso en cargas de motores AC, ya que puede conducir en ambos semi-ciclos de voltaje alterno.

En comparación con los relés, el TRIAC resulta ser más sensible a la tensión aplicada, a la corriente y a la disipación interna de potencia. Una mala operación puede dañar el dispositivo para siempre.

Efectos del ruido

Se define el ruido como toda señal eléctrica indeseada, que puede entrar al equipo por diferentes vías. El ruido abarca el espectro completo de frecuencia y no presenta una forma de onda determinada.

El ruido eléctrico puede ocasionarle serios problemas de funcionamiento a los equipos de estado sólido, a causa de los bajos niveles de señal con que funcionan.

El ruido puede corresponder a alguno de los tres tipos básicos que se indican:

- Ruido transmitido, propio de la señal original.
- Ruido inherente, producto de los elementos que se integran en un sistema de adquisición de datos.
- Ruido inducido, originado por las fuentes de poder, acoplamientos magnéticos y acoplamientos electrostáticos.

Algunas medidas que deben tenerse en cuenta para reducir el acoplamiento del ruido eléctrico son:

- Usar encapsulados metálicos adecuados (jaula Faraday).
- Canalizar las líneas de control de los dispositivos de estado sólido en forma separada de las líneas de poder.
- Utilizar cables apantallados y trenzados; que proporcionan un escudo adecuado contra el acoplamiento electrostático y magnético.

El empleo de filtros adecuados permitirá eliminar el ruido indeseado de la señal.

Consideraciones especiales

Los componentes de estado sólido presentan una alta confiabilidad cuando se utilizan en los rangos y condiciones de operación adecuados.

La vida media de un TRIAC puede ser, por ejemplo, de 450.000 horas o 50 años, considerando condiciones de operación típicas. Sin embargo, puede fallar en forma aleatoria, incluso si se emplea dentro de los rangos de operación de diseño.

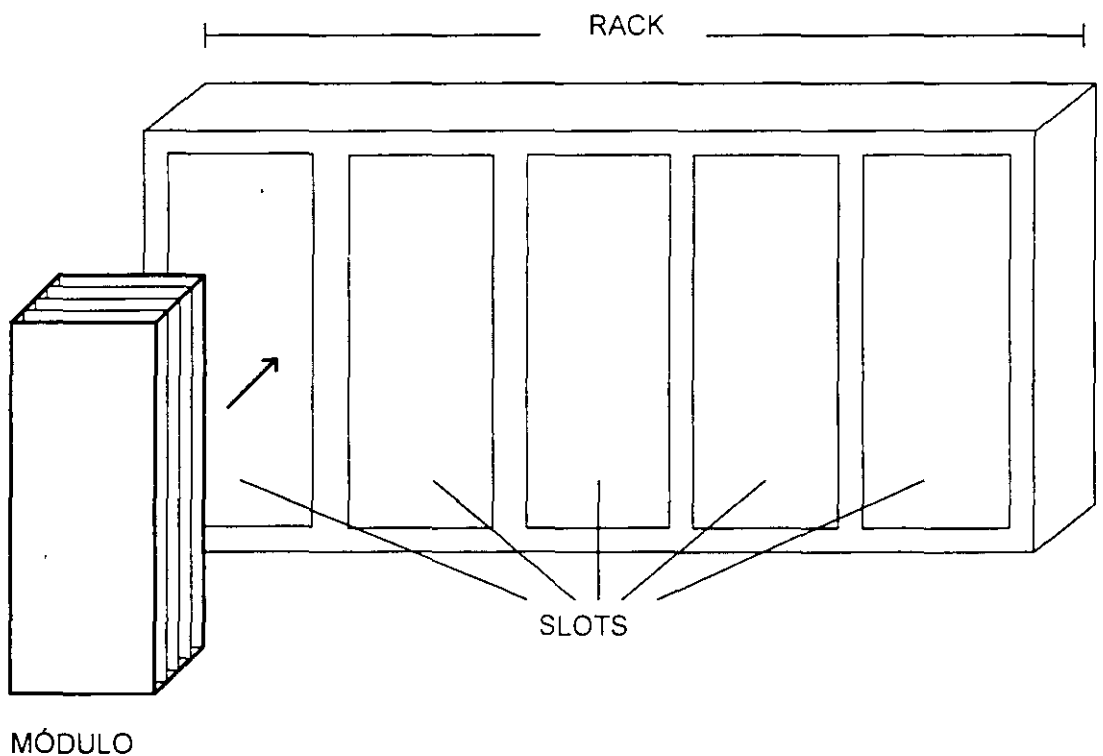
No es posible predecir cuándo va a fallar un componente de estado sólido cualquiera, como en el caso de los relés mecánicos, en los que observando su comportamiento se puede conocer el estado operacional.

Los controladores lógicos programables consideran las limitaciones y ventajas de los elementos de estado sólido, de modo que se minimizan los efectos del ruido. Generalmente, los PLC emplean rutinas de autodiagnósticos y verifican constantemente el funcionamiento correcto de los dispositivos de I/O.

3.3 Administración de entradas y salidas de un PLC

Bases del montaje

El montaje de los diversos módulos del PLC se realiza en *slots* ubicados en *racks*.



Los módulos básicos de un PLC son:

- Fuente de poder
- CPU
- Interfaces de entrada y salida

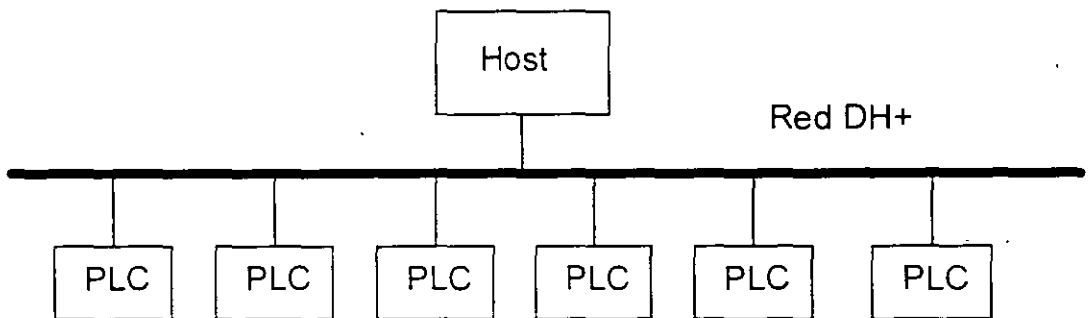
Dependiendo del modelo y la marca, existen en el mercado *racks* de diversos tamaños, como por ejemplo 4, 6, 8, 12, 14 y 16 *slots*. Según la aplicación se debe escoger el tamaño adecuado. Es posible instalar un módulo de ampliación, el que permite la conexión de un *rack* adicional.

Otros módulos existentes son:

- Módulos de comunicaciones (TCP/IP, DH+, etc.)
- Módulos de control de redundancia
- Módulos para conexión de *racks* remotos
- Módulos de interfaz hombre-máquina (teclado, monitores, etc.)
- Módulos de almacenamiento de información
- Módulos controladores PID

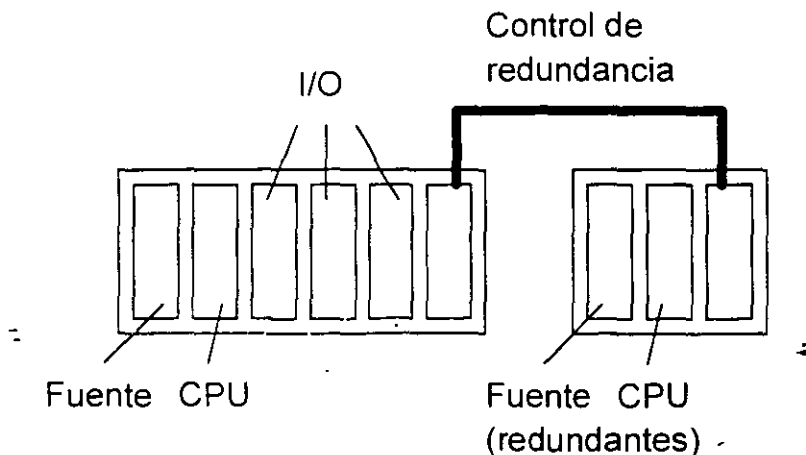
Módulos de comunicaciones

Permite la conexión del PLC a otros sistemas de información, tales como computadores y otros PLC. Existen por ejemplo redes tipo DataHiway para establecer una red de PLC conectados a un computador *Host*, utilizada comúnmente en sistemas de control distribuido.



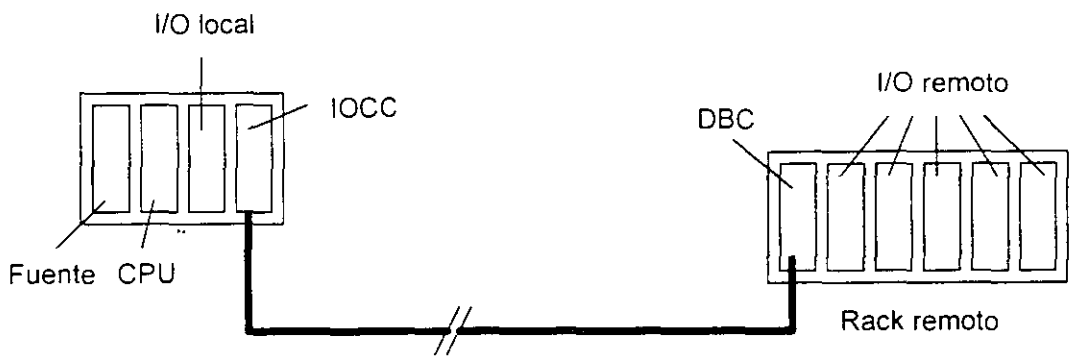
Módulos de control de redundancia

Son utilizados para asegurar la operación de un módulo redundante en caso de fallas. Generalmente se utiliza redundancia para el módulo de fuente de alimentación y el CPU.

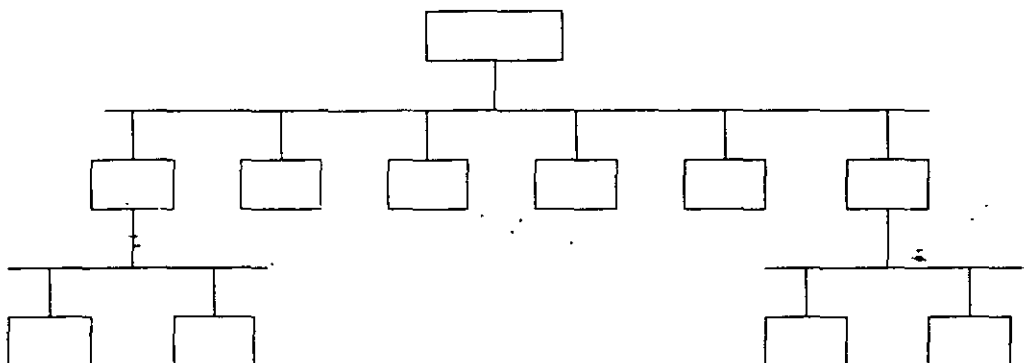


Módulos para conexión de *racks* remotos

En muchas aplicaciones los sensores y los actuadores están localizados a gran distancia del PLC. En estos casos se utilizan los *racks* remotos, los que son conectados por medio de un cable al *rack* central del PLC. Se consiguen distancias de 300 metros.

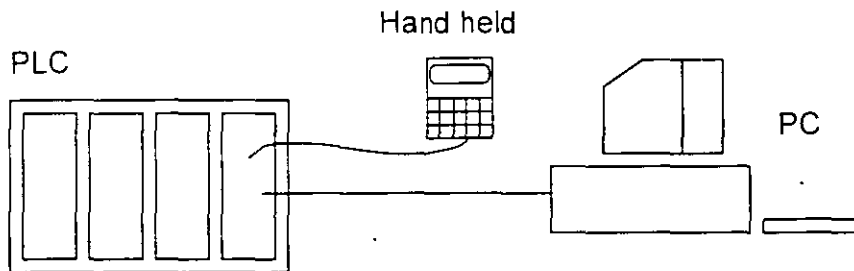


Para establecer esta comunicación se utiliza un módulo denominado canal controlador de entradas y salidas (IOCC) en el *rack* local y otro llamado controlador de base (DBC) en el *rack* remoto, al que se le puede conectar otro *rack* remoto, estableciéndose así una arquitectura distribuida con distintos niveles de jerarquía:



Módulos de interfaz hombre-máquina

Se utilizan para establecer la comunicación entre el PLC y el usuario. En la mayoría de los casos se emplea con este fin, un computador PC conectado serialmente, desde el cual se puede programar el PLC y ver los estados de los registros internos y los puntos de entrada/salida. En otros casos se usa un *Hand held monitor*, que es un dispositivo pequeño con teclas funcionales y pantalla de caracteres.

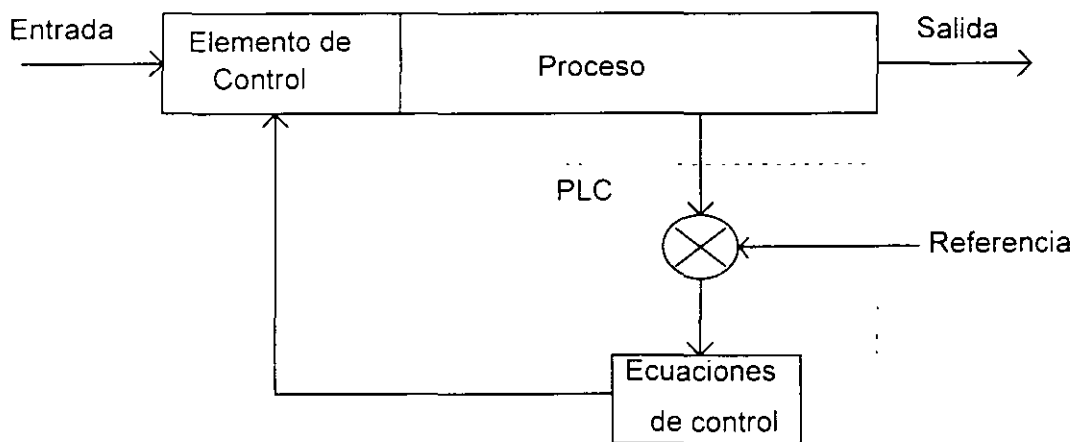


Módulos de almacenamiento de información

Por lo general se utilizan medios de almacenamiento magnéticos tales como cintas y discos, en los que se puede guardar información de los valores de los puntos de entrada/salida y registros internos.

Módulos controladores PID

Se utilizan en el control de procesos, en el que se pretende igualar una variable de salida de un proceso a una variable de referencia.

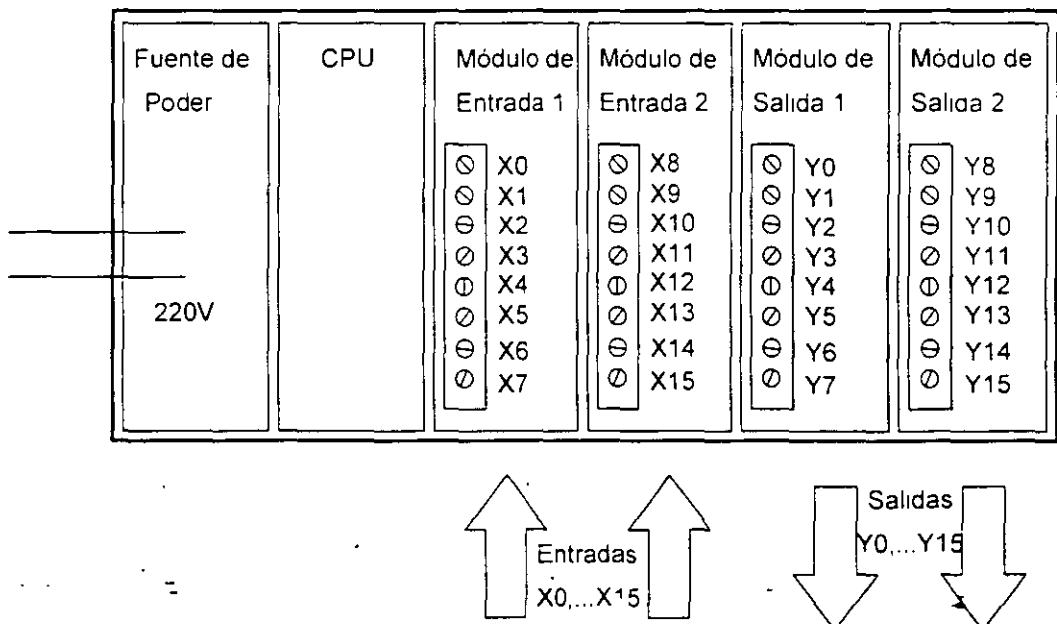


Puntos de entrada/salida

Los puntos del PLC son las entradas/salidas físicas que éste puede manejar. Cada punto tiene su representación interna en la memoria del PLC, en la que se utilizan números para identificarlos. Por lo general los módulos de entrada/salida vienen configurados en grupos de 8 puntos y pueden llegar hasta 1024, ampliables a más.

Los puntos de entrada son designados como X0, X1, X2, X3..., mientras que los puntos de salida se identifican como Y0, Y1, Y2, Y3...

A continuación se muestra una configuración básica de un PLC de 16 entradas y 16 salidas:

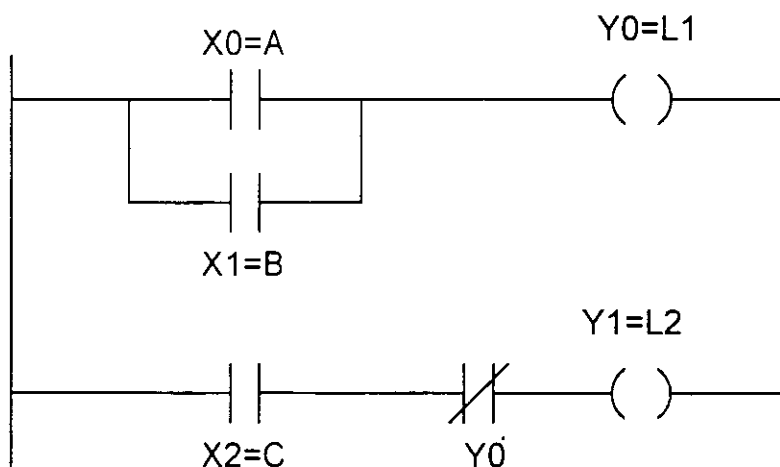


Al diseñar el programa se debe hacer referencia a las variables de entrada/salida que identifican los puntos del PLC.

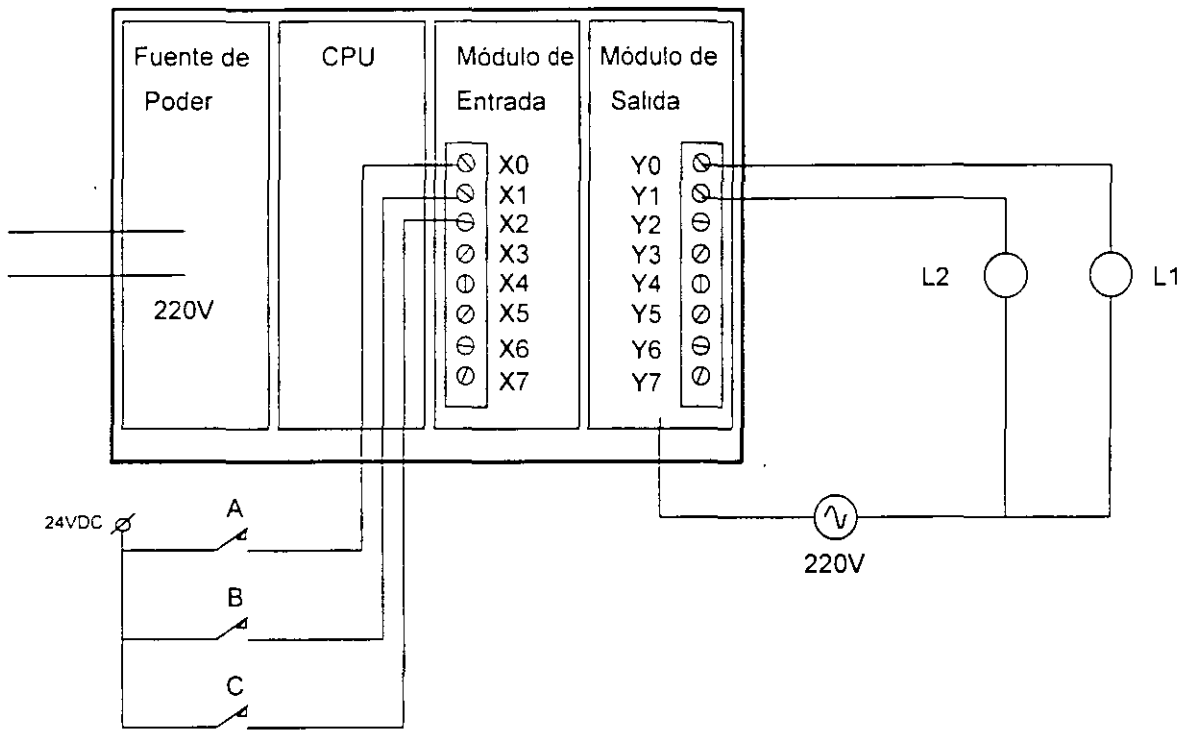
Ejemplo

- Se desea encender una lámpara L1 cuando se conecte el interruptor A o el interruptor B, y encender una lámpara L2 cuando L1 esté apagada y el interruptor C esté conectado.

Se distinguen las variables de entrada A, B y C, las que serán designadas como X0, X1 y X2; y las variables de salida L1 y L2, las que se identificarán como Y0 y Y1.



Las conexiones para este ejemplo se muestran a continuación:



La asignación de entradas y salidas se efectúa por medio del dispositivo de programación del PLC. Por lo general se utiliza un PC con interfaz gráfica que permita visualizar el diagrama escalera RLL (*Relay Ladder Logic*).

Registro imagen

Es un área de memoria del PLC reservada para mantener el estado de todas las entradas y salidas. Este registro se actualiza en forma permanente. Existen diversos registros:

- **Registro imagen discreto**

Corresponde a localizaciones de bits, donde se almacena el estado de todas las entradas/salidas digitales.

- **Registro imagen de relé control**

Son localizaciones de memoria de bits donde se guarda el estado de los Relés control.

- **Registro imagen de palabra**

Consiste en localizaciones de memoria, donde se registra el valor de cada palabra de entrada y salida.

En la programación de un PLC se utiliza también registros internos, que son de gran ayuda para almacenar datos intermedios. Estos registros son designados comúnmente como C0, C1, C2, ...

4. LENGUAJES DE PROGRAMACIÓN ORIENTADOS A PLC

4.1 Lenguajes de programación

Los lenguajes de programación ofrecen un conjunto de instrucciones con una determinada sintaxis para ejecutar una función.

Existen lenguajes de nivel bajo, intermedio y superior dependiendo del grado de comunicación que se tiene con la unidad de control de procesos (CPU) y el grado de complejidad de las instrucciones.

Otra clasificación de los lenguajes de programación son los lenguajes estructurados y los no estructurados, que se refieren a la forma de escribir y agrupar las instrucciones.

Un buen lenguaje de programación debe ser de fácil entendimiento, de tal forma que permita su modificación posterior si es que existen nuevos requerimientos.

Lenguajes de bajo nivel

Son los lenguajes que operan con instrucciones que controlan cada bit del CPU. Éstos son los lenguajes *Assembler* y de máquina. A manera de ejemplo, con estos lenguajes sólo se pueden sumar números de 8 ó 16 bits. Para realizar una suma de números de más bits es necesario descomponer el número en números primarios, sumarlo uno por uno guardando el arrastre de cada suma primaria para sumarlo con el siguiente número más significativo.

Ejemplo:

Suma 2+3 en Assembler de Z80

LD A, 03H	Carga 3 al acumulador A (A=3)
ADD A, 02H	Suma 2 al acumulador A (A=5)

Lenguajes de nivel intermedio

Estos lenguajes ofrecen un conjunto de instrucciones que pueden tanto comunicarse a nivel de bit con el microprocesador como ejecutar funciones de mayor grado de complejidad.

En los lenguajes de nivel intermedio se incorporan las funciones aritméticas, algunas funciones matemáticas (trigonométricas, raíz cuadrada, logaritmos, etc.) y funciones de manipulación de archivos en dispositivos de almacenamiento externo.

Ejemplos de lenguajes de nivel medio: C, FORTH.

Ejemplo:

Cálculo de 20! en C:

```
s=1;
for (i=2; i<=20; i++)
    s=s*i;
```


Lenguajes de nivel superior

Los lenguajes de nivel superior realizan con tan solo una instrucción una operación que con lenguajes de otro nivel se necesitaría fácilmente una docena de ellos.

Por ejemplo, con un lenguaje de nivel superior orientado al manejo de bases de datos, se puede con una sola instrucción ordenar alfabéticamente una lista de 10,000 nombres.

Ejemplos de lenguajes de nivel superior: PASCAL, FORTRAN, BASIC, dBASE, COBOL, SQL.

Ejemplo:

Ordenamiento de un directorio telefónico en dBASE

```
use telefono
index on nombre to telenom
```

Lenguajes estructurados y no estructurados

La diferencia fundamental entre la programación estructurada y la no estructurada radica en que la primera no acepta el comando de bifurcación. De esta forma, el programa se ejecuta sólo por secciones. Para realizar una bifurcación, es necesario recurrir a instrucciones condicionales que ejecutarán una sección del programa sólo si se cumple una determinada condición.

Por otra parte, el lenguaje no estructurado permite la bifurcación desde y hacia cualquier línea del programa.

Ejemplos de lenguajes no estructurados: BASIC, FORTRAN, *Assembler*:

Ejemplos de lenguajes estructurados: C, PASCAL, dBASE.

Ejemplo:

Cálculo de 20! en BASIC

```
10 S=1
20 I=2
30 S=S*I
40 IF I<=20 THEN 30
```

↑
NO ESTRUCTURADO

Cálculo de 20! en C

```
s=1;
for(i=2;i<=20;i++)
    s=s*i;
```

↑
ESTRUCTURADO

Lenguajes de programación orientados a PLC

El lenguaje de programación de un PLC permite la creación del programa que controlará su CPU.

Mediante este lenguaje el programador podrá comunicarse con el PLC y así dotarlo de un programa que controle las actividades que debe realizar.

Dependiendo del lenguaje de programación, es posible la realización del programa con distintos grados de dificultad.

Junto con el lenguaje de programación, los fabricantes suministran un software de ambiente de trabajo donde el usuario puede escribir sus programas. Estos softwares son amistosos y corren sobre computadores tipo PC bajo plataformas DOS o Windows.

Los métodos de programación más utilizados para PLC son:

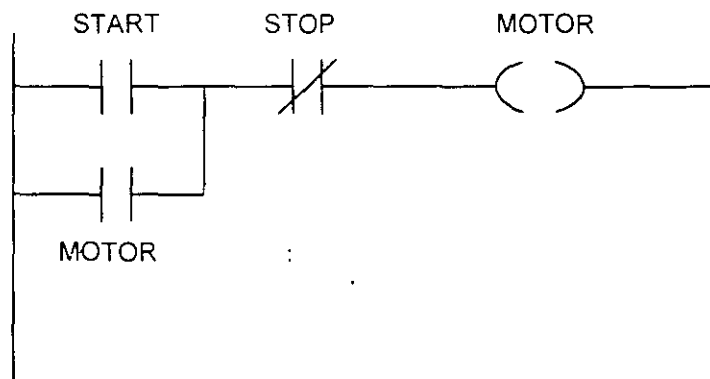
- Programación con diagrama escalera
- Programación con bloques funcionales
- Programación con lógica booleana

4.2 Programación con diagrama escalera

El diagrama escalera es uno de los más utilizados en la programación de PLC. Fue desarrollado a partir de los sistemas antiguos basados en relés. La continuidad de su utilización se debe principalmente a dos razones:

- Los técnicos encargados en darle mantenimiento a los PLC están familiarizados con este lenguaje.
- A pesar del desarrollo de los lenguajes de alto nivel, han sido pocos los lenguajes que han cumplido satisfactoriamente los requerimientos de control en tiempo real que incluyan la representación de los estados de los puntos de entrada y salida.

El nombre escalera proviene del uso de "rieles" y "peldaños" en el diagrama, como en este ejemplo de arranque de un motor.



En la gran mayoría de casos, las instrucciones para programar PLC pueden ser separadas en *básicas* y *expandidas*.

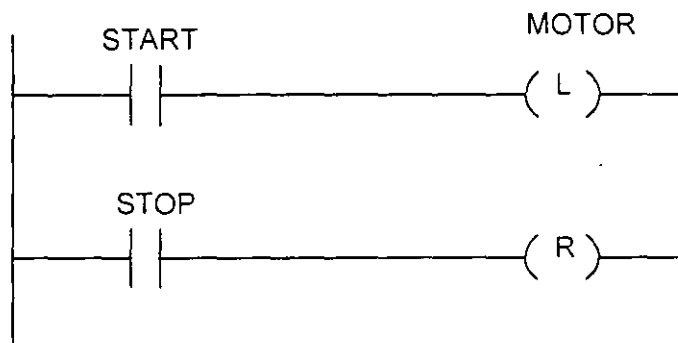
Instrucciones básicas:

Instrucciones básicas
RELAY
TIMER
COUNTER
LATCH
ONESHOT
I/O REG
REG I/O
BIN - BCD
BCD - BIN
ADD
SUB
COMPARE
MCR
SKIP

A continuación se explican algunas de ellas:

LATCH

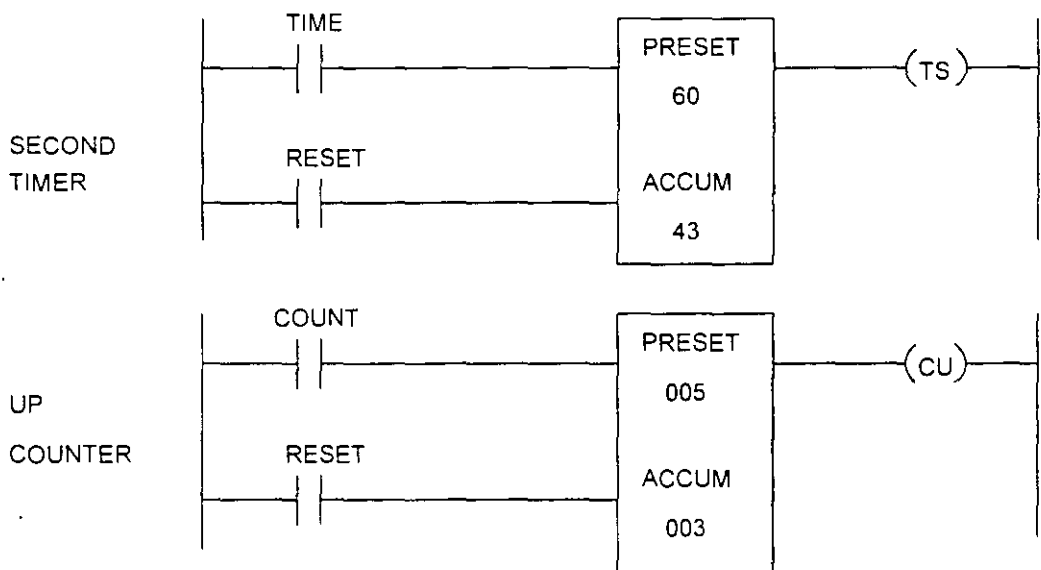
Mediante el empleo de *latches* es posible desarrollar el mismo diagrama anterior de arranque de un motor, en el que un simple contacto energiza y mantiene energizado el motor.



El *latch* retiene su estado lógico cuando se abre el contacto, es decir, basta un solo contacto momentáneo para que el *latch* quede energizado. Esta función es de gran uso en sistemas de seguridad, en los que por precaución un circuito lógico no debe empezar en el estado *on* después de reactivarse una falla eléctrica, sino que debe conectarse en forma manual.

TIMER y COUNTER

Estas instrucciones remplazan los contadores electromecánicos en aplicaciones que requieren de contadores y temporizadores de eventos discretos.

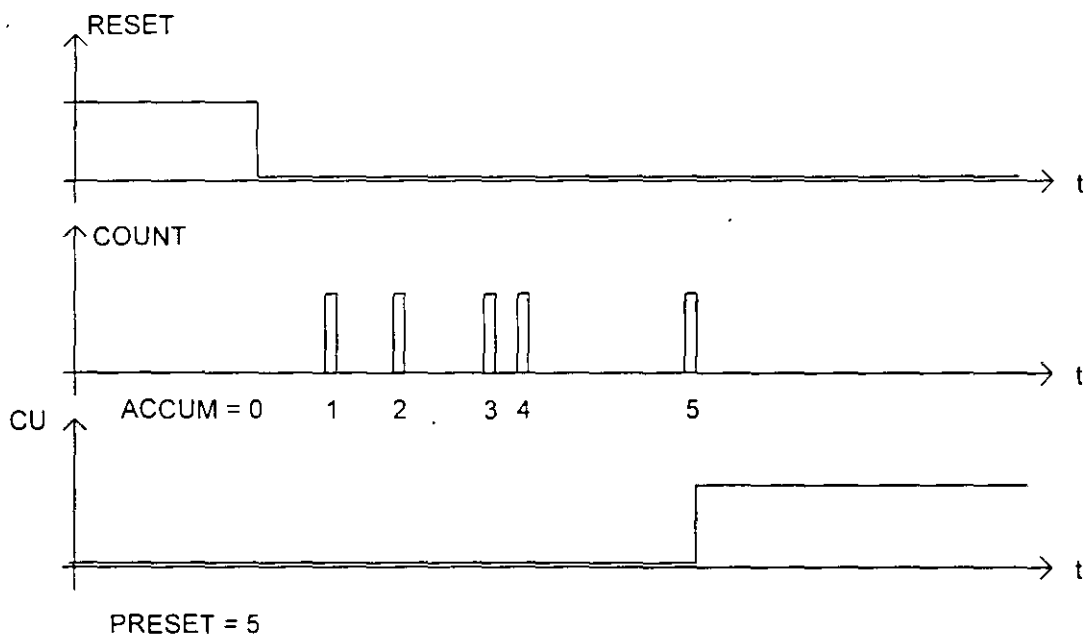


El **temporizador** opera de una manera similar al contador. Mientras el contacto TIME permanece cerrado, el valor del acumulador ACCUM se incrementa en uno por cada unidad de tiempo que pase. Esta unidad de tiempo es en algunos PLC 0.1 seg, mientras que en otros puede ser una unidad configurable.

Cuando el temporizador alcance el valor PRESET activará la salida TS. El contacto RESET hace que el valor del acumulador vuelva a 0.

El **contador** cuenta el número de contactos producidos en la entrada COUNT. Los contadores pueden contar hacia arriba: 0, 1, 2... ó hacia abajo 10, 9, 8, 7... El valor de la cuenta actual se almacena en el acumulador ACCUM. El valor del acumulador se hace 0 si el contacto RESET se cierra.

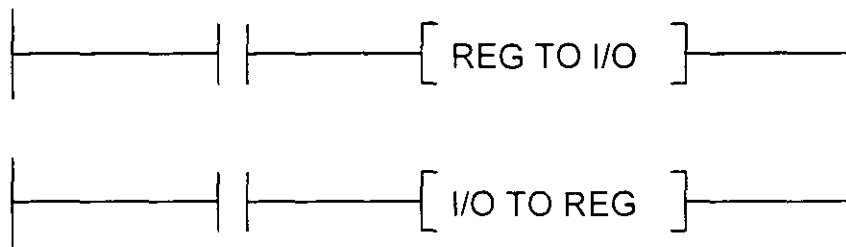
El contador cuenta hasta un valor de PRESET, y cuando lo alcanza activará la salida CU.



Ejemplo de un diagrama de tiempos de un contador

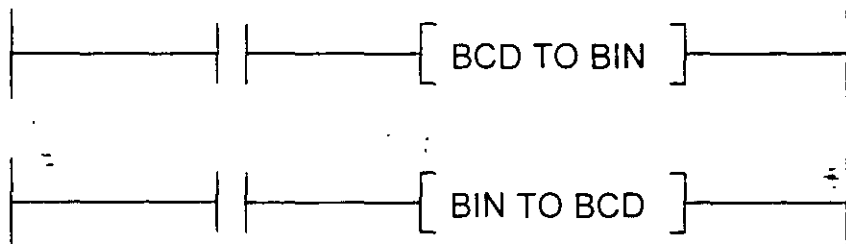
Instrucciones de entrada/salida

La instrucción I/O TO REG es utilizada para ingresar un punto de entrada a un registro del PLC, mientras que la instrucción REG TO I/O hace la operación contraria: pasa un registro a un punto de salida del PLC.



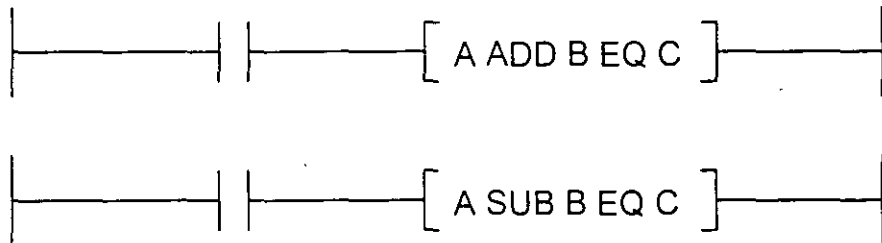
Instrucciones de conversión

Las instrucciones BCD TO BIN y BIN TO BCD son empleadas para convertir de código BCD a binario y binario a BCD respectivamente. Estas instrucciones se combinan con las de entrada y salida explicadas anteriormente.



Instrucciones aritméticas

Los PLC incluyen dentro de sus instrucciones, operaciones aritméticas sin signo. Los comandos utilizan los nemónicos ADD y SUB para la adición y sustracción respectivamente. En la figura los registros A y B son sumados o sustraídos, y el resultado se almacena en el registro C.



Instrucciones expandidas

Instrucciones expandidas	
MOVE	REM -FM - TOP
MOVE RIGHT 8	SORT
MOVE LEFT 8	AND
DP ADD	IOR
DP SUB	EOR
ADD X	INV
SUB X	MATRIX COMPARE
MPY	BIT SET
DVD	BIT CLEAR
GREATER THAN	SHIFT RT
TABLE - DEST	SHIFT LT
SRC - TABLE	DO SUB
MOVE TABLE	RETURN
ADD-TO-TOP	DO I/O
REM-FM-BOT	

Las instrucciones básicas contienen normalmente relés, *latches*, temporizadores, contadores, manipulación de registros y puntos de entrada/salida, conversiones y funciones matemáticas.

Debido a que los PLC contienen un microprocesador, es posible la incorporación de funciones más sofisticadas que las utilizadas en la lógica de relés.

Las instrucciones expandidas incluyen funciones tales como movimiento de datos, movimiento de tablas, administradores de listas, aritmética con signo y doble precisión, cálculos matriciales y ejecución de subrutinas.

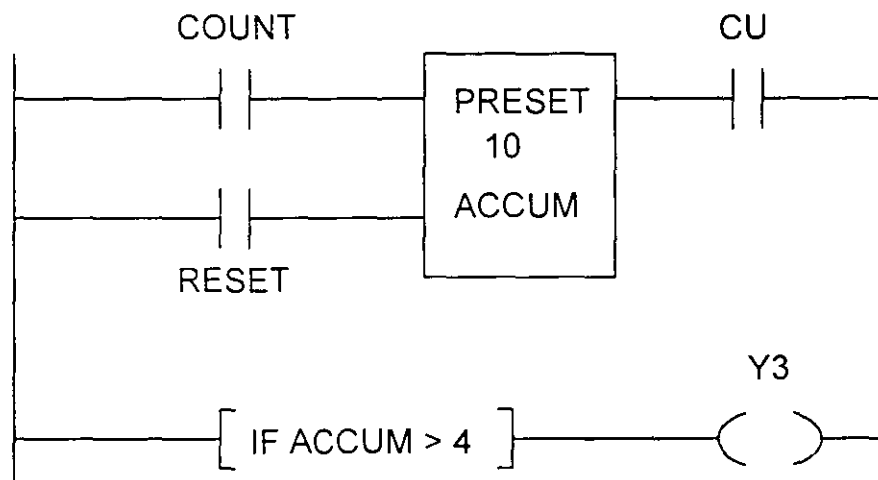
Instrucciones de movimiento de datos

Son utilizadas para copiar un registro o una porción de memoria a alguna localización de la memoria.

Instrucciones matemáticas avanzadas

Se incluyen operaciones aritméticas (suma, resta, multiplicación y división) de doble precisión con signo. Algunos PLC tienen otras funciones como raíz cuadrada y funciones trigonométricas.

La instrucción matemática GRATER THAN energiza la línea si la condición es verdadera. En el siguiente ejemplo, el acumulador ACCUM contiene la cuenta que lleva la caja *counter*. Si la cuenta es mayor que 4 se activa la salida Y3.

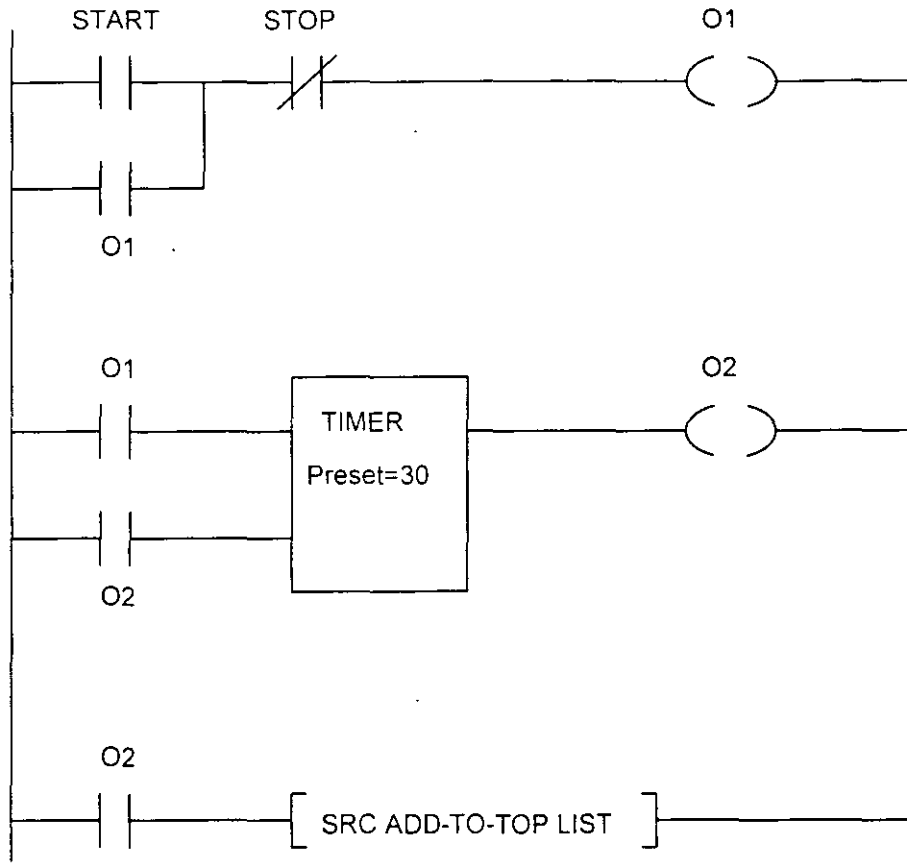


Instrucciones de tabla y de lista

Estas instrucciones permiten la creación y edición de tablas numéricas. De esta forma se pueden almacenar datos en una tabla, para que posteriormente puedan ser leídos y procesados. La información es almacenada en formato binario, pero puede ser convertida a decimal o a ASCII.

Las instrucciones de lista accesan los datos de forma secuencial con un puntero (FIFO o LIFO), mientras que las instrucciones de tabla permiten un acceso aleatorio.

Ejemplo:



En este ejemplo, al presionar el botón START se energiza la salida O1, que habilita el temporizador TIMER. Al pasar 30 segundos se activa la salida O2. Esta salida realimenta el TIMER para que reinicie su cuenta y activa la instrucción SRC ADD-TO-TOP LIST, que agregará al inicio de la lista el dato indicado por SRC, que podría ser un registro analógico de una temperatura de un proceso. Así se almacena cada 30 segundos la temperatura en una lista.

Instrucciones matriciales

Son utilizadas para realizar la operación OR (inclusivo y exclusivo) y AND de dos matrices binarias, el resultado se almacenará en una tercera matriz. Asimismo se pueden comparar dos matrices y tomar alguna decisión si son iguales.

Instrucciones de subrutina

Una subrutina es una porción del programa que puede ser ejecutada varias veces con distintos parámetros, desde distintas partes del programa.

4.3 Programación con bloques funcionales

Una de las formas más recientes de programar un PLC es a través de una carta gráfica de bloques funcionales. Este tipo de programación ha sido diseñado para describir, programar y documentar la secuencia del proceso de control.

En Europa, se ha comenzado a utilizar el lenguaje de programación llamado GRAFCET (creado en FRANCIA), orientado a la programación de PLC mediante bloques funcionales.

En la lógica combinacional, la programación con bloques funcionales es muy superior a otras formas de programación, mientras que los diagramas escalera y booleanos son mejores en lógica combinacional.

Debido a que hoy en día el control de procesos se programa principalmente con lógica secuencial, la programación con bloques funcionales será pronto el estándar para programar PLC.

Este lenguaje incluye un conjunto de símbolos y convenciones tales como pasos, transiciones, conectividades (también llamados enlaces) y condiciones.

Pasos

Son símbolos secuenciales individuales, representados por cuadrados numerados, los que pueden contener nemónicos que describen la función del paso.

Transiciones

Las transiciones describe movimiento de un paso a otro. Su representación es una línea horizontal corta.

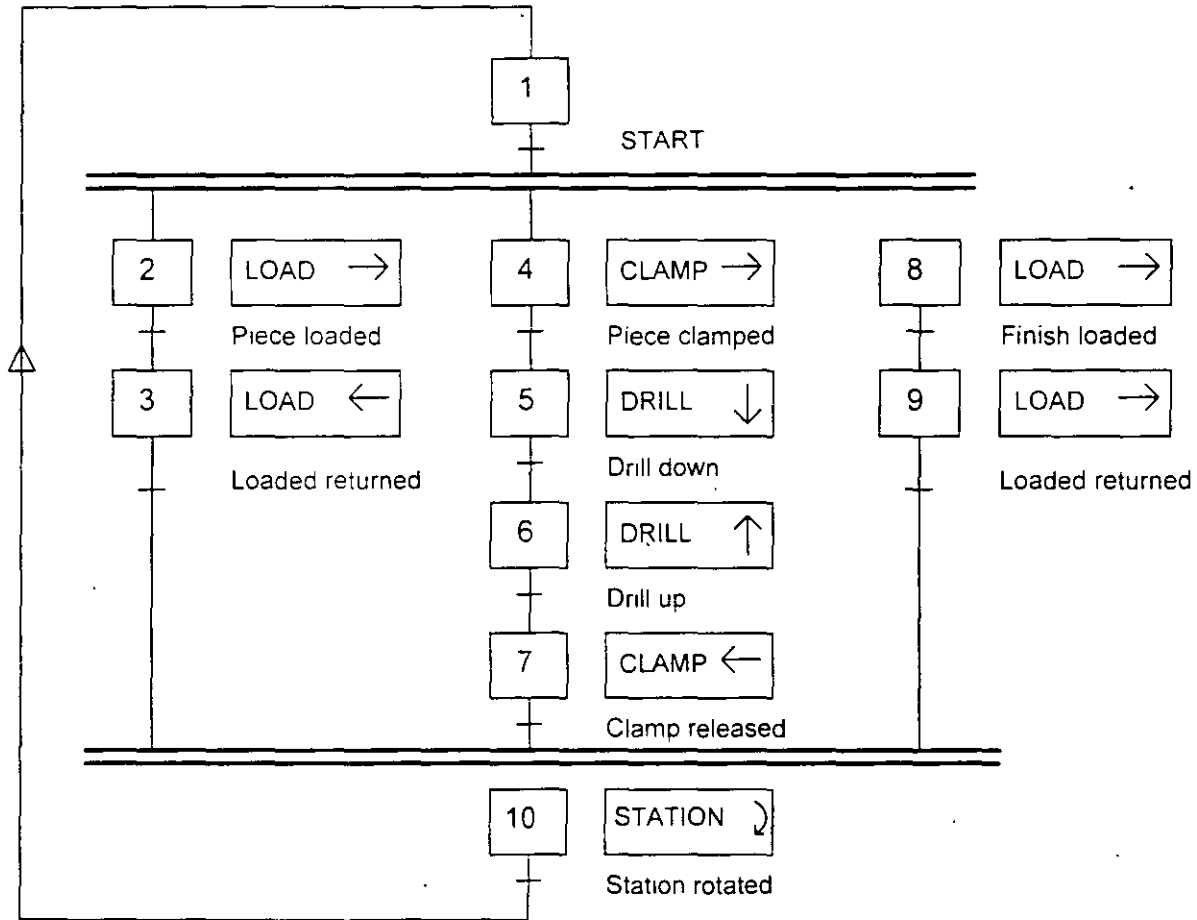
Enlaces

Los enlaces muestran el flujo del control, el que va desde arriba hacia abajo, salvo que se indique lo contrario.

Condiciones

Las condiciones están asociadas a las transiciones y deben ser escritas a la derecha.

Ejemplo



El ejemplo muestra lo fácil que puede ser programar y describir un control de un proceso de perforación por medio de un taladro.

La operación comienza con una pieza que es cargada, luego sujeta, perforada y removida, seguida de una estación que rota la pieza antes de que el proceso comience nuevamente.

Cada cuadrado contiene comandos de control que describen la entrada/salida discreta y/o las operaciones aritméticas que son programadas.

Este tipo de programación representa un gran vínculo entre el programador y el diseñador del proceso. Asimismo es una gran herramienta para:

- describir esquemáticamente el proceso,
- localizar fallas rápidamente,
- integrar fácilmente el sistema de control y el usuario

4.4 Programación con lógica booleana

La programación con lógica booleana incluye las funciones AND, OR y NOT para la lógica secuencial y las funciones TIMER, COUNTER y LATCH para la lógica combinacional.

Estas funciones son muy similares a las utilizadas en la programación con diagrama escalera. Específicamente:

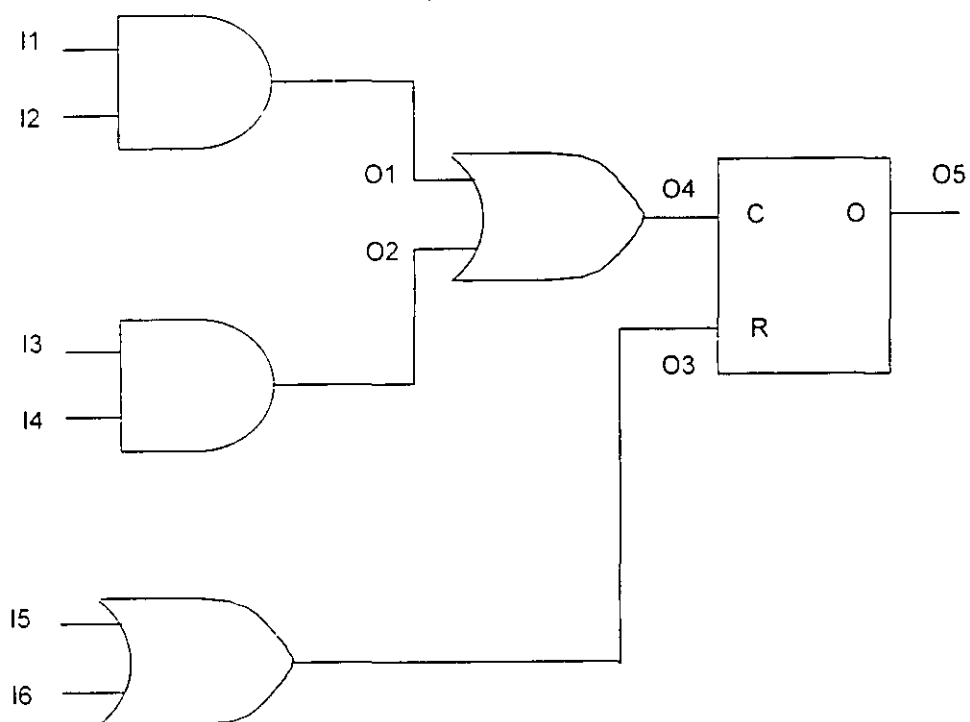
AND: Contactos en serie.

OR: Contactos en paralelo.

NOT: Contacto normalmente cerrado.

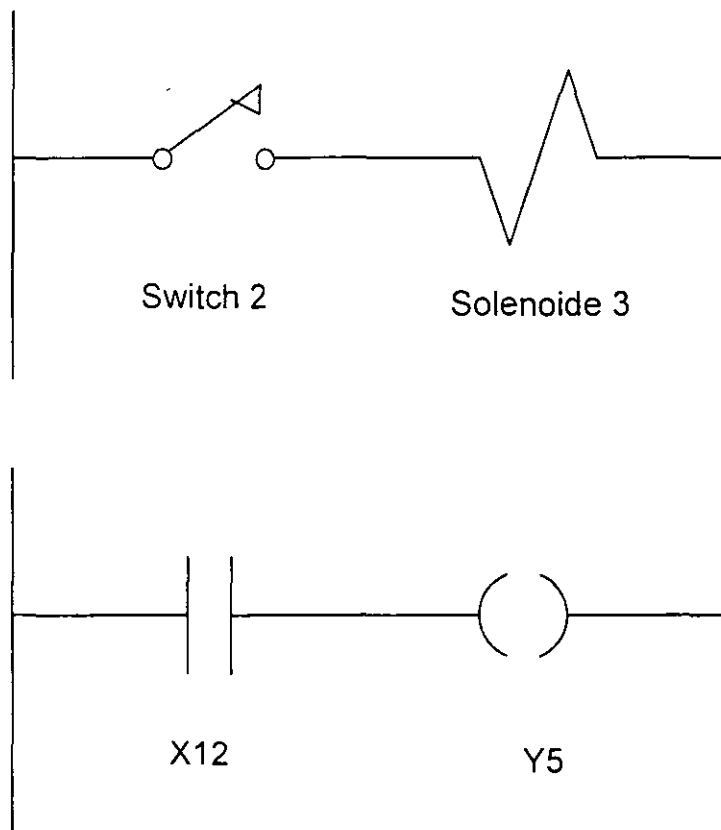
Las cajas tipo TIMER, COUNTER y LATCH son empleados de similar manera.

Algunas industrias europeas han optado por la programación booleana como estándar para el diseño del control lógico.



5. PROGRAMACIÓN DE UN PLC

Una forma usual de programar el PLC es utilizando el esquema *Relay Ladder Logic* (RLL), que es muy similar en forma e interpretación a los diagramas de escalera de relés.



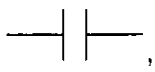
De acuerdo al diagrama escala, cuando el interruptor 2 se cierra, el solenoide 3 se energiza.

Para el programa RLL, el interruptor 2 está conectado a un terminal de módulo de entrada identificado como X12.

Cuando el PLC ejecuta el programa, envía una señal al terminal de módulo de salida identificado como Y5, el cual se encuentra conectado el solenoide 3.


En la figura del ejemplo, se utilizaron las instrucciones RLL de contacto y solenoide.

5.1 Contactos

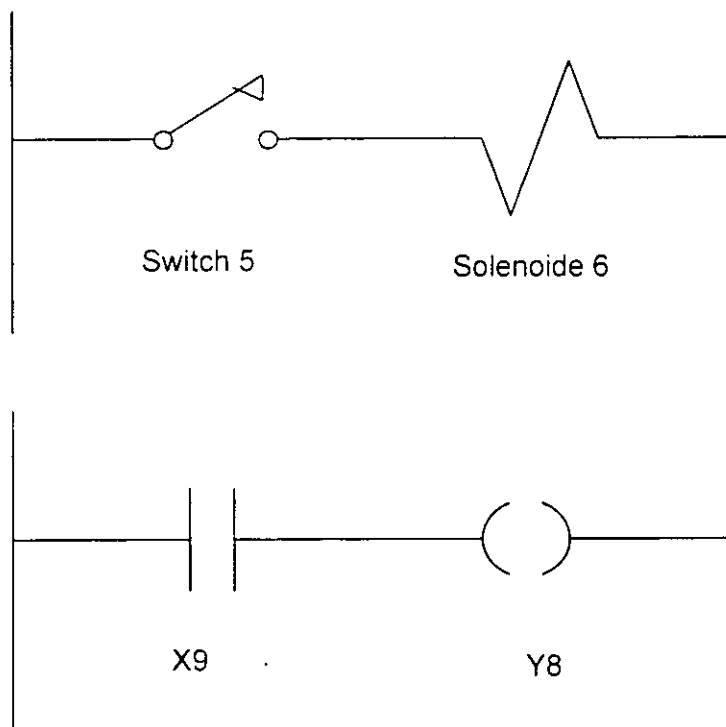
En el diagrama escala anterior, el solenoide está energizado cuando el interruptor se encuentra cerrado. En el programa RLL el *switch* se representa con el símbolo , que se denomina CONTACTO NORMALMENTE ABIERTO (NO). Esto significa que establece el flujo de energía cuando el interruptor se cierra. Si el *switch* se abre, no fluye corriente a través del contacto.

El módulo de entrada al cual se ha conectado el *switch*, detecta si éste se encuentra abierto o cerrado.

Un CONTACTO NORMALMENTE CERRADO (NC) puede representar la entrada de cualquier sensor, *switch* o el estado de la salida de otra etapa del programa.

Un contacto NC se representa con el símbolo , y conceptualmente invierte el estado de la entrada.

Por ejemplo en la figura, si el switch 5 está abierto, el solenoide 6 se encuentra energizado. En el programa RLL, el switch 5 está conectado al módulo de entrada X9 y la salida Y8 entrega el poder al solenoide 6.

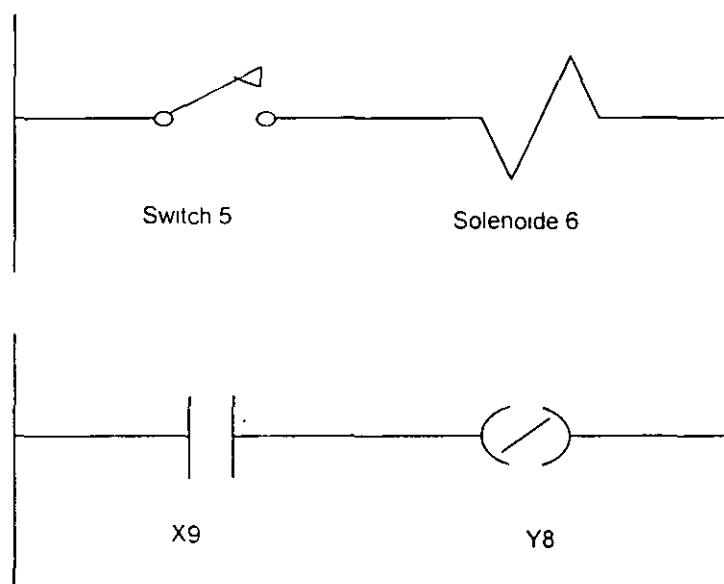


El PLC reconoce solamente si un contacto está abierto o cerrado, pero no puede determinar su concepción NA o NC. Por lo tanto, si la función debe ocurrir cuando el *switch* está abierto, la entrada de contacto se debe programar como normalmente cerrado NC.

5.2 Bobinas (solenoides)

En un Programa RLL, el dispositivo de salida es el solenoide cuyo símbolo es $\text{---}(\text{---})\text{---}$. Este símbolo es usado tanto para un dispositivo físico de salida externa, como para una salida interna que se emplea posteriormente en el programa.

La salida invertida se indica como $\text{---}(\text{---}/)\text{---}$. En este caso, al recibir la señal de salida, se desenergiza.

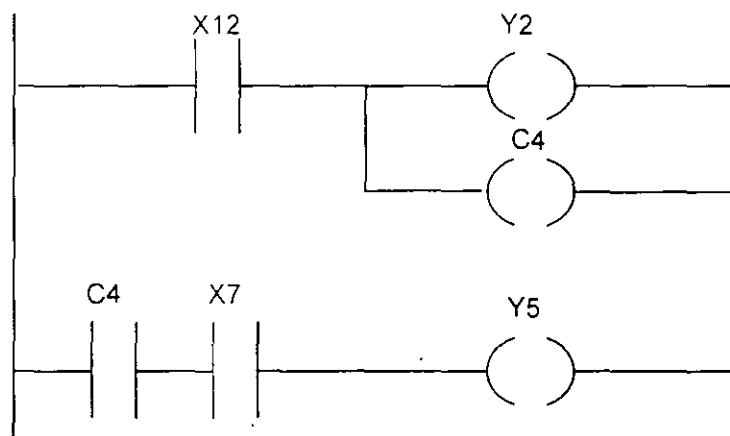


En la figura, si el *switch* 5 se cierra (entrada X9), entonces se desenergiza el solenoide 6 (salida Y8).

5.3 Relés de control

Estos elementos no existen físicamente como dispositivos de entrada o salida. Sin embargo, se ubican en la memoria del PLC y sirven como herramientas de programación para simular las entradas y salidas en el programa.

En un Programa RLL se representan mediante los mismos símbolos que las bobinas y los contactos.

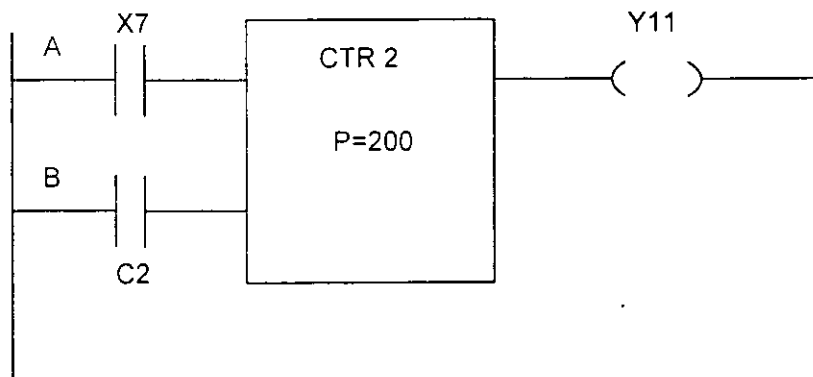


En el ejemplo, el relé de control C4 se energiza cuando X12 cambia a ON. El cambio de estado de C4 se registra en memoria. De igual forma, un relé de control se puede emplear como entrada en el Programa RLL; tal es el caso de C4 en la segunda línea. Si C4 y X7 se energizan, también lo hará Y5.

5.4 Cajas lógicas *RELAY LADDER*

Las cajas de instrucciones son funciones preprogramadas que amplían las capacidades de un programa más allá del conjunto de instrucciones RLL estándar.

Estas funciones permiten un empleo eficiente de la memoria del PLC y ahorran tiempo de programación.

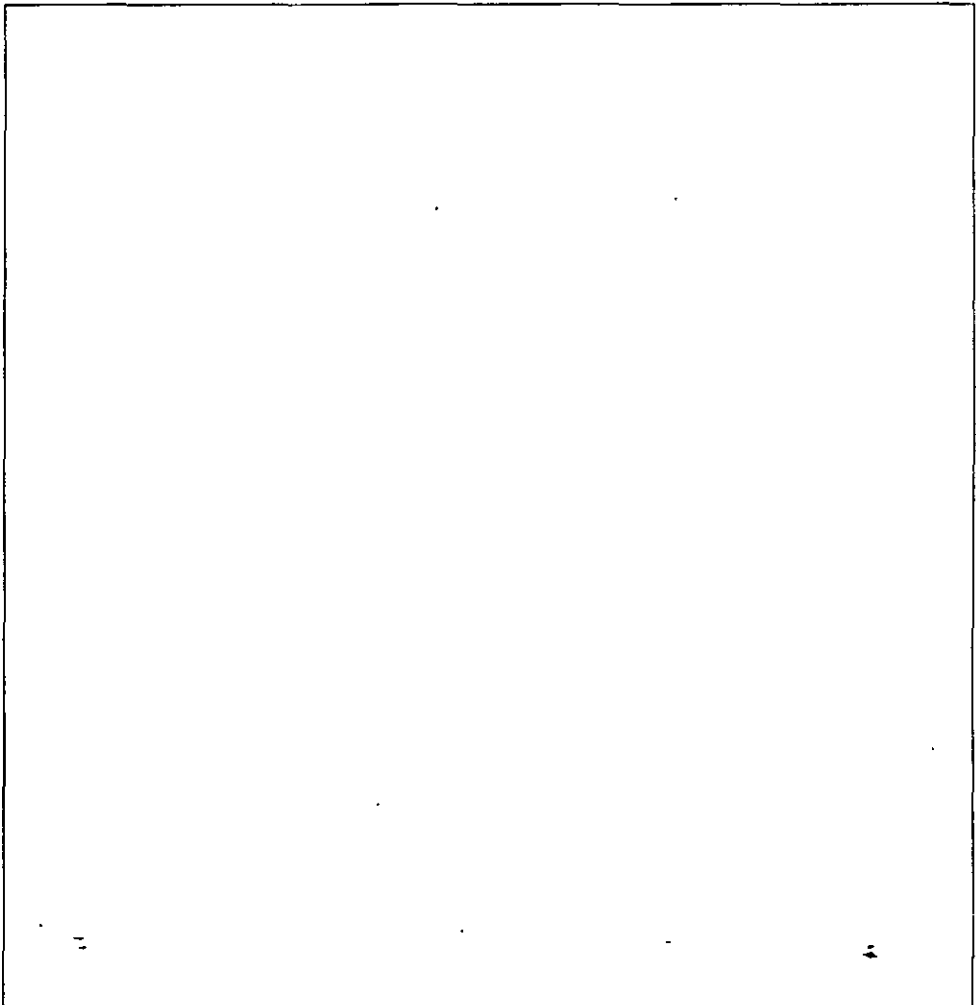


El contador de la figura corresponde a un ejemplo de caja de función.

El contador CTR 2 se habilita por medio de la línea de entrada inferior B. Las transiciones *Off-On* de la línea de entrada superior A se cuentan como pulsos. Una vez que la cuenta alcanza el valor prefijado, P=200, la bobina de salida Y11 es energizada.

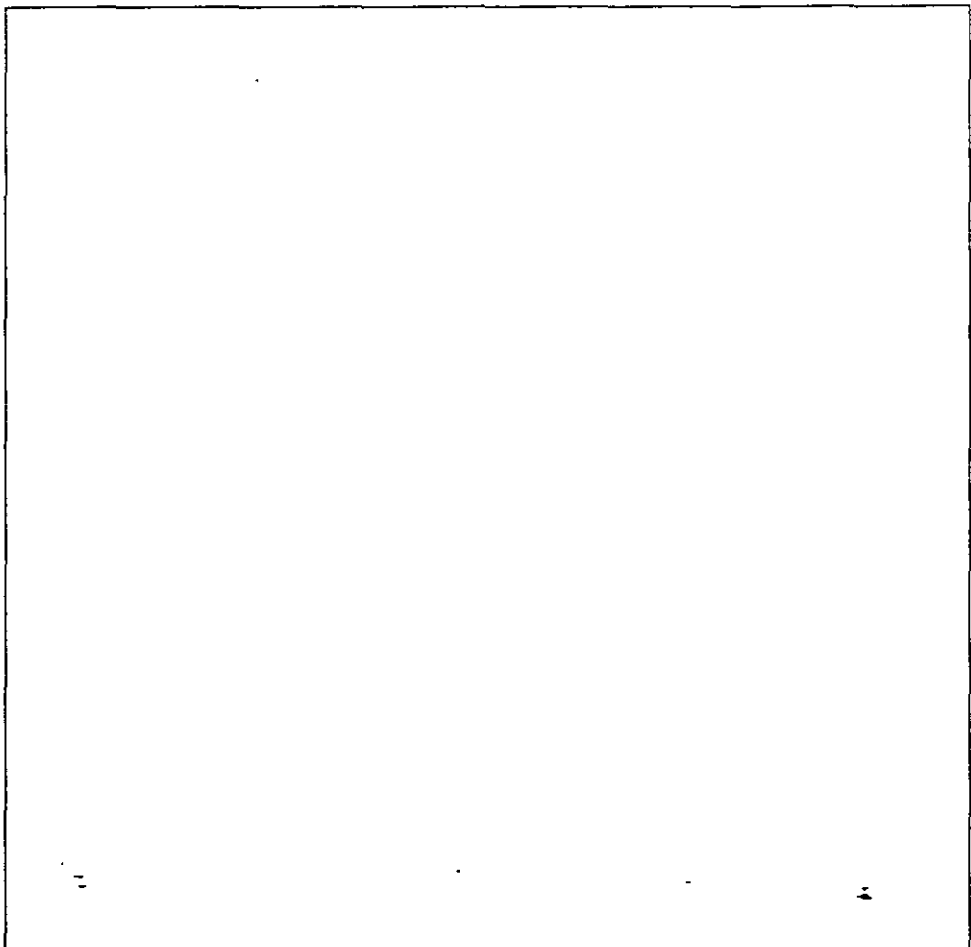
Ejercicio 1

Diseñar un diagrama RLD para un motor de 3 velocidades (V1, V2 y V3). El sistema cuenta con tres interruptores (S1, S2 y S3) que controlarán respectivamente cada una de las velocidades. Si están conectados 2 o más interruptores simultáneamente deberá activarse sólo la velocidad de menor rango. Adicionalmente el sistema debe contemplar un interruptor de apagado S0.



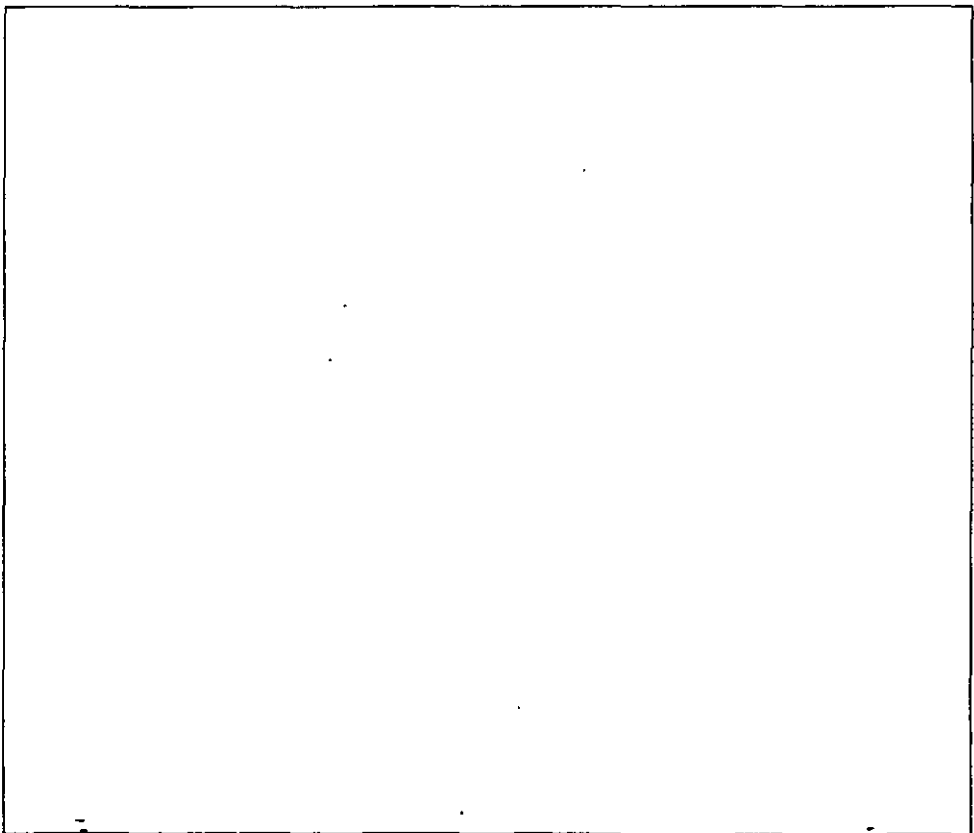
Ejercicio 2

Diseñar un diagrama RLD para un sistema que controle el encendido y apagado de un motor (M). El sistema debe contar con un pulsador de encendido (E) y uno de apagado (A). El motor se encenderá, y permanecerá encendido, cuando se presione el pulsador por un instante. De igual forma, el motor se apagará, y permanecerá apagado, cuando se presione el pulsador (A) por un instante.



Ejercicio 3

Diseñar un diagrama RLD para un sistema de alarma de una oficina. La oficina cuenta con una puerta (P) y una ventana (V). La alarma debe activar una sirena (S) cuando se abra la puerta o la ventana, y deberá permanecer activa si es que la puerta o la ventana se cierra. El sistema de alarma debe tener una luz indicadora Lp que señalará que fue la puerta la que activó la alarma y una luz Lv para la ventana. Adicionalmente deberá contar con un botón de encendido y apagado de alarma.



5.5 Diseño y documentación de programas

5.5.1 Definición de la aplicación

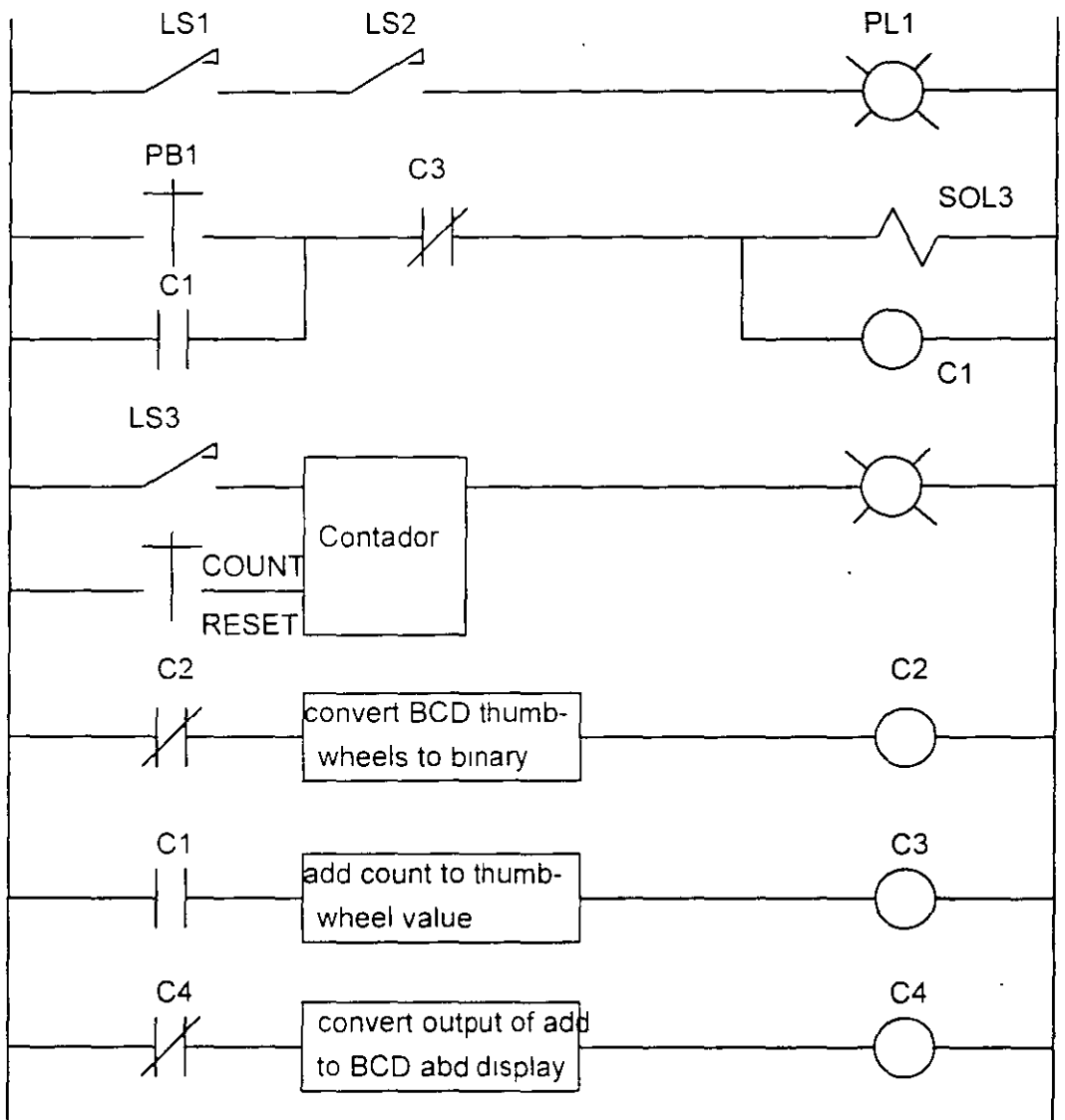
Lo primero que debe realizarse, es:

- Determinar las tareas que se requiere del equipo.
- Estimar los requisitos de tiempo.
- Estimar el orden en que se deben efectuar las tareas.

Previo a escribir el Programa RLL, se debe estar familiarizado con el equipo y su operación, y así determinar cómo automatizarlo.

5.5.2 Construcción del diagrama de relés RLD

Una vez definida la aplicación, se construye el diagrama RLD (Relay Ladder Diagram), donde se reúnen los requerimientos de operación. Este diagrama es una representación estándar de relés, interruptores, solenoides, motores, retardos de tiempo, lámparas, etc. que realizan la operación que se desea controlar.



Ejemplo de un diagrama RLD.

5.5.3 Asignación de identificadores

Se debe asignar la identificación a cada punto físico representado en el diagrama RLD, mediante una letra (X ó Y) y un número.

Una vez que el identificador se ha asignado al terminal, el dispositivo físico ahí conectado retiene su denominación hasta que se cambie a otro terminal.

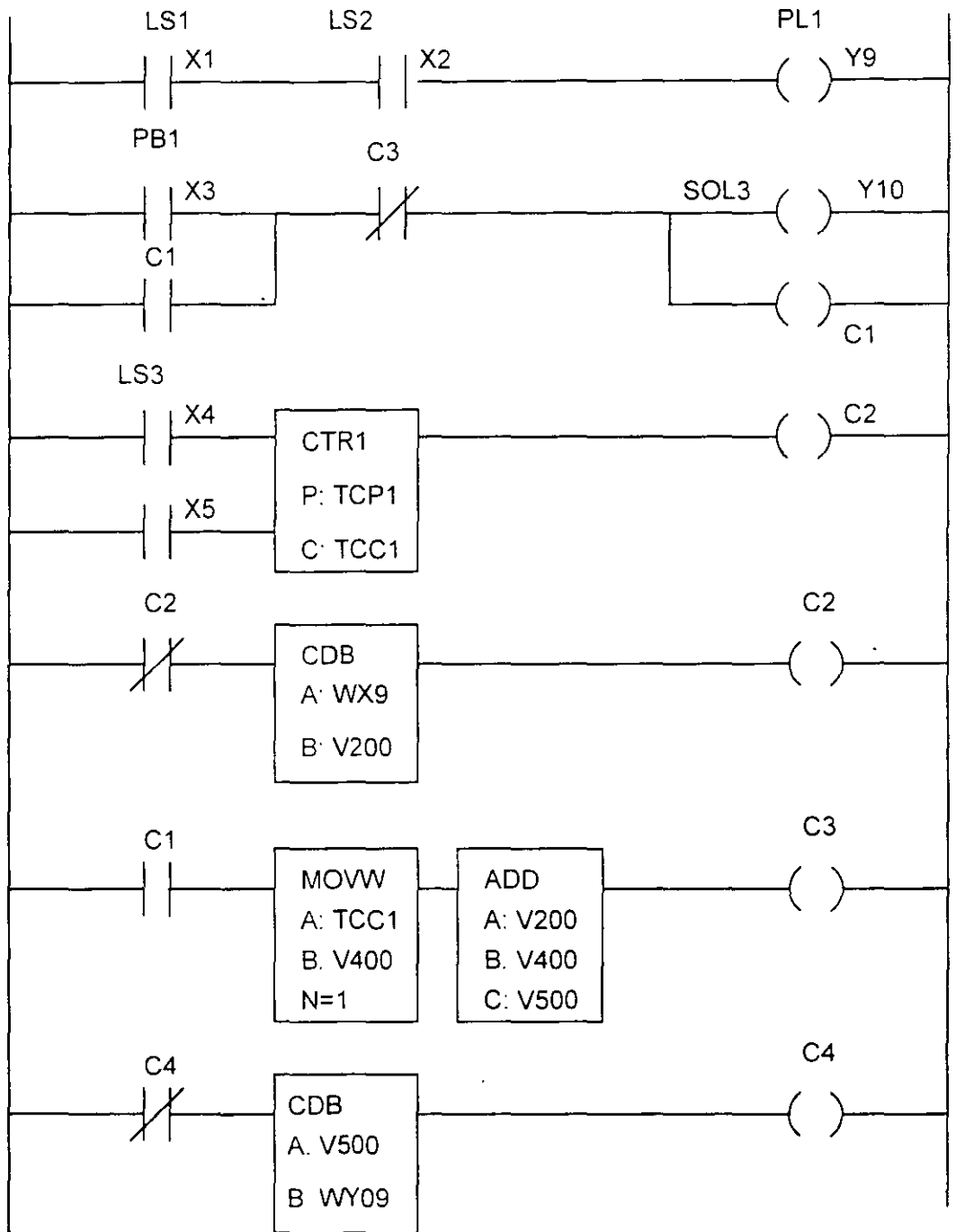
Base Assembly No. 01			
Identifier		Terminal Name	Slot Number
I/O Point type	Terminal designation		
X	1	LS1: part detect	1
X	2	LS2: part in place	1
X	3	PB1: cycle start	1
X	4	LS3: part count	1
	5		
	*		
Y	6	PL1: put part in place	2
Y	7	SOL3: clamp solenoid	2
Y	8	PL1: end of cycle	2
	*		
	*		

5.5.4 Construcción del diagrama RLL

Una vez asignadas las I/O y las localizaciones de memoria, se construye el diagrama RLL equivalente al diagrama RLD. Se cambian las designaciones mecánicas por los identificadores asignados.

El programa se almacena en memoria RAM. Cada instrucción se guarda, por lo general, como una palabra de 16 bits. El número de palabras por instrucción depende de:

- Tipo de instrucción y el número de referencia asignado.
- Localizaciones de variables V de memoria que se accesan dentro de la instrucción.
- Número de referencia de los relés de control que se accesan en la instrucción.
- Número de referencia de inicialización de temporizadores o valores que son accesados en la instrucción.



Ejemplo de un diagrama RLL.

Un bloque de memoria, memoria V, se asigna para las operaciones de cálculo interno.

Es recomendable llevar un registro con las posiciones de memoria V a medida que se diseña el programa. En la siguiente figura se muestra una manera conveniente de registrar las posiciones de memoria empleadas durante el diseño del programa.

		Comments
V	200	Binary value of thumbwheel input
V	201	
V	209	Value of current count in CTR1
V	213	Sum of thumbwheel input and CTR1 current
V	216	
V	217	
V	218	
V	219	

5.5.5 Diseño de diagnósticos en el programa

El PLC posee la capacidad de entregar información sobre el estado del software y del hardware. Esta información se guarda en formato de palabra y puede accesarse desde el dispositivo de programación.

De igual forma, las palabras de estado se pueden utilizar en el Programa RLL para facilitar la detección temprana de errores y dificultades en el hardware.

A manera de ejemplo se ilustra a continuación cómo un PLC Texas Instruments de las serie 500 informa sobre su estado de operación.

El PLC TI-500 posee un conjunto de palabras de estado de 16 bits. Cada palabra informa el estado de una operación específica. En muchos de los casos es necesario estudiar cada bit de la palabra. Algunas palabras de uso común son:

Palabra de estado 1 (STW01)

Informa sobre el estado de la batería del PLC, problemas de muestreo, puerta de comunicaciones, estado de los módulos I/O, y de módulos de funciones especiales. Bit = 0 indica que no hay problema, Bit = 1 señala que hay problema.

Bit	Problema
15	Batería baja.
14	Tiempo de scan muy corto.
13	Falla en puerta de comunicaciones.
12	Falla en I/O.

Palabra de estado 2 (STW02)

Informa sobre el estado de hasta 16 *racks*. LSB corresponde al *rack* 0 y MSB al *rack* 15. El bit respectivo toma el valor 1, cuando el *rack* ha fallado o no está, y 0 cuando no hay problema.

Palabra de estado 6 (STW06)

Informa el estado de la programación de la EPROM/EEPROM del PLC.

Palabras de estado 7 a 9 (STW07 - STW09)

STW07 entrega la dirección absoluta de memoria donde se detecta el primer error al tratar de programar la EPROM (EEPROM).

STW08 muestra el valor calculado de *checksum* para el programa RLL almacenado en la memoria EPROM. Este número se emplea para verificar que las copias de un programa sean iguales.

STW09 muestra el valor calculado de *checksum* para la EPROM completa: Programa RLL e información de memoria de configuración de I/O.

Palabra de estado 10 (STW10)

STW10 muestra, en código binario, el tiempo de muestreo del PLC.

Palabras de estado 11 a 18 (STW11 - STW18)

Informan el estado de los módulos I/O instalados en los *racks*. Cada bit corresponde a un módulo del *rack*. Bit = 0 indica que no hay módulo en el slot o está funcionando bien. Bit = 1 señala que el módulo del slot está en mal estado. Si existe un módulo que no corresponde a su configuración, éste se informa como si estuviese fallado.

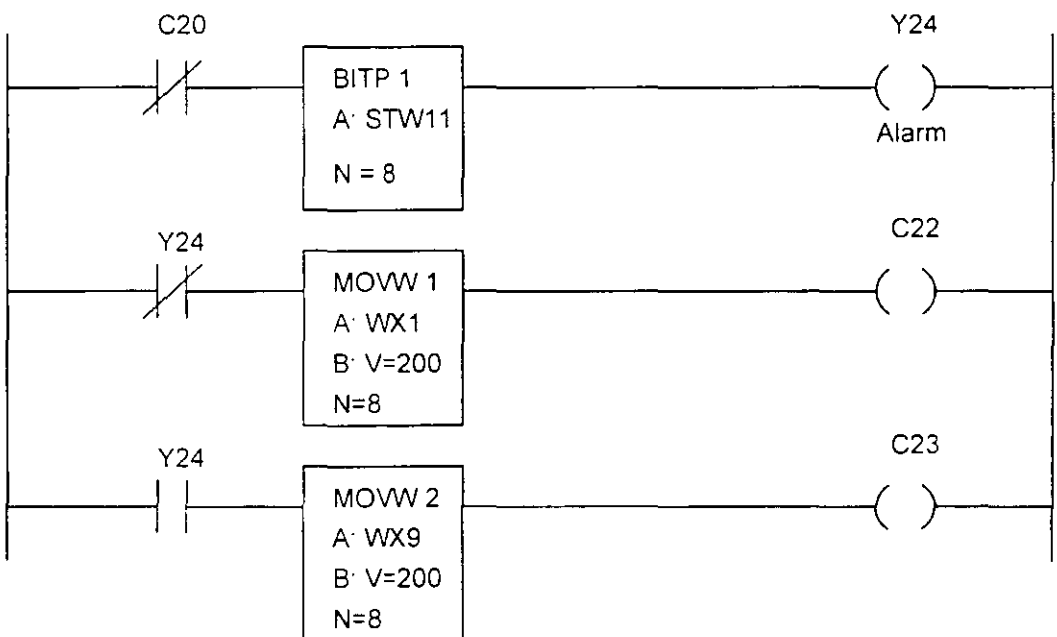
En una configuración distribuida, si algún *rack* pierde la comunicación con el PLC, en STW02 aparecerá 1 para el correspondiente bit y los bits en STW12 - STW18 muestran 0, incluso si los módulos del *rack* están fallados o mal configurados.

5.5.6 Empleo de las palabras de estado en el programa RLL

El programa que se muestra a continuación muestra un método para desconectar un módulo fallado y conectar el módulo de respaldo, ubicado en la misma base.

Módulo 1 en slot 1	Rack 1: WX1-WX8-STW11	BIT 8
Módulo 2 en Slot 2	Rack 2: WX9-WX16-STW12	BIT 15
Módulo 3 en Slot 3	Rack 1: Y17 - Y24	

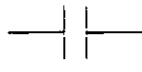
Y24 : Módulo de alarma de falla

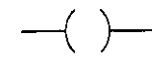


El estado del módulo de I/O N°1 se verifica con la instrucción BITP. Si BITP indica falla (bit 8.de STW11 en 1), Y24 se conecta (alarma). El programa desconecta el segundo módulo, para conectar el módulo de respaldo.

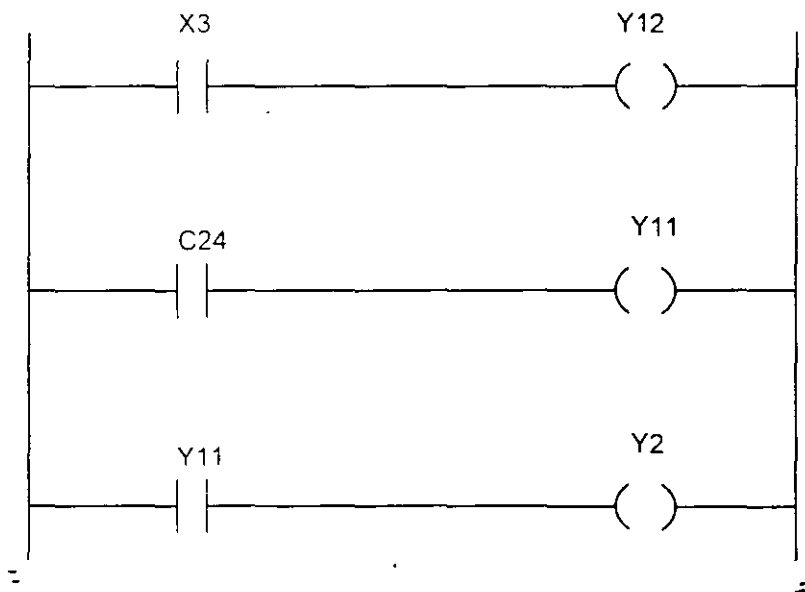
5.6 Funciones RLL

5.6.1 Contactos y bobinas

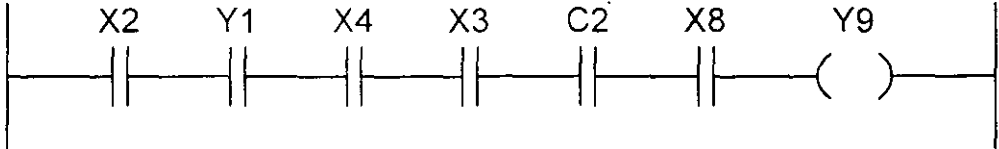
Representación de contactos: 

Representación de bobinas: 

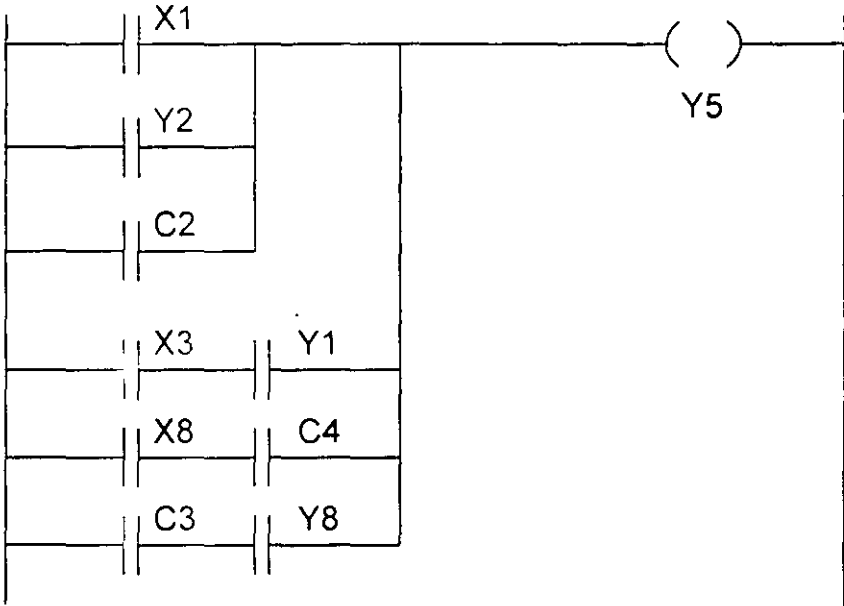
Las entradas y salidas físicas de módulos I/O se designan como X_n y Y_n respectivamente, donde n es el número de referencia, por lo general $n: 0-1023$. Mientras que los relés de control utilizados en lazos intermedio se denotan como C_n .



Conexión serie de contactos: ANDs



Conexión paralelo de contactos: ORs



5.6.3 Funciones en caja

Se representan por un rectángulo que contiene en su interior un identificador descriptor de la función (código nemónico) seguido por un número de de referencia. Dependiendo de la instrucción, este número puede variar de 1 a 32767.

Línea de 1 entrada con caja pequeña

Debe tener a lo menos un contacto antes de la caja y no más

6. ESPECIFICACIONES DE UN PLC INDUSTRIAL

Elementos a suministrar

- Unidad procesadora
- Unidad de memoria
- Módulos de entrada/salida
- Interfaz de comunicación
- *Rack* de entrada/salida
- Fuente de poder
- Periféricos
- Software de programación y documentación
- Manuales de hardware
- Repuestos
- Cables y conectores entre *racks*, fuente de alimentación y módulos de comunicación.

Códigos estándares

- IEC International Electrotechnical Mission
- ANSI American National Standards Institute
- IEEE Institut of Electrical and Electronics Engineers
- NEMA National Electrical Manufacture Association
- NEC National Electrical Code
- ISA Instrument Society of America
- ASME American Society of America
- ISO International Standard Organization
- SEC Superintendencia de Servicios Eléctricos y Combustibles

Condiciones de operación

- Ubicación geográfica de la aplicación
- Ambiente
- Instalación Interior-Exterior
- Altura
- Temperatura ambiente
- Humedad relativa
- Vibraciones
- Confiabilidad en la operación
- Suministro eléctrico

Requerimientos técnicos

- Modularidad y flexibilidad
- Capacidad de entradas/salidas
- Capacidad de la fuente de alimentación
- Tipo de señales a procesar
- Tiempo de ciclo
- Lenguaje de programación
- Indicadores: fuente, batería, status
- Tipos de instrucciones
- Características de las señales de entrada (discretas, analógicas, niveles de señal)
- Precisión de los conversores análogo-digitales
- Interfaz de operación

7. CONSIDERACIONES DE INSTALACION Y MONTAJE

7.1 Preparación del lugar de instalación

- Definición de los requerimientos de control.
- Determinar el número de PLC requeridos.
- Determinar disposición de paneles y tierras.

Los requerimientos de control se definen en términos del número de entradas y salidas. Posteriormente, se calculan los módulos de I/O y los *racks* que se necesitan. Una vez que se conocen los PLC, módulos de I/O y *racks* requeridos, se deben determinar la potencia necesaria para el funcionamiento correcto del PLC.

La potencia total requerida en la instalación, considerando el PLC, módulos de I/O, y módulos controladores, no debe exceder la capacidad disponible de la fuente de poder.

7.2 Consideraciones de seguridad

Al diseñar el sistema, se deben tener en cuenta las condiciones de seguridad del personal durante fallas. Los equipos conectados al PLC deben incluir *interlocks* y *switches* de seguridad, que prevengan la operación al producirse una falla.

- Debe existir un medio para desconectar la alimentación de energía a las cargas (salidas), independiente del PLC, para operaciones de rutina
- Debe existir un medio para desconectar la alimentación de energía a las salidas, para condiciones de emergencia.
- Se deben utilizar circuitos *by-pass* externos, para operaciones de partida o inicialización (cargas críticas).

7.3 Encapsulado (*Enclosure*)

Requerimientos mínimos

- Fácil acceso a componentes.
- Potencial de tierra común para el gabinete.
- Instalación en rieles o paneles verticales de seguridad.
- Cumplir estándares o normas eléctricas.
- Protección EMI.
- Restringir acceso a los equipos.
- Protección contra polvo y suciedad.
- Normas NEMA.

7.4 Consideraciones de temperatura

Se debe asegurar un adecuado flujo de aire, de modo que se obtenga una buena refrigeración del equipo.

Si la temperatura ambiente es alta, se debe utilizar ventilación forzada o acondicionamiento de aire. La temperatura máxima de operación típica es 60° C.

7.5 Consideraciones eléctricas

7.5.1 Tierras

Para obtener una operación adecuada, es fundamental contar con un buen sistema de conexión a tierra. Se recomienda la utilización de cable trenzado de cobre N°12 AWG o de mayor grosor en el retorno de tierra.

Algunas reglas para lograr un buen contacto eléctrico:

- Se deben emplear terminales adecuados en los extremos de los cables de tierra
- Es recomendable utilizar pernos de cobre para realizar la conexión al punto de tierra.
- La pintura, recubrimientos y el óxido impiden un buen contacto en los puntos de tierra. Se deben remover y emplear golillas dentadas para asegurar una buena continuidad y baja impedancia.

7.5.2 Alambrado

Algunas consideraciones que se deben tener en cuenta en el alambrado:

- Emplear cables de largo mínimo.
- No añadir cables.
- Evitar la proximidad de cables de alta potencia.
- Instalar cablería de entrada, salida y de otro tipo en paneles separados.
- Cuando sea posible, canalizar por separado los cables con señales DC y AC.
- Una impedancia de 0.1Ω o menor debe haber en la conexión a tierra de todos los componentes del sistema.
- Utilizar guías de cable.
- Proteger los cables desnudos.
- No utilizar el mismo cable de retorno de alimentación cuando las líneas son muy largas; de esta forma se minimiza la caída de voltaje.

7.5.3 Minimización del ruido eléctrico

Fuentes de ruido

El ruido puede ser conducido a través de los cables de señal o de alimentación, o puede ser irradiado por ondas electromagnéticas.

El acoplamiento electrostático se produce a través de las capacitancias parásitas existentes entre la línea de ruido y la línea de alimentación o señal. Este es el caso típico cuando se canalizan cables largos en un mismo *conduit*.

El acoplamiento magnético ocurre a través de las inductancias mutuas parásitas entre líneas

El ruido electromagnético irradiado es generalmente de alta frecuencia. Se debe tener especial cuidado en el sistema de control y su alambrado, ya que pueden actuar como antenas.

Las fuentes primarias de ruido en ambientes industriales son:

- motores grandes.
- máquinas soldadoras.
- contactores (*switch* con cargas electromagnéticas).
- máquinas de estado sólido

Eliminación del ruido

El empleo de supresores de ruido *snubbing* permite reducirlo en su origen. Son aplicables en dispositivos comandados por contactos mecánicos, y suprimen el arco en los contactos eléctricos (cargas inductivas).

Un tipo alternativo de supresor se logra con circuitos RC o varistores.

Aislación del ruido

Otra forma de manejar el problema de ruido, consiste en aislar el dispositivo que presenta problemas de ruido, de los cables y componentes electrónicos. Adicionalmente y en casos extremos, se emplean escudos electrostáticos.

Una medida complementaria, especial para cables con señales de valores bajos (TTL), se consigue con protecciones de malla y trenzado (12 vueltas/pie). Además, se debe mantener la separación física con los emisores.

8. APLICACIONES DE PLC

8.1 Sistema de control de un túnel

En un túnel de 2.5 km existen cuatro variables que deben ser controladas:

- Calidad del aire
- Velocidad de los vehículos
- Sentido del flujo vehicular
- Nivel del tanque de agua para apagar incendios

Arquitectura del sistema de control del túnel

Para llevar a cabo el control del túnel, se utiliza un sistema de control distribuido, en el que se emplea:

- un PLC con un *racks* remoto por cada 500 metros. Los *racks* remotos están encargados de tomar las señales de los sensores (medidores de temperatura, presión, CO, CO₂, opacidad, presencia de vehículos y nivel de agua del tanque) y accionar los dispositivos de salida (ventiladores, avisos luminosos, semáforos y bomba de agua).
- una red de computadores conectados al PLC a través de una de comunicación. Los computadores sirven de interfaz hombre - máquina. En ellos se visualizan los despliegues que indican el estado de operación del túnel.

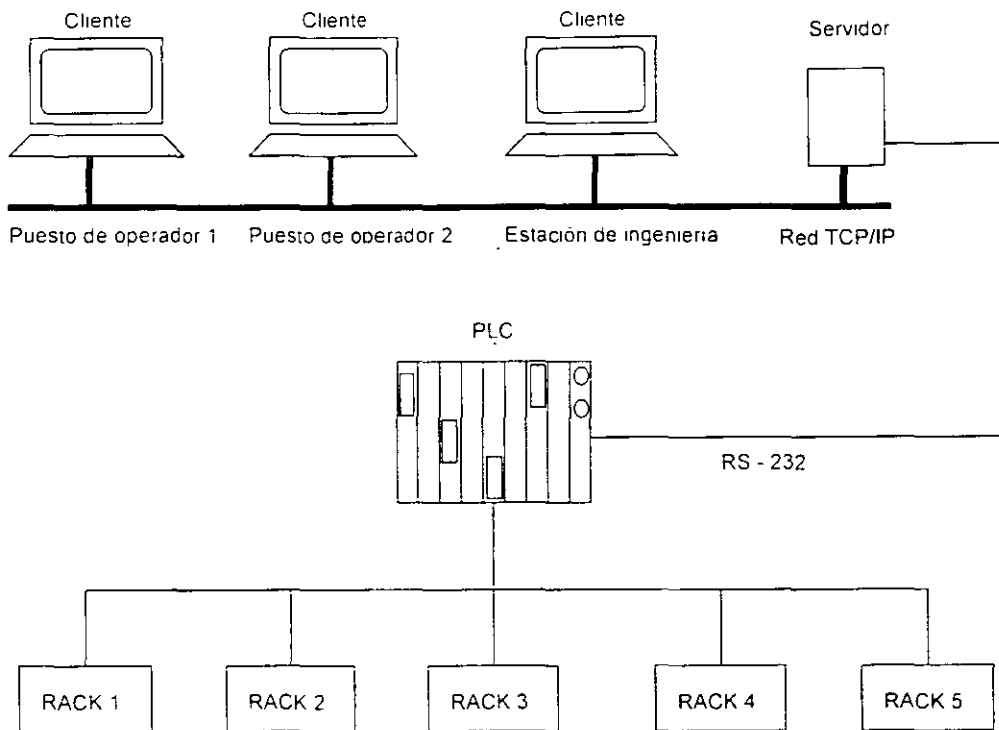


Diagrama de la arquitectura del sistema de control del túnel

Control de la calidad de aire del túnel

Es necesario que el nivel de la calidad del aire, que está en el interior del túnel, esté dentro de los límites adecuados para la salud de los usuarios. Con este fin se instalan a lo largo del túnel medidores de presión, temperatura, CO, CO₂ y opacidad, los que sirven para calcular una cifra de mérito que indica el grado de calidad del aire. Para mejorar la calidad del aire, el túnel dispone de ventiladores ubicados en la entrada y en la salida, los que renuevan el aire contaminado.

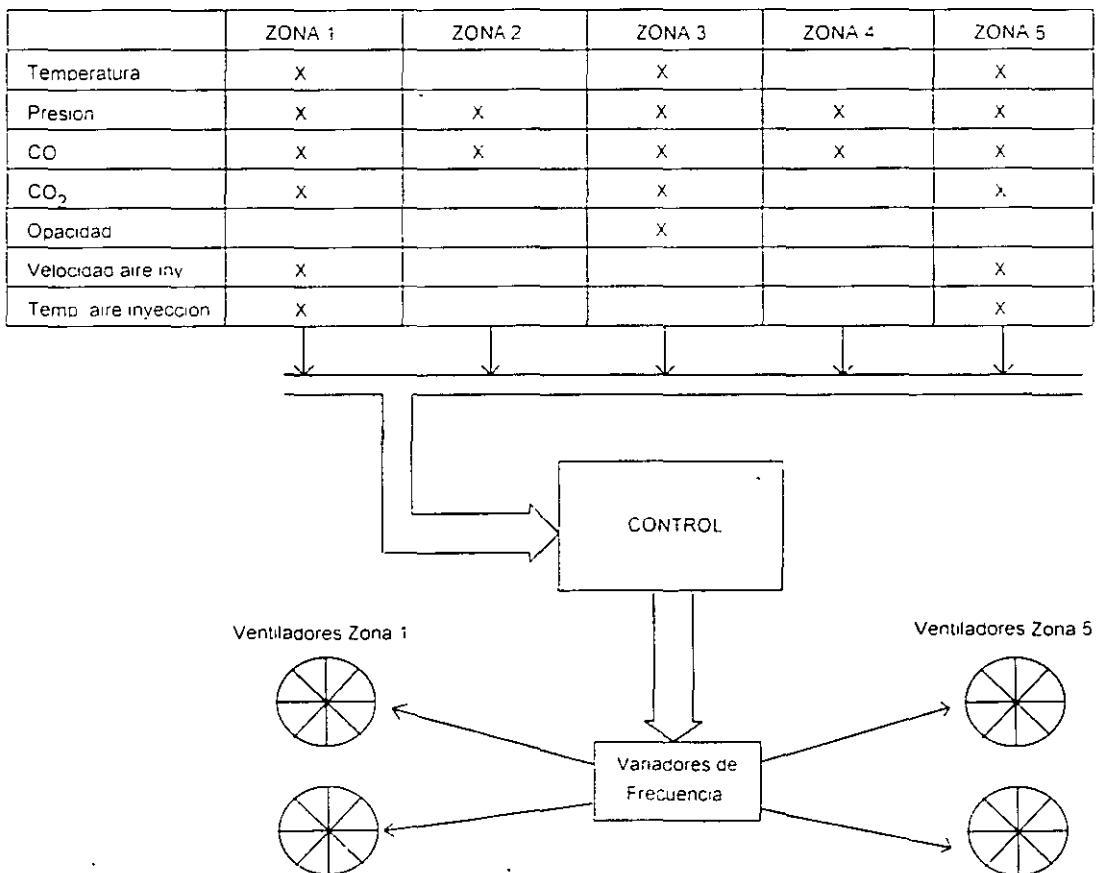


Diagrama de control de calidad de aire del túnel

Control de la velocidad de los vehículos del túnel

Para evitar accidentes, los vehículos que circulan al interior del túnel no pueden exceder la velocidad límite (por lo general 60 km/h). Con este fin se instalan a lo largo del túnel sensores de presencia de vehículos. Estos sensores son ubicados en parejas longitudinalmente al eje de circulación. La velocidad es calculada a partir de la diferencia de tiempos entre la detección de presencia vehicular del primer sensor y el segundo. Al existir un exceso de velocidad se deben encender avisos luminosos localizados al lado derecho de la vía que le indican al conductor que disminuya su velocidad.

Adicionalmente, se debe calcular la cantidad neta de vehículos presentes en el túnel, detectando atascamientos.

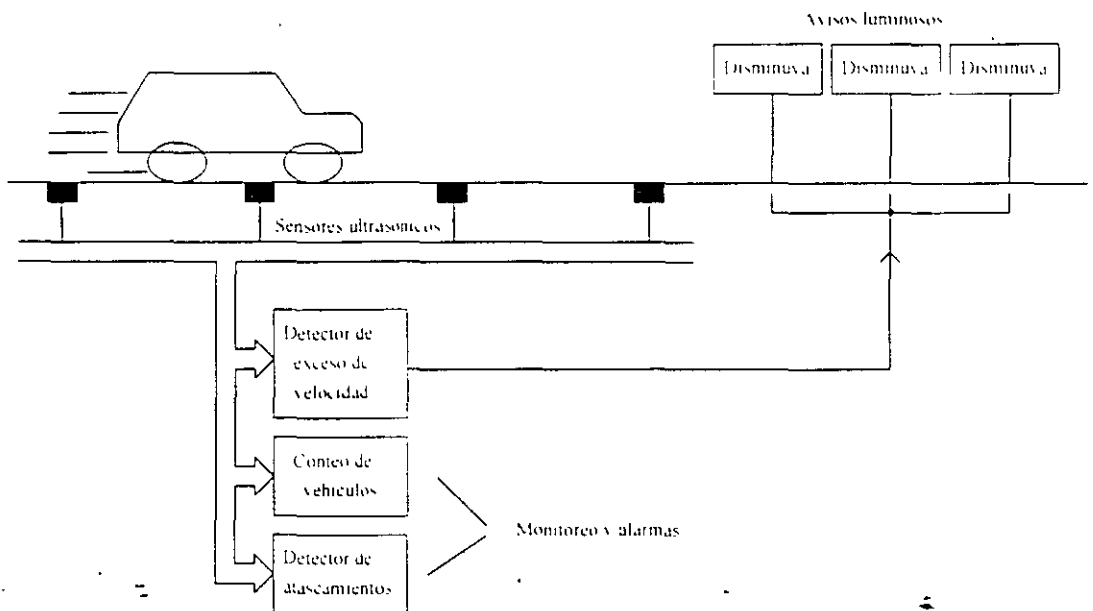
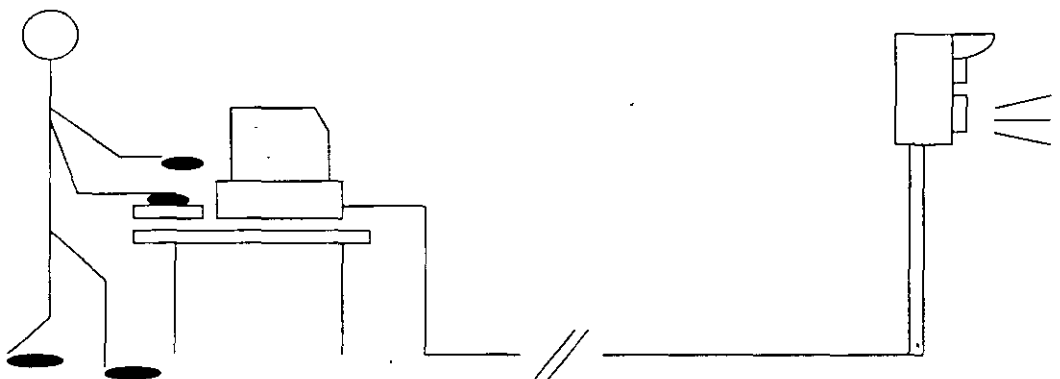


Diagrama de control de velocidad vehicular

Control del sentido del flujo vehicular del túnel

El tránsito normal de un túnel es el de doble vía. Sin embargo, debido al incremento del tráfico (comúnmente al inicio o final de días festivos), las autoridades de tránsito disponen que el flujo vehicular debe ser sólo en el sentido de mayor tráfico. Con este fin se dispone de semáforos en la entrada y en la salida que indican con luces verdes y rojas el sentido del tránsito de cada pista. Los semáforos son controlados por un operador.



Esquema de control de semáforos

Control de nivel del tanque de agua para apagar incendios

Se debe controlar el nivel de agua del estanque principal de agua, que se utiliza en caso de incendios. Este control se lleva a cabo con dos pozos (pozo 1 y pozo 2) y dos bombas de agua. Tanto los pozos como el estanque poseen sensores de niveles, los que gobiernan las bombas de agua apagándolas y encendiéndolas

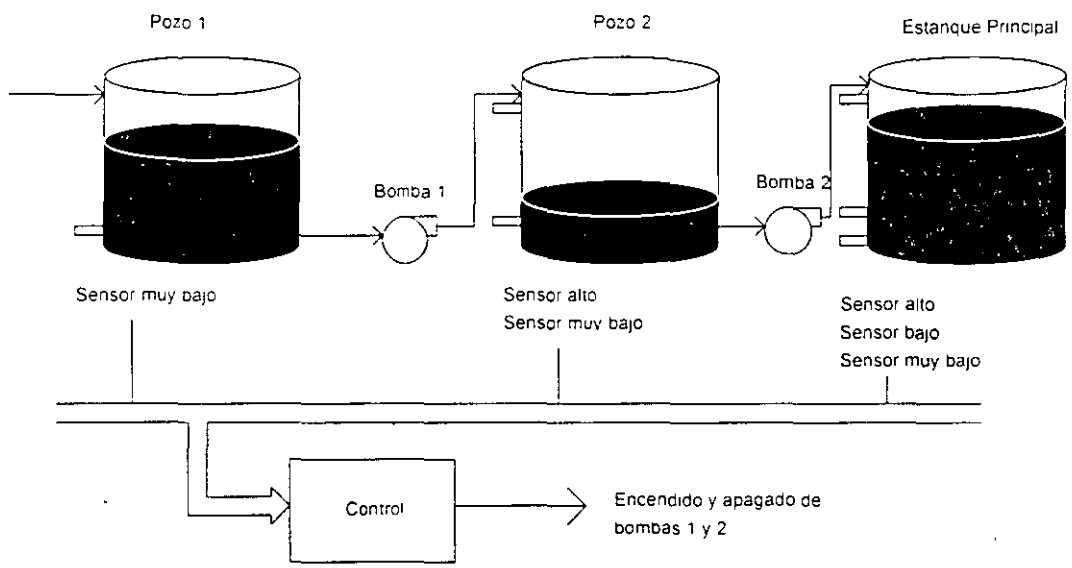


Diagrama del llenado de tanques de agua para incendio

Red de computadores

La red de computadores opera con un software de automatización SCAUT-3G sobre plataforma UNIX, en una red TCP/IP. El sistema opera con una configuración cliente - servidor de la siguiente manera:

- Un servidor, en el cual está instalado SCAUT-3G trabajando bajo el sistema operativo UNIX. El servidor tiene como funciones atender las comunicaciones (de la red y del PLC), mantener vigente la información en la base de datos en tiempo real, llevar registros históricos y estadísticos, analizar la información y realizar los monitoreos y controles respectivos.
- 3 clientes (o más), en los cuales trabajarán las interfaces de operación remotas de los puestos de operadores 1 y 2, y la estación de ingeniería, bajo ambiente Windows.

8.2 Supervisión de oleoductos

En el estado de Florida se desarrolló un sistema para monitorear y proteger un oleoducto que cruza los *Everglades*.

El diseño fue realizado utilizando un PLC y un sistema de comunicación digital por telemetría, alimentados de paneles solares.

Para detectar pérdidas en la línea o interrupciones en el suministro, se utiliza una combinación de "inteligencia" local y remota, que genera alarmas y, si las condiciones lo requieren, detiene el oleoducto.

Las funciones del sistema son:

- operar remotamente las bombas de impulsión (Diesel y eléctricas) de cada estación de bombeo
- supervisar los flujos
- calcular totales e índices operacionales

Cada unidad remota PLC actúa en forma independiente ante una situación anormal.

Monitoreo de pérdidas

La detección de pérdidas en las líneas de crudo se realiza bajo condiciones de estado estacionario.

El método consiste en establecer varias ventanas móviles de totalización de flujo, con diferentes lapsos de duración, para detectar en forma progresiva las mínimas discrepancias en el flujo sobre los períodos de tiempos mayores.

Cada PLC tiene conectado uno o más medidores de desplazamiento positivo. A través de ellos, cada uno ejecuta los cálculos de detección de pérdidas, considerando cada segmento de oleoducto en servicio (número de bombas en línea, estado de las válvulas, etc.)

El programa del PLC inhibe temporalmente la detección de pérdidas durante las situaciones transientes (partida o parada de bombas, etc.), informando de esta acción al operador del sistema, para que monitoree en forma manual.

La interfaz de operación se basa en un terminal gráfico de color, donde se despliegan reportes de estado de cada parte del sistema, se visualizan las alarmas y se cambian los parámetros de configuración (reinicialización de totalizadores, ajuste de factores de medición en los sensores).

9. EXPERIENCIAS DE LABORATORIO

9.1 Control de la marcha de un motor eléctrico

Se desea controlar la operación de un motor eléctrico que puede girar en sentido directo indefinidamente y en reversa sólo durante 60 segundos.

Para tales efectos se dispone del siguiente programa para el PLC

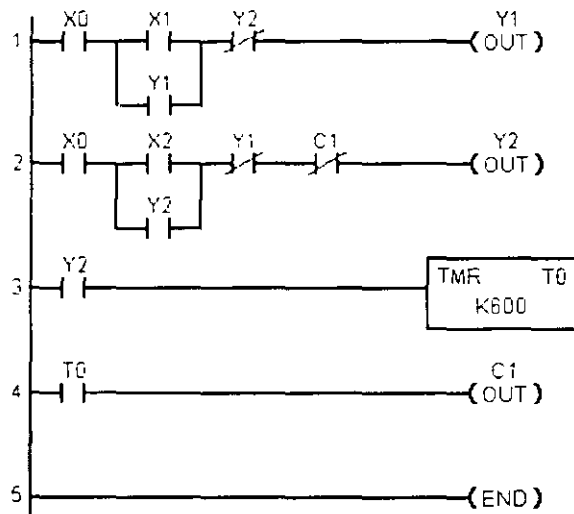


Diagrama escalera del control de marcha del motor

Analice el programa RLL diseñado para controlar el motor. Se utilizó la siguiente convención para las variables:

X0 : ON-OFF

X1 : Partida en sentido directo

X2 : Partida en reversa

Y1 : Motor girando en sentido directo

Y2 : Motor girando en reversa

9.2 Reconocedor de productos en una cinta transportadora

Una empresa manufacturera desea automatizar el sistema de cintas transportadoras. el que se describe por el esquema de la figura siguiente.

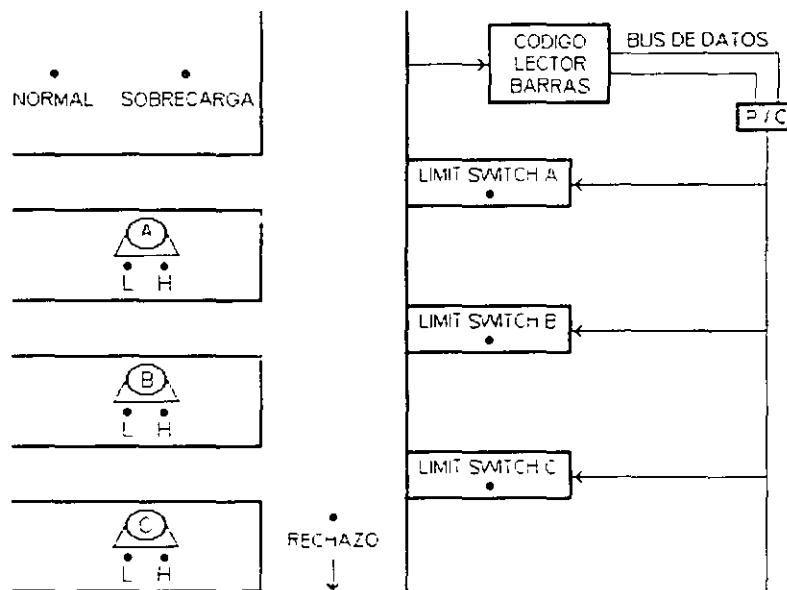


Diagrama del sistema de cintas transportadoras

El proceso contempla distintos productos que son transportados por la cinta central desde una etapa anterior de fabricación. Los productos son identificados por una lectora de código de barras para ser seleccionados. Para esto se entrega la lectura correspondiente en tres bits al PLC, donde

se decide cuál de los *switches limits* (A, B o C) debe ser accionado para que el producto siga su recorrido por la cinta transportadora lateral A, B o C respectivamente. Además, existe la posibilidad de rechazo en caso de que el código leído sea incorrecto o bien mientras no se reciba ningún producto.

a) Diseñe un programa RLL para controlar el proceso Utilice la siguiente convención para las variables.

X0 : ON-OFF (comienzo y fin de la simulación)

Código del producto	Bit 1	Bit 2	Bit 3
Variable	X1	X2	X3
Producto tipo A	1	0	0
Producto tipo B	1	0	1
Producto tipo C	1	1	0

Y0 : Limit switch A

Y1 : Limit switch B

Y2 : Limit switch C

Y3 : Rechazo

Y4 : Normal A

Y5 : Normal B

Y6 : Normal C

Y7 : Normal Total

- b) Ahora agregue al programa anterior las siguientes dos condiciones:

El motor A debe quedar fuera de servicio después de transportar 10 unidades.

Cuando el tiempo entre unidades de producto (en cada cinta lateral) sea inferior a 5 segundos debe activarse la señal de "sobrecarga" correspondiente. Es decir el motor operará en condición "normal" sólo si habiendo llegado un producto completa 5 segundos antes de la llegada del siguiente. Considere que "normal" y "sobrecarga" son complementos y que el panel de simulaciones enciende automáticamente la señal de "sobrecarga" cuando la correspondiente señal "normal" tiene nivel lógico cero. Considere además que:

Normal Total = AND (Normal A, Normal B, Normal C)

9.3 Control de un proceso de mezclado en la fabricación de galletas

Una fábrica de galletas desea automatizar una parte de su proceso productivo el que se puede esquematizar mediante la figura siguiente.

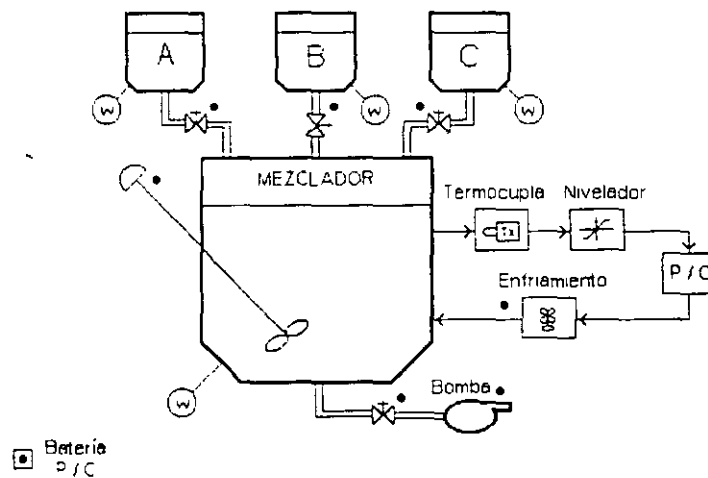


Diagrama del proceso de fabricación de galletas

A continuación se describe la secuencia de operaciones del proceso de mezclado en la fabricación de galletas:

- Se fijan valores de referencia W_A , W_B , y W_C para la carga de cada estanque de materia prima.
- Se sensa la temperatura inicial del mezclador.

- c) Se llenan los estanques hasta alcanzar los valores prefijados, siguiendo la secuencia A-B-C.
- d) Los estanques de materia prima se descargan al mezclador, siguiendo la misma secuencia.
- e) Se pone en marcha el agitador del mezclador durante un tiempo de un minuto, que corresponde al tiempo de mezclado de las materias primas.
- f) En caso de presentar el mezclador sobret temperatura debe actuar el dispositivo de enfriamiento hasta alcanzar un valor normal
- g) Finalizado el proceso de mezclado, el producto resultante se descarga con la ayuda de la bomba y la válvula de descarga a otra etapa del proceso productivo.

Se pide diseñar un programa RLL para simular íntegramente el control y el proceso, considerando los siguientes valores numéricos:

$$WA = 8 \text{ Kg} \quad WB = 9 \text{ Kg} \quad WC = 7 \text{ Kg}$$

Temperatura normal del mezclador : 20 °C

Sobret temperatura del mezclador : 50 °C

Asuma que la velocidad de descarga para cualquier estanque y del mezclador es de 1 Kg/seg, y que el sistema de enfriamiento es capaz de bajar la temperatura del mezclador a razón de 1 °C/seg. El sistema de enfriamiento sólo puede operar mientras esté funcionando el agitador.

Considere para el desarrollo del programa la siguiente convención para las variables:

ENTRADAS:

X0: Temperatura inicial del mezclador (condición normal o sobret temperatura)

X1: ON-OFF (puesta en marcha y fin del proceso)

SALIDAS:

Y0 : Válvula de descarga de estanque A

Y1 : Válvula de descarga de estanque B

Y2 : Válvula de descarga de estanque C

Y3 : Válvula de descarga de mezclador

Y4 : Agitador

Y5 : Sistema de enfriamiento

Y6 : Bomba

Y7 : Batería P/C (estado operativo del sistema)

El programa debe incluir las siguientes funciones.

controlar puesta en marcha y fin del proceso.

senzar temperatura del mezclador antes (y sólo antes) del inicio del proceso

indicar en led rojo el estado operativo del sistema.

indicar en led verde correspondiente el estado activo activo de las válvulas de descarga de los estanques A, B y C, y del mezclador.

indicar en led amarillo correspondiente el estado activo del agitador, bomba y sistema de enfriamiento.

efectuar secuencia descrita del proceso.

controlar sobretemperatura si es necesario.

Además se debe considerar configurar en la pantalla del PC una ventana (Watch) que indique los niveles de carga de los estanques A, B y C, y el nivel de carga y temperatura del mezclador.

9.4 Control de tráfico en una intersección

En la intersección de dos calles se cuenta con un sistema convencional de semáforos. Se trata de vías de un solo sentido, una con orientación sur a norte y la otra este a oeste. Debido a los problemas de congestión vehicular que suelen producirse a ciertas horas del día, se desea optimizar el sistema mediante un control de tráfico con PLC.

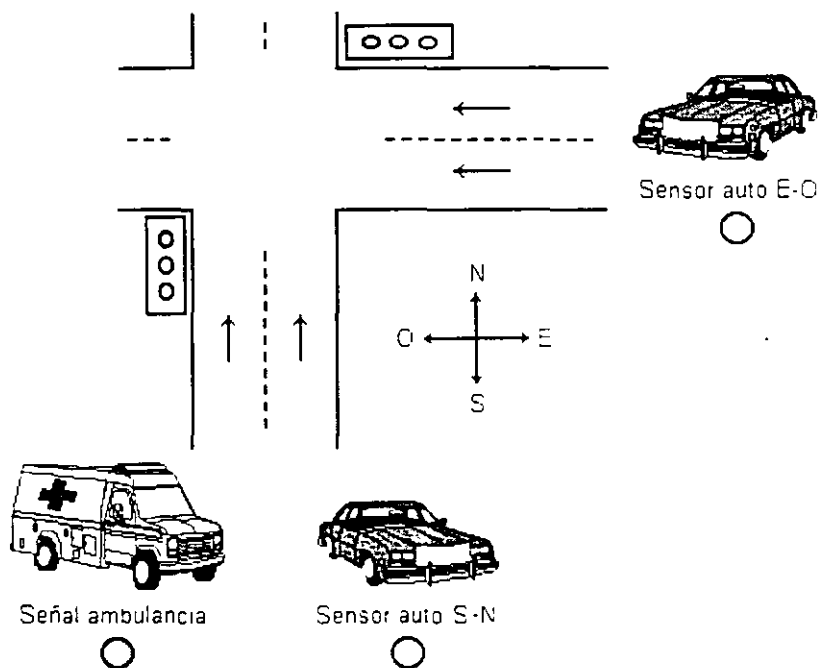


Diagrama de la intersección

Se dispone de sensores especiales que permiten detectar la llegada de vehículos. La presencia de un nuevo auto en una

de las vías es comunicada en forma binaria al PLC, de modo que el controlador conozca en todo momento la cola de autos para cada semáforo.

La temporización para la secuencia de los semáforos es:

Luz verde	:	43 segundos
Luz naranja	:	2 segundos
Luz roja	:	45 segundos

Los tiempos anteriores han sido fijados para un flujo alto de vehículos. En caso de ser menor este flujo, el controlador debe reducir la duración del ciclo de la siguiente forma:

Luz verde	:	28 segundos
Luz naranja	:	2 segundos
Luz roja	:	30 segundos

El criterio de decisión para flujo bajo es que la respectiva cola sea menor a 6 vehículos.

Se contempla que cada auto demora 3 segundos en atravesar el cruce y abandonar la cola.

Por la vía de sur a norte transitan con frecuencia ambulancias. El controlador debe dar prioridad a esta vía en caso de ser necesario. El PLC maneja un receptor que le permite captar

una señal binaria mandada por ambulancias que lleguen al cruce. (Se asume que la aparición de una ambulancia no altera la cola de autos.)

Considere para el desarrollo del programa para el PLC las siguientes funciones principales:

- partida y fin de la simulación
- temporización de los semáforos para flujo alto y bajo
- procesamiento de la información proveniente de los sensores para contabilizar las colas de autos
- actualización de la cola en caso de cruce de un auto
- prioridad para la vía sur a norte si es necesario

Utilice la siguiente convención para las variables:

ENTRADAS:

- X0 : ON-OFF (comienzo y fin de simulación)
- X1 : Llega auto por calle N-S
- X2 : Llega auto por calle E-O
- X3: Llega ambulancia por calle N-S

SALIDAS:

- Y0 : Luz verde para calle N-S
- Y1 : Luz naranja para calle N-S
- Y2 : Luz roja para calle N-S
- Y3 : Luz verde para calle E-O

Y4 : Luz naranja para calle E-O

Y5 . Luz roja para calle E-O

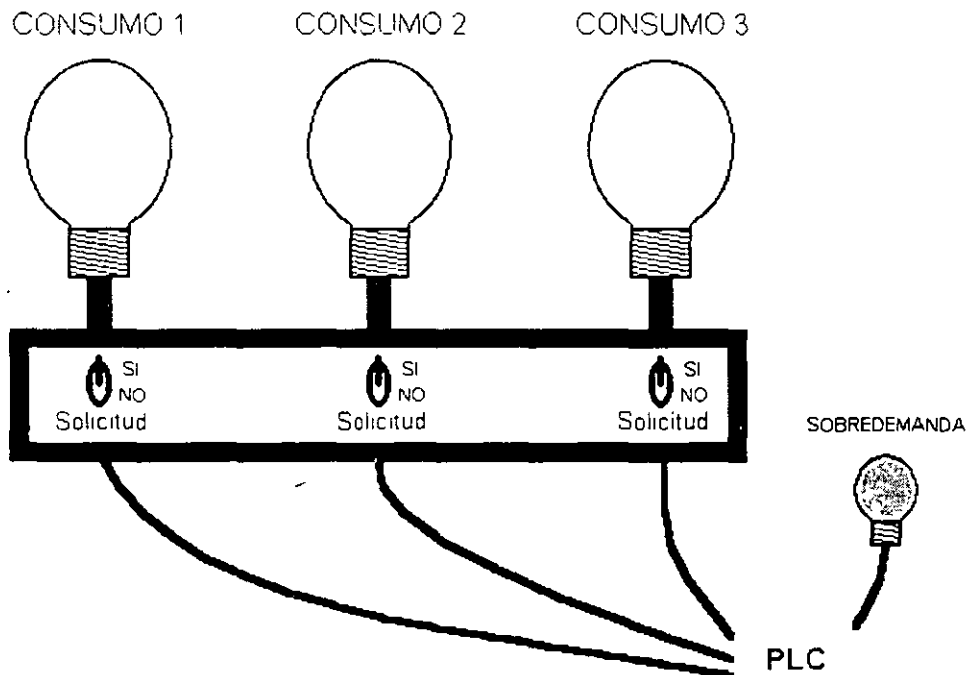
Y6 : Cruza auto por calle N-S

Y7 Cruza auto por calle E-O

9.5 Control de demanda de potencia

Un sistema de distribución eléctrica debe satisfacer las demandas de potencia de tres consumos. Las solicitudes se producen en forma aleatoria

Dada la capacidad del sistema, sólo se pueden habilitar como máximo dos consumos al mismo tiempo, por lo que es necesario implantar un sistema de control que regule los casos de sobredemanda.



Esquema del sistema de demanda de potencia

Para la toma de decisiones se desea realizar un control de la demanda de potencia empleando un PLC. Se dispone de señales binarias de solicitud de potencia de cada consumo. La habilitación de la potencia solicitada se efectúa en forma binaria.

Se ha diseñado la siguiente lista de reglas para el control:

Se habilita un consumo luego de existir una demanda y si la capacidad del sistema lo permite.

En caso de sobredemanda (solicitud de conexión simultánea de tres consumos) debe desconectarse el que lleve más tiempo habilitado.

Un consumo debe ser habilitado por al menos un minuto.

Un consumo que es desconectado por el controlador debe permanecer así por al menos un minuto.

Se debe generar una señal de alerta en caso de sobredemanda.

Desarrolle un programa RLL que realice el control descrito, considerando la siguiente convención para las variables:

ENTRADAS:

X0 : ON-OFF (comienzo y fin de simulación)

X1 : Demanda de potencia del consumo 1

X2 : Demanda de potencia del consumo 2

X3 : Demanda de potencia del consumo 3

SALIDAS:

Y0 : Sobredemanda de potencia del sistema

Y1 : Habilita potencia a consumo 1

Y2 . Habilita potencia a consumo 2

Y3 . Habilita potencia a consumo 3

10. TENDENCIAS Y PERSPECTIVAS

10.1 Introducción

La aparición del microprocesador, así como el uso de los computadores personales y sus lenguajes de programación de alto nivel, ha beneficiado enormemente el desarrollo de equipos de estado sólido utilizados para el control de procesos. El costo de este desarrollo continúa decreciendo a un ritmo acelerado, desplazándose así el uso de los controladores basados en relés.

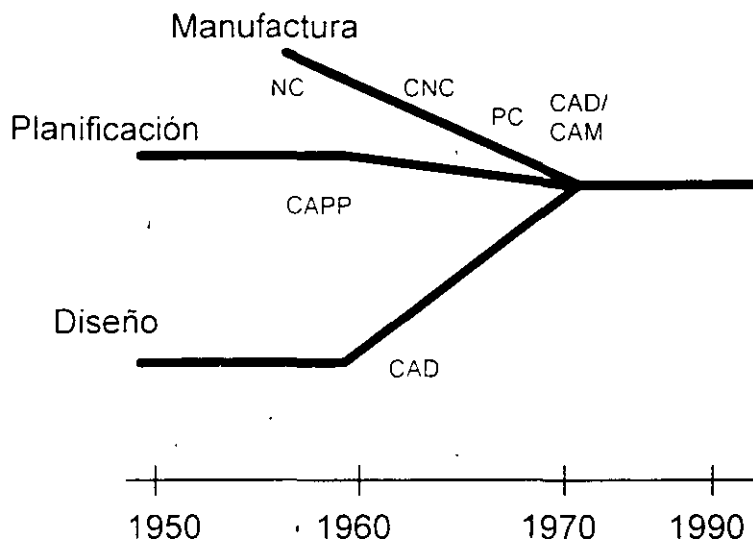
La realización de controladores programables que sean capaces de tomar muestras de señales analógicas, realizar cálculos sobre ellas y comunicar los resultados a un computador central distante, no era posible en 1970. Sin embargo hoy, un sistema de este tipo es común en cualquier planta.

Los aportes de la evolución de los controladores programables ha sido muy importante en el gran desarrollo de la manufactura mundial, que necesita producir a bajo costo y con alta calidad.

El hecho de que un PLC sea *programable*, significa que mediante modificaciones en su programa, el PLC puede

adaptarse a los requerimientos de la aplicación, los que tienden a variar con el tiempo.

La integración de equipos de inteligencia programable en la industria, tanto en la planta como en las oficinas y áreas de ingeniería, comunicados a través de una red eficiente, provee de información en tiempo real de las variables físicas y económicas de la producción



- CAD: Computer Aided Design
- CAM: Computer Aided Manufacturing
- CAPP: Computer Aided Planning Process
- CNC: Computer Numerical Control
- NC: Numerical Control
- PC: Personal Computer

A partir de la década de los '90 los ingenieros pueden diseñar y modificar productos utilizando los más recientes informes de clientes y análisis de tendencias de la demanda del mercado, comunicando sus resultados directamente a los aparatos de manufactura después de pasar por la ayuda computacional de planificación.

En este sentido, la tendencia es hacia la convergencia del diseño, la planificación y la manufactura, en una industria completamente automatizada

En el centro de la automatización industrial está el controlador programable, que con su estructura poderosa y de propósitos generales, brinda una capacidad real y eficiente para controlar procesos de manufactura combinada con la habilidad de recopilar y comunicar información rápidamente.

10.2 Tendencia del hardware

El hardware utilizado hasta nuestros días ha cambiado drásticamente desde la primera aparición del PLC en 1970. La invención y la aplicación del microprocesador han ayudado a esto.

Es muy probable que con este desarrollo acelerado, en un futuro próximo los controladores programables se llamarán de otra manera, tal como *unidad controladora*, ya que este término engloba mejor las nuevas funciones adquiridas.

CPU

El desarrollo de la CPU ha sido tanto en el número de bits que se puede procesar por cada ciclo de máquina (8 bits a mediados de los '70, 16 bits en la década de los '80 y ahora 32 bits) y velocidad de procesamiento, como en su arquitectura interna, dotando al microprocesador de nuevas instrucciones poderosas.

La tendencia del desarrollo de la CPU se puede resumir en:

- uso de microprocesadores de 32 bits
- arquitectura de procesos múltiples
- compartimiento de la memoria con otros controladores
- facilidades de autodiagnóstico

Sistemas de entradas y salidas

Los sistemas de entrada/salida tendrán la habilidad de interactuar con una gran variedad de sensores y actuadores de una manera altamente inteligente.

Los diagnósticos estarán rápidamente disponibles y serán usados de una manera que podrá ser de mayor entendimiento para los diseñadores de sistemas y para el personal en general.

Los puntos de entrada y salida estarán mejor distribuidos, de la misma manera como están distribuidos en el proceso.

Muchos de los sistemas de entrada y salida estarán incorporados en los mismos sensores y actuadores, de tal forma que en un futuro será posible prescindir de los bloques I/O

Los bloques de I/O contarán con cierta lógica rudimentaria de control, lo que permitirá ejecutar un control local. En este caso la CPU actuará sólo como supervisor

Existirá la posibilidad de configurar un punto como entrada o como salida ajustando los umbrales de corriente teniendo circuitos de protección común

Dispositivos de programación

Los dispositivos de programación serán adaptaciones de los computadores personales. Existirán versiones industriales que no sólo podrán programar cualquier unidad de control, sino también cualquier equipo inteligente del proceso, tales como controladores de robots, controladores numéricos, controladores de visión artificial. La comunicación podrá realizarse a través de una red local, o bien desde un computador portátil conectado directamente al PLC.

Los dispositivos de programación brindarán las siguientes facilidades en paralelo:

- simulación
- creación
- prueba
- depuración

Interfaces de operación

En las interfaces de operación será común el empleo de:

- pantallas gráficas a color sensibles al tacto
- módulos de voz
- animación de imágenes
- reconocimiento de voz

10.3 Software

El desarrollo del software está jugando un rol muy importante en la evolución de los PLC, tanto a nivel de sistemas operativos, como a nivel de lenguajes de programación.

Sistemas operativos

En la gran mayoría de los casos los sistemas operativos son exclusivos al controlador programable. Fueron diseñados para operar de una manera óptima pero no estándar.

Así como en los computadores personales tienden a tener sistemas operativos estándares, es probable que los controladores programables tengan la capacidad de escoger entre un número pequeño de sistemas operativos estándares.

La tendencia es utilizar un sistema operativo tipo multi-tarea (*multi-tasking*) que permite correr varias aplicaciones de forma concurrente.

Lenguajes de programación

Los programas se configurarán fácilmente para las aplicaciones industriales específicas por medio de herramientas de programación.

Con el propósito de obtener una mejor comunicación entre el diseñador del sistema y el controlador, cada vez se utilizarán más las técnicas gráficas.

Con lenguajes de programación de tipo gráfico, el operador podrá entender más fácilmente el programa y resolver eventuales problemas que se presenten.

Si es que no evolucionan los lenguajes de programación de diagrama escalera, bloques funcionales y lógica booleana, es muy probable que se desarrolle un nuevo lenguaje, el que deberá incluir combinaciones y conversiones de los lenguajes actuales mencionados.

10.4 Comunicaciones

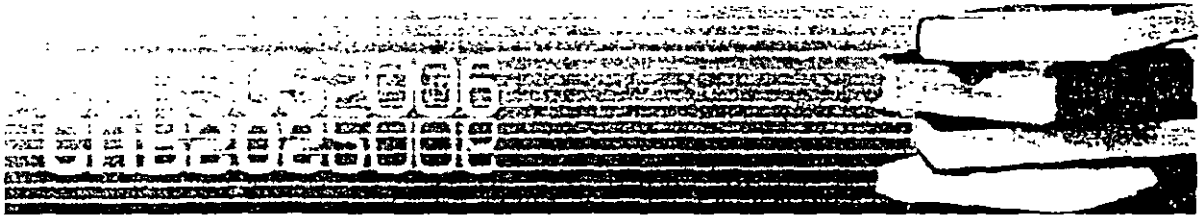
La tendencia es la de tener diversas redes de comunicación a distintos niveles jerárquicos. Esta es la estructura que se tiene en un sistema de control distribuido donde los sistemas de comunicación enlazan los controladores entre sí, con la interfaz-hombre máquina y con otros módulos que pertenezcan al sistema.

Para que la comunicación sea segura se emplean redes redundantes y protocolos estándares.

Es posible que en un futuro se utilice como medio de comunicación el aire, y la comunicación digital en ambientes industriales sea a través de ondas de radio, micro-ondas o rayos infrarrojos y no mediante cables. Lo que brindará una gran flexibilidad al sistema.



FACULTAD DE INGENIERÍA UNAM
DIVISIÓN DE EDUCACIÓN CONTINUA



CURSOS ABIERTOS

CIRCUITOS LÓGICOS PROGRAMABLES

CA 522

TEMA
ANEXO I

EXPOSITOR: Ing. José Luis Ramírez Gutiérrez y
Francisco Rodríguez Ramírez
Jueves 10 al viernes 25 de agosto de 2006
PALACIO DE MINERÍA

El Controlador Lógico Programable

Introducción

Un autómata programable industrial (API) o Programmable logic controller (PLC), es un equipo electrónico, programable en lenguaje no informático, diseñado para controlar en tiempo real y en ambiente de tipo industrial, procesos secuenciales.

Un PLC trabaja en base a la información recibida por los captadores y el programa lógico interno, actuando sobre los accionadores de la instalación.



Campos de aplicación

El PLC por sus especiales características de diseño tiene un campo de aplicación muy extenso. La constante evolución del hardware y software amplía constantemente este campo para poder satisfacer las necesidades que se detectan en el espectro de sus posibilidades reales.

Su utilización se da fundamentalmente en aquellas instalaciones en donde es necesario un proceso de maniobra, control, señalización, etc., por tanto, su aplicación abarca desde procesos de fabricación industriales de cualquier tipo a transformaciones industriales, control de instalaciones, etc.

Sus reducidas dimensiones, la extremada facilidad de su montaje, la posibilidad de almacenar los programas para su posterior y rápida utilización, la modificación o alteración de los mismos, etc., hace que su eficacia se aprecie fundamentalmente en procesos en que se producen necesidades tales como:

- Espacio reducido
- Procesos de producción periódicamente cambiantes
- Procesos secuenciales
- Maquinaria de procesos variables
- Instalaciones de procesos complejos y amplios
- Chequeo de programación centralizada de las partes del proceso

Ejemplos de aplicaciones generales

- Maniobra de máquinas
- Maquinaria industrial de plástico
- Maquinas transfer
- Maquinaria de embalajes
- Maniobra de instalaciones
 - Instalacion de aire acondicionado, calefacción...
 - Instalaciones de seguridad
- Señalización y control
 - Chequeo de programas
 - Señalización del estado de procesos

Ventajas e inconvenientes

No todos los automatats ofrecen las mismas ventajas sobre la lógica cableada, ello es debido, principalmente, a la variedad de modelos existentes en el mercado y las innovaciones técnicas que surgen constantemente. Tales consideraciones me obligan a referirme a las ventajas que proporciona un automata de tipo medio.

Ventajas

- Menor tiempo empleado en la elaboracion de proyectos debido a que:
 - No es necesario dibujar el esquema de contactos
 - No es necesario simplificar las ecuaciones lógicas, ya que, por lo general la capacidad de almacenamiento del modulo de memoria es lo suficientemente grande
 - La lista de materiales queda sensiblemente reducida, y al elaborar el presupuesto correspondiente eliminaremos parte del problema que supone el contar con diferentes proveedores, distintos plazos de entrega.
 - Posibilidad de introducir modificaciones sin cambiar el cableado ni añadir aparatos
 - Minimo espacio de ocupacion.
 - Menor coste de mano de obra de la instalación.
 - Economía de mantenimiento. Además de aumentar la fiabilidad del sistema, al eliminar contactos móviles, los mismos autómatas pueden indicar y detectar averias
 - Posibilidad de gobernar varias maquinas con un mismo autómata.
 - Menor tiempo para la puesta en funcionamiento del proceso al quedar reducido el tiempo cableado
 - Si por alguna razon la maquina queda fuera de servicio, el autómata sigue siendo util para otra maquina o sistema de producción.

Inconvenientes

- Como inconvenientes podríamos hablar, en primer lugar, de que hace falta un programador, lo que obliga a adiestrar a uno de los técnicos en tal sentido, pero hoy en dia ese inconveniente esta solucionado porque las universidades ya se encargan de dicho adiestramiento.

- El coste inicial también puede ser un inconveniente.

Funciones básicas de un PLC

- Detección:

Lectura de la señal de los captadores distribuidos por el sistema de fabricación.

- Mando

Elaborar y enviar las acciones al sistema mediante los accionadores y preaccionadores

- Dialogo hombre maquina

Mantener un diálogo con los operarios de producción, obedeciendo sus consignas e informando del estado del proceso

- Programacion

Para introducir, elaborar y cambiar el programa de aplicación del autómeta. El dialogo de programación debe permitir modificar el programa incluso con el autómeta controlando la maquina

Nuevas Funciones

- Redes de comunicación.

Permiten establecer comunicacion con otras partes de control. Las redes industriales permiten la comunicacion y el intercambio de datos entre autómetas a tiempo real. En unos cuantos milisegundos pueden enviarse telegramas e intercambiar tablas de memoria compartida

- Sistemas de supervisión

También los automatats permiten comunicarse con ordenadores provistos de programas de supervisión industrial. Esta comunicacion se realiza por una red industrial o por medio de una simple conexion por el puerto serie del ordenador.

- Control de procesos continuos

Además de dedicarse al control de sistemas de eventos discretos los autómetas llevan incorporadas funciones que permiten el control de procesos continuos. Disponen de módulos de entrada y salida analogicas y la posibilidad de ejecutar reguladores PID que estan programados en el automata

- Entradas- Salidas distribuidas

Los módulos de entrada salida no tienen por qué estar en el armario del autómata. Pueden estar distribuidos por la instalación, se comunican con la unidad central del autómata mediante un cable de red.

■ Buses de campo:

Mediante un solo cable de comunicación se pueden conectar al bus captadores y accionadores, reemplazando al cableado tradicional. El autómata consulta cíclicamente el estado de los captadores y actualiza el estado de los accionadores.

Estructura Externa

Introducción

El término estructura externa o configuración externa de un autómata programable industrial se refiere al aspecto físico exterior del mismo, bloques o elementos en que está dividido

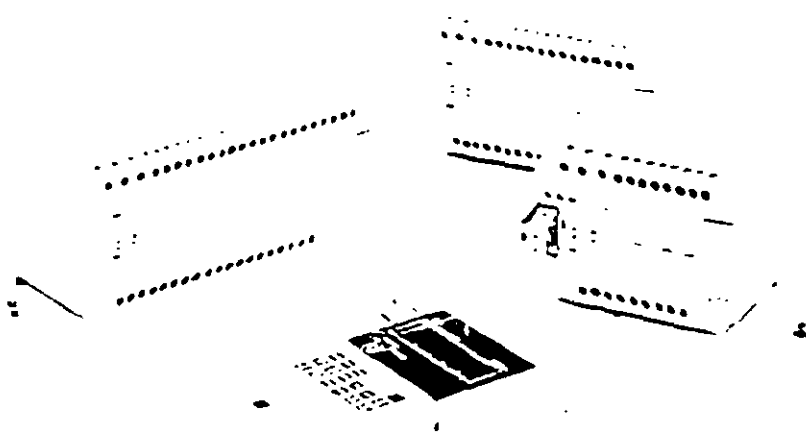
Actualmente son tres las estructuras más significativas que existen en el mercado

- Estructura compacta
 - Estructura semimodular (Estructura Americana)
 - Estructura modular (Estructura Europea)

Estructura compacta

Este tipo de automatismos se distingue por presentar en un solo bloque todos sus elementos, esto es, fuente de alimentación, CPU, memorias, entradas salidas, etc

Son los autómatas de gama baja o nanoautomatistas los que suelen tener una estructura compacta. Su potencia de proceso suele ser muy limitada dedicándose a controlar máquinas muy pequeñas o cuadros de mando



Estructura semimodular

Se caracteriza por separar las E/S del resto del autómata, de tal forma que en un bloque compacto están reunidas las CPU, memoria de usuario o de programa y fuente de alimentación y separadamente las unidades de E/S.

Son los automatismos de gama media los que suelen tener una estructura semimodular (Americana)

Estructura modular

Su característica principal es la de que existe un módulo para cada uno de los diferentes elementos que componen el autómata como puede ser una fuente de alimentación, CPU, E/S, etc. La sujeción de los mismos se hace por carril DIN, placa perforada o sobre RACK, en donde va alojado el BUS externo de unión de los distintos módulos que lo componen

Son los automatismos de gama alta los que suelen tener una estructura modular, que permiten una gran flexibilidad en su constitución.

Estructura Interna

Introducción

En este apartado vamos a estudiar la estructura interna de cada uno de los diferentes elementos que componen el autómata, las funciones y funcionamiento de cada una de ellas

El autómata está constituido por diferentes elementos, pero tres son los básicos:

- CPU
- Entradas
- Salidas

Con las partes mencionadas podemos decir que tenemos un autómata pero para que sea operativo son necesarios otros elementos tales como:

- Fuente de alimentación
- Interfaces
- La unidad o consola de programación
- Los dispositivos periféricos

El CPU =

Introducción

La CPU (Central Processing Unit) es la parte inteligente del sistema. Interpreta las instrucciones del programa de usuario y consulta el estado de las entradas. Dependiendo de dichos estados y del programa, ordena la activación de las salidas deseadas

La CPU está constituida por los siguientes elementos:

- Procesador
- Memoria monitor del sistema
- Circuitos auxiliares

Procesador

Está constituido por el microprocesador, el reloj (generador de onda cuadrada) y algún chip auxiliar

El microprocesador es un circuito integrado (chip), que realiza una gran cantidad de operaciones que podemos agrupar en

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

Para que el microprocesador pueda realizar todas estas operaciones está dotado de unos circuitos internos que son los siguientes

- *Circuitos de la unidad aritmética y lógica o ALU* Es la parte del μp donde se realizan los cálculos y las decisiones lógicas para controlar el autómata.
- *Circuitos de la unidad de control (UC) o Decodificador de instrucciones*: Decodifica las instrucciones leídas en memoria y se generan las señales de control.
- *Acumulador* Es la encargada de almacenar el resultado de la última operación realizada por el ALU
- *Flags* Flags, o indicadores de resultado, que pueden ser consultados por el programa.
- *Contador de programa* Encargada de la lectura de las instrucciones de usuario
- *Bus interno* No son circuitos en si, sino zonas conductoras en paralelo que transmiten datos, direcciones, instrucciones y señales de control entre las diferentes partes del μp .

Memoria monitor del sistema

Es una memoria de tipo ROM, y además del sistema operativo del autómata contiene las siguientes rutinas, incluidas por el fabricante.

- Inicialización tras puesta en tensión o reset.
- Rutinas de test y de respuesta a error de funcionamiento.
- Intercambio de información con unidades exteriores.
- Lectura y escritura en las interfaces de E/S.

Funciones básicas de la CPU

En la memoria ROM del sistema, el fabricante ha grabado una serie de programas ejecutivos, software del sistema y es a estos programas a los que accederá el μp para realizar las funciones

El software del sistema de cualquier automata consta de una serie de funciones básicas que realiza en determinados tiempos de cada ciclo

En general cada automata contiene y realiza las siguientes funciones:

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

Fuente de Alimentación

La fuente de alimentación proporciona las tensiones necesarias para el funcionamiento de los distintos circuitos del sistema

La alimentación a la CPU puede ser de continua a 24 Vcc, tensión muy frecuente en cuadros de distribución, o en alterna a 110/220 Vca. En cualquier caso es la propia CPU la que alimenta las interfaces conectadas a través del bus interno.

La alimentación a los circuitos E/S puede realizarse, según tipos, en alterna a 48/110/220 Vca o en continua a 12/24/48 Vcc

La fuente de alimentación del automata puede incorporar una batería tampón, que se utiliza para el mantenimiento de algunas posiciones internas y del programa usuario en memoria RAM, cuando falla la alimentación o se apaga el autómata.

Interfases

En el control de un proceso automatizado, es imprescindible un dialogo entre operador-maquina junto con una comunicación entre la máquina y el autómata, estas comunicaciones se establecieron por medio del conjunto de entradas y salidas del citado elemento

Los automatas son capaces de manejar tensiones y corrientes de nivel industrial, gracias a que disponen un bloque de circuitos de interfaz de E/S muy potente, que les permite conectarse directamente con los sensores y accionamientos del proceso.

De entre todos los tipos de interfaces que existen, las interfaces específicas permiten la conexión con elementos muy concretos del proceso de automatización. Se pueden distinguir entre ellas tres grupos bien diferenciados

- Entradas salidas especiales.
- Entradas salidas inteligentes.
- Procesadores periféricos inteligentes.

Las interfaces especiales del primer grupo se caracterizan por no influir en las variables de estado del proceso de automatización. Únicamente se encargan de adecuar las E/S. para que puedan ser inteligibles por la CPU, si son entradas, o para que puedan ser interpretadas correctamente por actuadores (motores, cilindros, etc.), en el caso de las salidas.

Las del segundo grupo admiten múltiples modos de configuración, por medio de unas combinaciones binarias situadas en la misma tarjeta. De esta forma se descarga de trabajo a la unidad central, con las ventajas que conlleva.

Los procesadores periféricos inteligentes, son módulos que incluyen su propio procesador, memorias y puntos auxiliares de entrada / salida. Estos procesadores contienen en origen un programa especializado en la ejecución de una tarea concreta, a la que le basta conocer los puntos de consigna y los parámetros de aplicación para ejecutar, de forma autónoma e independiente de la CPU principal, el programa de control.

Entradas y Salidas

Introducción

La sección de entradas mediante el interfaz, adapta y codifica de forma comprensible para la CPU las señales procedentes de los dispositivos de entrada o captadores.

Hay dos tipos de entradas:

- Entradas digitales
- Entradas analógicas

La sección de salida también mediante interfaz trabaja de forma inversa a las entradas, es decir, decodifica las señales procedentes de la CPU, y las amplifica y manda con ellas los dispositivos de salida o actuadores como lámparas, relés... aquí también existen unos interfaces de adaptación a las salidas de protección de circuitos internos.

Hay dos tipos de salidas

- Salidas digitales
- Salidas analógicas

Entradas digitales

Los módulos de entrada digitales permiten conectar al autómata captadores de tipo todo o nada como finales de carrera pulsadores .

Los módulos de entrada digitales trabajan con señales de tensión. por ejemplo cuando por una vía llegan 24 voltios se interpreta como un "1" y cuando llegan cero voltios se interpreta como un "0"

El proceso de adquisición de la señal digital consta de varias etapas.

- Protección contra sobretensiones
- Filtrado
- Puesta en forma de la onda
- Aislamiento galvánico o por optoacoplador.

Entradas analógicas

Los módulos de entrada analógicas permiten que los autómatas programables trabajen con accionadores de mando analógico y lean señales de tipo analógico como pueden ser la temperatura, la presión o el caudal

Los módulos de entradas analógicas convierten una magnitud analógica en un número que se deposita en una variable interna del automata. Lo que realiza es una conversión A/D, puesto que el automata solo trabaja con señales digitales. Esta conversión se realiza con una precisión o resolución determinada (número de bits) y cada cierto intervalo de tiempo (periodo muestreo)

Los módulos de entrada analógica pueden leer tensión o intensidad.

El proceso de adquisición de la señal analógica consta de varias etapas:

- Filtrado
- Conversión A/D
- Memoria interna

Salidas digitales

Un módulo de salida digital permite al automata programable actuar sobre los preaccionadores y accionadores que admitan órdenes de tipo todo o nada.

El valor binario de las salidas digitales se convierte en la apertura o cierre de un relé interno del automata en el caso de módulos de salidas a relé.

En los módulos estáticos (borno), los elementos que conmutan son los componentes electrónico como transistores o triacs, y en los módulos electromecánicos son contactos de relés internos al módulo.

Los módulos de salidas estáticas al suministrar tensión, solo pueden actuar sobre elementos que trabajan todos a la misma tensión, en cambio los módulos de salida electromecánicos, al ser libres de tensión, pueden actuar sobre elementos que trabajen a tensiones distintas

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

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

Salidas analógicas

Los módulos de salida analógica permiten que el valor de una variable numérica interna del automata se convierta en tensión o intensidad

Lo que realiza es una conversión D/A, puesto que el automata solo trabaja con señales digitales. Esta conversión se realiza con una precisión o resolución determinada (numero de bits) y cada cierto intervalo de tiempo (periodo muestreo)

Esta tensión o intensidad puede servir de referencia de mando para actuadores que admitan mando analógico como pueden ser los variadores de velocidad, las etapas de los tiristores de los hornos, reguladores de temperatura, ... permitiendo al autómata realiza funciones de regulación y control de procesos continuos

El proceso de envío de la señal analógica consta de varias etapas:

- Aislamiento galvanico
- Conversión D/A
- Circuitos de amplificación y adaptación
- Protección electrónica de la salida

Como hemos visto las señales analógicas sufren un gran proceso de adaptación tanto en los módulos de entrada como en los módulos de salida. Las funciones de conversión A/D y D/A que realiza son esenciales. Por ello los módulos de E/S analógicos se les considera módulos de E/S especiales

Memoria

Introducción

La memoria es el almacén donde el automata guarda todo cuanto necesita para ejecutar la tarea de control

Datos del proceso.

- Señales de planta, entradas y salidas
- Variables internas, de bit y de palabra
- Datos alfanumericos y constantes

Datos de control

- Instrucciones de usuario (programa)
- Configuración del automata (modo de funcionamiento, numero de e/s conectadas, ...)

Existen varios tipos de memorias:

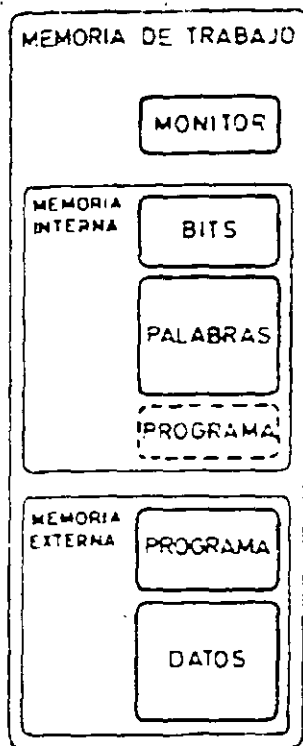
- RAM: Memoria de lectura y escritura
- ROM: Memoria de solo lectura, no reprogramable.
- EPROM: Memoria de solo lectura, reprogramables con borrado por ultravioletas.
- EEPROM: Memoria de solo lectura, alterables por medios electricos

La memoria RAM se utiliza principalmente como memoria interna, y unicamente como memoria de programa en el caso de que pueda asegurarse el mantenimiento de los datos con una bateria exterior

La memoria ROM se utiliza para almacenar el programa monitor del sistema como hemos visto en el apartado dedicado a la CPU

Las memorias EPROM se utilizan para almacenar el programa de usuario, una vez que ha sido convenientemente depurada

Las memorias EEPROM se emplean principalmente para almacenar programas, aunque en la actualidad es cada vez mas frecuente el uso de combinaciones RAM - EEPROM, utilizando estas ultimas como memorias de seguridad que salvan el contenido de las RAM. Una vez reanudada la alimentacion, el contenido de la EEPROM se vuelca sobre la RAM. Las soluciones de este tipo estan sustituyendo a las clásicas RAM + bateria puesto que presentan muchos menos problemas.



MAPAS DE MEMORIA

	SIEMENS 55-135U (CPU 946/947)	IZUMI MICRO-1
NO ACCESIBLE AL USUARIO		
	2K puntos* E/S 2K	28 puntos E/S 256
	256 256	80 45
	120 K**	— 1K2 (600 pasos)
	768 K**	SIN MEMORIA EXTERNA

* El autómata direcciona 8K puntos E/S adicionales sin imagen de proceso
 ** El usuario decide el reparto entre programas y datos

Memoria interna

En un autómata programable, la memoria interna es aquella que almacena el estado de las variables que maneja el autómata: entradas, salidas, contadores, relés internos, señales de estado, etc. Esta memoria interna se encuentra dividida en varias áreas, cada una de ellas con un cometido y características distintas.

La clasificación de la memoria interna no se realiza atendiendo a sus características de lectura y escritura, sino por el tipo de variables que almacena y el número de bits que ocupa la variable. Así, la memoria interna del autómata queda clasificada en las siguientes áreas:

Área de imágenes de entradas/salidas y Área interna (IR)

En esta área de memoria se encuentran:

- Los canales (registros) asociados a los terminales externos (entradas y salidas)
- Los relés (bit) internos (no correspondidos con el terminal externo), gestionados como relés de E/S.

- Los relés ES no usados pueden usarse como IR.
- No retienen estado frente a la falta de alimentación o cambio de modo de operación

Área especial (SR).

Son relés de señalización de funciones particulares como:

- Servicio (siempre ON, OFF)
- Diagnóstico (señalización o anomalías)
- Temporizaciones (relojes a varias frecuencias)
- Cálculo
- Comunicaciones
- Accesible en forma de bit o de canal
- No conservan su estado en caso de fallo de alimentación o cambio de modo

Área auxiliar (AR)

Contienen bits de control e información de recursos de PLC como: Puerto RS232C, puertos periféricos, casetes de memoria.

- Se dividen en dos bloques:

Señalización: Errores de configuración, datos del sistema.

Memorización y gestión de datos

- Es un área de retención.
- Accesible en forma de bit o de canal
- No conservan su estado en caso de fallo de alimentación o cambio de modo.

Área de enlace (LR)

- Se utilizan para el intercambio de datos entre dos PLC's unidos en forma PC Link(1:1).
- Dedicados al intercambio de información entre PLC's
- Si no se utilizan como LR pueden usarse como IR.
- Accesible en forma de bit o canal
- No conservan su estado en caso de fallo de alimentación o cambio de modo

Area de retencion (HIR)

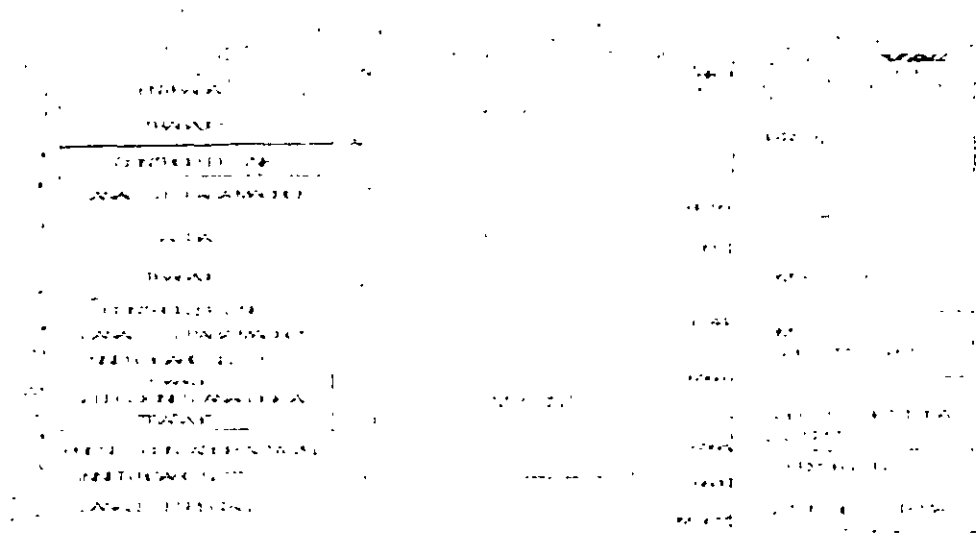
- Mantienen su estado ante fallos de alimentación o cambio de modo de PLC
- Son gestionados como los IR y direccionables como bit o como canal.

Area de temporizadores y contadores (TIM CNT).

- Es el area de memoria que simula el funcionamiento de estos dispositivos
- Son usados por el PLC para programar retardos y contajes.

Area de datos (DM)

- Se trata de memoria de 16 bits (palabra).
- Utilizable para gestion de valores numéricos
- Mantiene su estado ante cambios de modos de trabajo o fallo de alimentacion
- Direccionables como Canal(palabra)
- Esta área suele contener los parámetros de configuración del PLC(setup).



Las variables contenidas en la memoria interna, pueden ser consultadas y modificadas continuamente por el programa, cualquier número de veces. Esta actualización continua de los datos obliga a construir la memoria con dispositivos RAM.

Memoria de programa

La memoria de programa, normalmente externa y enchufable a la CPU mediante casete de memoria, almacena el programa escrito por el usuario para su aplicación.

Cada instrucción del usuario ocupa un paso o dirección del programa.

Las memorias de programa o memorias de usuario son siempre de tipo permanente RAM + batería o EPROM/EEPROM. Por lo general la mayoría de los fabricantes de automatismos ofrecen la posibilidad de utilizar memorias RAM con batería para la fase de desarrollo y depuración de los programas, y de pasar estos a memorias no volátiles EPROM o EEPROM una vez finalizada esta fase.

La ejecución del programa en el módulo es siempre prioritaria, de forma que si se da tensión al automatismo con un módulo conectado, la CPU ejecuta su programa y no el contenido en memoria RAM interna.

Funcionamiento

Introducción

Los automatismos programables son máquinas secuenciales que ejecutan correlativamente las instrucciones indicadas en el programa de usuario almacenado en su memoria, generando unas ordenes o señales de mando a partir de las señales de entrada leídas de la planta (aplicación). Al detectarse cambios en las señales, el autómata reacciona según el programa hasta obtener las ordenes de salida necesarias. Esta secuencia se ejecuta continuamente para conseguir el control actualizado del proceso.

La secuencia básica de operación del autómata se puede dividir en tres fases principales:

- Lectura de señales desde la interfaz de entradas.
- Procesado del programa para obtención de las señales de control.
- Escritura de señales en la interfaz de salidas.

A fin de optimizar el tiempo, la lectura y escritura de las señales se realiza a la vez para todas las entradas y salidas. Entonces, las entradas leídas de los módulos de entrada se guardan en una memoria temporal (Imagen entradas). A esta acude la CPU en la ejecución del programa, y según se va obteniendo las salidas, se guardan en otra memoria temporal (Imagen de salida). Una vez ejecutado el programa completo, estas imágenes de salida se transfieren todas a la vez al módulo de salida.

El automata realiza también otra serie de acciones que se van repitiendo periódicamente, definiendo un ciclo de operación. Dichas acciones se pueden observar en el diagrama de bloques de la figura 2.2.1

El PLC está en reposo y puede recibir o enviar el programa a un periférico

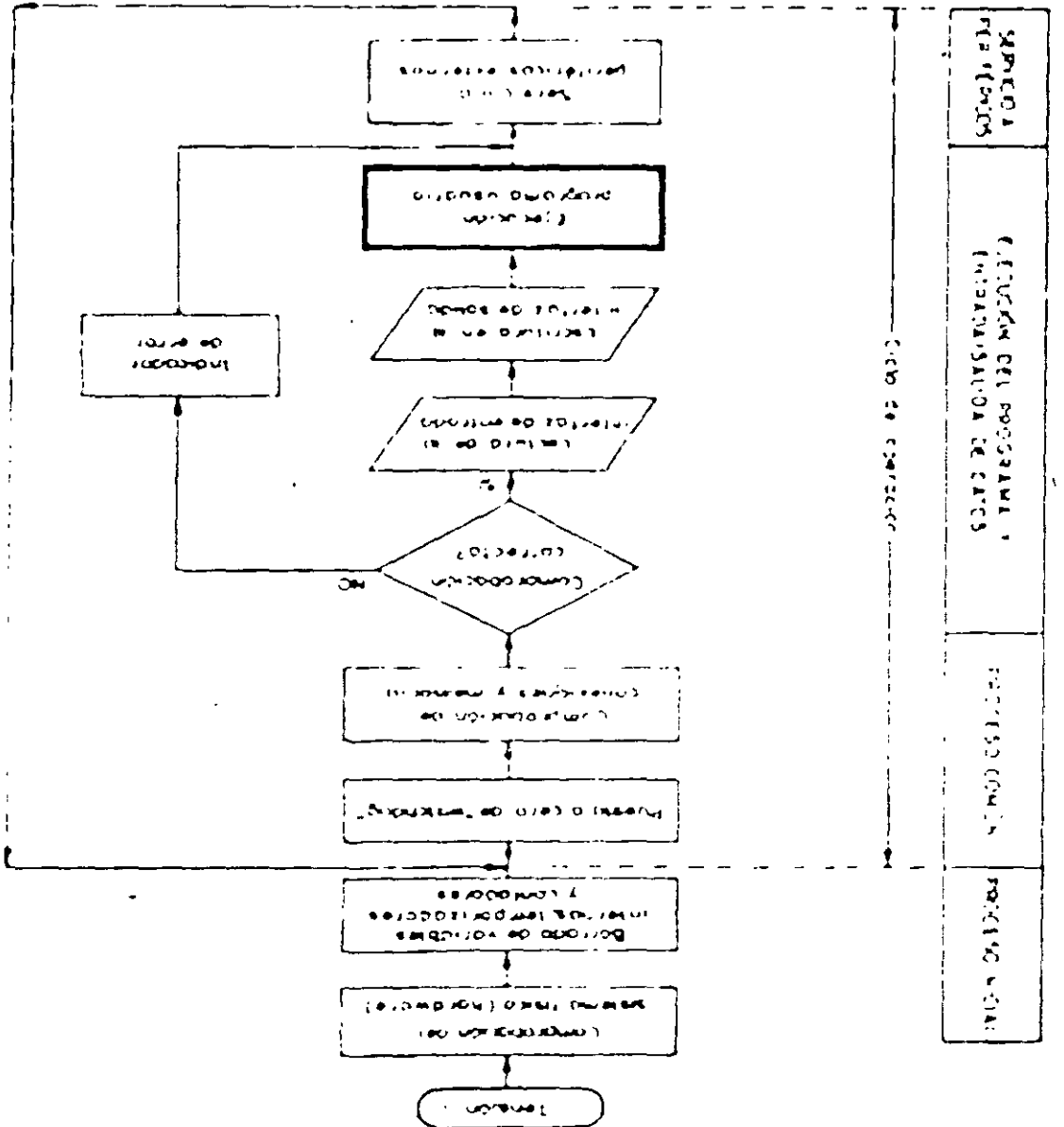
Program

diferentes.

El automata de OMRON CQM1H puede trabajar de tres formas

Modo de funcionamiento

Figura 2.2.1



El P.L.C ejecuta el programa que tiene en memoria.

Run

El PLC ejecuta el programa que tiene en memoria permitiendo el cambio de valores en los registros del mismo

Ciclo de funcionamiento

El funcionamiento del automata es, salvo el proceso inicial que sigue a un Reset, de tipo secuencial y ciclico, es decir, las operaciones tienen lugar una tras otra, y se van repitiendo continuamente mientras el automata esté bajo tensión

La figura 2.2.1 muestra esquemáticamente la secuencia de operaciones que ejecuta el automata, siendo las operaciones del *ciclo de operación* las que se repiten indefinidamente

El ciclo de funcionamiento se divide en dos partes como se puede observar en el esquema de diagrama de la figura 2.2.1 llamados *Proceso Inicial* y *Ciclo de Operación*

Proceso inicial

Como se muestra en la figura, antes de entrar en el ciclo de operación el autómata realiza una serie de acciones comunes, que tratan fundamentalmente de inicializar los estados del mismo y chequear el hardware. Estas rutinas de chequeo, incluidas en el programa monitor ROM, comprueban

- El bus de conexiones de las unidades de E/S.
- El nivel de la batería, si esta existe.
- La conexión de las memorias internas del sistema.
- El módulo de memoria exterior conectado, si existe.

Si se encontrara algún error en el chequeo, se activaría el LED de error y quedaría registrado el código del error

Compradas las conexiones, se inicializan las variables internas:

- Se ponen a OFF las posiciones de memoria interna (excepto las mantenidas o protegidas contra pérdidas de tensión)

- Se borran todas las posiciones de memoria imagen E/S.
- Se borran todos los contadores y temporizadores (excepto los mantenidos o protegidos contra pérdidas de tensión)

Trascurrido el *Proceso Inicial* y si no han aparecido errores el automata entra en el *Ciclo de Operación*

Ciclo de operación

Este ciclo puede considerarse dividido en tres bloques tal y como se puede observar en la figura 2.2.1. dichos bloques son

- Proceso Común
- Ejecucion del programa
- Servicio a perifericos

Proceso común

En este primer bloque se realizan los chequeos ciclicos de conexiones y de memoria de programa, protegiendo el sistema contra

- Errores de hardware (conexiones E/S, ausencia de memoria de programa, etc)
- Errores de sintaxis (programa imposible de ejecutar).

El chequeo ciclico de conexiones comprueba los siguientes puntos:

- Niveles de tensión de alimentación.
- Estado de la batería si existe.
- Buses de conexión con las interfaces.

El chequeo de la memoria de programa comprueba la integridad de la misma y los posibles errores de sintaxis y gramatica

- Mantenimiento de los datos, comprobados en el "checksum".

- Existencia de la instrucción END de fin de programa.
- Estructura de saltos y andamiento de bloque correctas.
- Códigos de instrucciones correctas

Ejecución del programa

En este segundo bloque se consultan los estados de las entradas y de las salidas y se elaboran las ordenes de mando o de salida a partir de ellos

El tiempo de ejecución de este bloque de operaciones es la suma del:

- Tiempo de acceso a interfaces de E/S
- Tiempo de escrutación de programa.

Y a su vez esto depende respectivamente de

- Numero y ubicacion de las interfaces de E/S
- Longitud del programa y tipo de CPU que lo procesa.

Servicio a periféricos

Este tercer y último bloque es unicamente atendido si hay pendiente algún intercambio con el exterior. En caso de haberlo, la CPU le dedica un tiempo limitado, de 1 a 2ms, en atender el intercambio de datos. Si este tiempo no fuera suficiente, el servicio queda interrumpido hasta el siguiente ciclo

Tiempo de ejecución y control en tiempo real

El tiempo total que el automata emplea para realizar un ciclo de operación se llama tiempo de ejecución de *ciclo de operación* o mas sencillamente tiempo de ciclo "Scan time"

Dicho tiempo depende de

- El numero de E/S involucradas
- La longitud del programa usuario
- El numero y tipo de periféricos conectados al autómata.

Los tiempos totales de ciclos son entonces la suma de tiempos empleados en realizar las distintas operaciones del ciclo como se puede ver en la figura 2.2.2:

- Autodiagnostico (Proceso común)
- Actualización de E/S (Ejecución del programa)

- Ejecución de programa.(Ejecución del programa)
- Servicio a periféricos.(Servicio a periféricos)

1	GESTION DE PROCESOS COMUNES	$T1 = 126 \text{ (ms) FLO}$
2	GESTION DE PERIFERICO:	$T2 = T1 + T3 + T4 \cdot 1000 \text{ (ms)}$ $S_1 T2 > 1 \text{ ms } T2 = 1 \text{ ms}$ $S_1 T2 > 1 \text{ ms } T2 \text{ va redondeado}$ $\text{por defecto a } 0.5 \text{ ms}$
3	EJECUCION DE INSTRUCCIONES	$T3 = \text{Suma de los tiempos de}$ $\text{ejecucion de las diversas}$ $\text{instrucciones del programa}$
4	ACTUACION DE E/S	$T4 = 0.09 + 10.07 \cdot N \text{ (ms)}$ $\text{donde } N = \text{numero de}$ GATE ARRAY

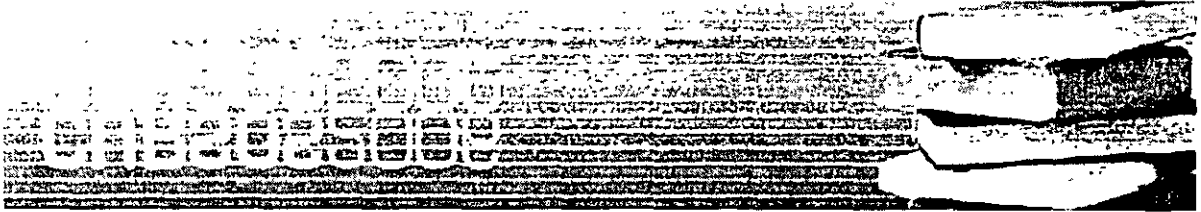
Figura 2.2.2

Los tiempos de ejecución de instrucciones se miden en unidades de microsegundos, resultando un tiempo de ejecución del programa variable en función del número e instrucciones contenidas. Precisamente el tiempo de ejecución es uno de los parámetros que caracterizan a un automata expresado normalmente en milisegundos por cada mil instrucciones ms/k1

$$\text{Tiempo total SCAN} = T1 + T2 + T3 + T4$$



FACULTAD DE INGENIERÍA UNAM
DIVISIÓN DE EDUCACIÓN CONTINUA



CURSOS ABIERTOS

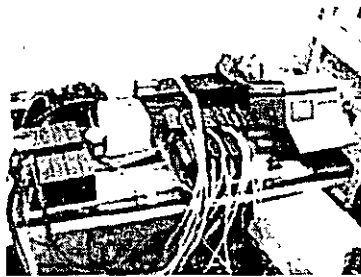
CIRCUITOS LÓGICOS PROGRAMABLES

CA 522

TEMA
ANEXO II

EXPOSITOR: Ing. José Luis Ramírez Gutiérrez y
Francisco Rodríguez Ramírez
Jueves 10 al viernes 25 de agosto de 2006
PALACIO DE MINERÍA

Controlador lógico programable



PLC

Los **controladores lógicos programables** o **PLC** (*Programmable Logic Controller* en sus siglas en inglés) son dispositivos electrónicos muy usados en Automatización Industrial.

Su historia se remonta a finales de la década de 1960, cuando la industria buscó en las nuevas tecnologías electrónicas una solución más eficiente para reemplazar los sistemas de control basados en circuitos eléctricos con relés, interruptores y otros componentes comúnmente utilizados para el control de los sistemas de lógica combinacional.

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

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

Existen varios lenguajes de programación, tradicionalmente los más utilizados son el diagrama de escalera, lista de instrucciones y programación por estados, aunque se han incorporado lenguajes más intuitivos que permiten implementar algoritmos complejos mediante simples diagramas de flujo más fáciles de interpretar y mantener.

En la programación se pueden incluir diferentes tipos de operandos, desde los más simples como lógica booleana, contadores, temporizadores, contactos, bobinas y operadores matemáticos, hasta operaciones más complejas como manejo de tablas (recetas), apuntadores, algoritmos PID y funciones de comunicación multiprotocolos que le permitirían interconectarse con otros dispositivos

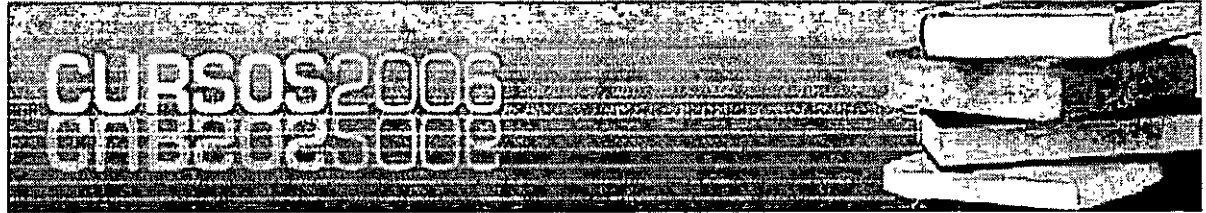
Automatización Industrial

Automatización Industrial. (Automatización: del griego antiguo: guiado por uno mismo) es el uso de sistemas o elementos computarizados para controlar maquinarias y/o procesos industriales substituyendo a operadores humanos. El alcance va mas allá que la simple mecanización de los procesos ya que ésta provee a operadores humanos

mecanismos para asistirlos en los esfuerzos físicos del trabajo, la automatización reduce ampliamente la necesidad sensorial y mental del humano. La automatización como una disciplina de la ingeniería es más amplia que un mero sistema de control, abarca la instrumentación industrial, que incluye los sensores y transmisores de campo, los sistemas de control y supervisión, los sistemas de transmisión y recolección de datos y las aplicaciones de software en tiempo real para supervisar y controlar las operaciones de plantas o procesos industriales.



FACULTAD DE INGENIERÍA UNAM
DIVISIÓN DE EDUCACIÓN CONTINUA



CURSOS ABIERTOS

CIRCUITOS LÓGICOS PROGRAMABLES

CA 522

TEMA
CONCEPTOS GENERALES DE
PROGRAMACION

EXPOSITOR: Ing. Francisco Rodríguez Ramírez
Del 14 al 25 de agosto de 2006
PALACIO DE MINERÍA

CONCEPTOS GENERALES DE PROGRAMACION

Antes de iniciar con el proceso de programación, es conveniente tener claro algunos conceptos preliminares respecto a la organización de los programas en la memoria del procesador.

Por otro lado, también es importante reconocer las diferentes representaciones de los lenguajes de programación, así como, su denominación en marcas de reconocido prestigio.

PROGRAMA, PROGRAMACION Y LENGUAJES DE PROGRAMACION

Desde el punto de vista del Procesador, un programa es un conjunto de instrucciones o proposiciones bien definidas que le dicen lo que tiene que hacer. Cada instrucción le indica: - qué operación realizará a continuación

- de dónde obtendrá los datos que necesita para realizarla

- dónde guardará los resultados de la operación.

Desde el punto de vista del usuario, un programa, son las especificaciones de un conjunto de operaciones que debe llevar a cabo el computador para lograr resolver una determinada tarea.

Un programa se escribe en un lenguaje de programación, estos lenguajes permiten simplificar la creación de programas debido a su fácil descripción de las instrucciones que ha de ejecutar el procesador; en algunos casos, agrupando varias instrucciones y dando un solo nombre al conjunto, de tal forma que la lista de operaciones se reduce considerablemente, resultando fácil la comprensión y resolución de programas. También varios cientos de instrucciones simples se pueden expresar con una lista de unas cuantas líneas.

Finalmente, a la acción de realizar un programa se le conoce como programación.

En conclusión, reuniendo estos tres conceptos podemos decir: Un programa se escribe en un lenguaje de programación y a la actividad de expresar un algoritmo en forma de programa se le denomina programación.

A menudo, el lenguaje de programación se denomina software de programación cuando se emplea un término genérico, a fin de distinguirlo del hardware.

CLASIFICACION DE LOS PROGRAMAS

Parte del programa lo escriben los usuarios para ejecutar tareas que deseamos automatizar, pero además existen otros programas ya escritos que permiten procesar los programas del usuario. A continuación, se definirán estos dos tipos de programas.

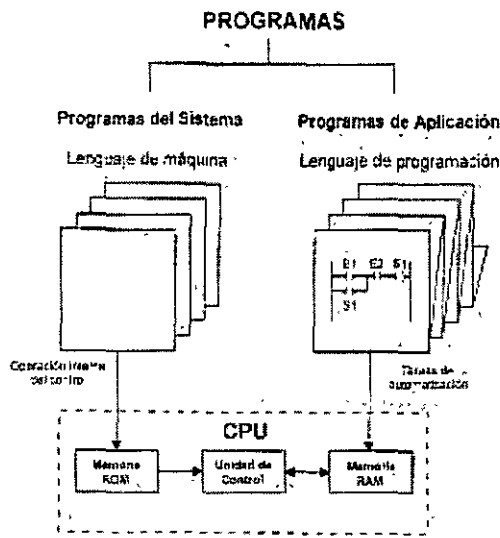
PROGRAMAS DEL SISTEMA

Existen cierto número de otros programas que proporcionan servicios vitales a los programas del usuario, esto es, realizan funciones operativas internas del controlador; estos programas, incluyendo los traductores de lenguaje reciben la denominación colectiva de programas del sistema o software del sistema. Un elemento notable de éste es el sistema operativo, cuyos servicios incluyen el manejo de los dispositivos de entrada y salida del PLC, el almacenamiento de la información durante largos períodos, organizar el procesamiento de los programas del usuario o aplicación, etc.

Estos programas están almacenados en memoria EPROM dentro de la CPU, por lo tanto no se pierden ni alteran en caso de pérdida de alimentación al equipo. El usuario No tiene acceso a ellos.

PROGRAMAS DE APLICACION DEL USUARIO

Es el conjunto de instrucciones o proposiciones que programa el usuario, con el fin de resolver tareas de automatización específica. Para ello, el usuario escribe el programa de acuerdo a la representación del lenguaje de programación que mejor se adapte a su trabajo, en todo caso, tenga un mejor dominio. Es importante señalar, que algunos fabricantes no emplean todos los tipos de representaciones de los lenguajes de programación, no obstante, el usuario tendrá que adaptarse a la representación que se disponga.



REPRESENTACION DE LOS LENGUAJES DE PROGRAMACION Y LA NORMA IEC 1131-3

En la actualidad cada fabricante diseña su propio lenguaje de programación, lo que significa, que existe una gran variedad comparable con la cantidad de PLCs que hay en el mercado.

Las formas que adopta el lenguaje de programación usado para realizar programas se denomina representación del lenguaje de programación.

Hasta el momento existen tres tipos de representaciones como las más difundidas a nivel mundial, las cuales cada fabricante la (s) emplea para su programación. estas son :

- Lista de instrucciones
- Plano defunciones y
- Diagrama contactos o plano de contactos

Es obvio, que la gran diversidad de lenguajes de programación da lugar a que cada fabricante tenga su propia representación, originando cierta incomodidad al usuario cuando programa más de un PLC.

Con el objetivo de uniformizar estas representaciones, se ha establecido una norma internacional IEC 1131-3 que se encarga de estandarizar los lenguajes de programación.

Esta norma contempla dos tipos de lenguajes de programación

- Lenguajes Gráficos
- Lenguajes Textuales

LENGUAJES GRAFICOS

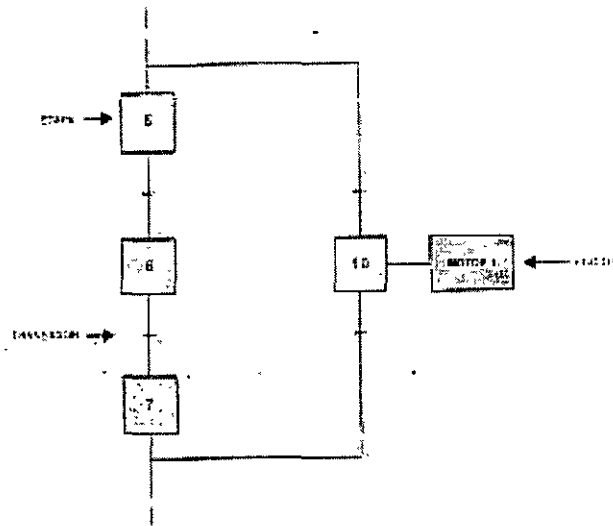
Se denomina lenguaje gráfico a la representación basada en símbolos gráficos, de tal forma que según la disposición en que se encuentran cada uno de estos símbolos Y en conformidad a su sintaxis que lo gobierna, expresa una lógica de mando y. control. Dentro de ellos tenemos

Carta de Funciones Secuenciales o Grafcet

El Grafcet es una representación de análisis gráfico donde se establecen las funciones de un sistema secuencial.

Este lenguaje consiste en una secuencia de etapas y transiciones, asociadas respectivamente con acciones y condiciones.

Las etapas representan las acciones a realizar y las transiciones las condiciones que deben cumplirse para ir desarrollando acciones. La Etapa - Transición es un conjunto indisociable.



Plano de Funciones

Es una representación gráfica orientada a las puertas lógicas AND, OR y sus combinaciones. Las funciones individuales se representan con un símbolo, donde su lado izquierdo se ubica las entradas y en el derecho las salidas. Los símbolos usados son iguales o semejantes a los que se utilizan en los esquemas de bloques en electrónica digital.

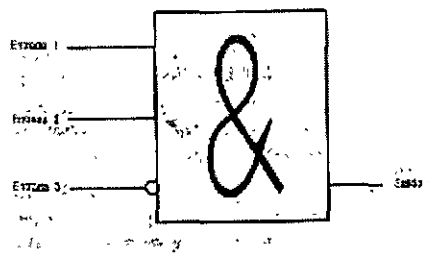
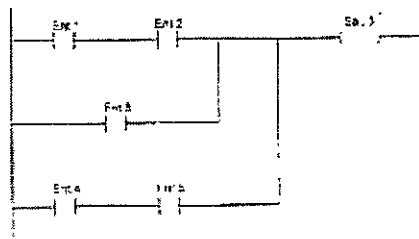


Diagrama de Contactos o Plano de Funciones

Es la representación gráfica que tiene cierta analogía a los esquemas de contactos según la norma Nema (USA).

Su estructura obedece a la semejanza que existe con los circuitos de control con lógica cableada, es decir, utiliza la misma representación de los contactos normalmente abiertos y normalmente cerrados, con la diferencia que su interpretación es totalmente diferente.

Además de los simples contactos que dispone, existen otros elementos que permiten realizar cálculos aritméticos, operaciones de comparación, implementar algoritmos de regulación, etc. Su gran difusión se debe por facilitar el trabajo a los usuarios



LENGUAJES TEXTUALES

Este tipo de lenguaje se refiere básicamente al conjunto de instrucciones compuesto de letras, códigos y números de acuerdo a una sintaxis establecida.

Se considera un lenguaje de menor nivel que los gráficos y por lo general se utilizan para programar pequeños PLCs cuyos programas no son muy complejos, o para programar instrucciones no programables en modo gráfico

Existen dos lenguajes diferentes en nivel y tipo de aplicación, ellos son

Lista de Instrucciones

Son instrucciones del tipo Booleanas, utilizando para su representación letras y números.

Dado que se usan abreviaturas nemotécnicas, no se requiere gran memoria para tareas de automatización.

La desventaja radica en la magnitud del trabajo que es necesario para su programación, especialmente si el programa consta de unos cientos de instrucciones.

Representación de un programa en lista de instrucciones para diferentes marcas de PLCs

Siemens (Simatic)	Telemecanique	General Electric
U E0.1	L I0.01	LD %I0001
U E0.2	A I0.02	AND %I0002
O E0.3	O I0.03	OR %I0003
= A3.1	= O3.01	OUT %Q0031

Texto Estructurado

Es un lenguaje del tipo booleano de alto nivel y estructurado, incluye las típicas sentencias de selección (IF-THEN-ELSE) y de interacción (FOR, WHILE Y REPEAT), además de otras funciones específicas para aplicaciones de control.

Su uso es ideal para aplicaciones en las que se requiere realizar cálculos matemáticos, comparaciones, emular protocolos, etc.

Programa en texto estructurado para un PLC marca Telemecanique TSX-07

```
LD [%MW10>100]

ST %Q0.3

AND [%MW20<=%MW35]

ST %Q0.2

LD %I0.2

OR [%MW30>=%MW40]

ST %Q0.4
```

DENOMINACION DE LOS LENGUAJES DE PROGRAMACION DE DIFERENTES PLCS

Cada fabricante ha nombrado mediante siglas o palabras compuestas a su lenguaje de programación o software de programación que lo identifica del resto de PLCs. A continuación se presenta una tabla donde se indican estos nombres.

LENGUAJE MARCA	GRAFICO			TEXTUAL	
	PLANO DE FUNCIONES	PLANO DE CONTACTOS	GRAFNET	LISTA DE INSTRUCCIONES	TEXTO ESTRUCTURADO
SIEMENS (Siemotica)	STEP 5	STEP 5, STEP 7	GRAPH 5, ST-GRAPH	STEP 5, STEP 7	STEP 5
SIEMENS (S)	TISOPT (MEL)		TISOPT (Máquina-Siema)	-	-
APC (Modicon)	MOOSOPT		-	MOOSOPT	-
KLÜCKNER MOELLER (Series PR30 - Series)	-	SUCOSOPT S 30	-	MOOSOPT S 30	-
TELEMECANIQUE	-	PL7 - 2	PL7 - 2	PL7 - 1	PL7 - 0
ALLEN BRADLEY	-	APS	-	-	-
GENERAL ELECTRIC (Pascal)	-	LOGICMASTER 30	-	-	LOGICMASTER 30

ESTRUCTURA DEL PROGRAMA DE APLICACION

Los Programas de aplicación se estructuran de acuerdo al modo como se procesan los programas (tareas). éstas pueden ser de dos tipos:

PROGRAMACION LINEAL

Se emplea para aplicaciones simples de automatización, su procesamiento es cíclico o secuencial y es suficiente programar las diferentes instrucciones en un solo bloque o sección de programación.

Un procesamiento cíclico o secuencial, consiste en la lectura, interpretación y ejecución de instrucción por instrucción, respetando el orden en que se han programado, salvo las instrucciones de salto. Para ejecutar las instrucciones se utilizan informaciones procedentes de la imagen de proceso de entradas (IPE), memorias internas, memorias intermedias, así como los datos actuales de los temporizadores y contadores. Los resultados se escriben en la imagen de proceso de salidas (IPS).

Después de la ejecución del programa se corre un ciclo de datos, esto significa el proceso durante el cual los datos de la IPS se transfieren a los módulos de salida, y simultáneamente, se transfieren a la IPE los datos actuales de los módulos de entrada. Con esta IPE actualizada, vuelve a lanzarse la ejecución del programa, lo que significa repetir todo el proceso desde el inicio.

Los PLCs que realizan solamente este tipo de procesamiento, están diseñados con microprocesadores del tipo (intel 8086/8088) que se caracterizan por su limitada capacidad para ejecutar un solo programa a la vez.

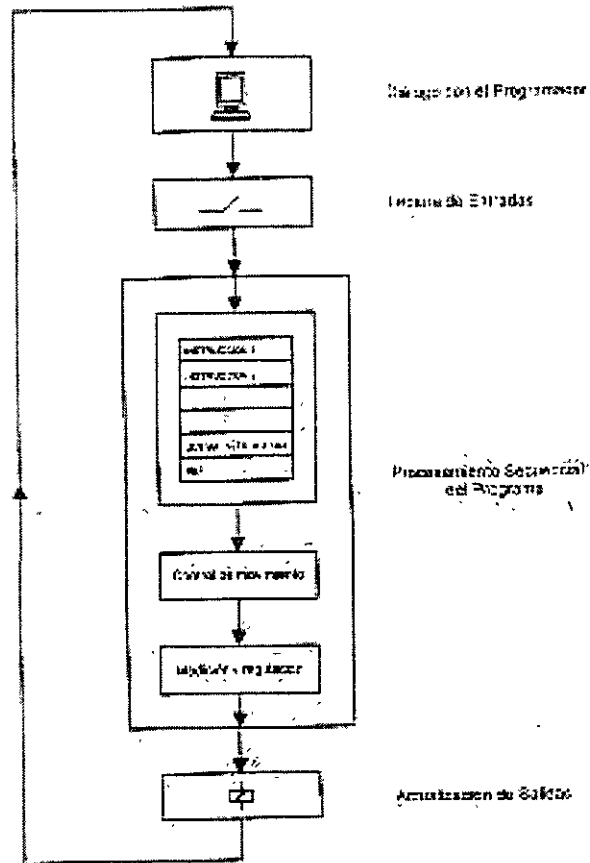
Estos tipos de PLCs son denominados también PLCs secuenciales, con capacidad además de ejecutar tareas de regulación, de comunicación, etc.

Sin embargo, esta forma de procesamiento dificulta notablemente el trabajo cuando se tiene que procesar diferentes funciones a la vez, y en algunos casos es casi imposible estructurar los programas debido a las siguientes desventajas:

- Incremento del tiempo de barrido, que es proporcional a la complejidad del programa.
- En extensos programas es muy tedioso su diagnóstico. Modificación y puesta a punto.
- Dificultad para la concepción del programa resultando complejo y difícil interpretarlo y actualizarlo.
- En muchos casos es indispensable el cumplimiento en tiempo real defunciones avanzadas tales como:

* medición analógica y regulación

- * servoposicionamiento
- * comunicación para el diálogo operador y control
- * funciones de monitoreo, etc.



PROGRAMACION ESTRUCTURADA

Cuando se desea programar tareas de automatización muy complejas donde utilizar una programación lineal resulta demasiado laborioso, es conveniente en este caso dividir el problema en partes, de tal forma, que interpretándolo y resolviéndolo en forma parcial mediante bloques y al final unir este conjunto de programas en uno solo, resulta significativamente más fácil para el usuario.

A esta filosofía de programación se le conoce con el nombre de Programación Estructurada, que consiste en la división del programa de aplicación en bloques que se caracterizan por una independencia funcional, donde cada bloque del programa realiza una tarea específica claramente definida.

La programación estructurada optimiza el tiempo de escaneo ya que no se ejecutan todos los bloques en cada ciclo de barrido, ejecutándose sólo los que están en actividad en el momento dado.

Las ventajas que se obtienen programando en forma estructurada son

La comprensión, solución, simulación y pruebas es mucho más fácil cuando un problema muy complejo es tratado por partes.

El diagnóstico de fallas y por ende su solución es también más fácil, dado que una vez identificado el bloque del programa donde se encuentra la falla, su corrección resulta más rápido que si se afrontara el programa global.

Los programas parciales pueden ejecutarse independientemente por equipos de programadores, cada grupo elaborando bloques individuales; además se pueden usar reiteradamente durante el escaneo del programa, o formar parte de otro programa de aplicación.

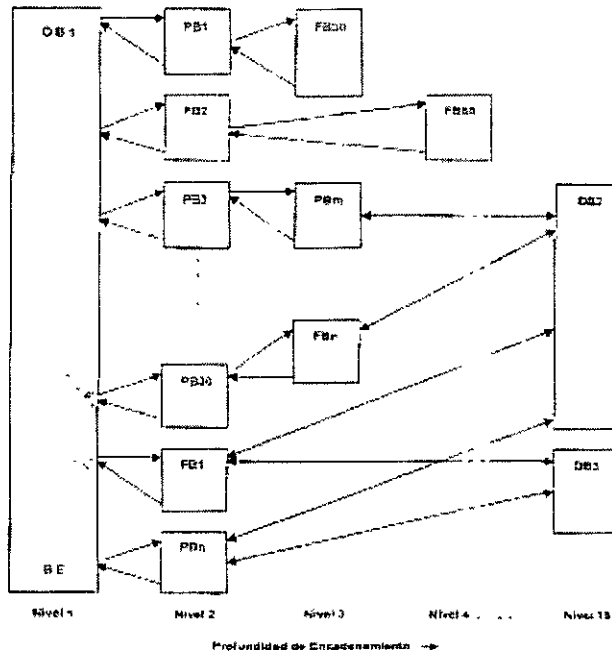
Se emplea mejor la capacidad de la memoria dado que pueden llamarse los bloques de programas las veces que se requiera sin que se tenga que programar repetidas veces.

Optimización del tiempo de barrido.

Por otro lado, dependiendo del tipo de procesador que disponga el PLC la programación estructurada puede aprovecharse con menor o mayor Eficiencia.

Este es el caso, como se mencionó anteriormente de los PLC diseñados en base a microprocesadores del tipo mono tarea, donde la programación estructurada compuesta por una serie de bloques de programación, se ejecuta en base al procesamiento secuencial o lineal de un bloque matriz, que viene hacer el núcleo de la estructura.

A continuación se puede ver un ejemplo de una programación estructurada cuya distancia medida por el número de bloques a los que "salta", se le conoce como Profundidad de Encadenamiento o Anidado. Con este tipo de microprocesador no se puede realizar en forma simultánea otras tareas como diálogo hombre-máquina, procesamiento analógico, etc.



OB: Módulo de Organización

PB Módulo de Programa

FB Módulo Funcional

DE3 Módulo de Datos

Sin embargo, hoy en día se cuenta con procesadores de mayor velocidad de procesamiento, mayor memoria y características adicionales que le permiten ejecutar a los PLCs programas más rápidamente, estos son los procesadores multifunción (286, 386, 486, etc.), con capacidad de ejecutar varios programas en forma simultánea tales como tareas de posicionamiento, medición analógica, tratamiento secuencial, diálogo, etc.

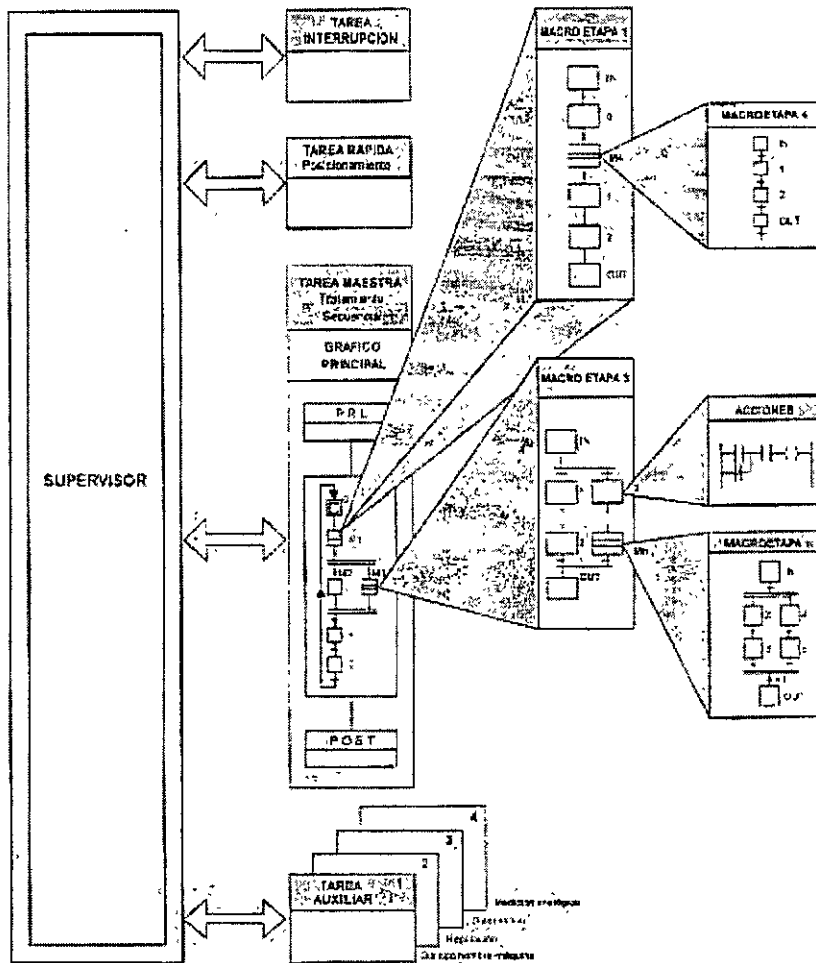
Los PLCs multifunción desarrollados en base a microprocesadores multitarea se caracterizan por su mayor velocidad para atender diferentes programas a la vez y en tiempo real, además por su mayor capacidad de memoria para ejecutar varios programas simultáneamente sin originar conflictos.

En la siguiente figura se muestra la estructura de la multitarea, donde el conjunto de programas o tareas son totalmente independientes, un supervisor gobierna la ejecución de las diferentes tareas.

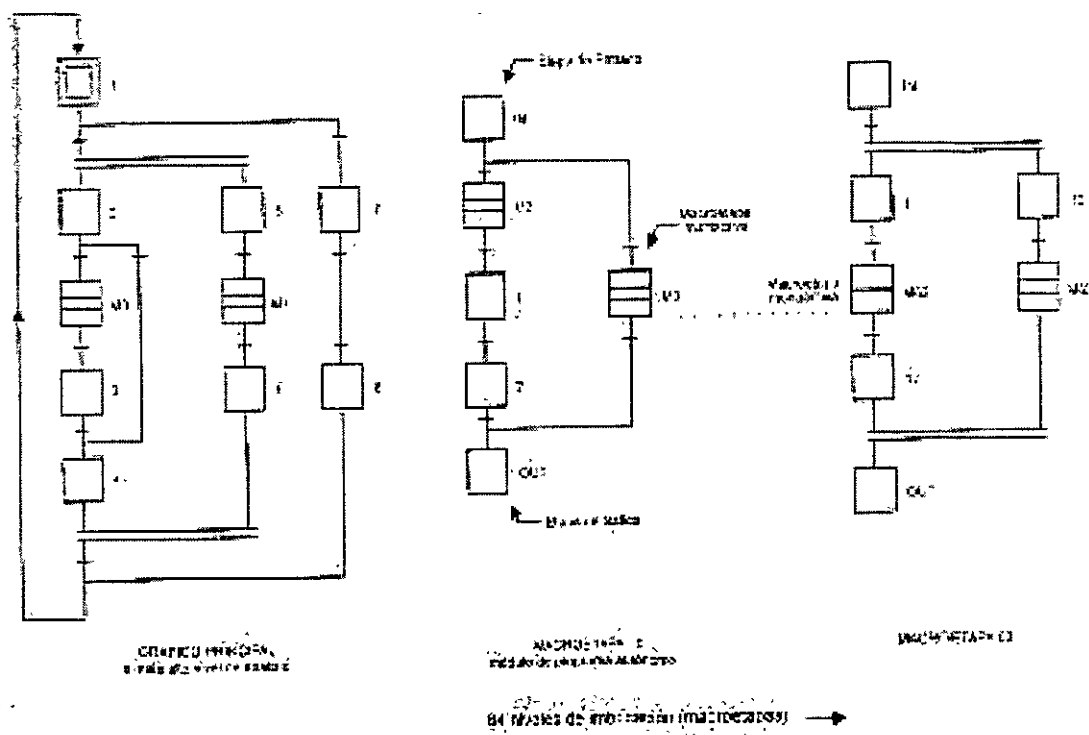
Así también, en estos procesadores la concepción del tratamiento secuencial es en base a la división en bloques de programas, algo así como subrutinas, que es básicamente el concepto de la programación estructurada.

En conclusión, la diferencia en el procesamiento de estos dos tipos de programas estructurados radica en que el primero funcionando con microprocesadores mono tarea, ejecutan los diversos módulos o bloques de programación según un procesamiento secuencial, es decir, uno a continuación del otro, mientras que el procesador multifunción además del procesamiento secuencial, puede ejecutar el programa estructurado independientemente si se ejecutó el bloque anterior. Esto significa, que si en algún momento durante el proceso de barrido del programa en el sistema de control se origina una contingencia, puede ejecutarse una tarea de interrupción sin tener que esperar el barrido total del programa

Programación estructurada con procesador multifunción (diagrama de bloques según lenguaje de programación PL7-3 de Telemecanique)



En la siguiente figura se muestra los bloques de programas en tratamiento secuencial y en Grafcet.



El número total de etapas entre el primer principal y las 64 microetapas es como máximo 512

INTRODUCCION A LA PROGRAMACION

Antes de empezar con la programación propiamente dicha, es necesario definir algunos conceptos que proporcionen al lector las bases suficientes para comprender de la manera más clara, el desarrollo de los temas que se tocarán más adelante en lo referente a la programación básica y avanzada, así por ejemplo, el lector deberá estar en condiciones de diferenciar una señal discreta de una analoga, representar las cantidades binarias, estructurar una instrucción de mando, tener presente las reglas básicas para las diferentes representaciones de los lenguajes de programación, etc.

Por consiguiente, el éxito que se tenga en lo sucesivo dependerá de lo aprendido en esta parte introductoria.

TIPOS DE SEÑALES

Existen dos tipos de señales bien definidas que un PLC puede procesar, estos son

SEÑAL DISCRETA

Este tipo de señal es conocido también con los siguientes nombres

- señal binaria

- señal digital
- señal lógica
- señal todo o nada (TON)

Se caracteriza porque sólo pueden adoptar uno de dos posibles estados o niveles. A estos dos estados posibles se le asocia para efectos del procesamiento el estado de señal "0" y el estado de señal "1". Así mismo, estos estados cuando se relaciona de acuerdo a su condición eléctrica se dice: no existe tensión y, existe tensión, la magnitud de la tensión no interesa ya que dependerá del diseño del componente electrónico que pueda asumir esta tensión nominal.

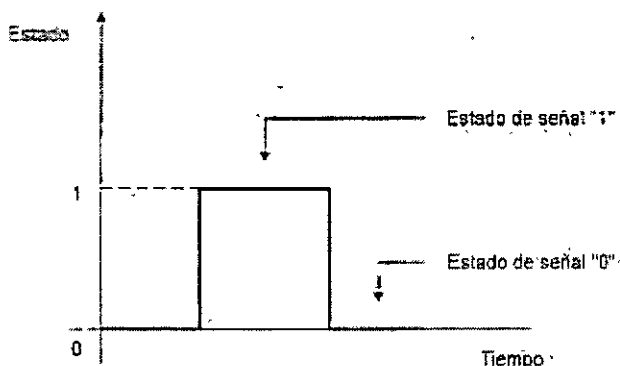
Como ejemplo se pueden citar aquellos dispositivos de campo de entrada y salida de donde provienen o se asigna una señal discreta con respecto a un PLC.

Entrada

- pulsador
- interruptor de posición
- interruptor fotoeléctrico, etc.

Salida

- contactor
- lámpara indicadora, etc.

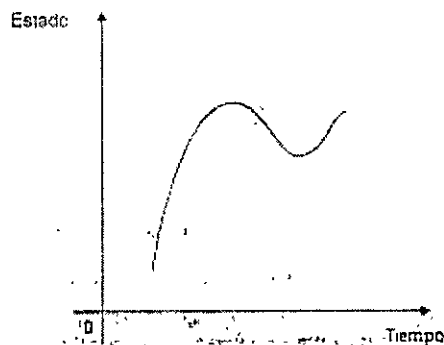


SEÑAL ANALOGA

Se conoce como señal analoga, aquella cuyo valor varía con el tiempo y en forma continua, pudiendo asumir un número infinito de valores entre sus límites mínimos y máximos.

A continuación se citan algunos parámetros físicos muy utilizados en los procesos industriales, tal que, en forma de señal análoga pueden ser controlados y medidos.

- temperatura
- velocidad
- presión
- flujo,
- nivel, etc.



REPRESENTACION DE LAS CANTIDADES BINARIAS

Dado que el PLC recibe la información proveniente del proceso ya sea en forma discreta o análoga, donde la información se almacena en forma de una agrupación binaria, es preciso por lo tanto, disponer de un medio de representación que facilite su manejo y mejore la capacidad de procesamiento.

Para ello se emplean con mayor frecuencia tres tipos de representación para la información. éstos son: bit, byte y palabra, en algunos casos se utilizan la doble palabra.

BIT

El bit es la unidad elemental de información donde sólo puede tomar dos valores un "1" ó un "0", es decir, un bit es suficiente para representar una señal binaria.

BYTE

El byte es una unidad compuesta por una agrupación ordenada de 8 bits, es decir, ocho dígitos binarios. Los bits se agrupan de derecha a izquierda tomando como número de bit del 0 al 7.

En un byte se puede representar el estado de hasta ocho señales binarias, puede usarse para almacenar un número cuya magnitud como máximo sería:

Número máximo de un byte = $11111111 = 2^8 - 1 = 255$

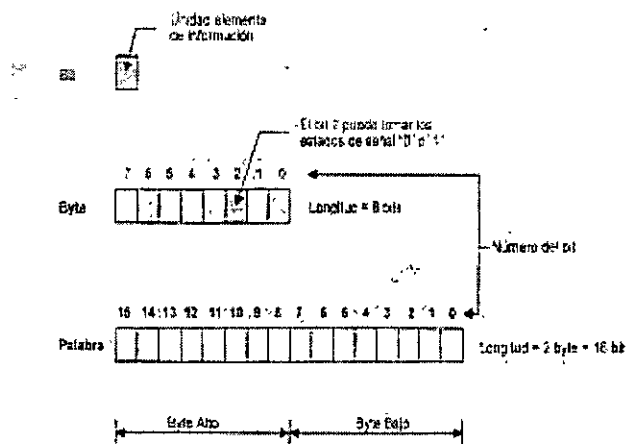
PALABRA

Para obtener mayor capacidad de procesamiento a veces se agrupan los bytes formando lo que se denomina las palabras.

La palabra es una unidad mayor compuesta de 16 bits = 2 bytes. Los bits de una palabra se agrupan de derecha a izquierda tomando como número de bit del 0 al 15.

En una palabra se pueden representar hasta 16 señales binarias, puede usarse para almacenar un número cuya magnitud como máximo sería

Número máximo en una Palabra = $2^{16} - 1 = 65535$



DIRECCIONAMIENTO DE BITS

Cuando se elabora un programa de control, se van indicando las diferentes instrucciones de mando donde en cada instrucción se indica que operación se debe ejecutar, también figura la dirección exacta del módulo y canal o terminal de conexión de las señales de E/S involucradas en el proceso.

El direccionamiento puede realizarse de dos formas

- Direccionamiento Fijo
- Direccionamiento Variable

DIRECCIONAMIENTO FIJO

Cuando la dirección de las señales de E/S queda determinada por la posición o puesto de enchufe en que están ubicados los módulos de E/S respecto a la CPU, se dice que el direccionamiento es fijo. Además, un direccionamiento fijo puede ser del tipo Octal (byte) o hexadecimal

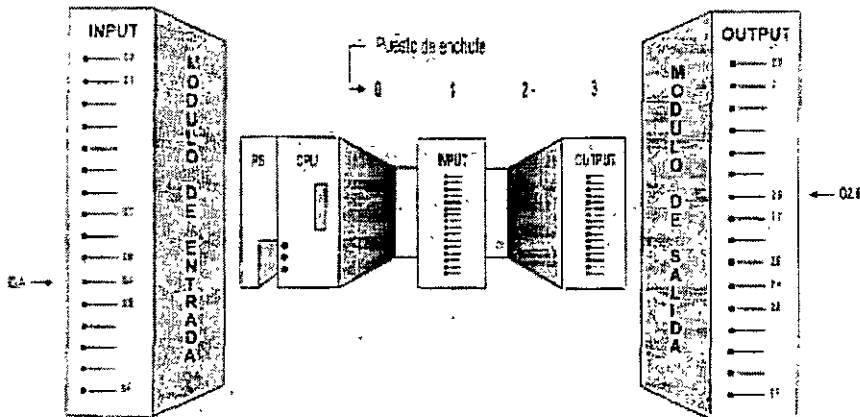
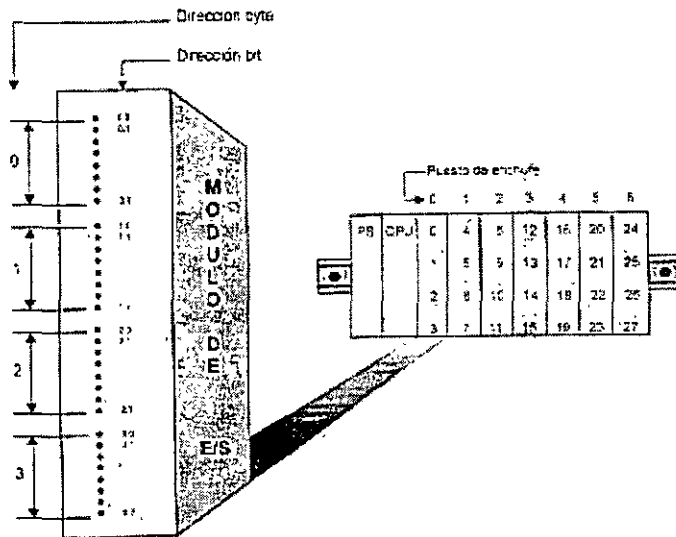
DIRECCIONAMIENTO FIJO DEL TIPO OCTAL (BYTE)

Un direccionamiento del tipo octal queda determinado cuando a cada módulo de E/S se le agrupa los terminales por bytes, es decir, en grupos de 8 bits del (0 al 7).

En este caso, en la dirección se especificará el byte correspondiente al terminal seleccionado y que pertenece al puesto de enchufe según L posición que ocupa.

DIRECCIONAMIENTO FIJO DEL TIPO HEXADECIMAL

Este direccionamiento se diferencia del anterior en el agrupamiento de los terminales, siendo para este caso del tipo hexadecimal, ósea en grupos de 16 bits del (0 al F).



PROGRAMACION EN LISTA DE INSTRUCCIONES

Es una forma sencilla de programar aplicaciones de automatización sin necesidad de requerir conocimientos previos de alguna materia, debido a que los programas están basados por instrucciones del tipo booleano con simbología elemental y precisa.

Algunas de las limitaciones que presenta esta forma de programar son:

- cuando se tiene muchas instrucciones es difícil entender rápidamente de lo que trata el programa
- un programa que consta de una gran cantidad de instrucciones es muy laborioso ingresarlas utilizando cualquier tipo de programador
- se emplea mayor tiempo en el diagnóstico y detección de fallas. etc.

No obstante, una de las ventajas que presenta, es que los programadores diseñados para este propósito no son muy costosos (hand-held) ni requieren softwares especiales como en el caso de las PCs.

En esta parte se reconocerá la estructura de una instrucción de mando con ejemplos para algunas marcas de PLC, y a continuación las operaciones binarias utilizando esta forma de representación

ESTRUCTURA DE UNA INSTRUCCION DE MANDO

Una instrucción de mando es la parte más pequeña de un programa y representa para el procesador una orden de trabajo.

Para que la instrucción de mando cumpla su función es necesario especificar dos partes : la parte operacional y la parte del operando.

INSTRUCCIÓN DE MANDO

OPERACION	OPERANDO
------------------	-----------------

Tipo	Dirección
-------------	------------------

La parte operacional representa lo que hay que hacer, esto significa la operación a ejecutar. Por ejemplo, ejecutar un(a)

- combinación binaria Y (And)

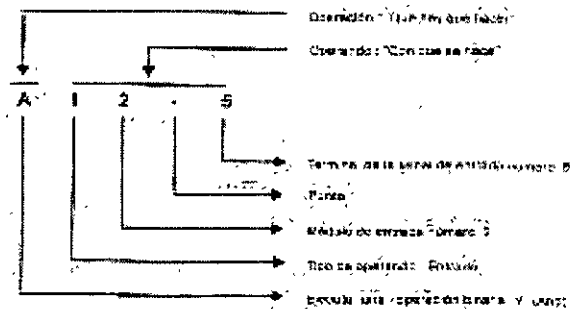
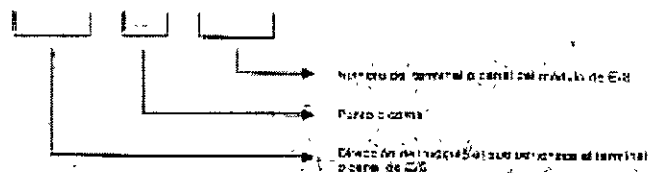
- combinación binaria O (Or)

- combinación binaria O-exclusiva (X0)
- operación de carga L (Load)
- operación de transferencia T (Transference)
- salto a una instrucción determinada JMPi (Jump)
- asignación a un resultado =, etc.

La parte del operando está compuesto por el tipo de operando y su dirección. El operando responde a la pregunta con que se hace la operación. El tipo de operando puede ser un (a)

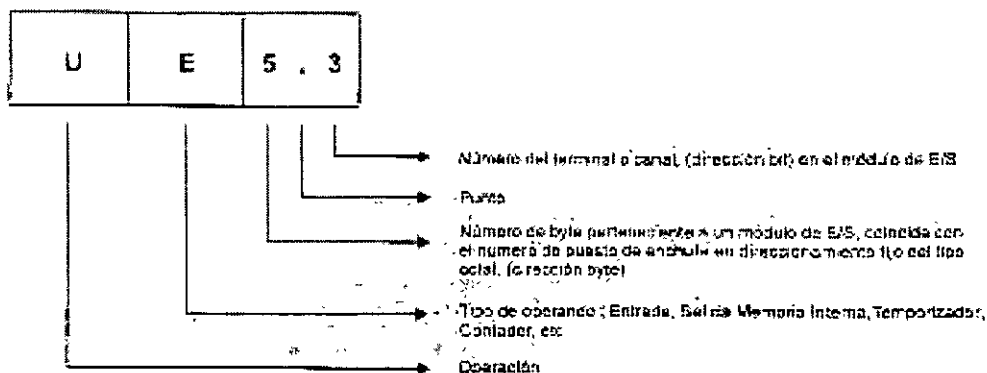
- entrada
- salida
- memoria interna
- dato
- temporizador
- contador, etc.

La dirección del operando se define según el tipo de direccionamiento que se emplee, fijo o variable y del número del terminal de los módulos de E/S.



EJEMPLOS DE INSTRUCCIONES DE MANDO PARA DIFERENTES MARCAS DE PLCs

A continuación se detalla para determinadas marcas de PLCs la estructura de su instrucción de mando dando algunos ejemplos para una mejor comprensión.



Ejemplos:

INSTRUCCIONES SIGNIFICADO

ALEMAN INGLES

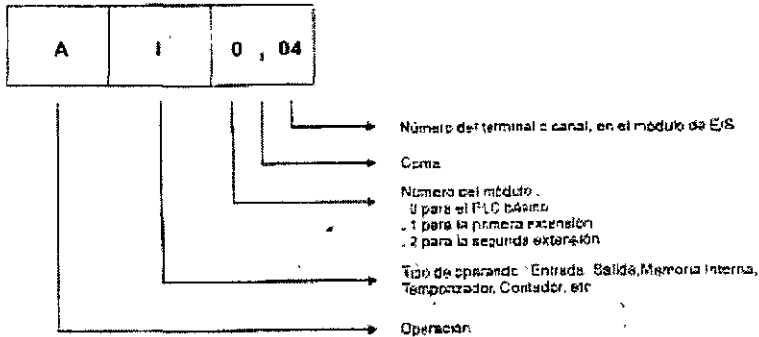
U E 5.3 A I 5.3 Lectura del estado de señal del canal 3, de un modulo de entradas digitales de 8 canales, enchufado en el puerto 5.

= A I 10.6 = Q 10.6 Salida del estado de señal por el canal 6, de un modulo de salida digital de 32 canales enchufado en el puerto 2, dirección byte 10.

ON M 3.7 ON F 3.7 Lectura del estado negado de la marca, con dirección 3 y dirección bit 7.

L EB 7 L IB 7 Lectura de los estados de señal de todo los canales, de un modulo digital de entrada de 8 canales enchufado en el puerto 7.

COMPACTOS



Ejemplos:

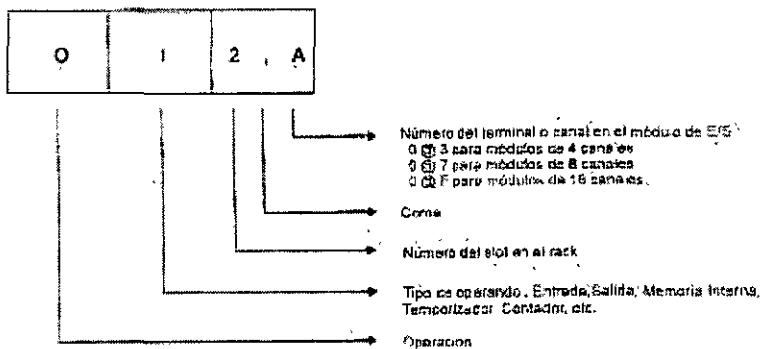
INSTRUCCIONES SIGNIFICADO

A 10.04 Lectura del estado de señal del canal 4, del modulo 0 (modulo básico)

= O2.07 Salida del estado de señal por el canal 7, del modulo 2 (modulo de segunda extensión)

L T5 Lectura del temporizador numero 5

MODULARES



Ejemplo:

INSTRUCCIONES SIGNIFICADO

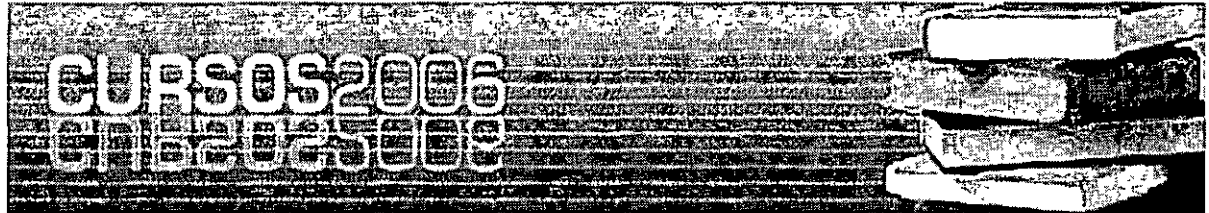
O I 2,A Lectura del estado de señal del canal 10, del modulo de entrada digital de 16 canales, enchufado en el puerto (slot) 2.

= O 14,2 Salida del estado de señal por el canal 2, del modulo de salida digital de 32 canales, enchufado en el puerto (slot) 14.

A C8 Lectura del contador numero 8.



FACULTAD DE INGENIERÍA UNAM
DIVISIÓN DE EDUCACIÓN CONTINUA



CURSOS ABIERTOS

CIRCUITOS LÓGICOS PROGRAMABLES

CA 522

TEMA
PROGRAMACIÓN DE PLC

EXPOSITOR: Ing. Francisco Rodríguez Ramírez
Del 14 al 25 de agosto de 2006
PALACIO DE MINERÍA

Programación de PLC

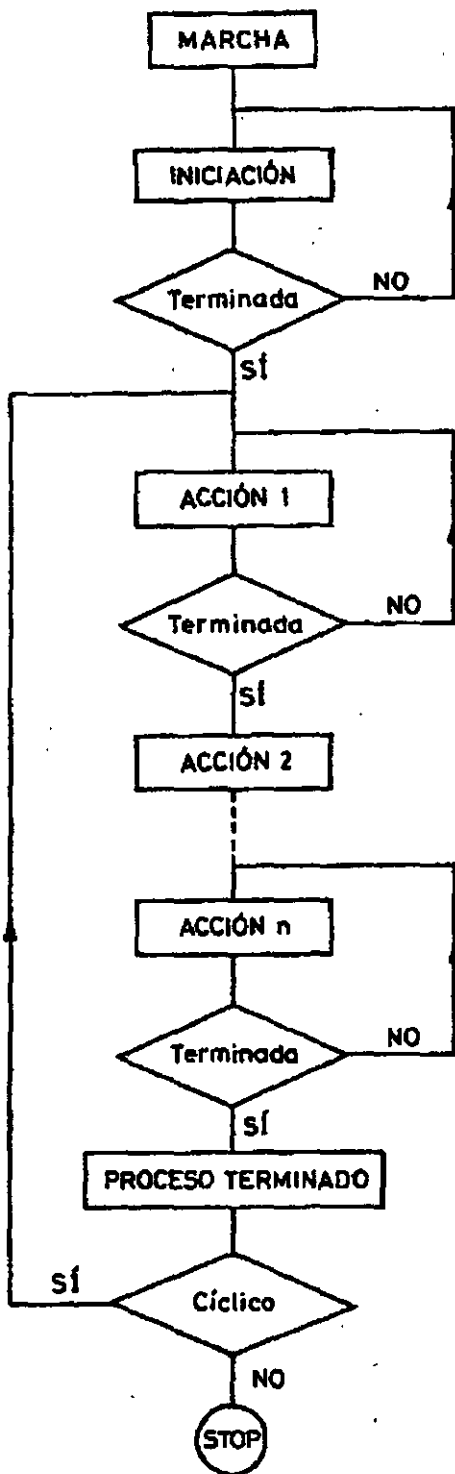
Introducción

El GRAFCET surge en Francia a mediados de los años 70, debido a la colaboración de algunos fabricantes de autómatas, como Telemecanique y Aper con dos organismos oficiales, AFCET (Asociación francesa para la cibernética, economía y técnica) y ADEPA (Agencia nacional para el desarrollo de la producción automatizada). Homologado en Francia, Alemania, y posteriormente por la comisión Electrónica Internacional (IEC 848, año 1988).

Actualmente es una herramienta imprescindible cuando se trata de automatizar procesos secuenciales de cierta complejidad con autómatas programables.

El GRAFCET es un diagrama funcional que describe la evolución del proceso que se quiere automatizar tal y como se muestra en la figura. Está definido por unos elementos gráficos y unas reglas de evolución que reflejan la dinámica del comportamiento del sistema.

Todo automatismo secuencial o concurrente se puede estructurar en una serie de etapas que representan estados o subestados del sistema en los cuales se realiza una o más acciones, así como transiciones, que son las condiciones que deben darse para pasar de una etapa a otra.



Elementos Gráficos

Las Etapas

Las etapas representan cada uno de los estados del sistema. El símbolo empleado para representar una etapa es un cuadrado con un número o símbolo en su interior que la identifica. Las etapas iniciales se representan por un cuadrado con doble línea. Cuando se recorre el gráfico de evolución, por cualquier camino posible, deben alternarse siempre una etapa y una transición.

REPRESENTACION DE ETAPAS:



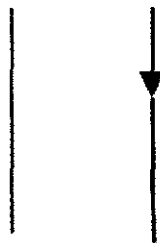
REPRESENTACION DE ETAPAS INICIALES:



Las acciones que llevan asociadas las etapas se representan con un rectángulo donde se indica el tipo de acción a realizar. Una etapa puede llevar asociadas varias acciones.

Las Líneas de Evolución

Las líneas de evolución unen entre si las etapas que representan actividades consecutivas. Las líneas se entenderan siempre orientadas de arriba abajo, a menos que se represente una flecha en sentido contrario. Dos líneas de evolución que se crucen debe de interpretarse, en principio que no están unidas.



Las Transiciones

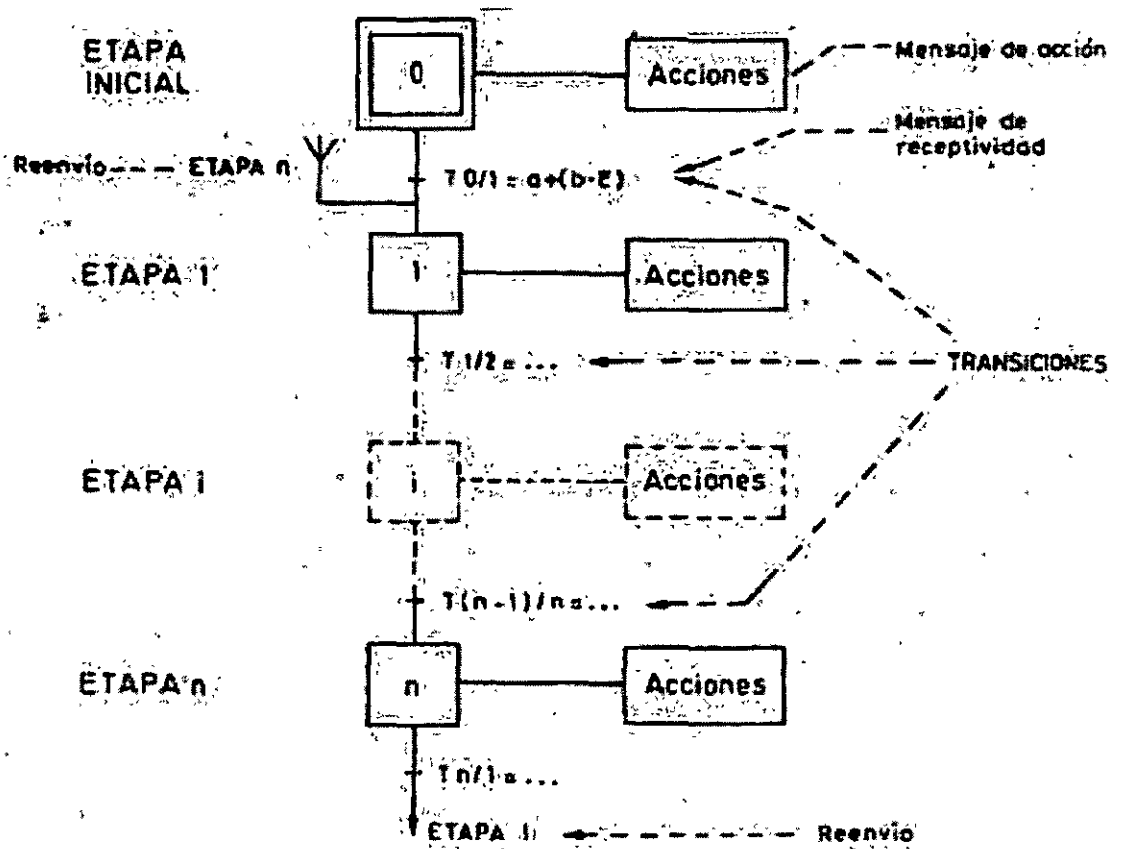
Las transiciones, representan las condiciones lógicas necesarias para que finalice la actividad de una etapa y se inicie la de la etapa o etapas inmediatamente consecutivas.

Gráficamente se representan por una línea cruzada sobre las líneas de evolución.



Los Reenvíos

Son símbolos en forma de flecha que indican la procedencia o destino de las líneas de evolución.



Reglas de Evolución

El proceso se descompone en etapas, que serán activadas de forma secuencial.

Una o varias acciones se asocian a cada etapa. Están acciones sólo están activas cuando la etapa está activa.

Una etapa se hace activa cuando la precedente lo está y la condición de transición entre ambas etapas ha sido activada.

La activación de una condición de transición implica la activación de la etapa siguiente y la desactivación de la etapa precedente.

La etapa inicial tiene que ser activada antes de que se inicie el ciclo del GRAFCET, un ciclo está formado por todas las etapas posteriores a la etapa inicial.

Existen procesos que requieren estructuras mas complejas, en las que se representan bucles, tomas de decisiones o tareas simultaneas que deben sincronizarse. Para estos casos el GRAFCET dispone de otras estructuras básicas a partir de las cuales pueden generarse los diagramas de dichos progresos.

Las tres estructuras básicas del GRAFCET, de las cuales pueden derivarse todas las demás, son:

- Secuencia lineal
- Convergencia y divergencia en o (subprocesos alternativos).
- Convergencia y divergencia en y (subprocesos simultáneos).

Secuencia Lineal

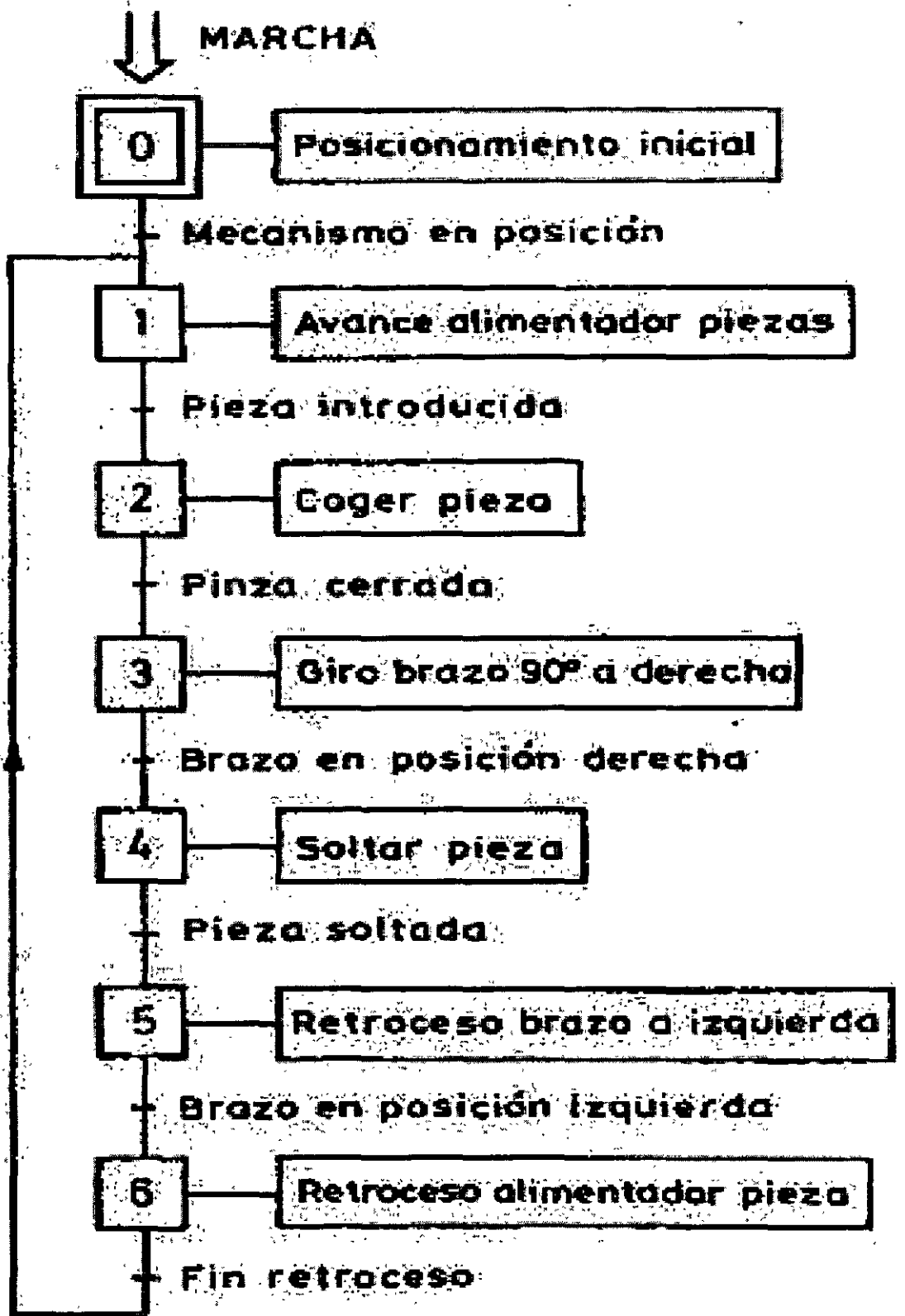
La secuencia lineal es la estructura más simple posible y consiste en una sucesión de etapas unidas consecutivamente por las líneas de evolución y condiciones de transición.

Dentro de un tramo de secuencia lineal, solamente una etapa debe estar activada en un instante determinado.

Se activa una etapa cuando se encuentra activada la anterior y se cumplan las condiciones de transición entre ambas.

La activación de una etapa implica automáticamente la desactivación de la etapa anterior.

Una secuencia lineal puede formar parte de una estructura más compleja.



Convergencia y divergencia en "o"

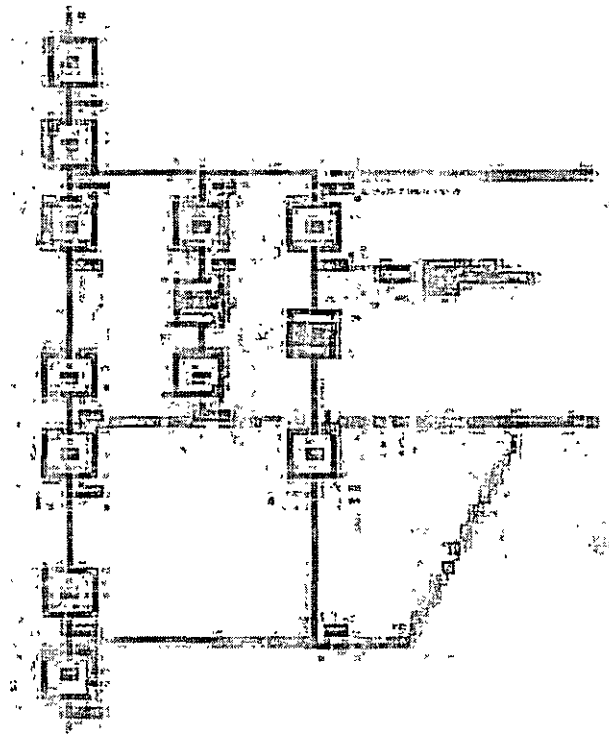
La divergencia y la convergencia en "o", a las que llamaremos conjuntamente bifurcación en "o", forman una estructura en la que existen los siguientes elementos:

- Una divergencia en "o", en la que se inician varios caminos o subprocesos alternativos posibles.
- Una serie de caminos alternativos con una macroestructura lineal, aunque pueden tener otras estructuras mas complejas.
- Una o mas confluencias en "o" de dichos caminos alternativos, de tal forma que la macroestructura debe ser globalmente cerrada.

Las propiedades básicas que cumple la estructura de bifurcación en "o" son las siguientes:

- A partir del punto de divergencia el proceso podrá evolucionar por distintos caminos alternativos, cada uno de ellos debe tener su propia condición de transición.
- Las condiciones de transición de los diversos caminos de divergencia han de ser excluyentes entre si, de forma que el proceso solo podrá progresar en cada caso por uno de ellos.

■ A nivel de gráfico global, los distintos caminos iniciados como divergencia en "o" deben confluir en uno o más puntos de convergencia en "o". Dicho de otra forma, la estructura debe ser totalmente cerrada y no pueden existir caminos abiertos, ya que esto denotaría situaciones sin posible salida.



La desactivación de la etapa previa a una divergencia se produce al activarse una cualquiera de las etapas siguientes, según una ecuación lógica en "o".

$$\text{RESET } B9 = Q10 + Q20 + Q30$$

("o" de todas las ramificaciones divergentes)

La activación de la etapa siguiente a una divergencia depende de la etapa previa y de la condición particular del camino activado, como si se tratara de una secuencia lineal:

$$\text{SET} \quad B10 = Q9 * (C9-1)$$

La activación de la etapa siguiente a una convergencia depende de las etapas previas según una ecuación en "o".

$$\text{SET} \quad B40 = Q19 * C19 + Q25 * C25$$

("o" de todas las ramificaciones concurrentes)

Convergencia y divergencia en y

La divergencia y convergencia en " y ", a las que llamaremos conjuntamente bifurcación en " y ", forman una estructura en las que existen los siguientes elementos:

■ Una divergencia en " y " en la que se inician varios caminos o subprocesos que deben iniciarse

simultáneamente cuando se cumpla una determinada condición de transición común.

- Una serie de caminos simultáneos con una macroestructura lineal, aunque pueden contener otras estructuras más complejas.

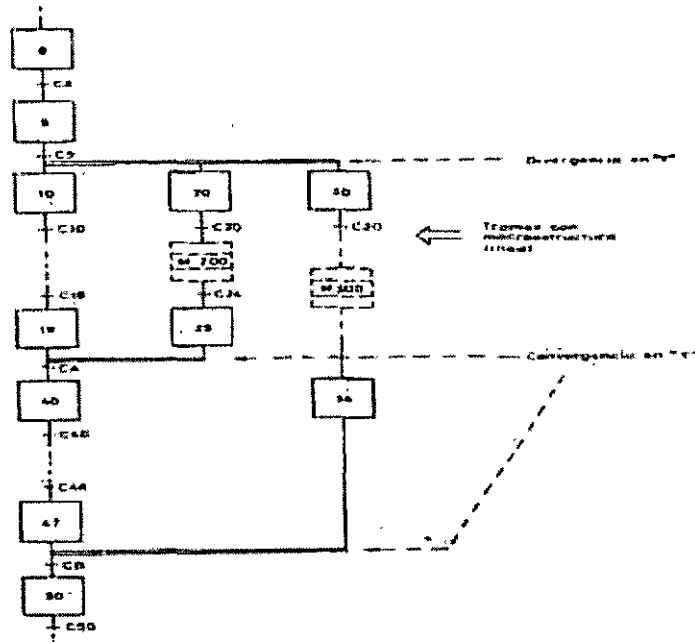
- Una o más confluencias en " y " de dichos caminos, de manera que la macroestructura debe ser globalmente cerrada

Las propiedades que cumplen las convergencias y divergencias en " y " son las siguientes:

- A partir del punto de divergencia el proceso evolucionara por varios caminos a la vez, ejecutando varias tareas simultáneamente.

- La condición de transición para iniciar las tareas simultaneas es única y común para todas ellas.

- La convergencia en "y " impone de por si una condición de transición: todas las tareas que confluyan deben haber terminado para que el proceso pueda continuar.



La etapa previa a una divergencia "y" no debe desactivarse hasta que se hayan activado todas las etapas siguientes, según una ecuación lógica en "y".

$$\text{RESET } B9 = Q10 * Q20 * Q30$$

("y" de todas las ramificaciones divergentes)

La activación de etapa inmediatamente después de una divergencia depende de que este activada la etapa inmediatamente anterior y de la condición de transición común.

$$\text{SET } B10 = Q9 * C9$$

$$\text{SET } B20 = Q9 * C9$$

$$\text{SET} \quad B30 = Q9 * C9$$

La activación a una etapa siguiente a una convergencia en "y" depende de que estén activadas todas las etapas previas.

$$\text{SET} \quad B40 = Q19 * Q25 * CA$$

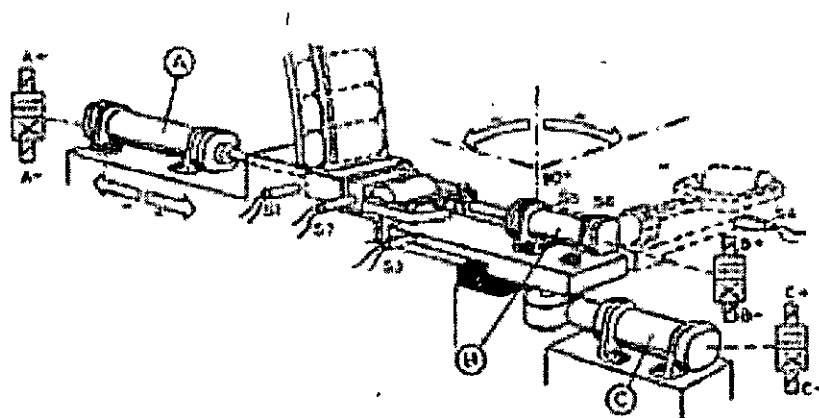
("y" todas las ramificaciones concurrentes)

$$\text{SET} \quad B50 = Q47 * Q34 * CB$$

Ejercicio de Diseño

Introducción

Para ilustrar el método de diseño basado en GRAFCET vamos a desarrollar un ejemplo. Se trata del diseño de un automatismo para control de manipulador de la figura 3.1.8.



A+ Empuje alimentador S1-
Final retroceso alimentador

A- Retroceso alimentador S2-
Final avance alimentador

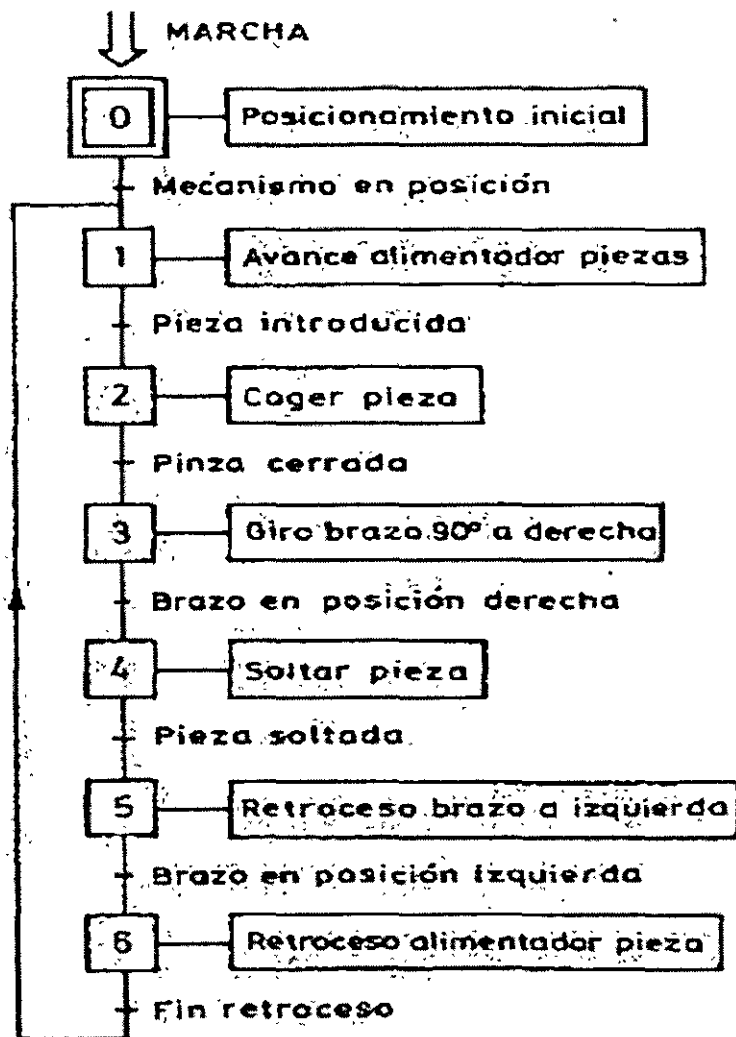
B+ Cierre pinza S3-
Brazo en posición izquierda

B- Apertura pinza S4-
Brazo en posición derecha

C+ Giro brazo a derecha Pinza abierta	S5-
C- Giro brazo a izquierda Pinza cerrada	S6-

1ª fase GRAFCETfuncional

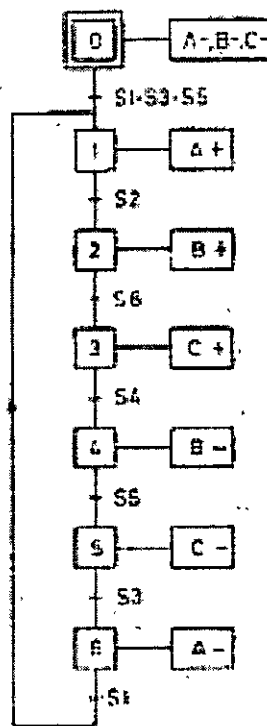
En esta primera fase se realiza el GRAFCET como una simple sucesión de acciones a desarrollar, sin definir la forma ni los medios empleados para ejecutarlas, indicando al lado de las etapas cada una de las acciones a desarrollar y entre ellas las condiciones de transición.



2ª fase GRAFCET sensores y accionamientos

Esta segunda fase es prácticamente igual a la primera. Como se puede observar en el GRAFCET de esta fase, las únicas diferencias con respecto al GRAFCET anterior son las especificaciones de los detalles indicando cual es la tecnología empleada para implementar el automatismo.

La tecnología empleada pueden ser los accionamientos dedicados a ejecutar las distintas operaciones (cilindros, motores, electroválvulas, etc.) y los sensores (pulsadores, finales de carrera, etc.) destinados a suministrar las receptividades que nos permitirán formular las condiciones de transición.



3ª fase diseño del sistema de control

Una vez obtenido el gráfico de control, conteniendo todos los accionamientos y sensores este puede ser utilizado para el diseño del sistema de control.

Pasos a seguir para el diseño:



Realizar una tabla de asignación de variables.

TABLA DE ASIGNACIÓN DE VARIABLES		
VARIABLES DE ENTRADA		ENTRADA AL AUTÓMATA
Final de carrera	S1	0001
Final de carrera	S2	0002
Final de carrera	S3	0003
Final de carrera	S4	0004
Final de carrera	S5	0005
Final de carrera	S6	0006
ETAPAS DEL GRAFCET		RELES INTERNOS DE MEMORIA DEL AUTÓMATA
	E0	1000
	E1	1001
	E2	1002
	E3	1003
	E4	1004
	E5	1005
	E6	1006
ACCIONES DEL GRAFCET		SALIDAS DEL AUTÓMATA
	Alimentador	0900
	Pinza	0901
	Brazo	0902

Como se puede observar en la tabla, la asignación al autómata de diferentes entradas, relés internos de memoria y salidas las representamos con unos números concretos. Estos números serán diferentes según el autómata utilizado puesto que cada fabricante asigna con diferente numeración las entradas, salidas y relés internos. En nuestro caso

hemos utilizado el tipo de asignación utilizada en el autómata OMRON CQMIH.

Realizar el programa usuario en el lenguaje de programación seleccionado.



Diseño de la parte secuencial

El modo de realizar el diseño de la parte secuencial consiste asignar a cada etapa un biestable de tipo R-S, cuyas condiciones de “ set ” y “ reset ” a partir de las condiciones de transición indicadas en el gráfico.

Condiciones de “ set ” del biestable de la etapa X:

La activación del biestable de una etapa X tiene lugar cuando la etapa o etapas previas están activadas y se cumplen las condiciones de transición entre dichas etapas y la etapa X.

Así en nuestro ejemplo, la etapa 1 puede resultar activada a partir de la etapa 0 o de la etapa 6, con las correspondientes condiciones de transición.

$$\text{SETQ1} = \text{Q0} * \text{S1} * \text{S3} * \text{S5} + \text{Q6} * \text{S1}$$

Condiciones de “ reset ” del biestable de la etapa X.

La desactivación del biestable de una etapa tiene lugar cuando la etapa o etapas posteriores quedan activadas.

Así en nuestro ejemplo la desactivación de la etapa 1 debe producirse tan pronto como se active la etapa 2.

$$\text{RESET } Q1 = Q2$$

Aplicando este procedimiento a cada una de las etapas, se obtiene el esquema lógico de la parte secuencial del proceso que estamos realizando.



Diseño de la parte combinacional:

En esta fase se diseñan las acciones a desarrollar en cada etapa del proceso y se obtiene el esquema lógico utilizando las salidas de los biestables.

En el ejemplo que nos ocupa tendremos que las ecuaciones lógicas para cada una de las salidas a controlar son:

$$\text{Electroválvula } A^+ : A^+ = Q1$$

$$\text{Electroválvula } A^- : A^- = Q0 + Q6$$

$$\text{Electroválvula } B^+ : B^+ = Q2$$

$$\text{Electroválvula } B^- : B^- = Q0 + Q4$$

$$\text{Electroválvula } C^+ : C^+ = Q3$$

$$\text{Electroválvula } C^- : C^- = Q0 + Q5$$

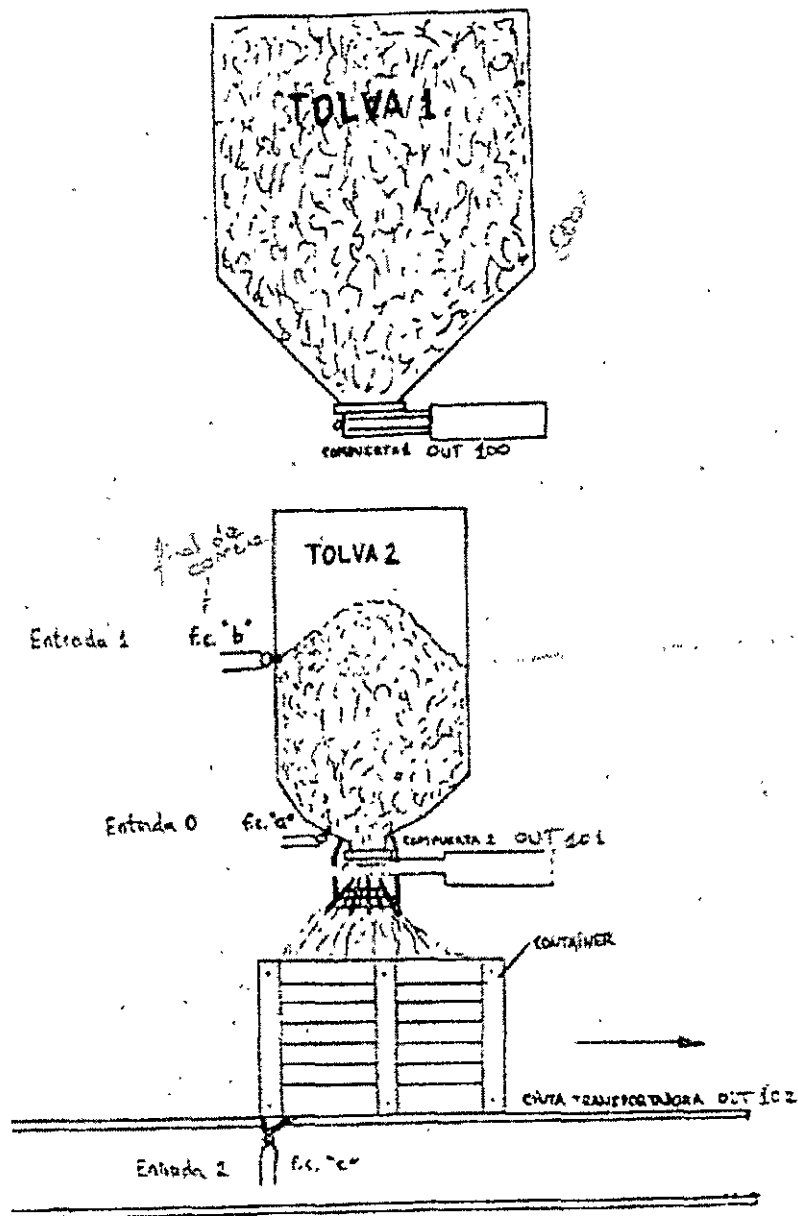
Ejercicio

Funcionamiento

En primer lugar tenemos el deposito 1 (tolva 1), el cual contiene piedrilla en gran cantidad. Esta tolva, mediante el accionamiento de la compuerta 1 (cilindro neumático), permite llenar otra tolva que está justo debajo de ella, es la llamada tolva 2.

La tolva 2 se llenara justo con la capacidad del container a llenar. El fin de carrera "b" (entrada 1), detecta la tolva 2 llena, y el "a" (entrada 0) vacía. Lo mismo que la tolva 1, esta también posee su correspondiente compuerta.

El container, es manipulado por una cinta transportadora accionada por un motor trifásico, el cual para debajo de la tolva 2 (para que sea llenado) cuando lo detecta el fin de carrera "c" (entrada 2), y cuando es llenado sigue avanzando hasta que el próximo container accione el "c", y así sucesivamente



Funcionamiento y comentarios del programa

Al accionar el pulsador de marcha "M" (entrada 10) activo un relé interno biestable (1100), y lo desactivo con el pulsador de parada "P" (entrada 4).

Al dar la marcha, la tolva 1 abre la compuerta 1 (OUT 100=1) y llena la otra tolva, la tolva 2, a no

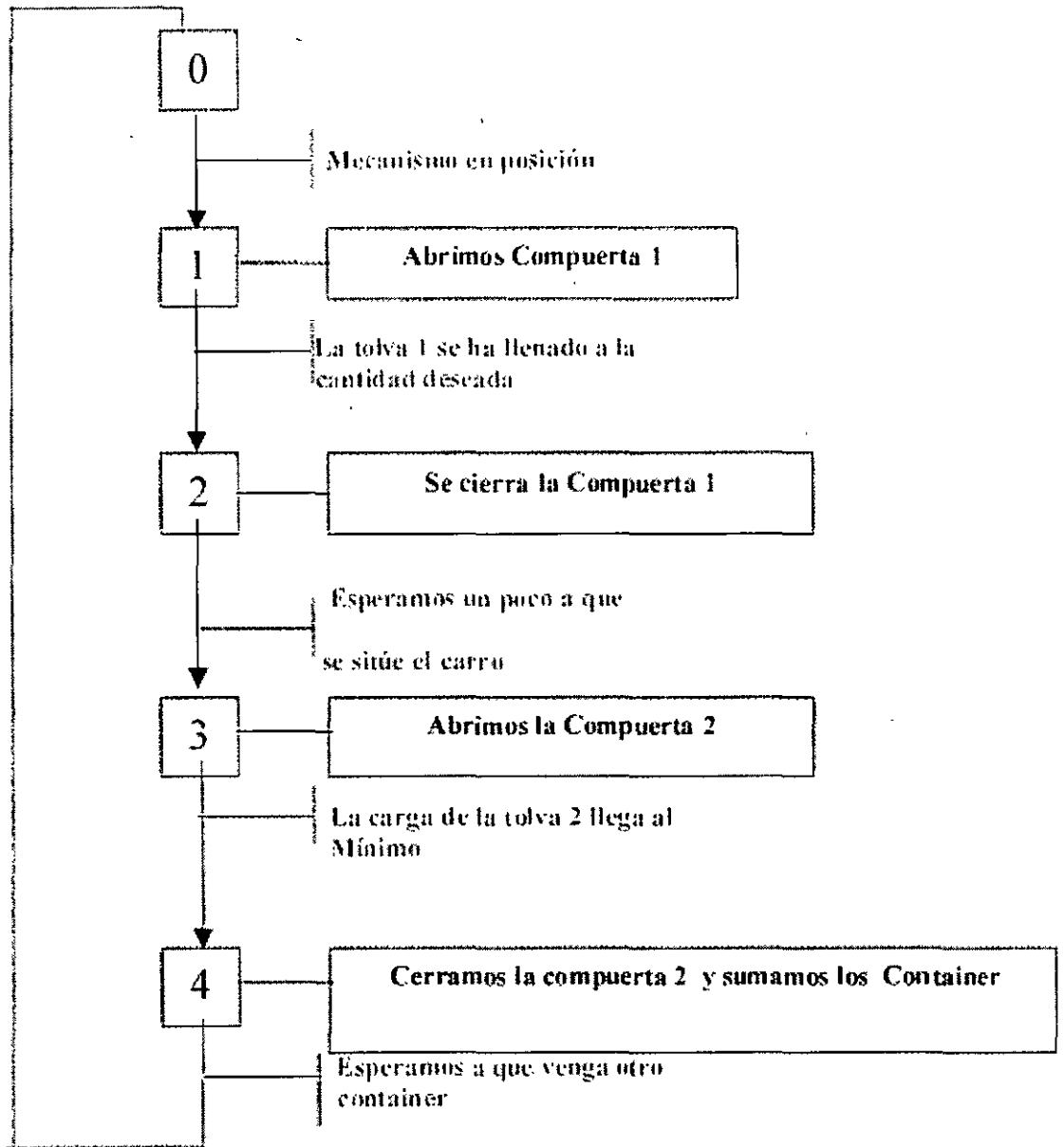
ser que: ya se encuentre llena (entrada 1), que el contador haya contabilizado los 50 containeres cargados, y que la tolva 2 este llenando el container.

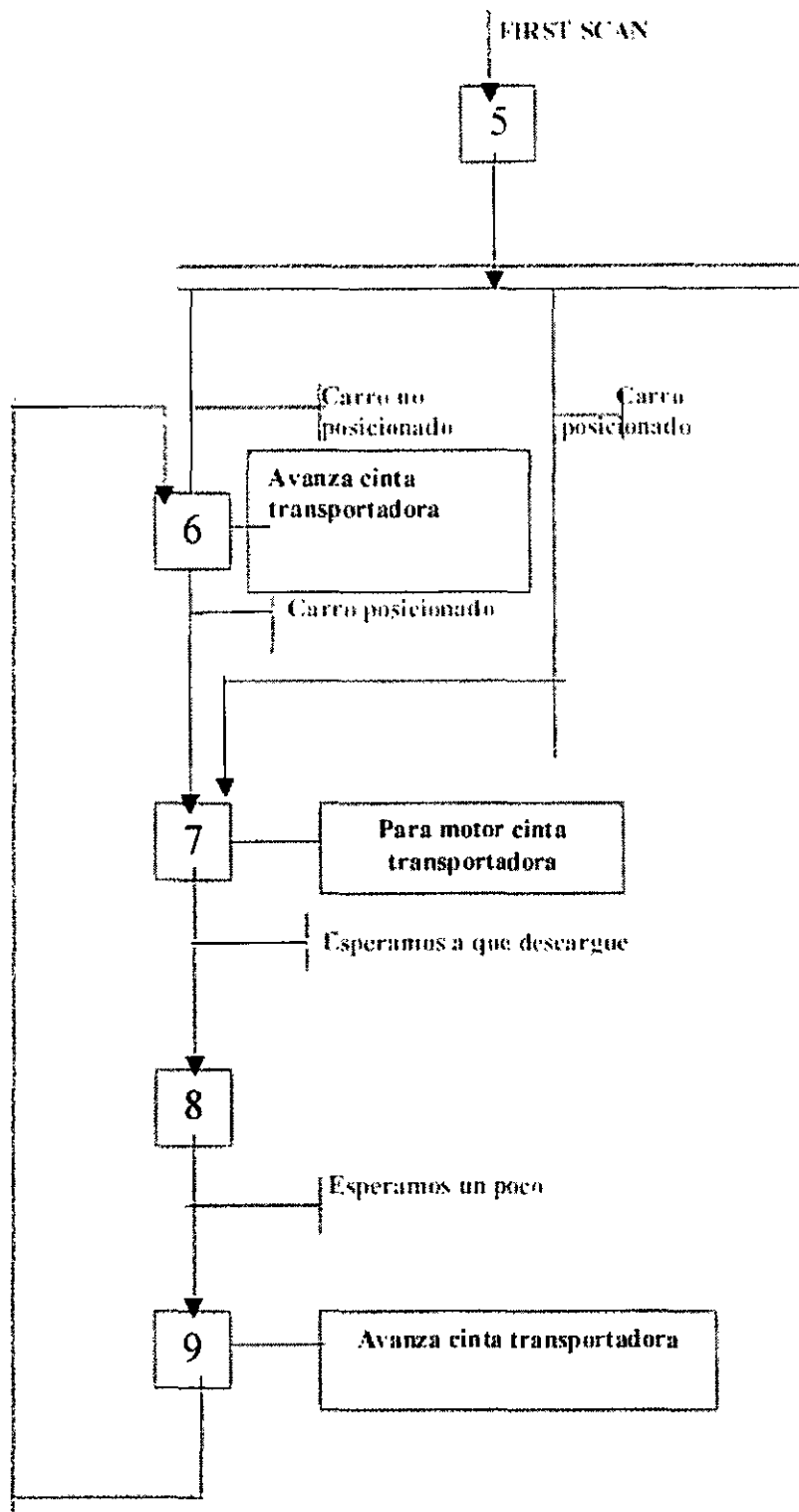
Cuando la tolva 2 adquiere su carga, la tolva 1 cierra su compuerta ($OUT\ 100=0$) y al cabo de 0,3 sg, si el container se encuentra debajo de esta tolva (entrada 2), entonces se abre la compuerta 2 ($OUT\ 101=1$) para llenar el container: esta compuerta se cierra ($OUT\ 101=0$) cuando la carga de la tolva 2 llega a su mínimo (entrada 0).

Cuando las dos tolvas tengan sus compuertas cerradas, la cinta transportara los containeres ($OUT\ 102=1$), hasta que la próxima vacía se situé debajo de la tolva 2.

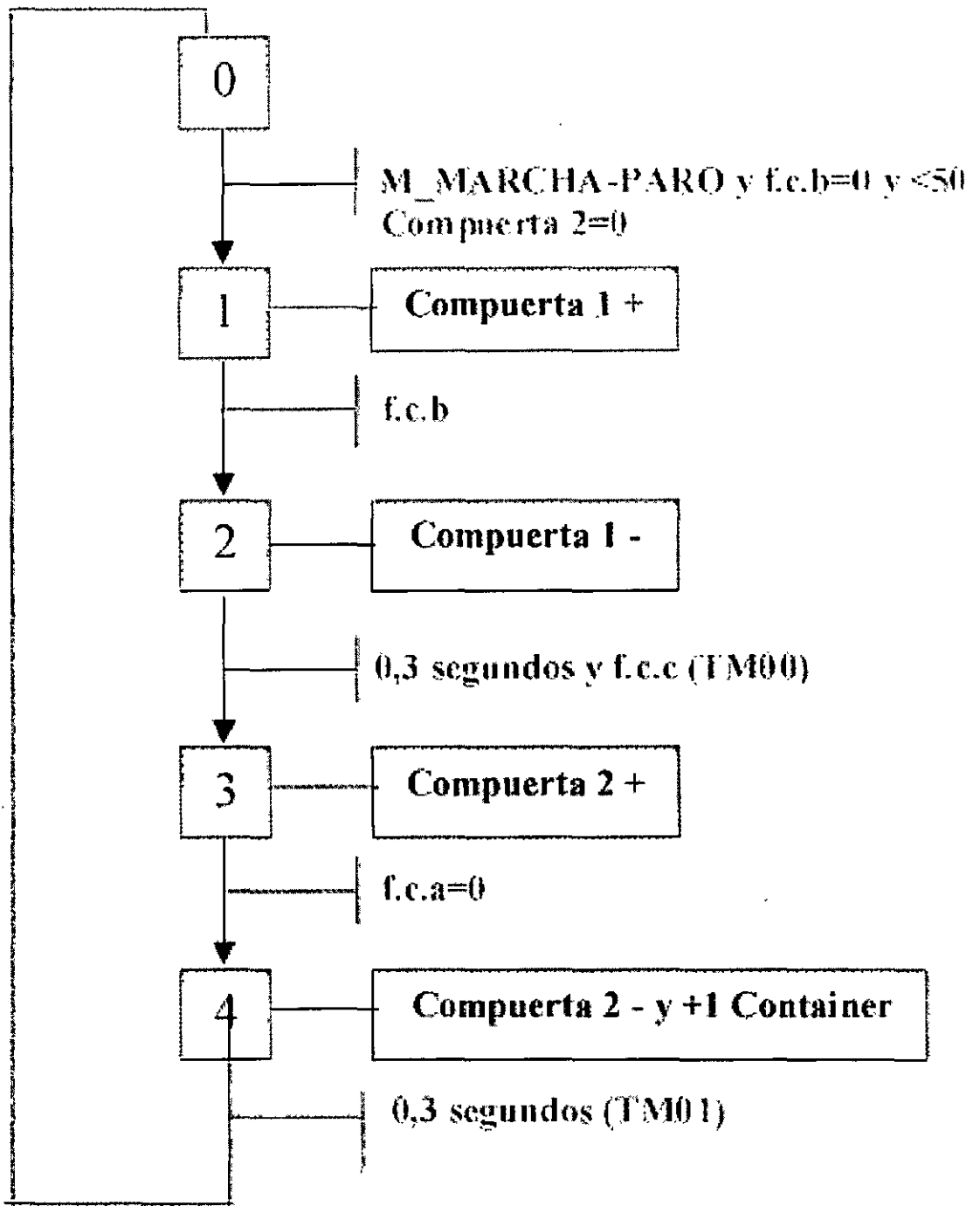
Solución

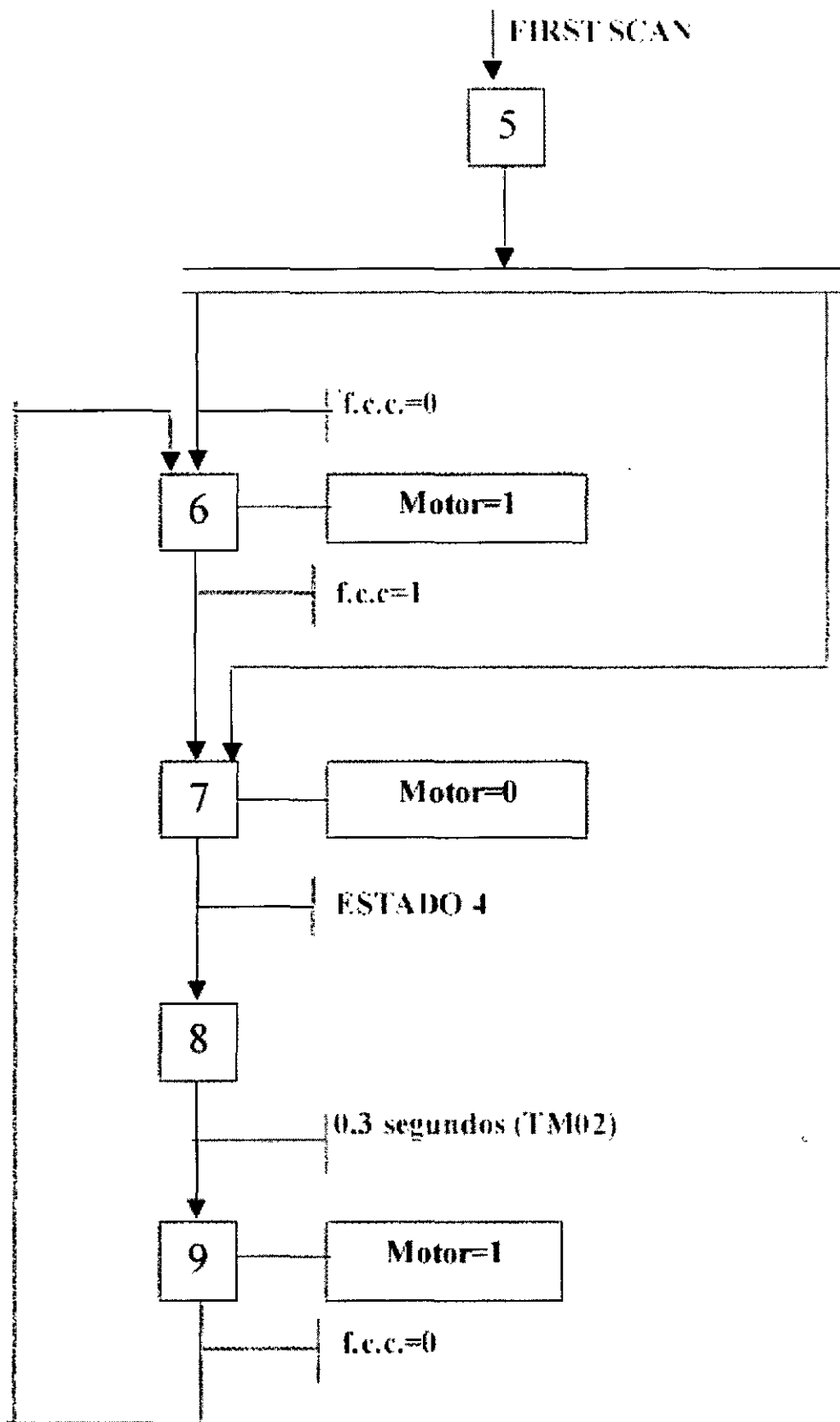
1 GRAFCET funcional





Grafcet 2





3 Asignación de variables

TABLA DE ASIGNACIÓN DE VARIABLES	
VARIABLES DE ENTRADA	ENTRADA AL AUTÓMATA
Final de carrera a "f.c.a"	0000
Final de carrera b "f.c.b"	0001
Final de carrera c "f.c.c"	0002
Pulsador de marcha	0004
Pulsador de parada	0010
ETAPAS DEL GRAFCET	RELES INTERNOS DE MEMORIA DEL AUTÓMATA
E0	1000
E1	1001
E2	1002
E3	1003
E4	1004
E5	1005
E6	1006
E7	1007
E8	1008
E9	1009
M/P (Marcha Paro)	1100
ACCIONES DEL GRAFCET	SALIDAS DEL AUTÓMATA
Compuerta Tolva1	100
Compuerta Tolva2	101
Motor cinta transportadora	102

4 Diseño del sistema de control

Red1

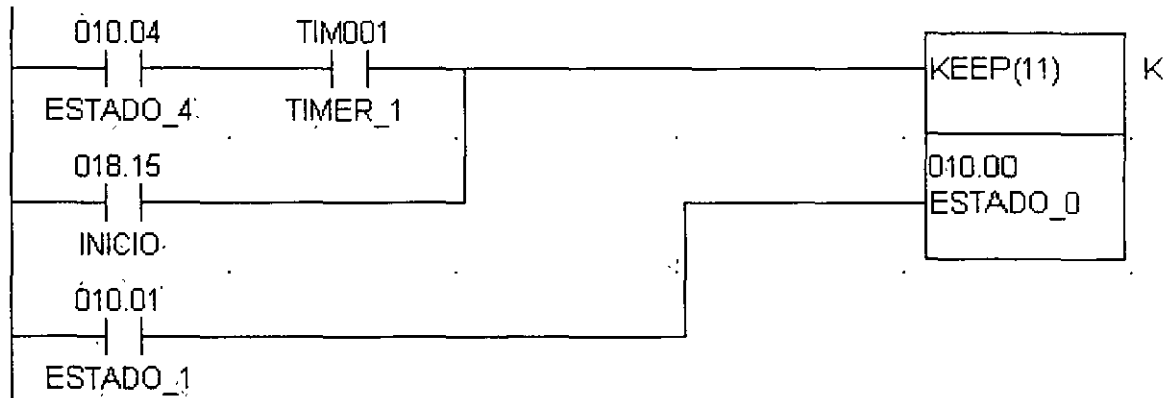
LD 010.04

AND TIM001

OR 018.15

LD 010.01

KEEP 010.00



Red 2

LD 010.00

AND 011.00

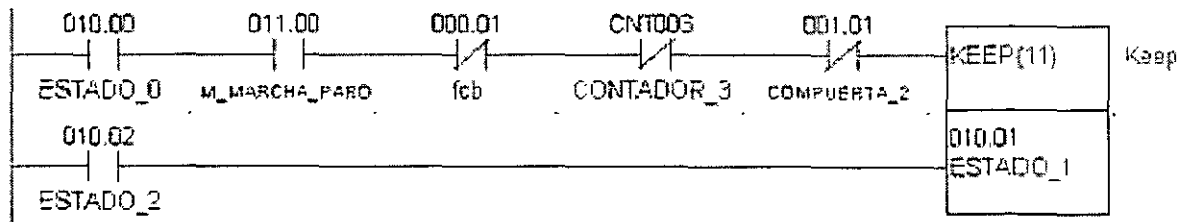
AND NOT 000.01

AND NOT CNT003

AND NOT 001.01

LD 010.02

KEEP 010.01



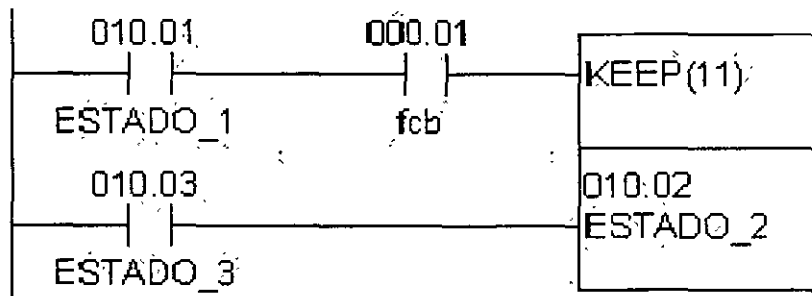
Red 3

LD 010.01

AND 000.01

LD 010.03

KEEP 010.02



Red 4

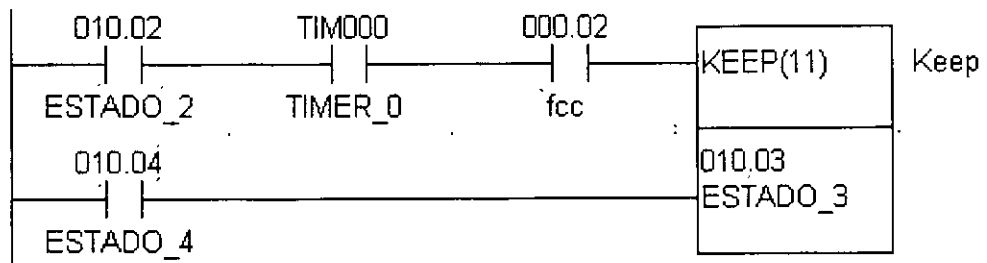
LD 010.02

AND TIM000

AND 000.02

LD 010.04

KEEP 010.03



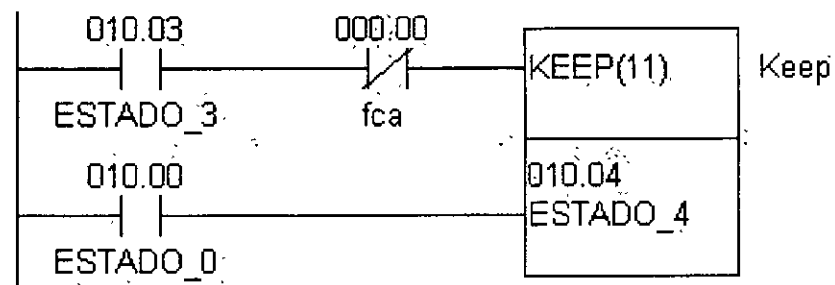
Red 5

LD 010.03

AND NOT 000.00

LD 010.00

KEEP 010.04



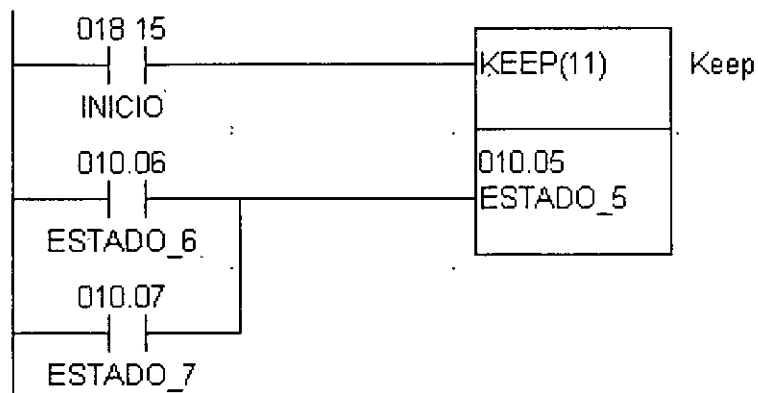
Red 6

LD 018.15

LD 010.06

OR 010.07

KEEP 010.05



Red 7

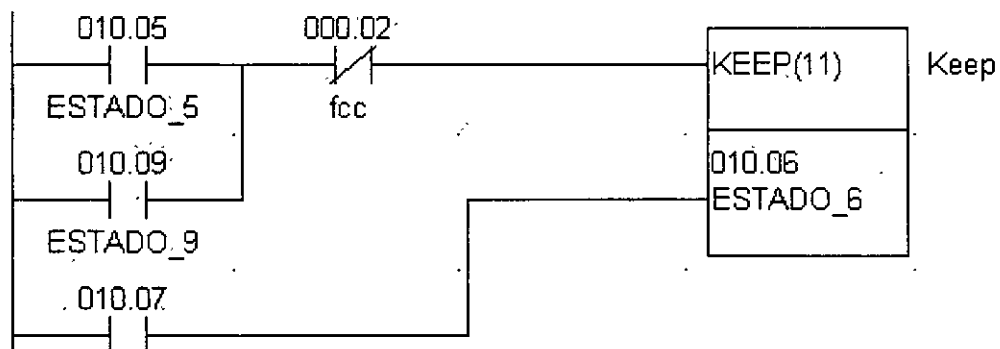
LD 010.05

OR 010.09

AND NOT 000.02

LD 010.07

KEEP 010.06



Red 8

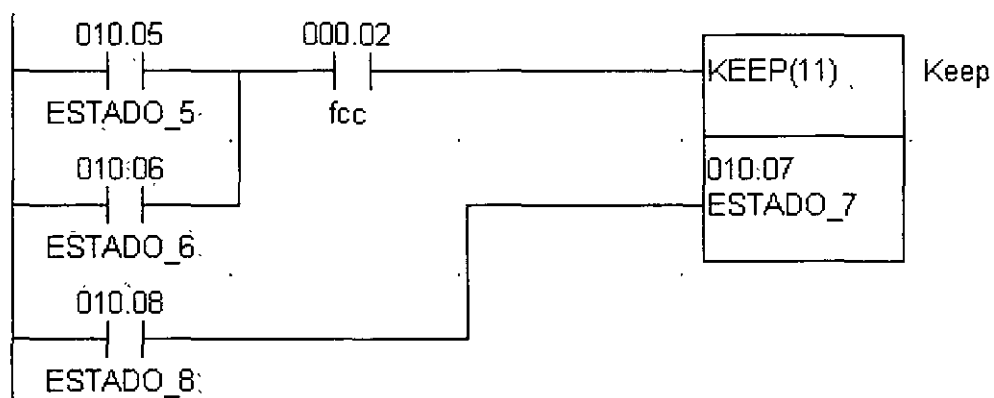
LD 010.05

OR 010.06

AND 000.02

LD 010.08

KEEP 010.07



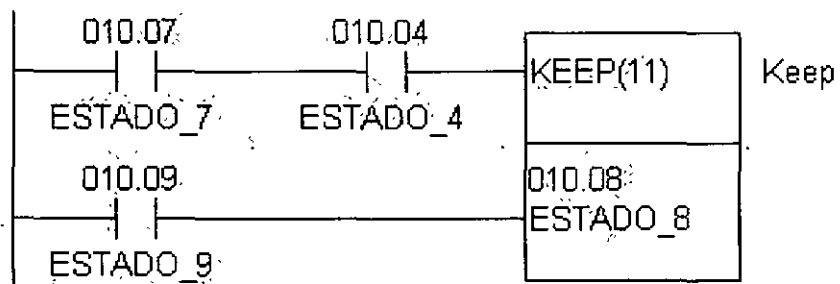
Red 9

LD 010.07

AND 010.04

LD 010.09

KEEP 010.08



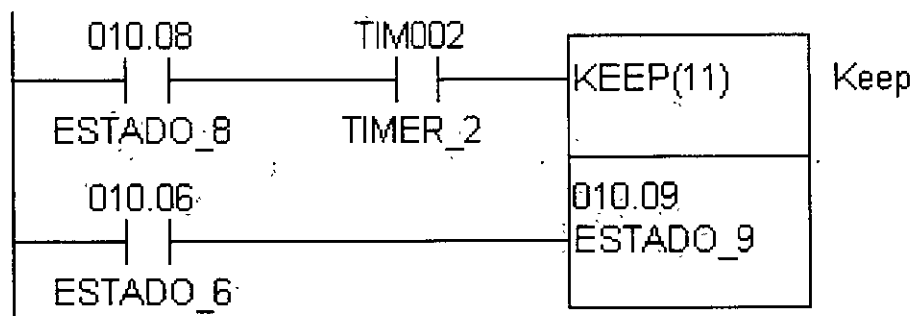
Red 10

LD 010.08

AND TIM002

LD 010.06

KEEP 010.09

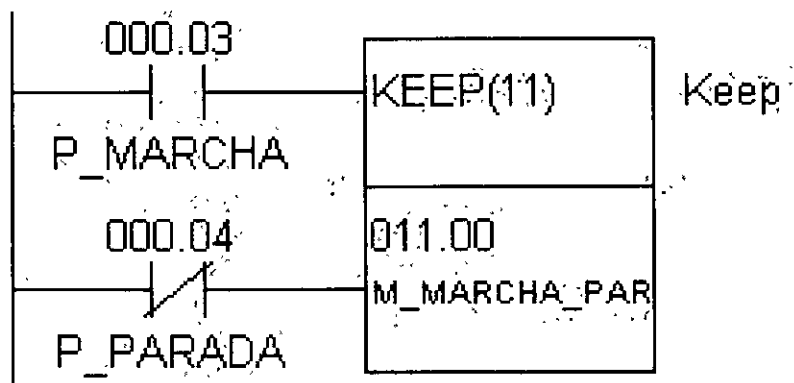


Red 11

LD 000.03

LD NOT 000.04

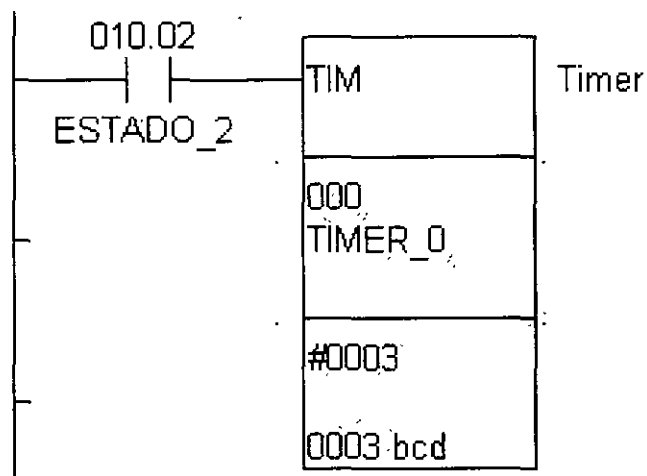
KEEP 011.00



Red 12

LD 010.02

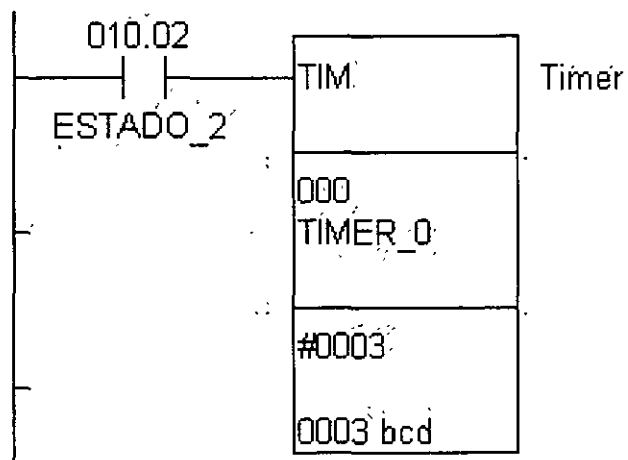
TIM 000 #0003



Red 13

LD 010.04

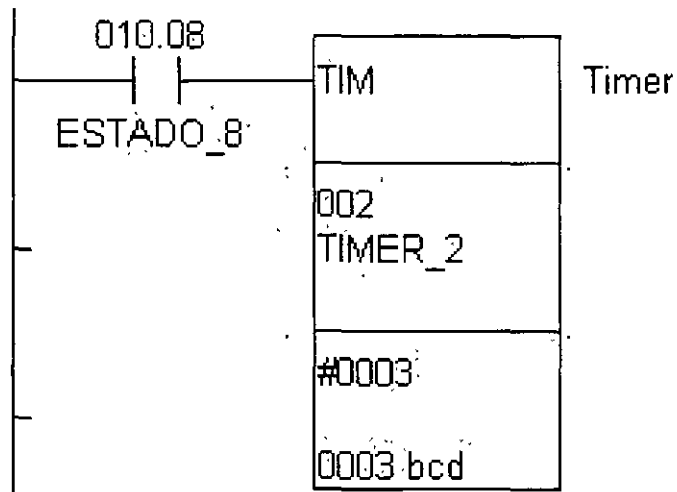
TIM 001 #0003



Red 14

LD 010.08

TIM 002 #0003

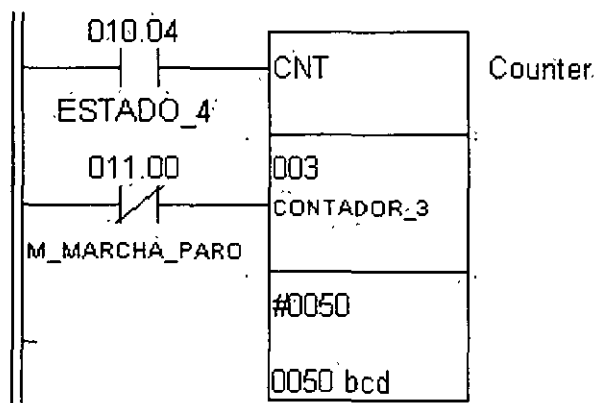


Red 15

LD 010.04

LD NOT 011.00

CNT 003 #0050

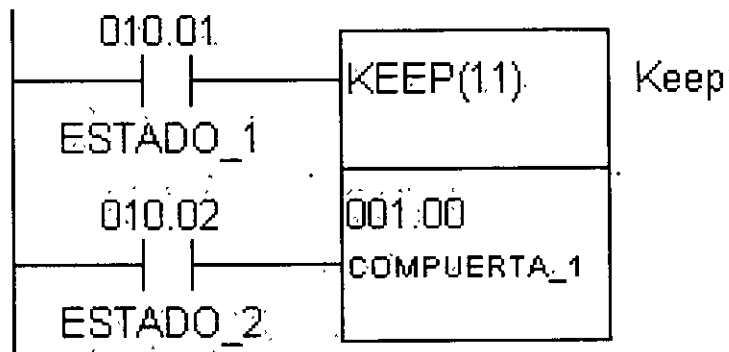


Red 16

LD 010.01

LD 010.02

KEEP 001.00

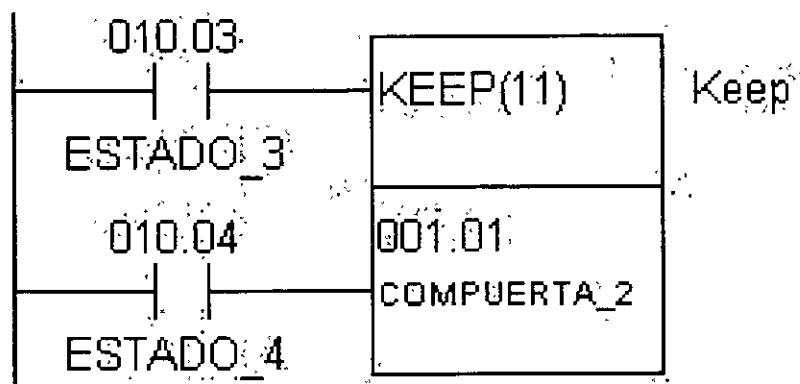


Red 17

LD 010.03

LD 010.04

KEEP 001.01



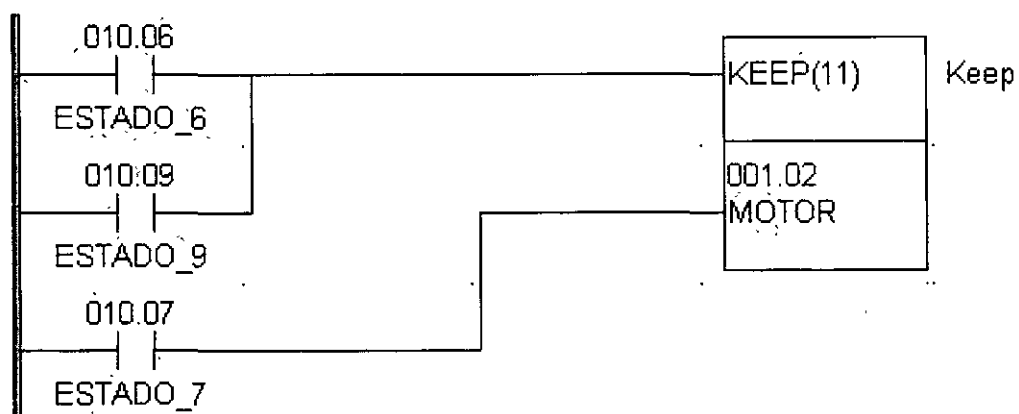
Red 18

LD 010.06

OR 010.09

LD 010.07

KEEP 001.02



Red 19

END

