

11126

**UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO 96**

---

**FACULTAD DE ESTUDIOS  
SUPERIORES CUAUTITLAN**

**DISEÑO, OPERACIÓN Y PROTECCION DE INSTACIONE  
ELECTRICAS INDUSTRIALES.  
LA PROGRAMACION EN STEP 5 DE UN MINIAUTOMATA  
SIMATIC S5 PARA RESOLVER TAREAS DE  
AUTOMATIZACION SENCILLAS"**

**TRABAJO DE SEMINARIO  
QUE PARA OBTENER EL TÍTULO DE:  
INGENIERO MECANICO ELECTRICISTA**

**P R E S E N T A :**  
**RAUL / ZUÑIGA ANGULO**

**ASESOR:**  
**M.I. BENJAMIN CONTRERAS SANTA CRUZ**

**CUAUTITLAN IZCALLI, EDO. DE MEXICO 2003**



Universidad Nacional  
Autónoma de México



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

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

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

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

2

**FACULTAD DE ESTUDIOS SUPERIORES CUAUTITLAN**  
**UNIDAD DE LA ADMINISTRACION ESCOLAR**  
**DEPARTAMENTO DE EXAMENES PROFESIONALES**



GOBIERNO NACIONAL  
SECRETARIA DE EDUCACION  
UNIDAD DE LA ADMINISTRACION ESCOLAR  
CUAUTITLAN

**DR. JUAN ANTONIO MONTARAZ CRESPO**  
**DIRECTOR DE LA FES CUAUTITLAN**  
**PRESENTE**

U. N. A. M.  
FACULTAD DE ESTUDIOS  
SUPERIORES-CUAUTITLAN



DEPARTAMENTO DE  
EXAMENES PROFESIONALES

ATN: Q. Ma. del Carmen García Mijares  
Jefe del Departamento de Exámenes  
Profesionales de la FES Cuautitlán

Con base en el art. 51 del Reglamento de Exámenes Profesionales de la FES-Cuautitlán, nos permitimos comunicar a usted que revisamos el Trabajo de Seminario:  
Diseño, operación y protección de instalaciones eléctricas industriales.  
La programación en STEP 5 de un miniautomata SIMATIC S5 para resolver  
tareas de automatización sencillas.

que presenta el pasante: Raúl Zúñiga Angulo  
con número de cuenta: 09324092 - 9 para obtener el título de :  
Ingeniero Mecánico Electricista

Considerando que dicho trabajo reúne los requisitos necesarios para ser discutido en el EXÁMEN PROFESIONAL correspondiente, otorgamos nuestro VISTO BUENO.

**ATENTAMENTE**  
**"POR MI RAZA HABLARA EL ESPIRITU"**

Cuautitlán Izcalli, Méx. a 9 de Septiembre de 2003

MODULO

PROFESOR

FIRMA

MODULO	PROFESOR	FIRMA
<u>I</u>	<u>M.I. Benjamin Contreras Santacruz</u>	<u>[Firma]</u>
<u>II</u>	<u>Ing. Ricardo Ramírez Verdeja</u>	<u>[Firma]</u>
<u>IV</u>	<u>José Gustavo Orozco Hernández</u>	<u>[Firma]</u>

TESIS CON  
FALLA DE ORIGEN

## Agradecimientos

Gracias a mi Dios y a la Virgen, de Guadalupe por haberme mostrado el camino correcto en los momentos más difíciles de mi existencia.

A ti mamacita, por haberme apoyado en todos los momentos difíciles de mi vida, por todo tu amor, por haber sido siempre un motivo de lucha y esfuerzo, y por haber confiado en mí, te amo mamacita.

A ti Chiquita, por ser así, por compartir conmigo mi proyecto de vida, por que nos amamos muchísimo, por ser una mujer excepcional, por todo lo que he realizado desde que te conocí, además de todo lo que representas en mí, gracias.

Padre, te agradezco el haber forjado gran parte de mi carácter, darme el coraje y la determinación para no desistir y que en este momento entiendo, gracias padre.

A ti Tere, por tu ejemplo de ser humano, por tu apoyo, por ser siempre así conmigo y por que te quiero mucho, gracias hermana.

A mis hermanos: José, Francisco, Ángeles, y Patricia.

Gracias a todas las personas que estuvieron incondicionalmente durante este proceso de formación, y que me dieron siempre un aliento para salir adelante y por las que estaré siempre agradecido.

Gracias a cada uno de mis maestros, a los que están y a los que ya no están, por haberme formado y dado las herramientas para poder así realizarme como profesionalista, muchísimas gracias.

Gracias a la Universidad Nacional Autónoma de México y en particular a la FES Cuautitlán por haberme brindado la formación académica, por lo que siempre estaré comprometido a ponerla en lo más alto, a donde quiera que vaya, GRACIAS.

Raúl

TESIS CON  
FALLA DE ORIGEN

## INDICE.

<b>Introducción</b> .....	1
<b>Capítulo 1. Conceptos básicos de los PLC's</b> .....	3
1.1 Diferencia entre control por cableado y control por programa .....	3
1.1.1 Lógica cableada .....	4
1.1.2 Lógica programada .....	5
1.2 Estructura de un PLC .....	5
1.2.1 Unidad de control .....	6
1.2.2 Memoria .....	6
1.2.3 Memorias internas .....	6
1.2.4 Memoria de programa .....	7
1.2.5 Tarjetas de entrada y salida .....	7
1.2.6 Bus de datos .....	9
1.2.7 Fuente de alimentación .....	9
1.3 La separación galvánica en los módulos de entrada - salida .....	9
<b>Capítulo 2. Automatas Programables SIMATIC S5</b> .....	10
2.1 Miniautomatas S5-90U y S5-95U .....	10
2.2 Automata S5-100U .....	12
2.3 Automata S5-115U .....	14
2.4 Automatas SIMATIC S5 135U y 155U .....	18
<b>Capítulo 3. Lenguaje de programación STEP 5</b> .....	22
3.1 Formas de representación .....	22
3.2 Programa .....	23
3.2.1 Programas del sistema .....	23
3.2.2 Programas de aplicación .....	23
3.3 Bloque .....	24
3.3.1 Bloques de procesamiento .....	24
3.3.1.1 Bloque de programa .....	24
3.3.1.2 Bloques de funciones .....	25
3.3.1.3 Bloques de organización .....	25
3.3.1.4 Bloques de secuencia .....	25
3.3.2 Bloques de almacenamiento .....	25
3.3.2.1 Bloques de datos .....	25
3.4 Elaboración del programa de aplicación .....	25
3.4.1 Filosofía de programación en STEP 5 .....	27
3.5 Juego de operaciones .....	28
3.6 Parámetros de programación de bloques .....	28
3.7 Instrucción .....	28
<b>Capítulo 4. Operaciones binarias de STEP 5: Operaciones AND y OR</b> .....	30
4.1 Consulta a las salidas .....	31
4.2 La combinación "Y" delante de "O" .....	32
4.3 La combinación "O" delante de "Y" .....	33
<b>Capítulo 5. Programación en STEP 5: Arranque del Software STEP 5</b> .....	35
5.1 Concepto de manejo: Teclado .....	36
5.1.1 Teclas funcionales Help e Info .....	36
5.1.2 Superficie de manejo .....	36

**TESIS CON  
FALLA DE ORIGEN**

5.1.3 Layout de las máscaras S5 .....	36
5.2 De la selección de paquete a la selección de la función .....	37
5.2.1 Primer paso: Selección de paquete .....	38
5.2.2 Segundo paso: Parametrizar los ajustes previos .....	38
5.2.2.1 Aclaraciones de los conceptos de la máscara de ajustes previos .....	40
5.2.3 Tercer paso: Elección de función .....	43
5.3 Entrada de bloques .....	43
5.4 Salida de bloques .....	45
<b>Capítulo 6. Correcciones en LAD y CSF .....</b>	<b>46</b>
6.1 Corrección a lo largo de un segmento .....	46
6.1.1 Particularidades .....	47
6.2 Correcciones en LAD .....	47
6.3 Correcciones en CSF .....	55
6.4 Insertar, añadir y borrar segmentos .....	55
<b>Capítulo 7. Funciones auxiliares para la gestión de bloques: Modos de operación ....</b>	<b>57</b>
7.1 Selección de las funciones auxiliares .....	58
7.2 Transferencia de bloques .....	58
7.2.1 Pasos de manejo de la función Transferir .....	59
7.3 Borrar bloques .....	60
7.3.1 Pasos de manejo para la función borrar .....	60
7.4 Salida del directorio .....	61
7.4.1 Pasos de manejo de la función Directorio .....	61
7.5 Modificar un archivo de programa ajustado .....	62
<b>Capítulo 8. Operaciones básicas de STEP 5: Circuitos de automantenimiento .....</b>	<b>63</b>
8.1 Funciones de memoria R - S .....	63
8.1.1 RS Memorias .....	64
8.2 Temporizadores (Timer's) .....	65
8.2.1 Tipos de temporizadores .....	67
8.2.1.1 Impulso .....	68
8.2.1.2 Impulso prolongado .....	68
8.2.1.3 Retardo a la conexión .....	68
8.2.1.4 Retardo a la conexión memorizada .....	68
8.2.1.5 Retardo a la desconexión .....	68
8.2.2 Ajuste de tiempo (Carga) .....	69
8.3 Funciones de carga y transferencia .....	69
8.4 Contadores (Counters) .....	70
8.4.1 Conteo hacia delante .....	72
8.4.2 Conteo hacia atrás .....	72
8.4.3 Carga de un contador .....	72
8.4.4 Borrado de un contador .....	72
8.4.5 Consulta digital de un contador .....	73
8.4.6 Consulta binaria del estado de señal de un contador "Q" .....	73
8.5 Funciones de comparación .....	73
8.6 Funciones aritméticas .....	75
8.7 Operaciones de llamada de bloque .....	75
8.8 Operaciones de terminación de bloque .....	75
8.9 Operación Stop "STP" .....	76
<b>Capítulo 9. Bloques funcionales .....</b>	<b>77</b>

9.1 Estructura de un bloque de funciones .....	77
9.1.1 Cabecera de bloque .....	77
9.1.2 Cabeza de bloque .....	77
9.1.3 Cuerpo del bloque .....	78
9.2 Creación de un FB .....	78
9.3 Programación de un bloque de funciones .....	79
9.4 Llamada de un bloque funcional .....	81
9.5 Ejemplo de llamada y parametrización de un FB .....	82
<b>Capítulo 10. Ejemplos de programación en STEP 5 .....</b>	<b>83</b>
<b>Conclusiones .....</b>	<b>95</b>
<b>Abreviaturas .....</b>	<b>96</b>
<b>Bibliografía .....</b>	<b>97</b>

TESIS CON  
FALLA DE ORIGEN

## Introducción

El objetivo del siguiente trabajo es dar a conocer a detalle la programación en STEP 5 de los Controladores Lógicos Programables (PLC's) de la familia SIMATIC S5, fabricados por la compañía Alemana SIEMENS, para llevar a cabo la automatización de algún sistema.

El lenguaje de programación STEP 5 es un lenguaje de programación con el que se formulan las funciones de automatización en el PLC SIMATIC S5.

Los PLC's operados por el lenguaje de programación STEP 5 son compactos y son adaptables a diversos ambientes y el costo va de acuerdo al número de entradas y salidas.

En el **primer capítulo** se hace referencia a las ventajas que ofrecen las PLC's en la automatización de un sistema así como también conocer las características internas como externas de éstos.

Es importante conocer las características internas como externas de los PLC's para saber cuál de éstos se adapta a nuestras necesidades de automatización, en el **segundo capítulo** se dan las características de cada uno de los PLC's de toda la familia SIMATIC S5, desde los más pequeños (en tamaño y en entradas-salidas) hasta los más grandes.

En el **capítulo tres** se explica el uso del lenguaje de programación para PLC's: STEP 5, desde las formas de representación (Diagrama de escalera o contactos (LAD), Bloques de Funciones (CSF) y Lista de instrucciones (STL)), hasta la unidad más pequeña de un programa: la instrucción.

Dado que el principio básico de los PLC's se basa en las operaciones "AND" y "OR", que no son más que conexiones en serie y en paralelo, respectivamente, entre dos conectores, en el **cuarto capítulo** se señalan las diferentes combinaciones entre éstas operaciones.

En el **capítulo 5** se muestra la instalación de STEP 5, los ajustes previos a la programación, así como también el concepto y pasos de manejo, la configuración del teclado, superficie de manejo, hasta mostrar las características de cada máscara o visualización en pantalla.

Por lo regular una vez creado un programa, cabe la posibilidad de que se quiera ampliar o realizar algún tipo de modificación para el óptimo rendimiento del proceso, o simplemente porque no funciona correctamente. En el **capítulo sexto** se definen estas posibilidades como también se muestran algunos ejemplos de las correcciones en diagrama de escalera y de bloques con funciones.

En el **séptimo capítulo**, se señalan las funciones auxiliares como son la transferencia, el borrado y la salida del directorio de los bloques de programación en STEP 5.

En el **capítulo octavo** se muestran en forma detallada las funciones de encclavamiento, memorización, temporización, carga y transferencia, conteo y comparación, aritméticas, salto, terminación de bloque y otras operaciones.

TESIS CON  
FALLA DE ORIGEN

En el **capítulo noveno** se definen las características de los bloques funcionales, que es un bloque de programación de **STEP 5**, con la ventaja de que puede ser llamado una o varias veces por otros bloques, y la única desventaja es, de que se programa en lista de instrucciones.

En **décimo** y último **capítulo** se muestran algunos ejemplos de programación en **STEP 5** de algunos procesos de automatización.

TESIS CON  
FALLA DE ORIGEN

## CAPÍTULO 1

### Conceptos básicos

Según el manual de Siemens un PLC es "un autómatas programable o controlador lógico programable (Programmable Logic Controller PLC), es un dispositivo electrónico diseñado para el control de máquinas y procesos. Utiliza una memoria programable para almacenar instrucciones y ejecutar funciones específicas que incluyen control de activación y desactivación (ON / OFF), temporización, conteo, secuencia, manejo de datos y regulación"<sup>1</sup>.

Las amplias ventajas de todo tipo que ofrece respecto a la lógica cableada lo convierte en el elemento imprescindible en el campo de control a cualquier nivel.

Por su parte, Simón Andre en su libro *Autómatas Programables*<sup>2</sup> menciona que los automatismos son dispositivos que permiten a las máquinas o a las instalaciones funcionar automáticamente.

Asimismo, afirma que un automatismo bien concebido tiene las siguientes características:

- Simplifica considerablemente el trabajo del hombre a quien libera de la necesidad de estar permanentemente situado frente a la máquina, pudiendo dedicarse a otras actividades más nobles
- Aumenta la seguridad del personal
- Permite economizar material y energía
- Tiene flexibilidad
- Incrementa la producción y la productividad
- Facilita los cambios en los procesos de fabricación permitiendo pasar de una cantidad a de un tipo de producción a otro
- Elimina las tareas complejas, peligrosas, pesadas o indeseadas, haciéndolas ejecutar por la máquina

Los PLC proporcionan ahorro en los costos de material, instalación, localización y corrección de problemas y mano de obra, al reducir el cableado y los correspondientes errores de cableado, además, de que ocupan menos espacios que los sistemas de control por relés. Su capacidad para ser programados aumentó notablemente su flexibilidad cuando se modifican los sistemas de control.

#### 1.1 Diferencia entre control por cableado y control por programa.

La clave principal para la aceptación de los PLC en los sistemas de control es la forma de programarlos. La programación básica de los PLC está basada en los diagramas eléctricos de control convencional (lógica cableada).

<sup>1</sup> Manual de Siemens, p. 2

<sup>2</sup> Simón, Anare, *Autómatas Programables, Programación, Automatismo y Lógica programada*, Paraninfo, España, 1988, p. 13

Una tarea de automatización puede realizarse utilizando la técnica de control por cableado o la técnica de control por programa. Con cualquiera de estas dos técnicas es posible implementar las secuencias de control necesarias para resolver la tarea de automatización.

### 1.1.1 Lógica cableada

Para Andre Simón<sup>3</sup>, el automatismo se realiza mediante módulos interrelacionados. El funcionamiento no depende más que del cableado. Los módulos pueden ser electromagnéticos, eléctricos, neumáticos o fluidicos.

Desde el punto de vista del funcionamiento, la lógica cableada es más rápida, pero las interconexiones entre módulos imponen limitaciones de complejidad y volumen a los equipos.

En la técnica de control por cableado o mejor conocida como control convencional, la unión física de diferentes elementos tales como bolones, pulsadores, relés, contactores, etc. Es la que determina la lógica o secuencia según la cual trabaja el control.

A continuación se da un ejemplo del control por cableado:

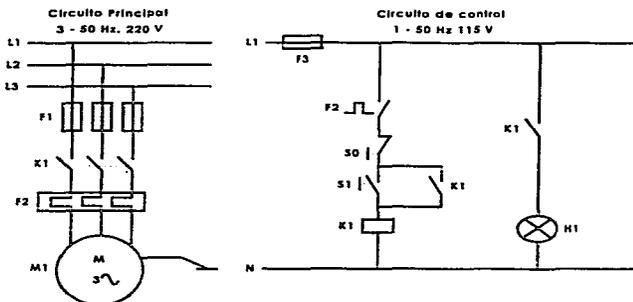


Fig. 1.1 Control por cableado

Donde:  
 S0 es el pulsador de paro  
 S1 es el pulsador de arranque  
 F2 es el relevador bimetálico  
 K1 es el contactor del motor  
 H1 es la lámpara

<sup>3</sup> Ibidem, p. 21

TESIS CON  
 FALLA DE ORIGEN

La figura 1.1, muestra el circuito de fuerza de un motor y su circuito de control por cableado. En el circuito de control se observa como las conexiones serie y paralelo, hechas entre los elementos "Hardware" permiten comandar el motor y señalar su estado de operación. La lógica de control es la siguiente:

1. **Conexión.-** Al pulsar el botón de arranque (**S1**) se acciona el contactor **K1**, el contacto de auto retención **K1** se cierra y el motor queda conectado.
2. **Desconexión.-** Pulsando el botón de paro (**S0**), o al accionarse el bimetálico (**F2**) se abre el contactor **K1** y el motor se desconecta.
3. **Señalización.-** Al accionarse el contacto auxiliar **K1** se conecta la lámpara **H1**, señalizando el estado de operación "conectado" del motor.

Si se deseara modificar la lógica de control, se necesitarían hacer cambios en el cableado, reorganizando los diversos elementos que participan en el circuito de control.

En la técnica de control por cableado o control convencional, modificar una secuencia implica descablear y recablear para obtener lo que se desea.

Cuando en el control por cableado, existen, decenas y centenas de señales, hacer modificaciones al cableado resulta un problema relevante, para estos casos se utiliza la técnica de control por programa, ya que el correspondiente cableado es independiente de la lógica o secuencia de control deseada.

### 1.1.2 Lógica programada

Andre Simón<sup>4</sup> menciona que "un automatismo que utilice la lógica programada se presentará bajo la forma de un ordenador, microordenador o autómatas programable (PLC). Su funcionamiento depende del programa grabado en una memoria".

En el control por programa, los contactos de los emisores del proceso y los elementos finales de control se conectan a los bornes del PLC.

### 1.2 Estructura de un PLC

Acercas de la estructura de un PLC Josep Balcells y José Luis Romeral afirman en el libro *Autómatas programables*<sup>5</sup>, que éste se compone de los siguientes bloques funcionales:

- Unidad central de proceso o de control (CPU)
- Memorias internas
- Memoria de programa
- Tarjetas o puntos de entrada
- Tarjetas o puntos de salida
- Bus de datos
- Fuente de alimentación

TESIS CON  
FALLA DE ORIGEN

<sup>4</sup> Simón, Andre, Op. cit., p. 21

<sup>5</sup> Balcells, Josep, Romeral, José Luis, *Autómatas Programables*, Alfa omega, México, 1998, p. 67

### 1.2.1 Unidad de control

La **unidad de control** consulta el estado de las entradas y recoge de la memoria de programa la secuencia de instrucciones a ejecutar, elaborando a partir de ella las señales de salida u órdenes que se enviarán al proceso. Durante la ejecución del programa, las instrucciones son procesadas en serie, una tras otra. La **unidad de control** es también la responsable de actualizar continuamente los temporizadores y contadores internos que hayan sido programados.

### 1.2.2 Memoria

Para Alber Mayol i Badía<sup>6</sup>, a diferencia de otros equipos programables el PLC "dispone de una memoria perfectamente organizada en áreas de trabajo específicas", como se muestra en el esquema siguiente, que representaría la organización típica de la memoria de un Automata:

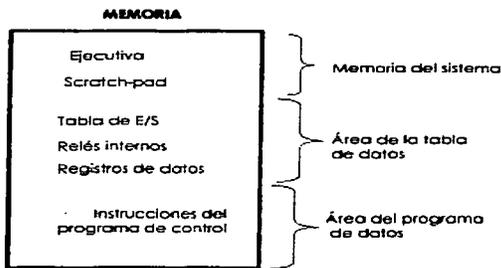


Fig. 1.2 Mapa de memoria del Automata

La memoria está organizada en palabras y registros, constituidos por cierto número de bits: 8, 12, 16 o 32, dependiendo del tipo de procesador que se emplee. Cada una de las palabras o registros constituye una información completa que define una instrucción o un dato numérico, o bien un grupo de estados de E/S, en función del área de memoria a que pertenezca.

#### 1.2.3 Memorias Internas

La memoria del automata contiene todos los datos e instrucciones que necesita para ejecutar la tarea de control.

<sup>6</sup> Mayol i Badía, Alber, *Automatas Programables*, Marcombo, España, 1992, p. 36

Según Josep Ballcells y José Luis Romeral<sup>7</sup>, la *memoria interna* es la que se encarga de almacenar datos intermedios de cálculo y variables internas que no aparecen directamente sobre las salidas, así como un reflejo o imagen de los últimos estados leídos sobre las señales de entrada o enviados a las señales de salida.

#### 1.2.4 Memoria de programa

La *memoria de programa* contiene la secuencia de operaciones que deben realizarse sobre las señales de entrada para obtener las señales de salida, así como los parámetros de configuración del autómatas. Por ello, si hay que introducir alguna variación sobre el sistema de control basta generalmente con modificar el contenido de esta memoria.

#### 1.2.5 Tarjetas de entrada y salida

Las *tarjetas de entrada y salida* construyen la interfase entre los emisores de señal y los actuadores de la máquina o proceso que se va a controlar.

Los interfaces de entrada y salida establecen la comunicación entre la unidad central y el proceso, filtrando, adaptando y codificando de forma comprensible para dicha unidad las señales procedentes de los elementos de entrada, y decodificando y amplificando las señales generadas durante la ejecución del programa antes de enviarlas a los elementos de salida.

Dada la enorme cantidad de variantes que se presentan en las señales de proceso existen también un gran número de tipos de interfaces, tanto de entradas como de salidas.

Por su parte, Simón Andre<sup>8</sup>, dice que las Entradas/Salidas enlazan el autómatas con los órganos exteriores del automatismo o de la instalación,

Los módulos de interface de entrada proporcionan la adaptación de nivel, el aislamiento y filtrado de las señales procedentes de los diferentes captadores situados en la máquina. Se unen a los órganos de control:

- *Manuales*: pulsadores, conmutadores, selectores, combinados, etc.
- *Automáticos*: Finales de carrera de control mecánico, detectores de proximidad inductivos o capacitivos, indicadores de velocidad, de nivel o de presión, etc.

Las señales de entrada, de procedencia y naturaleza diversas: alternas o continuas, de diferentes polaridades, binarias, numéricas o analógicas, etc., deberán ser transformadas antes de su introducción a la unidad de proceso del autómatas. Por otro lado, deberán estar protegidas frente a sobrevoltajes y parásitos que en determinadas situaciones podrían ser considerados como señal y tratados como tal.

Los módulos de interface de salida realizan la adecuación de las señales de salida y su aislamiento de la electrónica del autómatas: Transmiten las órdenes a los organismos gobernados: motores (a través de contactores), electroválvulas eléctricas, hidráulicas o neumáticas, impresores, visores de señalización, etc. En función de la naturaleza de los

<sup>7</sup> Ballcells, Josep, Romeral, José Luis, Op. cit., p. 67

<sup>8</sup> Simón, Andre, Op. cit., p. 72

accionadores, las señales de salida son amplificadas en los módulos estáticos de corriente alterna o continua o gobiernan directamente relés electromagnéticos\*.

Según Ballcells y Romeral<sup>9</sup> estas interfases se clasifican en diferentes formas:

- **Por el tipo de señales:**
  - Digitales de un bit
  - Digitales de varios bits
  - Analógicas
- **Por la tensión de alimentación:**
  - De corriente continua (estáticas de 24/110 Vcc)
  - De corriente continua a colector abierto (PNP o NPN)
  - De corriente alterna (60/110/220 Vca)
  - Salidas por relé (libres de tensión)
- **Por el aislamiento:**
  - Con separación galvánica (opto-acopladores)
  - Con acoplamiento directo
- **Por la forma de comunicación con la unidad central:**
  - Comunicación serie
  - Comunicación paralelo
- **Por la ubicación:**
  - Locales
  - Remotos

En la gama de pequeños autómatas, el tipo de interfases disponibles suele ser más limitado, siendo las más frecuentes, clasificadas por entradas y salidas, las siguientes:

- **Tarjetas de Entradas:**
  - Corriente continua a 24 o 48 Vcc
  - Corriente alterna a 110 o 220 Vca
  - Analógicas de 0-10 Vcc o 4-20 mA

Las tarjetas de entrada reciben las señales del proceso y las adaptan a los niveles de señal internos del equipo. Estas señales ya accionadas viajan a través de un bus hasta el CPU. Este ejecuta el programa de control y en base a él se producen señales de comando o de respuesta.

- **Tarjetas de Salidas:**
  - Por relé
  - Estáticas por triac a 220 Vca máximo
  - Colector abierto para 24 o 48 Vcc
  - Analógicas de 0-10 V o 4-20 mA

Las tarjetas de salidas toman las señales de respuesta elaboradas por el CPU y realizan la conversión en sentido contrario. Es decir a partir del nivel de señal interno las

<sup>9</sup> Ballcells, Josep, Romeral, José Luis, Op. cit., p. 73

tarjetas producen el nivel adecuado para alimentar relés, electroválvulas, lámparas y en general elementos finales de control.

### 1.2.6 Bus de datos

El bus de datos es la vía por la que se intercambian todas las señales entre la Cpu y las tarjetas de entrada y salida.

### 1.2.7 Fuente de alimentación

De acuerdo a Ballcells y Romera<sup>10</sup> la fuente de alimentación proporciona las tensiones necesarias para el funcionamiento de los distintos circuitos del sistema.

Un autómata programable está formado por bloques que requieren niveles de tensión y de potencia diferentes y que, además, están sometidos a condiciones ambientales de ruido electromagnético también distintas.

Por todo ello es frecuente que la alimentación se obtenga de varias fuentes separadas, procurando independizar las siguientes partes del circuito:

- Unidad central e interfaces E/S (alimentación autómata)
- Alimentación de entradas
- Alimentación de salidas (cargas) de tipo electromagnético

La tensión de operación normalmente es mucho menor que la tensión de suministro, del orden de 5 a 24 V. Sin embargo los circuitos de los emisores y los circuitos de los dispositivos de control requieren una tensión de trabajo superior (24 V, 115 V, 220 V, etc.) Esta tensión es suministrada por fuentes de alimentación externas, no por la fuente de alimentación del PLC.

## 1.3 La separación galvánica en los módulos de entrada - salida

La separación galvánica permite la transmisión de señales entre diferentes partes de una instalación sin tensión de referencia común.

Este módulo se realiza por la actuación centralizada de tensiones de referencia independientes. Para la aplicación práctica en autómatas programables esto implica una separación clara entre la parte "noble" de proceso y los circuitos de potencia (entradas/salidas).

Andre Simón<sup>11</sup> señala que "la separación galvánica permite un funcionamiento sin toma de tierra lo que resulta particularmente interesante para los circuitos de entrada/salida pero que va en detrimento de la fiabilidad".

<sup>10</sup> Ibidem, pp. 73-74

<sup>11</sup> Simón, Anare, Op. cit., p. 74

## CAPÍTULO 2

### Autómatas programables SIMATIC S5

SIMATIC es una familia de autómatas programables (PLC) formada básicamente por 6 modelos, en los que cada uno de estos modelos incluye toda una serie de componentes elementales.

Desde el miniautómata compacto hasta el PLC superdotado para las gamas media y alta. Hay autómatas SIMATIC para cualquier exigencia y cualquier tarea en todos los tipos y tamaños.

Todos tienen en común su elevada capacidad de procesamiento, modularidad de diseño y alta resistencia a las condiciones ambientales y mecánicas.

#### 2.1 Miniautómatas

Los miniautómatas S5 - 90U y S5 - 95U son PLC's compactos diseñados para resolver de forma económica tareas de automatización sencillas.

Los miniautómatas S5 - 90U y S5 - 95U se presentan dentro de un robusto gabinete de plástico, este gabinete se puede enganchar sobre un perfil normalizado de 35mm.

Ambos PLC's pueden ampliarse con todos los módulos periféricos del S5 - 100U.

El S5 - 90U es ampliable con la tarjeta de interfase IM90.

El S5 - 95U es ampliable directamente con todos los módulos periféricos del S5 - 100U.

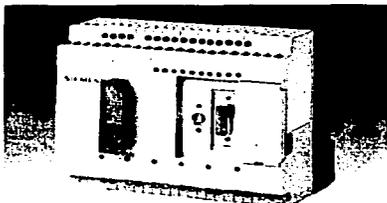


Fig. 2.1 Autómata S5 - 90U

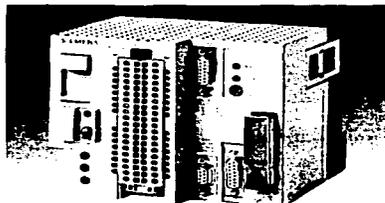


Fig. 2.2 Autómata S5 - 95U

**Datos técnicos de los miniautómatas S5 - 90U Y S5 - 95U**

	S5 - 90U	S5 - 95U
<b>Campo de aplicación</b>	Es utilizable para mandos con una estructura sencilla y que disponen de muy poco espacio para su montaje. Por ejemplo: plantas embotelladoras, mando de puertas, filtros, mando de bombas, trenes de lavado, climatización, sistemas de transporte, medio ambiente, etc.	
<b>Volumen de funciones</b>	Combinaciones binarias, intrucciones con paréntesis, asignaciones de resultado, memorización, conteo, formación de temporizaciones, carga, transferencia, comparación, salto, llamada a bloques, funciones especiales, suma, resta, bloques funcionales integrados para multiplicación / división, procesamiento de valores analógicos y conversión de código.	
<b>Tiempo de ejecución para 1024 instrucciones binarias</b>	2 ms	2ms
<b>Memoria de programa</b>	4 Kbytes RAM, interna o cartucho EPROM / EEPROM	16 Kbytes, 4 Kbytes más para datos o cartucho EPROM/EEPROM
<b>Alimentación de sensores</b>	DC 24 V (100 mA) desde el PLC	-
<b>Tensión de alimentación PLC</b>	AC 115 V/230 V	DC 24 V
<b>Banderas</b>	1024, de ellas 512 remanentes	2048, de ellas 512 remanentes
<b>Temporización, 0.01 a 9990 s</b>	32	128
<b>Contadores, 0 a 999</b>	32, de ellos 8 remanentes	128, de ellos 8 remanentes
<b>Periferia integrada</b>		
Entradas digitales	8: DC 24 V, con sep. galvánica	16: DC 24 V, con sep. galvánica
Entradas de contadores	1: DC 24V, con sep. galvánica <sup>1)</sup>	2: DC 24V, sin sep. galvánica
Entradas de alarmas	1: DC 24V, con sep. galvánica <sup>1)</sup>	4: DC 24V, sin sep. galvánica <sup>1)</sup>
Entradas analógicas	-	8: 0 a 10V, sin sep. galvánica <sup>1)</sup>
Salidas digitales	6: contactos de relés, con sep. Galvánica	16: DC 24V, con sep. galvánica, máx. 24 V, 0.5 A
Salidas analógicas		0: a 10V ó 0 a 20mA sin sep. Galvánica
<b>Periferia externa (ampliación)</b>		
digital	máx. 160 entradas / salidas	máx. 448 entradas / salidas
analógica	máx. 16 entradas / salidas	máx. 32 entradas / salidas
<b>Posibilidades de expansión</b>	máxima 6 módulos S5 - 100U, vía interfase IM 90	máximo 32 módulos S5 - 100U, acoplamiento directo

1) Utilizables también como entradas digitales

Tabla 2.1: Datos técnicos de los miniautómatas S5-90U y S5-95U

**TESIS CON  
FALLA DE ORIGEN**

## 2.2 Autómata S5 - 100U

El autómata S5 - 100U es un miniautómata modular para resolver económicamente tareas de automatización sencillas o de gama media.

Un autómata S5 - 100U se compone fundamentalmente de:

- Unidad central de procesos CPU
- Elementos de bus
- Tarjetas de entradas / salidas
- Fuente de alimentación

La fuente de alimentación, la CPU y los elementos de bus se enganchan directamente sobre un riel de 35mm. Las tarjetas se insertan en los "SLOTS" disponibles en los elementos de bus.

### Datos técnicos de los miniautómatas S5 - 100U

Unidad central	CPU 100	CPU 102	CPU 103
<b>Volúmen de funciones</b>	Combinaciones binarias, instrucciones con paréntesis, asignaciones de resultado, memorización, contaje, formación de temporizaciones, carga, transferencia, comparación, salto, llamada a bloques, funciones especiales, cálculo, combinación de palabras.		
<b>Memoria de programa</b>			
RAM interna	2 Kbytes	4 Kbytes	20 Kbytes
o cartucho de memoria EPROM / EEPROM	2 Kbytes (útiles)	4 Kbytes (útiles)	20 Kbytes (útiles)
<b>Tiempo de procesamiento para 1024 instrucciones binarias</b>	70 ms	7 ms	0,8 ms
<b>Contadores</b>	1024; de ellas 512 remanentes	1024; de ellas 512 remanentes	2048; de ellas 512 remanentes
<b>Temporizadores, 0.001 a 9990 s</b>	16	32	128
<b>Contadores</b>	16; de ellos 8 remanentes	32; de ellos 8 remanentes	128; de ellos 8 remanentes
<b>Entradas / Salidas direccionales</b>			
digitales máximo	384	448	448
analógicas máximo	8	16	32
<b>Posibilidades de expansión</b>	Hasta cuatro filas y con un máximo de 32 módulos periféricos		

Tabla 2.2 Datos técnicos del miniautómata S5-100U

TESIS CON  
FALLA DE ORIGEN

Las siguientes tablas muestran algunas de las tarjetas digitales de entradas y salidas más utilizadas en el sistema modular S5 100U.

**Entrada digital**

Entrada digital 4ESS	420 - 8MA11	421 - 8MA12	422 - 8MA11	430 - 8MB11	430 - 8MC11	430 - 8MD11
Entradas	4	8	16	4	4	4
Separación galvanizada en grupos de	no/-	no/-	no/-	si/4	si/4	si/4
Tensión de entrada	DC 24V	DC 24V	DC 24V	DC 24 / 40V	AC 115V	AC 230V
Intensidad de entrada para "1". tip.	7 mA	7 mA	4.5 mA	4.5 / 7.5 mA	DC 6 mA	DC 2.5 mA
					AC 14 mA	AC 16 mA
Entrada digital 4ESS	431 - 8MA11	431 - 8MC11	431 - 8MD11	431 - 8FA11	433 - 8MA11	
Entradas	8	8	8	8	8	
Separación galvanizada en grupos de	si / 8	si / 8	si / 8	si / 8	si / 8	
Tensión de entrada	DC 24V	UC 115V	UC 230V	DC 24V	DC 5 a 24 V	
Intensidad de entrada para "1". tip.	8.7 mA	DC 2.5 mA	DC 1.8 mA	9 mA	1.1 mA / 5 V	
		AC 12 mA	AC 16 mA		5.1 mA / 24 V	

**Salida digital**

Salida digital 4ESS	440 - 8MA11	440 - 8MA21	441 - 8MA11	450 - 8MB11	450 - 8MD11	451 - 8MA11
Salidas	4	4	8	4	4	8
Separación galvanizada en grupos de	no/-	no/-	no/-	si/4	si/4	si/4
Tensión de alimentación	DC 24 V	DC 24 V	DC 24 V	DC 24 / 40 V	AC 115 / 230 V	DC 24 V
Intensidad de salida para "1"	0.5 A	2 A	0.5 A	0.5 A	1 A	1 A
Protección contra cortocircuitos	Electrónica	Electrónica	No	Electrónica	Fusible	Electrónica
Salida digital 4ESS	450 - 8FA11	451 - 8MD	453 - 8MA11	451 - 8MR12	452 - 8MR11	
Salidas	4	8	8	8	4	
Separación galvanizada en grupos de	si / 4	si / 8	si / 8	si / 2	si / 1	
Tensión de alimentación	DC 24 V	AC 115 / 230 V	DC 5 a 24 V	DC 24 V	DC 24 V	
Intensidad de salida para "1"	2 A	0.5 A	0.1 A	Capacidad de carga de los contactos carga óhmica a DC 30 V / AC 250 V		
				1.5 A / 3 A		
				2.5 A / 5 A		
				Carga inductiva a DC 30 V / AC 250 V		
				0.5 A / 0.5 A		
				0.3 A / 1.5 A		
Protección contra cortocircuitos	Electrónica	Fusible	No	—	—	

Tabla 2.3 Datos técnicos de entradas y salidas del miniaturizado S5-100U

### 2.3 Autómata S5 - 115U

El controlador SIMATIC S5 - 115U es un equipo robusto que permite su utilización en servicios rudos y condiciones ambientales difíciles. Es un controlador adecuado para la gama media.

Un autómata S5 - 115U se compone de un aparato central (ZG) y, según la necesidad, aparatos de ampliación.

El aparato central y de ampliación, son la base de una configuración. El aparato central incluye siempre una fuente de alimentación y la CPU. También soportan a las tarjetas de entradas y salidas.

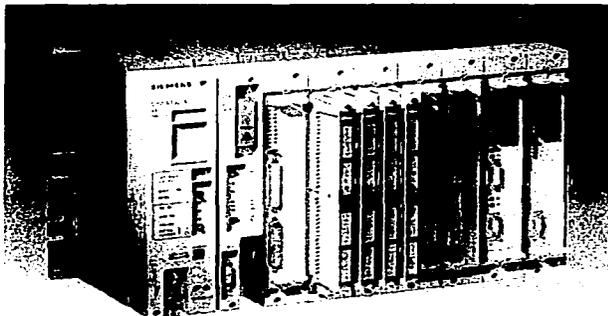


Fig. 2.3 Autómata modular SIMATIC S5 - 115U

### Datos técnicos de los autómatas S5 - 115U

Tipo	Unidades centrales				
	CPU 941	CPU 942	CPU 943	CPU 944	CPU 945
<b>Tamaño de memoria</b>					
- Total máximo de ellos:	18 Kbytes	42 Kbytes	48 Kbytes	96 Kbytes	256 / 384 Kbytes
- Memoria interna RAM	2 Kbytes	10 Kbytes	48 Kbytes	96 Kbytes	256 / 384 Kbytes
- Cartucho de memoria RAM	16 Kbytes	32 Kbytes	-	-	-
- Cartucho de memoria EPROM	16 Kbytes	32 Kbytes	64 Kbytes	128 Kbytes	-
- Tarjeta de memoria Flash-EPROM	-	-	-	-	256 / 512 Kbytes
<b>Banderas en total</b>	2048	2048	2048	2048	2048
de ellas remanentes a voluntad	01/1024/2048	01/1024/2048	01/1024/2048	01/1024/2048	01/1024/2048
<b>Banderas 3 en total</b>	-	-	-	-	32768
de ellas remanentes a voluntad	-	-	-	-	01/16384/32768
<b>Temporizaciones (0.001...9990 s) en total</b>	128	128	128	128	256
de ellas remanentes a voluntad	0/64/128	0/64/128	0/64/128	0/64/128	0/64/256
<b>Contadores (1...999, adelante/atrás) en total</b>	128	128	128	128	256
de ellas remanentes a voluntad	0/64/128	0/64/128	0/64/128	0/64/128	0/64/256
<b>Entradas/ salidas direccionables</b>					
Digitales	4096/4096	4096/4096	4096/4096	4096/4096	4096/4096
Análogas	256/256	256/256	256/256	256/256	256/256
<b>Tempo de procesamiento</b>					
Para 1024 instrucciones binarias	1.6 ms	1.6 ms	0.8 ms	0.8 ms	0.1 ms
Para programación de usuario típico (1024 instrucciones) para ciclo básico (además)	10 ms	10 ms	5 ms	1.5 ms	0.3 ms

Tabla 2.4 Datos técnicos de los autómatas S5-115U

TESIS CON  
 FALLA DE ORIGEN

Las siguientes tablas muestran algunas de las tarjetas digitales más comúnmente utilizadas, en el sistema modular **SS 115U**.

### Entrada digital

Entrada digital 6ESS	420 - 7LA11	430 - 7LA12	431 - 7LA11	432 - 7LA11	434 - 4UA12
Cantidad de Entradas	32	32	16	16	32
Separación galvanizada	no	sí	sí	sí	sí
en grupos de	-	8	4	4	32
Tensión de entrada	DC 24V	DC 24V	DC 24 / 48V	AC 24 / 60V	+ 5 (TTL) + 15 V (CMOS). Señales de sensores NAMUR
Intensidad de entrada para "1" tip.	8.5 mA	8.5 mA	4.5 / 10.5 mA	9 / 10 mA	0.1 mA (TTL). 0.3 mA (CMOS) = 2.1 mA (sensores NAMUR)
Conector frontal	46 polos	46 polos	24 polos	24 polos	42 polos

Entrada digital 6ESS	434 - 7LA12	435 - 7LA11	435 - 7LB11	435 - 7LC11	436 - 7LA11	436 - 7LB11	436 - 7LC11
Cantidad de Entradas	8	16	16	8	16	16	8
Separación galvanizada	sí						
en grupos de	1	4	2	1	4	2	1
Tensión de entrada	DC 24V	AC 115V	AC 115V	AC 115V	AC 230V	AC 230V	AC 230V
Intensidad de entrada para "1" tip.	8.5 mA	DC 6 mA	DC 6 mA	DC 6 mA	DC 2.2 mA	DC 2.2 mA	DC 2.2 mA
Conector frontal	46 polos	24 polos					

Tabla 2.5 Datos técnicos de las tarjetas de entrada del automático SS-115U

TESIS CON  
FALLA DE ORIGEN

## Salida digital

Salida digital 6ES5	441 - 7LA11	451 - 7LA11 451 - 7LA21	453 - 7LA11	454 - 7LA11	454 - 7LB11	455 - 7LA11
Cantidad de Salidas	32	32	16	16	8	16
Separación galvanizada en grupos de	no	sí	sí	sí	sí	sí
	—	8	8	4	1	2
Tensión de alimentación	DC 24 V	DC 24 V	DC 24/48/60 V	DC 24 V	DC 24 V	AC 40/115 V
Intensidad de salida para "1"	0.5 A	0.5 A	0.5 A	2 A	2 A	2 A por cada grupo
Protección contra cortocircuitos	Electrónica	Electrónica	Electrónica	Electrónica	Fusible	Fusible
Conector frontal	46 polos	46 polos	24 polos	24 polos	24 polos	24 polos

Salida digital 6ES5	456 - 7LA11	456 - 7LB11	457 - 7LA11	458 - 7LA11	458 - 7LB11	458 - 7LC11
Salidas			Interr. M	Contactos en relés		
Cantidad	16	8	32	16	8	16
Separación galvanizada	sí	sí	sí	sí	sí	sí
en grupos de	4	1	8	1	1	4
Tensión de alimentación	AC 115/230 V	AC 115/230 V	DC 5/12/24 V	DC 24 V		
Intensidad de salida para "1"	1 A	2 A	0.1 A	Capacidad máxima de carga de los contactos con carga óhmica		
				10 W/0.5A/ UC 30V	5A/AC 250	5A/AC 250
				Con carga inductiva:		
				No Admisible	1.5 A/ AC 250 V 0.5 A/ DC 30V	1.5 A/ AC 250 V 0.5 A/ DC 30V
Protección contra cortocircuitos	Fusible	Fusible	sin	sin	sin	sin
Conector frontal	24 polos	24 polos	46 polos	46 polos	24 polos	46 polos

Tabla 2.6 Datos técnica de las tarjetas de salida del automático 55-1164

TESIS CON  
FALLA DE ORIGEN

#### 2.4 Autómatas SIMATIC S5 135U y 155U

Estos autómatas son aparatos multiprocesadores en operación compacta, con fuente de alimentación y ventiladores.

Los procesadores centrales se descargan gracias a las tarjetas procesadoras de señal. La comunicación es sencilla con otros autómatas y ordenadores por medio de interfaces integradas, procesadores de comunicación y redes locales propias.

Se cuenta con la técnica de control centralizado o descentralizado (máximo 23m)

El autómata S5 - 155U tiene el Juego de instrucciones *STEP 5* más extenso y admite la máxima configuración de memoria.

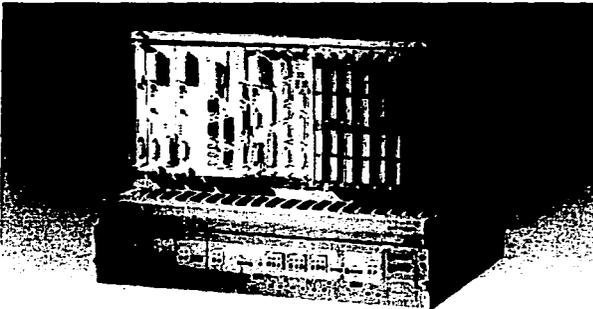


Fig. 2.4 Autómata S5 - 135U

**Datos técnicos de los autómatas S5 - 135U y S5 - 155U**

<b>Tipo</b>	<b>CPU 922</b>	<b>CPU 928</b>	<b>CPU 928 B</b>	<b>CPU 948</b>
Utilizable en S5-	135U, 155U	135U, 155U	135U, 155U	155U
<b>Tamaño de memoria</b>				
Memoria interna RAM	22 Kbytes	46 Kbytes	46 Kbytes	640 / 1664 Kbytes
Cartucho de memoria RAM / EPROM	64 Kbytes	64 Kbytes	64 Kbytes	—
CP 516	8 Mbytes	8 Mbytes	8 Mbytes	8 Mbytes
CP 581	120 Mbytes	120 Mbytes	120 Mbytes	120 Mbytes
<b>Entradas / salidas direccionables</b>				
Digitales con imagen de proceso	1024 / 1024	1024 / 1024	1024 / 1024	1024 / 1024
Además, sin imagen de proceso	3072 / 3072	3072 / 3072	3072 / 3072	3072 / 3072
Además, en acceso directo a memoria		4096 / 4096	4096 / 4096	4096 / 4096
Además, en direccionamiento por páginas	518152 / 518152	518152 / 518152	518152 / 518152	518152 / 518152
Análogicas	192 / 192	192 / 192	192 / 192	192 / 192
Además, en acceso directo a memoria	256 / 256	256 / 256	256 / 256	256 / 256
Además, en direccionamiento por páginas	32130 / 32130	32130 / 32130	32130 / 32130	32130 / 32130
<b>Banderas (remanentes)</b>	2048	2048	2048	2048
<b>Banderas S (remanentes)</b>	Ninguna	Ninguna	8192	32768
<b>Temporizadores (0.01 a 9990 s)</b>	128	256	256	256
<b>Contadores (0 a 999)</b>	128	256	256	256
<b>Tiempo de procesamiento para 1024 instrucciones binarias</b>	20 ms	1.1 ms	0.6 ms	0.18 ms
<b>programa de aplicación típico (1024 instrucciones)</b>	20 ms	7.5 ms	0.9 ms	0.4 ms

Tabla 2.7 Datos técnicos de las CPU de los autómatas S5-135U y S5-155U

TESIS CON  
FALLA DE ORIGEN

Las tablas siguientes muestran las tarjetas digitales de entrada y salida más comúnmente utilizadas en los autómatas S5 - 135U y S5 - 155U.

### Entrada digital

Entrada digital 6ES5	420 - 4UA13	430 - UA13	431 - 4UA12	432 - 4UA12
<b>Entradas</b>				Con señal conjunta
Cantidad	32	32	16	32
Separación galvánica	no	sí	sí	sí
En grupos de	-	32	1	8
<b>Tensión de entrada</b>	<b>DC 24 V</b>	<b>DC 24 V</b>	<b>DC 24 / 48 / 60 V</b>	<b>DC 24 V</b>
<b>Intensidad de entrada para "1", tip.</b>	8.5 mA	7 mA	4.5 mA (a 24 V) 6.5 mA (a 48 V) 7.5 mA (a 60 V)	8.5 mA
<b>Conector frontal</b>	42 polos	72 polos	42 polos	42 polos
<b>Espacio necesario</b>	1	1	1	1
Entrada digital 6ES5	434 - 4UA12	435 - UA12	436 - UA12	436 - 4UB12
<b>Entradas</b>				
Cantidad	32	16	16	8
Separación galvánica	sí	sí	sí	sí
En grupos de	32	8	8	1
<b>Tensión de entrada</b>	Señales TTL (+5 V), señales CMOS (+15 V), señales de sensores NAMUR	<b>AC 24 / 48 / 60 V</b>	<b>AC 115/230 V</b>	<b>AC 115/230 V</b>
<b>Intensidad de entrada para "0"</b>	1 mA (TTL) 3 mA (CMOS) = 1.2 mA (NAMUR)	-	-	-
<b>para "1"</b>	0.1 mA (TTL) 0.3 mA (CMOS) = 2.1 mA (NAMUR)	15 mA (a 48 V) 20 mA (a 60 V)	15 mA (a 115 V) 25 mA (a 230 V)	15 mA (a 115 V) 25 mA (a 230 V)
<b>Conector frontal</b>	42 polos	20 polos	20 polos	20 polos
<b>Espacio necesario</b>	1	2	2	2

Tabla 2.0 Datos técnicos de las tarjetas de entrada de los autómatas S5-135U y S5-155U

TESIS CON  
FALLA DE ORIGEN

## Salida digital

Salida digital 6ES5	441 - 4UA13	451 - 4UA13	453 - 4UA12	455 - 4UA12	454 - 4UA13
<b>Salidas</b>					
Cantidad	32	32	16	16	16
Separación galvanizada en grupos de	no	sí	sí	sí	sí
Tensión de alimentación	DC 24 V	DC 24 V	DC 24 V	DC 24/48/60 V	DC 24 V
Intensidad de salida para "1"	0.5 A	0.5 A	2 A	2 A	2 A
Protección contra cortocircuitos	Electrónica	Electrónica	Electrónica	Electrónica	Electrónica
Conector frontal	42 polos	42 polos	42 polos	20 polos	25/42 polos
Espacio necesario	1	1	2	2	2

Salida digital 6ES5	456 - 4UA12	456 - 4UB12	457 - 4UA12	458 - 4UA12	458 - 4UC11
<b>Salidas</b>				Cont de relés	Cont de relés
Cantidad	16	8	16	16	16
Separación galvanizada en grupos de	sí	sí	sí	sí	sí
Tensión de alimentación	8	1	1	1	8
Intensidad de salida para "1"	AC 115/230 V	AC 115/230 V	DC 24/48/60 V	DC 24 V	DC 24 V
	2 A	2 A	0.5 A	Capacidad de datos (con módulo contactos) DC 60 V, AC 48 V Carga óhmica 0.5 A Carga inductiva 0.05 A	Carga de los contactos de protección de DC 110 V AC 250 V Carga óhmica 5 A Carga inductiva 1.5 A
Protección contra cortocircuitos	Fusible	Fusible	Electrónica	-	-
Conector frontal	20 polos	20 polos	42 polos	42 polos	42 polos
Espacio necesario	2	2	2	1	1

Tabla 2.9 Datos técnicos de las tarjetas de salida de los autómatas 55-135U y 55-155U

TESIS CON  
FALLA DE ORIGEN

## CAPÍTULO 3

### Lenguaje de programación STEP 5

El *STEP 5* es un lenguaje de programación con el que se formulan las funciones de automatización en el controlador programable *SIMATIC 5S*, a continuación se muestra un esquema:

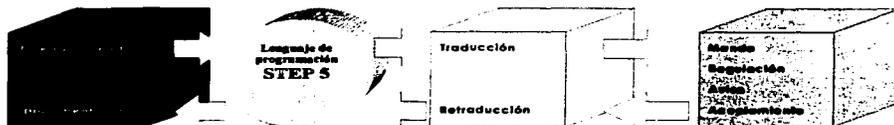


Fig. 3.1 Lenguaje de programación STEP 5

#### 3.1 Formas de representación.

Existen 3 formas de representación del STEP 5. Cada usuario puede elegir la forma de representación que le sea más cómoda. Esto facilita considerablemente el uso y aprendizaje del lenguaje.

1. **Diagrama de contactos o LAD (Ladder Diagram).** Representación gráfica en forma de diagramas de circuito, similar a los esquemas de circuitos usados en control convencional.
2. **Bloques de funciones o CSF (Control System Flowchart).** Representación gráfica mediante bloques de funciones utilizando los símbolos normalizados según DIN 40 700.
3. **Lista de instrucciones o STL (Statement List).** Representación por medio de instrucciones según DIN 19 239. Esta forma se aproxima al lenguaje máquina del controlador.

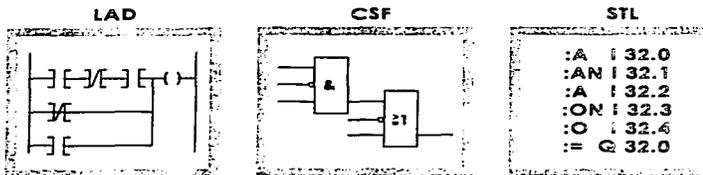


Fig. 3.2 Formas de representación en STEP 5

TESIS CON  
FALLA DE ORIGEN

### 3.2 Programa

Los programas escritos en el lenguaje STEP 5 se clasifican en dos grandes grupos:

1. Programas del sistema
2. Programas de aplicación.

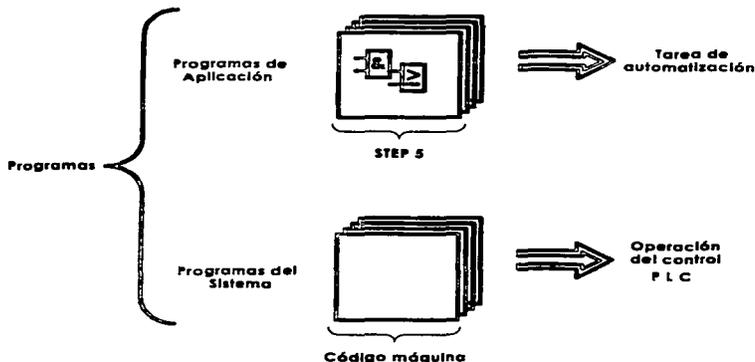


Fig. 3.3 Tipos de Programas en el STEP 5

#### 3.2.1 Programas del sistema

Es el conjunto de todas las instrucciones y declaraciones para realizar las funciones operativas internas del controlador. Por ejemplo: asegurar los datos en caso de caída de alimentación; organizar el procesamiento de los programas de aplicación, etc.

Estos programas están almacenados en memoria EPROM dentro del CPU. No se pierden ni se alteran en caso de falta de alimentación al equipo. El usuario no tiene acceso a ellos.

#### 3.2.2 Programas de aplicación

Es el conjunto de instrucciones y declaraciones para elaborar las señales de control que resuelven una tarea de automatización específica. El usuario escribe estos programas usando el lenguaje de programación STEP 5.

En un programa de aplicación se pueden distinguir una serie de partes llamados bloques. El número y tipo de bloques que componen un programa de aplicación depende de la tarea de automatización y los criterios de programación.

### 3.3 Bloque

Un bloque es una parte del programa de aplicación que se distingue por tener una función, estructura o finalidad específica. En STEP 5 los bloques se clasifican en: bloques de procesamiento y bloques de almacenamiento.

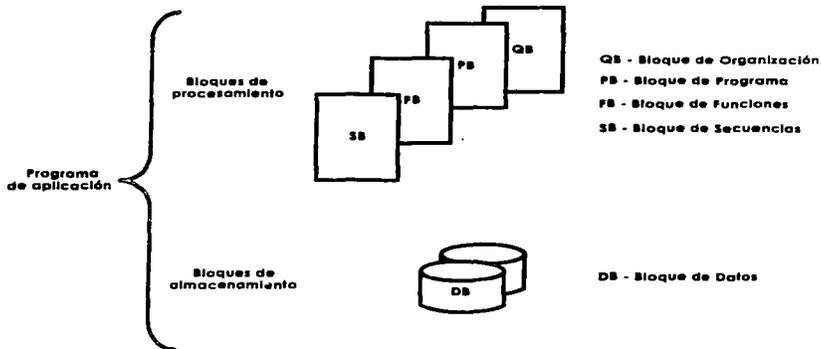


Fig. 3.4 Tipos de Bloques en los Programas de Aplicación

#### 3.3.1 Bloques de procesamiento

Los bloques de procesamiento contienen instrucciones para procesar o elaborar una tarea determinada. Se subdividen a su vez en:

- Bloques de Programa (PB)
- Bloques de Funciones (FB)
- Bloques de Organización (OB)
- Bloques de Secuencia (SB)

##### 3.3.1.1 Bloque de Programa (PB)

Los bloques de programa "PB", sirven para implementar el programa de aplicación que resuelve la tarea de automatización específica, utilizando operaciones básicas del STEP 5. Un programa de aplicación puede estar formado de uno o más BLOQUES DE PROGRAMA. La división del programa de aplicación en BLOQUES DE PROGRAMA se hace siguiendo criterios de división funcional y/o tecnológica de la tarea de automatización.

TESIS CON  
FALLA DE ORIGEN

### 3.3.1.2 Bloques de Funciones (FB)

Los bloques de funciones "FB", sirven para implementar tareas de control en las que se utilicen operaciones avanzadas del STEP 5, o bien, tareas de control que se repitan a menudo o son muy complejas.

Existen FB's estándar y FB's de usuario. Los primeros se suministran completos, listos para su utilización, y los segundos los crea el usuario en su totalidad.

### 3.3.1.3 Bloques de Organización (OB)

Se utilizan para administrar la ejecución del programa de aplicación. Por ejemplo, el bloque de organización No. 1 "OB1" se encarga de la ejecución cíclica del programa de aplicación, en él se listan todos los bloques de programa (FB) o de funciones (FB) que se han de ejecutar. Existen OB's que se procesan en forma cíclica, otros cuyo procesamiento se controla por alarmas, o bien, por tiempo.

Los OB's realizan una función definida dentro del programa de aplicación. Existen dos tipos de OB's, los que programa el usuario y los llama automáticamente el sistema operativo y, otros que se encuentran ya programados y los tiene que llamar el usuario.

### 3.3.1.4 Bloque de Secuencia (SB)

Para implementar funciones o tareas de tipo secuencia. Se trata de bloques de funciones que organizan la ejecución de una secuencia en lenguaje de programación GRAPH 5.

## 3.3.2 Bloques de almacenamiento

Los bloques de almacenamiento NO contienen instrucciones, sirven únicamente para almacenar información. Existe un solo tipo de bloques de almacenamiento el Bloque de Datos (DB).

### 3.3.2.1 Bloque de Datos (DB)

Para almacenar información. A través de este tipo de bloque el usuario archiva datos fijos o variables con los que trabaja el programa de usuario.

## 3.4 Elaboración del programa de aplicación

La elaboración de un programa de aplicación está controlada en el PLC por los programas del sistema. El control se realiza a través de los Bloques de Organización (OB). Un Bloque de Organización como el OB1 es ejecutado cíclicamente por el sistema, independientemente del resto de los programas en el controlador. Los programas del sistema "buscan" al OB1 en la memoria del controlador: si el OB1 existe, éste será ejecutado.

TESIS CON  
FALLA DE ORIGEN

En el **OB1** se programan instrucciones de "llamada" a los demás bloques del programa de aplicación (**PB**, **FB**), etc. Los bloques se elaboran en el orden en que son llamados en el **OB1**.

**IMPORTANTE:** Si un bloque no es llamado en forma primaria por el **OB1** o no pertenece a una línea de llamado que nace en el **OB1**, este bloque **NO** será procesado por el controlador, aún cuando se encuentre en la memoria.

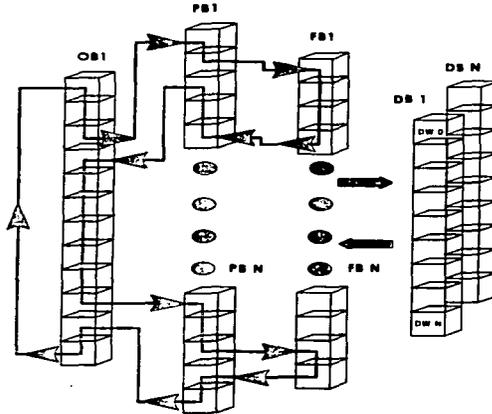


Fig. 3.5 Elaboración del programa de aplicación

Al llegar a una instrucción de "llamada" el procesamiento en el **OB1** se detiene y continúa en el bloque indicado. Una vez elaborado dicho bloque el procesamiento regresa al **OB1** y continúa en el punto donde fue suspendido.

En los módulos distintos al **OB1** también pueden programarse instrucciones de llamada a bloques del mismo o diferente tipo. Con cada llamada el procesamiento se aleja cada vez más de la elaboración del programa **OB1**. La distancia medida por el número de módulos a los que se "salta" desde el **OB1** se le conoce como *Profundidad de Encadenamiento*.

La forma en que los bloques son llamados entre sí da lugar a una organización y estructura definida del programa de aplicación.

### 3.4.1 Filosofía de programación en STEP 5

En STEP 5 los programas de aplicación (los que resuelven una tarea de automatización específica), están compuestos de bloques de uno o varios tipos (OB, PB, SB y DB). Esta división en partes de un programa de aplicación no es circunstancial, responde a una filosofía de programación definida, que ofrece ventajas y beneficios.

En Programación, a la metodología de dividir una tarea global en partes independientes se conoce como Programación Estructurada. En STEP 5, Programación Estructurada significa dividir el programa de aplicación en bloques que se caracterizan por una independencia funcional o tecnológica.

Cada bloque del programa realiza una tarea específica, claramente definida.

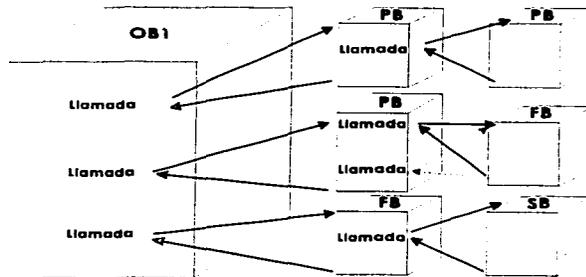


Fig. 3.6 Programación Estructurada en STEP 5

La modificación y corrección de un programa de aplicación es más fácil dentro del concepto de programación estructurada. Al conocerse claramente la tarea (responsabilidad) de cada bloque de programa, pueden ubicarse con facilidad la procedencia de errores, o determinarse cuales bloques deben ser sustituidos para reestructurar de tal o cual manera el programa de aplicación.

En tareas de automatización, extensas o complejas, resulta de gran utilidad dividir en partes la tarea global. Ya que esto facilita las labores de comprensión, manipulación y prueba del programa de aplicación.

Gracias al concepto de programación estructurada, los bloques que componen un programa de aplicación pueden entre otras cosas:

- Ejecutarse en forma independiente
- Utilizarse reiteradamente a lo largo de un programa
- Formar parte (al menos en estructura) de otro programa de aplicación

TESIS CON  
FALLA DE ORIGEN

### 3.5 Juego de operaciones

En los controladores SIMATIC 55U es posible programar una gran variedad de instrucciones.

Entre más potente es un controlador, mayor número de instrucciones pueden ser utilizadas. Al conjunto de instrucciones realizables por un controlador se le conoce como juego de operaciones. En cualquier Juego de operaciones STEP5 se distinguen 3 tipos de operaciones:

1. **Operaciones básicas.** Las operaciones básicas comprenden funciones ejecutables en bloques de organización, de programa, de paso y de funciones. Con excepción de la suma, resta y las operaciones organizativas, pueden programarse en las 3 formas de representación (**LAD, CSF, STL**).
2. **Operaciones Complementarias.** Las operaciones complementarias comprenden funciones complejas tales como, instrucciones de sustitución, funciones de prueba de bit, operaciones de desplazamiento y transformación. Sólo pueden programarse en la forma de representación (**STL**).
3. **Operaciones del Sistema.** Las operaciones del sistema acceden directamente al sistema operativo. Sólo deben utilizarse los programadores expertos. Sólo pueden programarse en la forma de representación (**STL**).

### 3.6 Parámetros de programación de Bloques

Cuando se programa cualquiera de los bloques **OB, PB, SB, FB** y **DB** es necesario tomar en cuenta ciertos parámetros de programación tales como:

- El conjunto de operaciones que pueden ser programadas en cada bloque;
- Las formas de representación en las que puede programarse el bloque;
- La longitud máxima por bloque;
- Los números de bloques válidos, etc.

### 3.7 Instrucción

Es la unidad automática más pequeña de un programa y constituye un orden de trabajo para el procesador. Una instrucción se compone de una parte operacional y un operando.

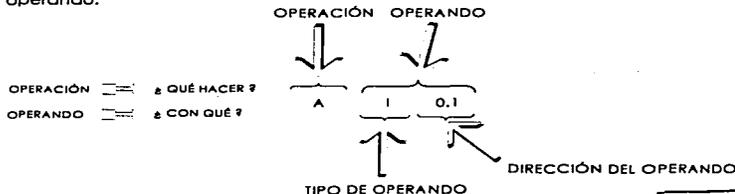


Fig. 3.7 Constitución de una instrucción en STEP 5

La parte operacional describe el trabajo o función a realizar, es decir, indica al procesador, "que hay que hacer". Por ejemplo:

<b>A</b>	Formar una combinación binaria " Y "
<b>OW</b>	Formar una combinación digital " O "
<b>=</b>	Asignar un resultado
<b>C</b>	Llamar a un bloque determinado
<b>JU</b>	Saltar a un bloque determinado
<b>SD</b>	Arrancar un timer "On Delay", etc.

La parte del operando indica "Con qué se ejecuta la operación". Por ejemplo:

<b>I 1.1</b>	Señal bit de entrada en el byte 1, bit 1
<b>FW 3</b>	Bandera palabra con byte de inicio 3
<b>Q 2.3</b>	Señal de bit de salida en el byte 2, bit 3
<b>DB 3</b>	Bloque de datos número 3
<b>C 4</b>	Contador número 4
<b>T 5</b>	Timer número 5
<b>PB 20</b>	Bloque de programa número 20

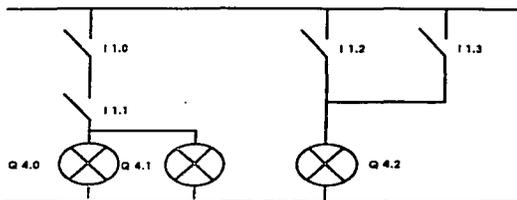
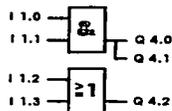
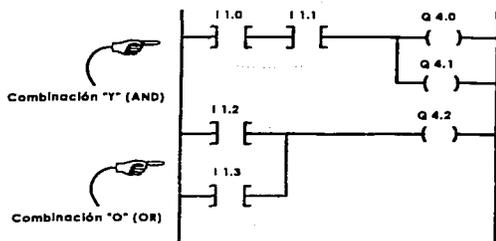
Un operando queda definido por 2 informaciones: la identificación del tipo de operando o tipo de señal y su parámetro o dirección.

TESIS CON  
FALLA DE ORIGEN

## CAPÍTULO 4

### Operaciones binarias de STEP 5.

#### Operaciones AND y OR



Segment 1

```

:A I 1.0
:A I 1.1
:= Q 4.0
:= Q 4.1
***

```

Segment 2

```

:O I 1.2
:O I 1.3
:= Q 4.2
:BE

```

Fig. 4.1 Combinación "Y", "O" y Asignación

La combinación "Y" corresponde a una conexión en serie de contactos de un diagrama de escalera. Las salidas Q 4.0 y Q 4.1 de la función "Y" toman sólo el estado de señal "1" (están conectadas), si en todos los operandos consultados (I 1.0; I 1.1) se verifica el resultado de la consulta "1". Si uno de los operandos tiene asignado el resultado de la consulta "0", las salidas toman el estado de señal "0", es decir, están desconectadas.

La cantidad de operandos consultados NO viene limitada por el Controlador Programable sino por el ancho de la pantalla o del papel cuando se realiza la representación en LAD y CSF; Diagrama de Contactos y Diagrama de Funciones, respectivamente. Por ejemplo, La conexión en serie representada en LAD admite como

TESIS CON  
FALLA DE ORIGEN

máximo 7 contactos y una asignación. En la conexión mixta en CSF admite como máximo 6 símbolos colocados uno detrás de otro.

La combinación "0" corresponde a la conexión en paralelo de contactos. La salida Q 4.2 de la función "0" toma el estado de señal "1" (conectada), si en uno de los operandos consultados (I 1.2; I 2.3) se verifica el resultado de la consulta "1". Cuando todos los operandos consultados tienen asignado el resultado de la consulta "0" originan en la salida el estado de la señal "0" (desconectada).

#### 4.1 Consulta a las salidas

Para el encendido de las lámparas conectadas a las salidas Q 4.0 y Q 4.1 rigen diferentes condiciones, de manera que ambos casos y para cada salida se deben proveer de un segmento propio o de un símbolo de combinación. Como el PLC puede consultar no sólo el estado de la señal de las entradas sino también el de las salidas, se consulta en la combinación "Y" para la salida Q 4.1 el estado de señal de la salida Q 4.0.

Esta consulta Q 4.0 corresponde a la combinación "Y" de las entradas I 1.0 e I 1.1.

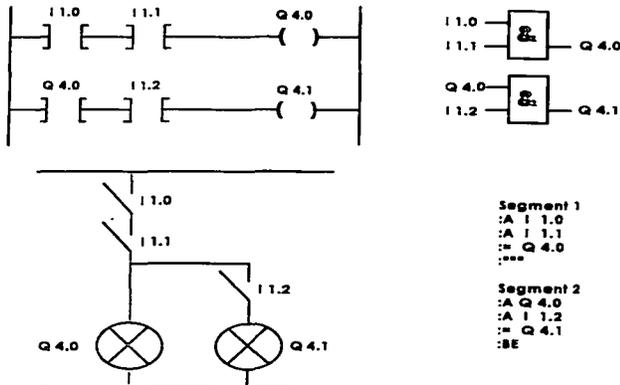


Fig. 4.2 Consulta a las salidas

TESIS CON  
FALLA DE ORIGEN

## 4.2 La combinación "Y" delante de "O"

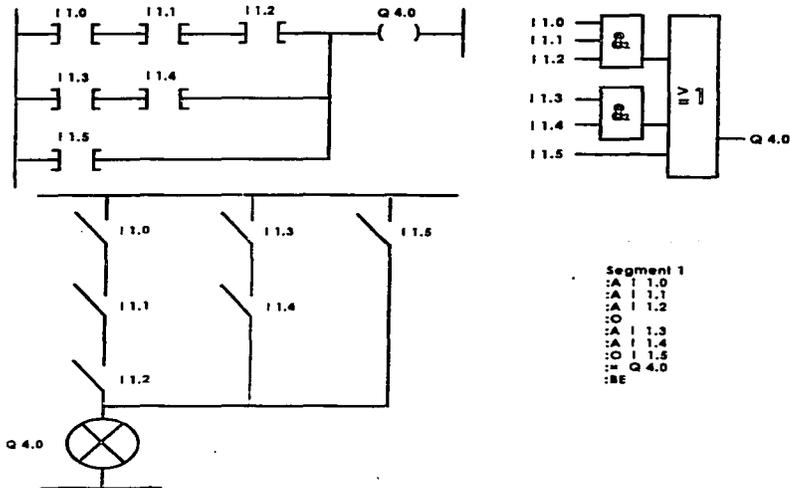


Fig. 4.3 Combinación "Y" delante "O"

La combinación "Y" delante de "O": Consiste en una conexión en paralelo de varios contactos en serie.

Estas derivaciones compuestas de conexiones en serie y en paralelo conducen en la salida Q 4.0 al estado de señal "1" (conectada), si al menos en una de ellas los contactos conectados en serie o el contacto individual I 1.5 están cerrados (estado de señal "1").

Estas funciones "Y" delante de "O" se pueden programar sin utilizar los paréntesis en lista de instrucciones, sin embargo, las derivaciones en paralelo, deben separarse introduciendo el símbolo "O" ("O" sin operando). Para ello se elaboran primero las funciones "Y" y de acuerdo a los resultados obtenidos se genera el resultado de la función "O".

Mientras que la primera función "Y" (I 1.0, I 1.1, I 1.2) está unida con la siguiente función "Y" (I 1.3, I 1.4) a través del símbolo individual "O", el operando I 1.5 puede

TESIS CON  
FALLA DE ORIGEN

programarse al final de un módulo básico delante de la asignación directamente con el símbolo "O I 1.5".

Hay que tener presente que:

El símbolo individual "O" se precisa siempre que sucedan después de una combinación "Y" otras combinaciones "Y", y que estén combinadas según una función "O", o bien, que después de un operando suceda una combinación "Y".

Si al término de una combinación "Y" delante de "O" se suceden sólo operandos combinados según la función "O" no se separan con el símbolo "O".

#### 4.3 La combinación "O" delante de "Y"

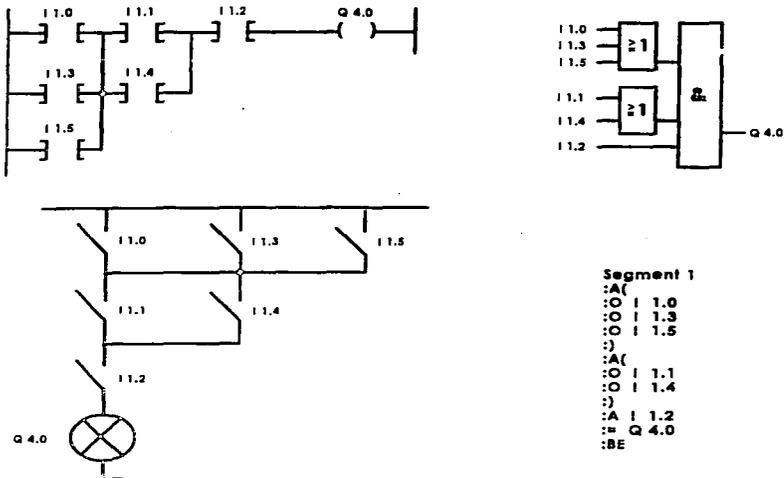


Fig. 4.4 Combinación "O" delante de "Y"

La combinación "O" delante de "Y" consiste de una conexión en serie de varios contactos en paralelo.

Esta combinación compuesta de conexiones en serie y en paralelo tiene en la salida **Q 4.0** el estado de señal "1" (conectado), si en cada una de las dos derivaciones en paralelo están cerrados al menos uno de los contactos y el contacto individual **I 1.2** tiene estado de señal "1".

En estas combinaciones "O" delante de "Y", las funciones "O" que puedan contener también una función "Y" delante de "O" pueden recopilarse por medio de los paréntesis siguiendo las reglas del álgebra de Boole. Con esto se establece que se elaborarán las funciones "O" delante de "Y".

Las operaciones "A (" y ") se programan independientemente. El número de operaciones "abrir paréntesis" debe ser igual a las de "cerrar paréntesis".

Dentro de una expresión entre paréntesis "A (" ... ")" puede haber una combinación "O" o una "Y" delante de "O" de cualquier extensión. En la figura 4.5 se muestra un ejemplo:

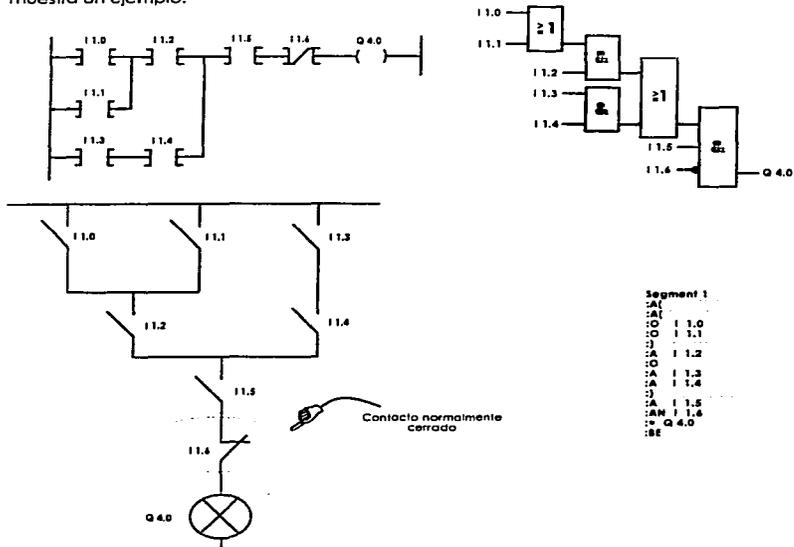


Fig. 4.5 Combinación "O" o una "Y" delante de "O"

## CAPÍTULO 5

### Programando en STEP 5

#### Arranque del Software STEP 5

El sistema operativo **MS - DOS** y el paquete **STEP 5** se encuentran en el disco duro, unidad C (en el caso en que se encuentren en otra unidad se indica la letra correspondiente de la unidad donde se encuentren los paquetes mencionados).

- El software del paquete STEP 5 se encuentra en el directorio **\$5\_100**.
- El programador debe estar conectado y en la pantalla aparece: C:\>
- Introducir la secuencia de caracteres **CD \$5\_100**
- Accionar **Enter**. (El programador despliega **C:\\$5\_100>**)
- Introducir la secuencia de caracteres **\$5**
- Accionar **Enter**.
- De las opciones de idioma seleccionar el idioma inglés (**E**)

El programador visualiza en la pantalla la mascara de SELECCIÓN DE PAQUETE (SELECT PACKAGE)

**SELECCION DE PAQUETE**
**SIMATIC S5/KOMI**

---

LAD, CSF, STL \_\_\_\_\_ V.x.x C: \$5PX\$01X.CMD

**TTY / AS 511 - INTERFAS (ESTANDAR)**

<b>F1</b>	<b>F2</b>	<b>F3</b>	<b>F4</b>	<b>F5</b>	<b>F6</b>	<b>F7</b>	<b>F8</b>
PAQUETE	PROG SER	INFO	VERSION	INTERFASE	UNIDAD	NUEVA SEL	RETORNO

Fig. 5.1 SELECT PACKAGE

**TESIS CON  
FALLA DE ORIGEN**

El **STEP 5** ofrece una única superficie de manejo para selección y arranque de todos los paquetes S5.

### 5.1 Concepto de manejo: Teclado

El aparato de programación con el software del paquete STEP 5 sirve de ayuda tanto al usuario que aún no tenga experiencia como al que ya haya trabajado con él por medio de una guía de operario confortable. En caso de que se haya realizado una entrada errónea, el aparato de programación visualiza en la pantalla el aviso correspondiente explicándole como tiene que realizar el manejo de forma correcta.

Mediante el teclado se puede comunicar con el aparato de programación. Una vez arrancado el paquete STEP 5 se le asigna a algunas combinaciones de teclas del programador una función de SIMATIC. Las teclas de un programador son igualmente asignables de forma variable, es decir, la función de las teclas es asignada por el software que ha sido activado. Lo mismo sucede con el STEP 5; una vez cargado el paquete STEP 5 adoptan las teclas funciones específicas de S5. De esta forma se realiza una configuración de programas y una utilización de programas simple y rápida.

#### 5.1.1 Teclas funcionales Help e info

Por medio de la tecla **HELP** () se pueden recibir en la pantalla informaciones o ayudas adicionales para la entrada, en la línea de comandos en la que se encuentra el cursor, o sobre los pasos de manejo siguientes. Con la tecla funcional **INFO** en la máscara **SELECCIÓN DE PAQUETE (SELECT PACKAGE)** se pueden visualizar aclaraciones sobre los paquetes o en la máscara **AJUSTES PREVIOS (PRESETS)** aclaraciones sobre los campos parciales en los que se encuentra el cursor.

La ventaja de este concepto de manejo reside en que se puede familiarizar rápidamente con este aparato, aún en caso de que no lo conozca.

#### 5.1.2 Superficie de manejo

La superficie de manejo se realiza por medio de guía de menú. De acuerdo con esto, los comandos no se tienen que introducir carácter a carácter, sino que solamente hay que accionar una tecla funcional.

#### 5.1.3 Layout de las máscaras S5

Los programas de servicio y los paquetes S5 tienen básicamente la misma estructura de pantalla. La imagen también llamada máscara está subdividida en cuatro zonas.

TESIS CON  
FALLA DE ORIGEN

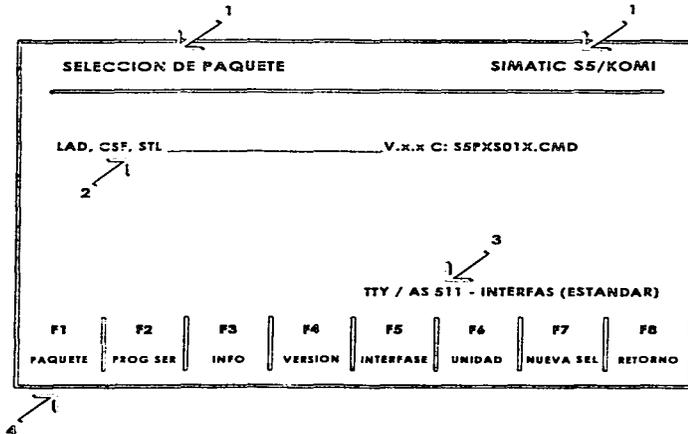


Fig. 5.2 Máscara de selección de paquete

- En la primera zona se dan informaciones sobre los componentes software, el bloque o la función con la que se trabaja en ese momento.
- La segunda, es la zona media con 19 líneas. Es la propia zona de trabajo.
- La tercera zona es la línea de avisos, que ofrece informaciones sobre la función que se desarrolla en ese momento, avisos de error o perturbaciones.
- La cuarta zona visualiza la función que se puede ejecutar accionando una de las 8 teclas funcionales.

#### 5.2 De la selección de paquete a la selección de la función

Después de arrancar el software STEP 5, de la selección de paquete hasta la llamada de una función, solamente se precisan tres pasos de manejo.

TESIS CON  
FALLA DE ORIGEN

1. Primer paso:  
Selección de  
paquete

2. Segundo paso:  
Parametrizar la  
máscara de  
ajustes previos

3. Tercer paso:  
Elección de función.

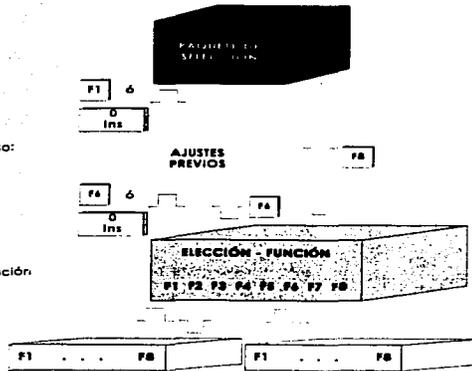


Fig. 5.3 Pasos de manejo

### 5.2.1 Primer paso: Selección de paquete (SELECT PACKAGE)

Punto de partida:

El STEP 5 ha sido arrancado y se ha visualizado en la pantalla la máscara de SELECCIÓN DE PAQUETE.

Seleccionar paquete

En caso de que estén visualizados varios paquetes:

- Posicionar el cursor en el paquete deseado
- Accionar la tecla F: (PACKAGE) o la tecla de aceptación (la tecla "0 ins" de la zona numérica del teclado)

Salte la máscara AJUSTES PREVIOS (PRESETS). El programador espera la parametrización de los ajustes previos.

### 5.2.2 Segundo paso: Parametrizar los ajustes previos (PRESETS)

Ajustes previos de paquete S5 y de los programas de servicio S5

TESIS CON  
FALLA DE ORIGEN

La máscara AJUSTES PREVIOS (PRESETS) contiene aquellos parámetros que se tienen que determinar antes del principio de la propia programación (Edición). Con ayuda de las teclas del cursor se seleccionan los diferentes campos de entrada.

Los nombres de archivo, por ejemplo ARCHIVO DE PROGRAMA (PROGRAM FILE) se introducen directamente a través del teclado. En otros campos de entrada, por ejemplo REPRESENTACIÓN (REPRESENT), se selecciona a partir de diferentes alternativas el ajuste deseado, por ejemplo STL. Estas alternativas las visualiza el programador en el campo de entrada; para lo cual tiene usted que accionar una varios veces la tecla funcional F3 (SELECT). Indicaciones sobre los campos de entrada se reciben a través de la tecla funcional F7 (INFO).

Rellenar la máscara de ajustes previos se llama parametrizar. Los parámetros preseleccionados los acepta el programador automáticamente de paquete a paquete. Los parámetros de la máscara de ajustes previos pueden ser modificados en cada momento para lo cual se tiene que llamar la máscara de ajustes previos a partir de la selección de la función del paquete correspondiente.

#### Punto de partida

Ha sido seleccionado el paquete deseado y el programador visualiza en la pantalla la máscara de ajustes previos.

AJUSTES PREVIOS		SIMATIC 55/PSS01	
REPRESENTAC.	: LAD	ACH. PROGRAMA	: ST.550
SIMBOLICOS	: NO	ACH. SIMBOL.	
COMENTARIOS	: SI	ACH. PIE PAG.	:
PIE DE PAGINA	: NO	ACH. IMPRES.	:
SUMA SEGURO	: NO	ACH. CAMINO	:
MODULO SERVICIO	: OFF		
NOMBRE CAMINO	:		

F1	F2	F3	F4	F6	F6	F7	F8
		ELEGIR			ACEPTAR	INFO	

Fig. 5.3 Máscara de ajustes previos

**F3 (SELECT).**- Se muestran las operaciones (alternativas) en el posicionamiento del cursor.

**F6 (ENTER).**- Da por válidos los parámetros seleccionados y visualizados y realiza la selección de función.

**F7 (INFO).**- Con esta tecla se reciben aclaraciones de conceptos sobre el campo parcial en que está el cursor.

TESIS CON  
FALLA DE ORIGEN

### 5.2.2.1 Aclaraciones de conceptos de la máscara de ajustes previos

#### Archivo de programa (PROGRAM FILE)

55-DOS procesa en los disquetes o en el disco duro varios programas de usuario, que están memorizados en el ARCHIVO DE PROGRAMA. Los nombres de los ARCHIVOS DE PROGRAMA deben ser diferentes. El usuario debe introducir aquí la unidad de disco (A: o C:) y un nombre de 6 caracteres (sólo letras, cifras o el carácter "@"). El primer carácter (una letra), por ejemplo, C: PRUEBA; La secuencia de caracteres "ST.S5D" las coloca el programador automáticamente.

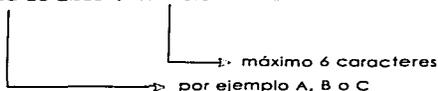
ST.S5D designa archivos con programas de usuario. Esta secuencia de caracteres no puede ser sobrescrita por el usuario.

Si no se indica ningún nombre, el programador introduce como nombre C:#####ST.S5D. Si se introducen menos de 6 caracteres, el programador los rellena con "@"

Todos los comandos posteriores, como por ejemplo: ENTRADA (INPUT), SALIDA (OUTPUT), TRANSFERIR (TRANSFER), BORRAR (DELETE), los procesa solamente el ARCHIVO DE PROGRAMA aquí ajustado.

Sintaxis:

< Unidad de disco > : < Nombre > ST.S5D



#### Posibilidad de selección y atributos

PRO (PROTEGIDO) El archivo está abierto en modo lectura.  
RW (READ/WRITE) Modo lectura - escritura.

#### Representación (REPRESENT)

Los bloques pueden ser editados en una de las tres formas de representación.

#### Opciones

L&D Diagrama de escalera  
CFS Diagrama de funciones lógicas  
STL Lista de instrucciones

#### Archivo simbólico y símbolos (SYMBOLS)

El paquete STEP 5 ARCHIVO SIMBOLICO no es parte integrante del paquete STEP 5 V.3.0.

En el ARCHIVO SIMBOLICO deposita el programador la lista de correspondencias.

TESIS CON  
FALLA DE ORIGEN

**Comentarios (COMMENTS)**

Bajo el concepto de COMENTARIO se incluyen los comentarios de instrucciones de líneas, de segmentos y de instalaciones.

**Opciones**

- **YES** Los comentarios salen en la pantalla o impresora
- **NO** Los comentarios no salen en la pantalla o impresora, a excepción de comentarios de operandos

**Pie de página (FOOTER)**

El pie de página es una parte del texto que el programador incluye en la parte inferior de cada página que se visualiza con impresora. El contenido del texto puede ser cualquiera. Se puede elegir entre pies de página con amplitud de 80 o 132 caracteres. El pie de página se edita con el programador con el programa de servicio FOOTER. El pie de página se memoriza en el ARCHIVO PIE PAG. (FOOTER FILE), pudiéndole llamar el usuario en la máscara de AJUSTES PREVIOS (PRESETS) en cada paquete. Con esto se puede documentar de forma clara el programa de usuario.

**Opciones**

- SI {
- 80 CARACTERES.- Al imprimir, el programador visualiza el pie de página en la parte inferior de la hoja con 80 caracteres
  - 132 CARACTERES.- Al imprimir, el programador visualiza el pie de página en la parte inferior de la hoja con 132 caracteres

NO Impresión sin pie de página

**Archivo pie pág. (FOOTER FILE)**

El pie de página lo edita el usuario en el programador y lo memoriza en un ARCHIVO PIE. El usuario puede llamar en la máscara AJUSTES PREVIOS, en cada paquete, el pie de página deseado, para lo cual se seleccionará el nombre del archivo PIE PAG. Estos son del tipo:

- = F1.INI para amplitud de pie de 80 caracteres
- = F2.INI para amplitud de pie de 132 caracteres

**Archivo de impresora (PRINTER FILE)**

Todos los parámetros de impresora son memorizados en un ARCHIVO DE IMPRESORA, a los que puede acceder el usuario. Estos archivos se pueden modificar y memorizar de nuevo, bajo un nuevo nombre. Cada paquete S5 accede al mismo ARCHIVO DE IMPRESORA, en caso de que no sea llamado y dado por válido otro ARCHIVO DE IMPRESORA en la máscara AJUSTES PREVIOS del paquete que en ese

**TESIS CON  
FALLA DE ORIGEN**

momento elabora el usuario. El ARCHIVO DE IMPRESORA válido en cada momento lo puede visualizar el usuario en el programador llamando la máscara A.JUSTES PREVIOS.  
Los archivos de impresora son del tipo DR.INI

#### **Suma de seguridad (CHEC SUM)**

La suma de seguridad es un método de comprobación de que se ha realizado la transferencia de un bloque del PC al programador de forma correcta. La suma de seguridad de un bloque es generada por el programador e incluida en el bloque. Durante la transferencia del PC al programador, es comprobada por éste último. En caso de que la transferencia sea defectuosa el programador visualiza un aviso.

#### **Modo de servicio (MODE)**

Aquí se determina si la programación y comprobación del programa de usuario se debe realizar en comunicación o no con el autómata.

##### **Opciones**

**OFF** La programación se realiza sin conexión con el autómata

**ON** [NO MOD]

La programación y comprobación se realiza con conexión al autómata. No se puede realizar modificaciones o borrado del programa de mando en el autómata

**ON** [NO IN STOP]

Sólo se puede modificar el programa de mando estando el autómata en STOP

**ON** [MOD IN CYCLE] (Opción por default)

También se pueden realizar modificaciones durante el procesamiento cíclico del programa del autómata

##### **NOTA:**

*Después de una perturbación de la comunicación con el autómata se puede establecer una nueva comunicación con una secuencia de selecciones ON u OFF.*

TESIS CON  
FALLA DE ORIGEN



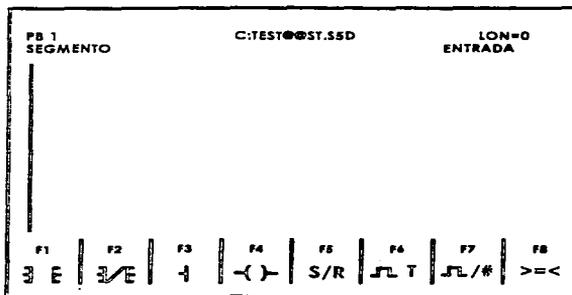
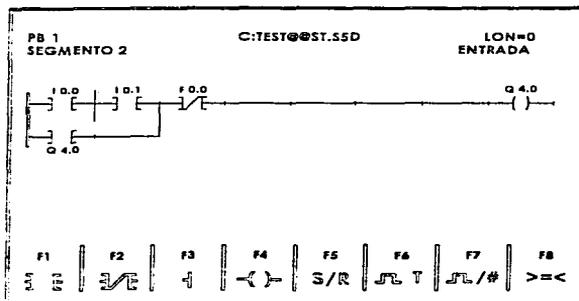


Fig 5.4 Máscara del editor de escalera

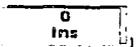
#### Significado de las teclas funcionales

Las explicaciones sobre la función de las teclas funcionales F1 a F8 se pueden recibir por medio de la tecla HELP (  $\leftarrow$  Backspace  $\rightarrow$  ).



Finalizar y memorizar el bloque

Accionar la tecla de aceptación (



TESIS CON  
FALLA DE ORIGEN

El programador visualiza la máscara de **SELECT FUNCTION**.

#### 5.4 Salida de bloques (OUTPUT / BLOCK)

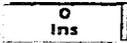
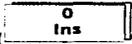
Visualizar el bloque de programa en la pantalla

- Punto de partida: **SELECT FUNCTION**
- Se selecciona ahora el modo **SALIDA (OUTPUT)**
- Accionar la tecla **F2 (OUTPUT)** y enseguida accionar la tecla **F2 (BLOCK)**
- Rellenar la línea de comando:

OUTPUT DEVICE: **FD**    BLOCK: **PB1**

SEARCH:

PTR:

- Accionar la tecla de **aceptación** (). En la pantalla aparece el primer segmento del bloque
- Con las teclas (de la zona numérica del teclado) "+" y "-", se puede avanzar o retroceder segmentos
- Con la tecla funcional "F7", se puede cambiar la forma de representación del segmento (**LAD, CSF ó STL**)
- Para salir de visualización se pulsa la tecla de **aceptación** ()

TESIS CON  
FALLA DE ORIGEN

## CAPÍTULO 6

### Correcciones en LAD y CSF

En este capítulo se describe mediante ejemplos las posibles correcciones a lo largo de un segmento o un bloque.

En el modo de representación **LAD (Diagrama de Contactos)** o **CSF (Bloque de Funciones)** existen las siguientes posibilidades de corrección:

#### Sobreescribir

Operandos	(Modo CORRECCIÓN o INSERCIÓN)
Entradas	(Modo CORRECCIÓN o INSERCIÓN)
Elementos de funciones	(Modo CORRECCIÓN o INSERCIÓN)
Contactos	(Modo CORRECCIÓN o INSERCIÓN)

#### Inserción de

Entradas	(Modo CORRECCIÓN o INSERCIÓN)
Elementos de funciones	(Modo CORRECCIÓN o INSERCIÓN)
Contactos	(Modo CORRECCIÓN o INSERCIÓN)
Segmentos	(Modo SALIDA)

#### Borrado de

Entradas	(Modo CORRECCIÓN o INSERCIÓN)
Elementos de funciones	(Modo CORRECCIÓN o INSERCIÓN)
Contactos	(Modo CORRECCIÓN o INSERCIÓN)
Segmentos	(Modo SALIDA)

#### 6.1. Corrección a lo largo de un segmento

Las correcciones a lo largo de un segmento se pueden realizar si el programador se encuentra dentro del editor. El editor se puede llamar desde una de las funciones siguientes:

- ENTRADA/BLOQUE
- SALIDA/BLOQUE/CORRECCIÓN
- SALIDA/BLOQUE/INSERCIÓN

Se pueden utilizar en el modo de edición todas las funciones de edición que son explicadas en el Capítulo 5 en el tema "Entrada de bloques" (INPUT / BLOCK).

<p><b>TESIS CON FALLA DE ORIGEN</b></p>
---

### 6.1.1 Particularidades

#### Modo corrección

Si el programador se encuentra en el editor, se pueden realizar todas las correcciones en el modo **ENTRADA (INPUT/BLOCK)**

En el modo **SALIDA (OUTPUT/BLOCK)**, se tiene que llamar adicionalmente la función

5

**CORRECCIÓN** por medio de la tecla especial **CORR** (La tecla de la zona numérica del teclado), para cambiar al modo editor.

#### Búsqueda

El puesto de corrección deseado se puede encontrar de forma rápida y sencilla por medio de la función **BÚSQUEDA**

#### Comprobación de sintaxis

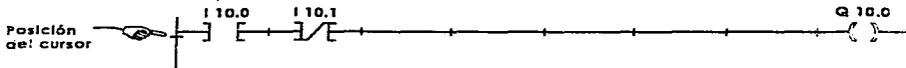
El cursor sólo abandona el campo de entrada si la entrada ha sido correcta.

### 6.2 Correcciones en LAD (Diagrama de Contactos)

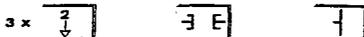
#### Ejemplos:

#### CONTACTO EN PARALELO

Situación de partida:



Entrada de:



Visualización:



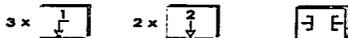
TESIS CON  
FALLA DE ORIGEN

### INSERCIÓN DE UN CONTACTO EN PARALELO

Situación de partida:



Entrada de:

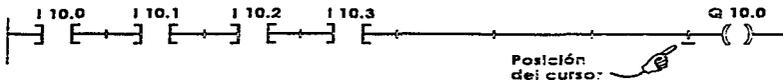


Visualización:



### SALIDA ADICIONAL

Situación de partida:

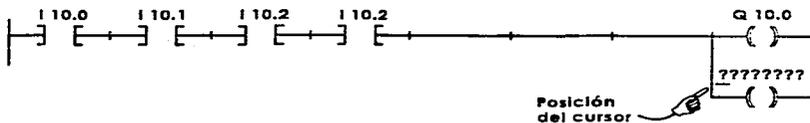


Entrada de:



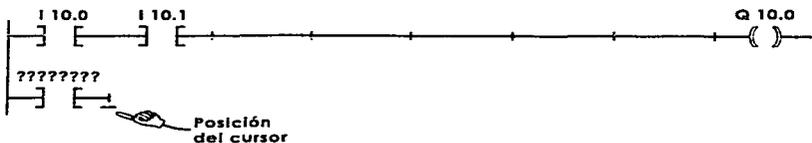
TESIS CON  
FALLA DE ORIGEN

Visualización:



ELEMENTO VACÍO

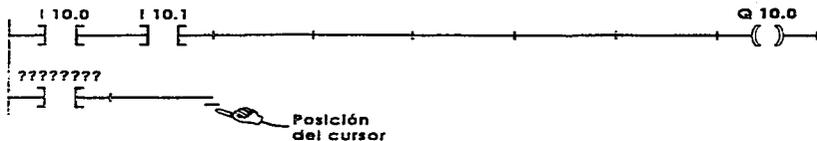
Situación de partida:



Entrada de:

Tecla del tabulador  
en esa dirección

Visualización:



TESIS CON  
FALLA DE ORIGEN

### INSERCIÓN DE UN CONTACTO EN SERIE

Situación de partida:

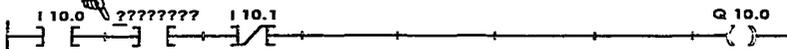


Entrada de:



Visualización:

Posición  
del cursor

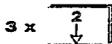


### CONTACTO EN PARALELO

Situación de partida:



Entrada de:



TESIS CON  
FALLA DE ORIGEN

Visualización:

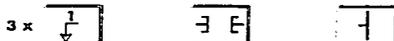


## INSERCIÓN DE UN CONTACTO EN PARALELO

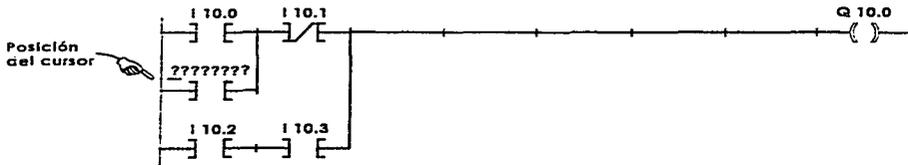
Situación de partida:



Entrada de:



Visualización:



TESIS CON  
FALLA DE ORIGEN

## INVERSIÓN DE UN CONTACTO

Situación de partida:



Entrada de:

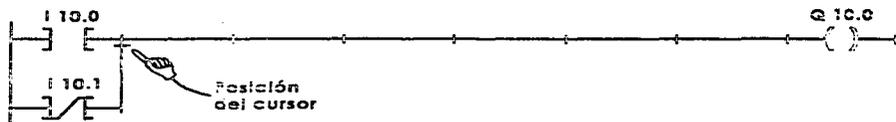


Visualización:

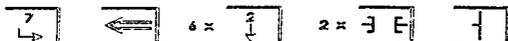


## CIRCUITO PARALELO

Situación de partida:

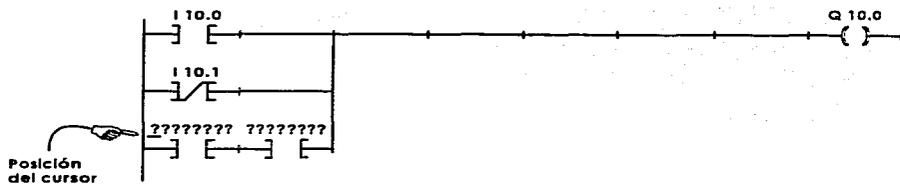


Entrada de:



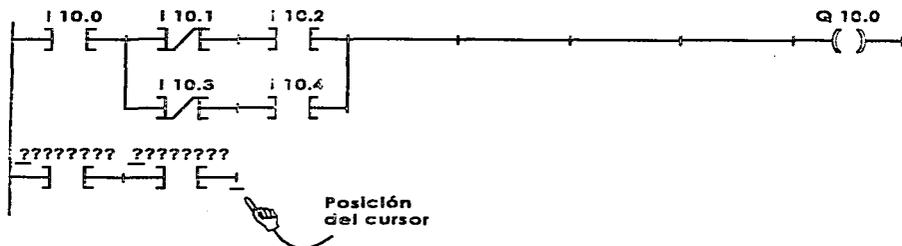
TESIS CON  
FALLA DE ORIGEN

Visualización:



## CIRCUITO PUENTE

Situación de partida:

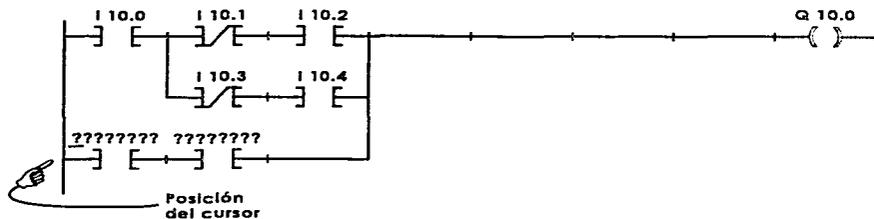


Entrada de:



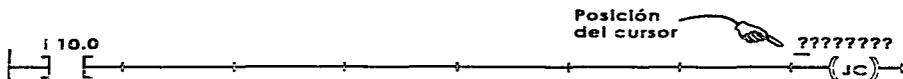
TESIS CON  
FALLA DE ORIGEN

Visualización:

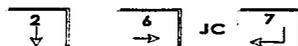


## SALIDAS ESPECIALES

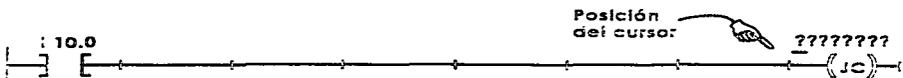
Situación de partida:



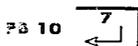
Entrada de:



Visualización:

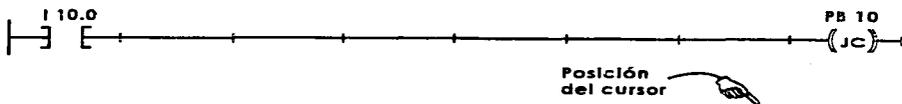


Después:



TESIS CON  
FALLA DE ORIGEN

Visualización:

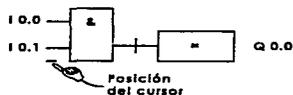


### 6.3 Correcciones en CSF (Bloque de Funciones)

Ejemplos:

#### INSERCIÓN DE ENTRADAS ADICIONALES

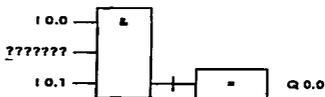
Situación de partida:



Entrada de:

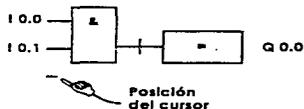


Visualización:



ó también:

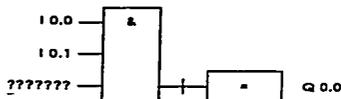
Situación de partida:



Entrada de:



Visualización:



#### INSERCIÓN DE UNA OPERACIÓN ADICIONAL

### 6.6 Insertar, añadir y borrar segmentos

Punto de partida:

El programador se encuentra en el modo SALIDA DE BLOQUES (OUTPUT / BLOCK)

Inserta: segmentos

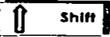
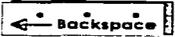
- Por medio de las teclas "+" o "-", se visualiza el segmento, en donde se desea realizar la inserción. Antes de ese segmento se inserta el nuevo segmento.

TESIS CON  
FALLA DE ORIGEN

- Accionar la tecla "Insertar segmento" ( Shift  Fin). El programador visualiza el nuevo segmento. Los segmentos son renumerados de nuevo.
- Introducir las restantes instrucciones.

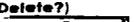
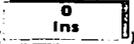
#### Añadir segmentos

- Seleccionar el último segmento

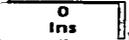
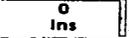
- Accionar la tecla añadir segmento ( Shift  Backspace)
- Se añade un segmento

#### Borrar segmentos

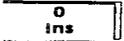
- Seleccionar el segmento

- Accionar la tecla "Borrar segmento" ( Shift  Supr)
- El programador pregunta: ¿Borrar? ( Delete?)
- Sí: Accionar la tecla de aceptación ( Ins)
- No: Accionar la tecla de Interrupción ( Esc)

#### Memorizar bloques corregidos

- Accionar la tecla de aceptación ( Ins)
- El programador visualiza el aviso: *Ya en archivo destino. ¿sobrescribir?*
- Sí: Accionar la tecla de aceptación ( Ins)
- No: Accionar la tecla de Interrupción ( Esc)

#### Interrumpir las correcciones

- Accionar la tecla de aceptación ( Ins)

TESIS CON  
FALLA DE ORIGEN

## CAPÍTULO 7

### Funciones auxiliares para la gestión de bloques.

#### MODOS DE OPERACIÓN

Según Balcells y Romeral en su libro *Autómatas Programables*<sup>12</sup>, afirman que un autómata bajo tensión puede mantenerse en uno de los siguientes estados de funcionamiento:

- **RUN:** El autómata ejecuta normalmente el programa de usuario contenido en su memoria.
  - Las salidas evolucionan *ON/OFF* según el estado de las entradas y las órdenes del programa
  - Los temporizadores y contadores programados operan con normalidad

Con el autómata en **RUN**, el programa de usuario se ejecuta siguiendo una secuencia o ciclo de operación que pasa por las siguientes etapas:

- Chequeos del sistema
  - Lectura de señales desde la interfaz de entradas
  - Escritura de señales en la interfaz de salidas
  - El procesamiento del programa para la obtención de las señales de control
- **STOP:** La ejecución del programa se detiene por orden del usuario.
    - Las salidas pasan a estado *OFF*
    - Las posiciones internas (relés, registros), contadores y temporizadores mantienen su estado en memoria interna
  - **ERROR:** El autómata detiene la ejecución por un error de funcionamiento, y queda bloqueado hasta que se corrige el error.
    - Las salidas pasan a estado *OFF*
    - Corregido el error, el autómata sale de este modo bien por reset de puesta en tensión u ordenado desde la *CPU*, bien por comando enviado desde la unidad de programación

Balcells y Romeral afirman también que con las funciones auxiliares son gestionados los bloques y los archivos de documentación así como los archivos de programa. Con las funciones auxiliares se pueden ejecutar las siguientes funciones.

- Transferencia de bloques y archivos de documentación
- Borrado de bloques y archivos de documentación, borrado total del PC
- Salida del índice (Directorio)
- Cambio de archivo de programa ajustado previamente

<sup>12</sup>Balcells, Joseph, Romeral, José Luis, Op. cit., p.p. 76 y 88

### 7.1 Selección de las funciones auxiliares

**Punto de partida:** El programador se encuentra en el modo **SELECCIÓN DE FUNCIÓN (SELECT / FUNCTION)**

Accionar la tecla **F7 (AUX FCT)**. El programador visualiza las funciones auxiliares

FUNCIONES A U X		SIMATIC S5/OSSOA
<b>F1 TRANSF.</b>	:	TRANFERENCIA DE MÓDULOS
<b>F2 BORRAR</b>	:	BORRADO DE MÓDULOS O APARATOS (AG, PG, FD)
<b>F3 DIR</b>	:	DIRECTORIO (RESUMEN DE CONTENIDO) DEL AG, FD
<b>F6 ACH. PRG</b>	:	MODIFICAR EL FICHERO DE PROGRAMA AJUSTADO
<b>F8 RETORNO</b>	:	REGRESO A LA SELECCIÓN DE FUNCIÓN
<b>F1</b>		<b>F2</b>
TRANSF.		BORRAR
<b>F3</b>		<b>F4</b>
DIR		<b>F5</b>
<b>F6</b>		<b>F7</b>
ACH PRG		<b>F8</b>
		RETORNO

Fig. 7.1 Mascar de Selección de función

- **F1 (TRANSFER)** Transferencia de bloques y archivos de documentación
- **F2 (DELETE)** Borrado de bloques y archivos de documentación, borrado total del PC
- **F3 (DIR)** Salida del índice (Directorio)
- **F6 (PRG FILE)** Cambiar el archivo de programa preseleccionado
- **F8 (RETURN)** Regreso a la elección de función

### 7.2 Transferencia de bloques

Con esta función se pueden transferir:

- Bloques individuales
- Una zona de bloques de un tipo de bloque
- Todos los bloques de un tipo de bloque

TESIS CON  
FALLA DE ORIGEN

- Un grupo de bloques (excepto los archivos de documentación) de un archivo de programa
- Uno o todos los archivos de documentación (archivo DOC)
- El archivo de programa completo

Los aparatos que pueden utilizarse como fuente o destino de intercambio de bloque son:

- **PC** El autómata programable se puede seleccionar como fuente o destino. El PC puede encontrarse en **RUN** o en **STOP**
- **FD** Como fuente o destino se selecciona el archivo de programa preseleccionado
- **ARCHIVO** Como destino o fuente de la transferencia se puede seleccionar cualquier archivo de programa

#### NOTA IMPORTANTE

- **Archivo de documentación**

*Los archivos de documentación no pueden ser memorizados en un PC, por consiguiente, el PC no puede ser utilizado nunca como fuente o destino en el caso de transferencia de un archivo de documentación.*

- **Bloques de comentario**

*Los bloques de comentario no se pueden memorizar en el PC, por consiguiente, en el caso de transferencia de bloque de comentario no se puede seleccionar el PC como destino o fuente.*

### 7.2.1 Pasos de manejo de la función Transferir

#### Visualizar el bloque de programa en la pantalla

- Punto de partida: **SELECT FUNCTION**
- Se selecciona ahora el modo **FUNCIONES AUXILIARES F7 (AUX FCT)**
- Accionar la tecla **F7 (TRANSFER)** y el programador visualiza la línea de comandos:

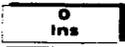
**TRANSFER SOURCE:                      BLOCK:                      TO DEST:                      BLOCK:**

- Rellenar la línea de comando:
- Accionar la tecla de **aceptación**: (la tecla "**0 Ins**" de la zona numérica del teclado)

**Posibles errores**

TESIS CON  
FALLA DE ORIGEN

*PB ya en archivo destino, ¿sobrescribir? Un bloque existe ya en el archivo destino. Son posibles las siguientes reacciones:*

- Sí. Accionar la tecla de aceptación ()
- No Accionar la tecla de interrupción ()

*PC no es posible como aparato. En los paquetes offline no puede realizarse ningún intercambio de datos en el PC.*

### 7.3 Borrar bloques

Lo que se puede borrar:

- Bloques individuales
- Una zona de bloques de un mismo tipo de bloque
- Bloques de un mismo tipo (sólo en FD)
- Todos los bloques (solamente con borrado total del PC)
- Todo el archivo de programa (sólo en FD)

Los bloques pueden ser borrados desde:

- **PC** Son borrados en el autómatas todos los bloques indicados
- **FD** Son borrados los bloques indicados del archivo de programa seleccionado

#### NOTA IMPORTANTE

##### **Borrado total del PC**

*La función con la que se borran todos los bloques en el PC, se llama BORRADO TOTAL (ALL). El PC tiene que estar en STOP y tiene que existir online (conexión del PLC con el PC) con el programador.*

##### **Borrar en FD**

*En el dispositivo se borra el archivo que haya sido introducido en la máscara de ajustes previos como archivo de programa*

#### 7.3.1 Pasos de manejo de la función BORRAR

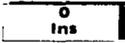
Visualizar el bloque de programa en la pantalla

TESIS CON  
FALLA DE ORIGEN

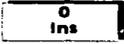
- Punto de partida: **SELECT FUNCTION**
- Se selecciona ahora el modo **FUNCIONES AUXILIARES F7 (AUX FCT)**
- Accionar la tecla **F2 (DELETE)** y el programador visualiza la línea de comandos:

**DELETE FROM SOURCE:                      BLOCK:**

- Rellenar la línea de comando:

- Accionar la tecla de **aceptación** ()

*El programador visualiza la pregunta Delete?*

- Sí.      Accionar la tecla de aceptación ()
- No      Accionar la tecla de interrupción ()

#### **7.4 Salida del directorio**

El usuario puede visualizar en la función directorio:

- La lista de bloques del archivo de programa seleccionado
- La lista de bloques contenidos en la memoria del autómata
- Lista de archivos \$S de una unidad de disco

**Para cada bloque se indica:**

- El tipo de bloque
- El número de bloque
- El archivo de documentación (no en el PC)
- La longitud del bloque (no en el PC)
- El número de biblioteca (no en el PC)
- La dirección de principio en el PC (sólo en el PC)

##### **7.4.1 Pasos de manejo de la función DIRECTORIO**

**Visualizar el bloque de programa en la pantalla**

**TESIS CON  
FALLA DE ORIGEN**

- Punto de partida: **SELECT FUNCTION**
- Se selecciona ahora el modo **FUNCIONES AUXILIARES F7 (AUX FCT)**
- Accionar la tecla **F3 (DIR)** y el programador visualiza la línea de comandos:

**OUTPUT DIR\_FROM SOURCE:                      BLOCK:                      PTR:**

- Rellenar la línea de comando:

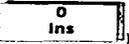
- Accionar la tecla de aceptación (  )

El programador visualiza el directorio seleccionado en la pantalla, en la impresora.

### 7.5 Modificar un archivo de programa ajustado

Con la función **F6 (PRG FILE)** se puede modificar el archivo de programa ajustado sin tener que llamar la máscara AJUSTES PREVIOS.

- Llamar la función auxiliar **F6 (PRG FILE)**
- Modificar el nombre del archivo de programa

- Accionar la tecla de aceptación (  )

TESIS CON  
FALLA DE ORIGEN

## CAPÍTULO 8

### Operaciones básicas de STEP 5.

#### Circuitos de automantenimiento

La conexión tradicional para realizar una función de memoria en control convencional, es la de un circuito de automantenimiento.

Se lleva a cabo a través de la conexión de un contacto normalmente abierto del contactor, el cual está en paralelo con un pulsador de arranque.



Preferencia a la desconexión

Preferencia a la conexión

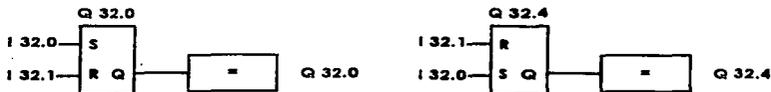
Fig. 8.1 Circuitos de automantenimiento

Para el funcionamiento del contactor existen 2 variantes:

1. **Preferencia a la desconexión** El pulsador de paro quedará en serie directamente con la bobina, de tal manera que mientras el botón de paro esté accionado jamás se podrá energizar la bobina
2. **Preferencia a la conexión** Se disponen en serie el pulsador de paro y el contacto de autoretención y ellos a su vez en paralelo con el pulsador de arranque

#### 8.1 Funciones de memoria R - S

Una función de memoria R - S, se representa con un rectángulo con una entrada de carga "S" y una entrada de borrado "R". Un cambio de señal de 0 a 1 en la entrada "S" significa *conexión*. Por lo contrario, un cambio de señal de 0 a 1 en la entrada "R" significa *desconexión*. En caso de que se aplique un estado de señal 0 simultáneamente en "R" y "S", no varía el estado de señal que hubiera con anterioridad.



## Segment 1

```

:A I 32.0
:S Q 32.1
:A I 32.1
:R Q 32.0
:A Q 32.0
:= Q 32.0
:***

```

## Segment 2

```

:A I 32.1
:R Q 32.4
:A I 32.0
:S Q 32.4
:A Q 32.4
:= Q 32.4
:***

```

a) Memoria con preferencia al RESET

b) Memoria con preferencia al SET

Fig. 8.2 Operaciones de memoria

## SR MEMORIAS (F5)

F1: R/S: Memoria con preferencia al SET

F2: S/R: Memoria con preferencia al RESET

S	R	Q (Salida binaria)
0	0	Estado de señal sin cambio
0	1	0
1	0	1
1	1	0 con S/R 1 con R/S

Tabla 8.1 Salidas binarias

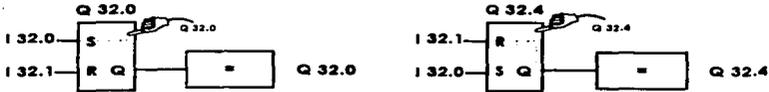
Como se observa en la tabla anterior, si "R" y "S" tienen simultáneamente el estado de señal "1" tiene preferencia:

- El SET si la memoria es R/S
- El RESET si la memoria es S/R

En la programación, en una elaboración cíclica las últimas instrucciones programadas son elaboradas en último lugar. En el caso a) (Memoria con preferencia al RESET) se ejecuta primero el SET, la salida se conecta, posteriormente se ejecuta el RESET y la salida se desconecta. En el caso b) (Memoria con preferencia al SET) ocurre totalmente todo lo contrario.

TESIS CON  
 FALLA DE ORIGEN

**NOTA:** En la programación de lista de instrucciones (STL), como se observa en los casos a) y b), los operandos de las salidas (Q 32.0 y Q32.4) se consideran en serie con las salidas (Q 32.0 y Q32.4) respectivamente, como se muestra a continuación:



#### Segment 1

```

:A I 32.0
:S Q 32.1
:A I 32.1
:R Q 32.0
:A Q 32.0
:= Q 32.0
:***

```

#### Segment 2

```

:A I 32.1
:R Q 32.4
:A I 32.0
:S Q 32.4
:A Q 32.4
:= Q 32.4
:***

```

a) Memoria con preferencia al **RESET**

b) Memoria con preferencia al **SET**

Fig. B.3 Consideración en la programación en STL de las memorias

### B.2 Temporizadores (TIMER'S)

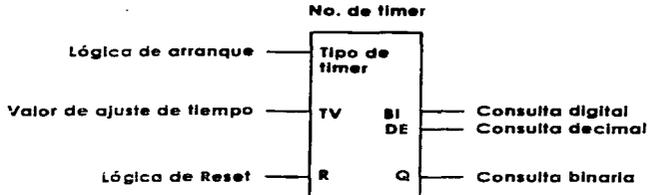


Fig. B.4 Timer

Cada uno de los temporizadores, está contenido en la memoria en forma de 16 bits (2 bytes), palabra de tiempo. LA palabra de tiempo contiene, entre otros datos, el valor del tiempo real y la escala del tiempo, en la cual, y a través de la unidad de control, reduce el valor del tiempo a la unidad (seg.).

TESIS CON  
FALLA DE ORIGEN

El tiempo de temporización se puede expresar por medio de una constante de tiempo (KT), la cual tiene la siguiente estructura:

$$KT = (n)(I)$$

Donde:

**n** = valor del tiempo real que va desde 0 a 999

**I** = base de tiempo que va desde 0 a 3, que equivalen a:

0 = 0.01 seg

1 = 0.1 seg

2 = 1 seg

3 = 10 seg

El valor del tiempo real en segundos, se obtiene multiplicando el valor del tiempo por la base del tiempo, así por mostrar algunos ejemplos:

KT 750.0 equivale a 7.5 seg.

KT 350.1 equivale a 35 seg.

KT 210.2 equivale a 210 seg.

KT 56.3 equivale a 560 seg.

Dado que:

$$KT = (n)(I) = (750)\left\{\overset{0}{0.01}\right\} = 7.5 \text{ seg.}$$

$$KT = (n)(I) = (350)\left\{\overset{1}{0.1}\right\} = 35 \text{ seg.}$$

$$KT = (n)(I) = (210)\left\{\overset{2}{1}\right\} = 210 \text{ seg.}$$

$$KT = (n)(I) = (56)\left\{\overset{3}{10}\right\} = 560 \text{ seg.}$$

A continuación se muestra un ejemplo de un timer SP (Pulso) y su correspondiente lista de instrucciones:

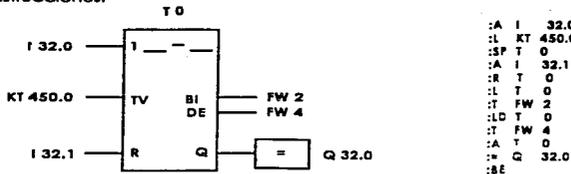


Fig. 8.5 Programación de un timer

TESIS CON  
 FALLA DE ORIGEN

### 8.2.1 Tipos de temporizadores (TIMER'S) (F6)

- Punto de partida: **SELECT FUNCTION**
- A partir de la máscara de "SELECT FUNCTION" se presiona **F1 (INPUT)** y enseguida **F1 (BLOCK)**. Aparece la siguiente línea de comandos:

**INPUT DEVICE:**

**BLOCK:**

- Se rellena la línea de comandos con los siguientes caracteres:

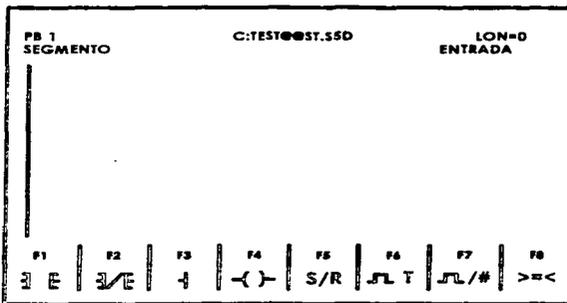
**FD:** Se refiere al archivo de programa ajustado previamente

**PB1:** Ésta es la designación del bloque

**INPUT DEVICE: FD**

**BLOCK: PB1**

- Se acciona la tecla de **aceptación** (la tecla "0 Ins" de la zona numérica del teclado)
- En la pantalla aparece la máscara del editor de escalera (**LAD**)



- Se acciona la tecla **F6** y aparece la máscara de **TIMER FUNCTIONS**

F1 SF (I - I) : Start timer as pulse  
 F2 SE (I - V) : Start timer as extended pulse  
 F3 SD (TI - IO) : Start timer as ON delay  
 F4 SS (TI - IS) : Start timer as stored ON delay  
 F5 SF (OI - IT) : Start timer as OFF delay

TESIS CON  
FALLA DE ORIGEN

### 8.2.1.1 Impulso (SP)

En el caso de que el resultado de la combinación sea "1" se pone en marcha el temporizador. Este timer mantiene activada su salida todo el tiempo que el timer este conectado, al terminar de contar regresa la salida a cero. Para que este timer lleve a cabo la operación descrita, es necesario que la señal de arranque se mantenga.

### 8.2.1.2 Impulso prolongado (SE)

Con el resultado de la combinación igual a "1" pone en marcha el temporizador. Este timer realiza una función similar a la del timer de impulso. Mientras que el timer **SP** necesita que la señal de arranque este presente en todo momento, el timer **SE** solo necesita un flanco positivo (de "0" a "1") en su entrada de arranque para realizar la misma función que el timer **SP**. Si se produce un flanco positivo (de "0" a "1") en la entrada de arranque mientras está corriendo el tiempo, se arranca otra vez desde el principio.

### 8.2.1.3 Retardo a la conexión (SD)

Con un resultado de la combinación "1" se pone en marcha el temporizador. Este timer activa su salida cuando el tiempo ha transcurrido, siempre y cuando la señal de arranque se mantenga en "1". Cuando la señal de arranque regresa a "0" (cero) la salida de este timer también se va a "0" cero.

### 8.2.1.4 Retardo a la conexión memorizada (SS)

Con un resultado de la combinación "1" y en la primera elaboración se pone en marcha el temporizador. Este timer realiza una función muy similar a la de un timer **SD**. Mientras que el timer **SD** necesita que la señal de arranque este presente en todo momento, el timer **SS** solo necesita un flanco positivo (de "0" a "1") en la entrada de arranque para realizar la misma función que el timer **SD**. Si se produce otro flanco positivo (de "0" a "1") en la entrada de arranque mientras está corriendo el tiempo, se arranca otra vez desde el principio. El estado de la señal de salida regresa a "0" (cero) si se borra el temporizador con la función **R** (reset).

### 8.2.1.5 Retardo a la desconexión (SF)

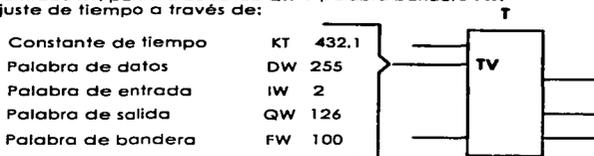
Con un resultado de la combinación "0" (cero) se pone en marcha el temporizador. Mientras la señal de arranque sea "1", o bien, el timer este contando, la salida del timer se encuentra en "1". La salida de este timer regresa a "0" (cero) cuando ha transcurrido el tiempo.

**NOTA:** La operación **Reset (R)** pone en ceros el timer; salida y tiempo. La operación **Reset** tiene prioridad ante cualquier otra operación del timer.

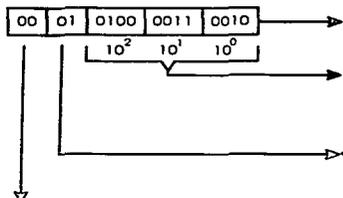
TESIS CON  
 FALLA DE ORIGEN

### 8.2.2 Ajuste de tiempo (CARGA)

El tiempo **TV** se puede ajustar como una constante **KT**, palabra de datos **DW**, palabra de entrada **IW**, palabra de salida **QW** o palabra bandera **FW**.  
Ajuste de tiempo a través de:



El tiempo **TV** debe corresponder a la estructura de la palabra para el ajuste de tiempo.



Estructura de la palabra para un valor dado de tiempo  
(  $432 \times 0.1 \text{ s} = 43.2 \text{ s}$  )

Valor del tiempo  
0 - 999 ( BCD )

Base del tiempo en BCD

0: 0.01 s  
1: 0.1 s  
2: 1 s  
3: 10 s

Sin ningún significado para el ajuste de tiempo

#### Ajuste de tiempo (carga)

Una palabra de tiempo está compuesta, empezando por la izquierda, de dos bits de estado que son utilizados por el procesador para la elaboración de la función de tiempo, sin embargo, no tiene ningún significado para el ajuste de tiempo. A continuación vienen dos bits utilizadas para establecer la base de tiempo, después vienen tres grupos de cuatro bits para indicar el valor numérico del tiempo en código BCD.

### 3.3 Funciones de carga y transferencia

Con las operaciones de carga y transferencia el lenguaje de programación STEP 5 hace posible el intercambio de informaciones entre los diferentes operandos del STEP 5, como son: entradas y salidas (periférica), la imagen de proceso de las entradas y salidas, la memoria de banderas, tiempos y contadores así como os bloques de datos. Este intercambio de información no se realiza directamente, sino a través de un acumulador (ACCU). Este es un registro especial en el procesador que actúa como una Memoria Intermedia.

TESIS CON  
FALLA DE ORIGEN

En el intercambio de información se distingue perfectamente el sentido del flujo de la información. Si se realiza desde la memoria al **ACCU** se le denomina *carga* y si se realiza desde el **ACCU** a la memoria se le denominará como *transferencia*. En la familia de PLC's *SIMATIC 3S*, existen 2 acumuladores: **ACCU 1** y **ACCU 2**.

Con la operación de carga el contenido de la memoria es copiado y cargado en el **ACCU 1**. Si se realiza una segunda carga se sobrescribe sobre el **ACCU 1** y el contenido anterior de éste, pasa al **ACCU 2**, y el contenido de éste último se pierde.

Con la operación de transferencia se copia el contenido del **ACCU 1** quedando a disposición de futuras operaciones de transferencia. Dicho contenido es cargado en la memoria del correspondiente receptor, sobrescribiéndose sobre el contenido anterior.

Las operaciones de carga y transferencia son operaciones absolutas e independientes de cualquiera que sea el resultado de la combinación, siendo ejecutadas en cada ciclo.

Las operaciones de carga y transferencia sólo se pueden representar en **listas de instrucciones (STL)** a excepción de las operaciones de **tiempo, cómputo y comparación**.

### 8.4 Contadores (COUNTERS)

- Punto de partida: **SELECT FUNCTION**
- A partir de la máscara de "SELECT FUNCTION" se presiona **F1 (INPUT)** y enseguida **F1 (BLOCK)**. Aparece la siguiente línea de comandos:

**INPUT DEVICE:**

**BLOCK:**

- Se rellena la línea de comandos con los siguientes caracteres:

**FD:** Se refiere al archivo de programa ajustado previamente

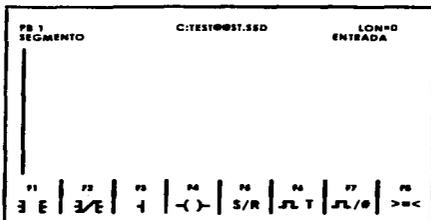
**PB1:** Ésto es la designación del bloque

**INPUT DEVICE: FD**

**BLOCK: PB1**

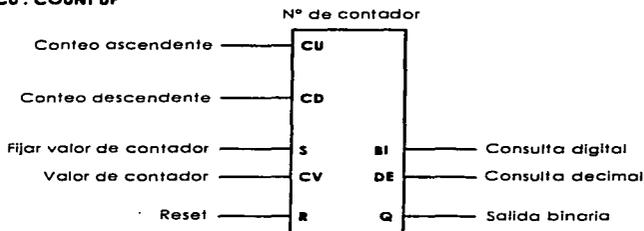
- Se acciona la tecla de **aceptación** (la tecla "**0 Ins**" de la zona numérica del teclado)
- En la pantalla aparece la máscara del editor de escalera (**LAD**)

TESIS CON  
 FALLA DE ORIGEN



- Se acciona la tecla **F7** y aparece la máscara de **COUNTER FUNCTIONS**

**F1 CD: COUNT DOWN**  
**F2 CU: COUNT UP**



**Fig. 8.6 Representación gráfica de un contador**

La técnica utiliza diversos tipos de contadores para los diferentes procesos de cómputo, como por ejemplo el conteo de estaciones de vigilancia de sacos, coches, lotes, vaso, etc.

Cada uno de los contadores está asignado en la memoria como una palabra de 16 bits, palabra de cómputo.

Al programar en lista de instrucciones se tiene que tener en cuenta la secuencia de órdenes a continuación se muestra un ejemplo con su correspondiente programación en Bloques de funciones (CSF):

TESIS CON  
 FALLA DE ORIGEN

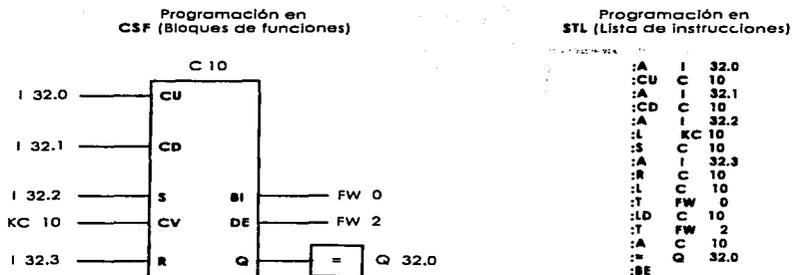


Fig. 8.7 Contador

### 8.4.1 Conteo hacia adelante = CU

El valor numérico del contador se aumenta en 1. La función se realiza cuando se detecta un cambio de flanco positivo (de "0" a "1") en la entrada correspondiente CU.

Al alcanzar dicho valor numérico el límite superior de 999 ya no sigue aumentando y los cambios de estado de señal en la entrada de este contador no surten efecto en lo sucesivo. No existe arrastre de valores.

### 8.4.2 Conteo hacia atrás = CD

El valor numérico del contador disminuye en 1. La función se realiza cuando se detecta un cambio de flanco positivo (de "0" a "1") en la entrada correspondiente CD.

Al alcanzar dicho valor numérico el límite inferior de 0, ya no sigue disminuyendo y los cambios de estado de señal en la entrada del contador hacia atrás no tienen efecto en lo sucesivo (sólo valores numéricos positivos). No existe arrastre de valores.

### 8.4.3 Carga de un contador

Un contador se carga con el valor depositado en "CV" cuando en su entrada de carga "S" se detecta un cambio de flanco positivo (de "0" a "1").

### 8.4.4 Borrado de un contador "R"

Un contador queda borrado cuando en la entrada del borrado o antes de la operación de borrado aparece un (Resultado Lógico de la Operación) RLO = 1. Con un (Resultado Lógico de la Operación) RLO = 0 permanece invariable. El borrado es una

operación independiente. Al cumplirse la condición de borrado el contador no puede cargarse ni efectuar funciones de conteo.

#### 8.4.5 Consulta digital de un contador

Los valores digitales de un contador pueden cargarse en el acumulador como numero DUAL (BI) o DECIMAL (DE) y desde aquí transferirlos a otra zona de operandos.

Se debe distinguir, especialmente en la programación de lista de instrucciones, entre el valor L, para la consulta de "BI" y LD, (codificación de la carga) para la consulta de "DE".

#### 8.4.6 Consulta binaria del estado de señal de un contador "Q"

Se puede consultar el estado bit de un contador. Cuando el contador tiene un valor de cero el estado bit del contador es 0. Cuando el contador tiene un valor distinto de cero, el estado bit del contador es 1.

Los estados de señal pueden consultarse con A C, O C, AN C y ON C.

### 8.5 Funciones de comparación

- Punto de partida: **SELECT FUNCTION**
- A partir de la máscara de "SELECT FUNCTION" se presiona F1 (INPUT) y enseguida F1 (BLOCK). Aparece la siguiente línea de comandos:

**INPUT DEVICE:**

**BLOCK:**

- Se rellena la línea de comandos con los siguientes caracteres:

**FD:** Se refiere al archivo de programa ajustado previamente

**PB1:** Ésta es la designación del bloque

**INPUT DEVICE: FD**

**BLOCK: PB1**

- Se acciona la tecla de **aceptación** (la tecla  del teclado)
- En la pantalla aparece la máscara del editor de escalera (LAD)

TESIS CON  
 FALLA DE ORIGEN



- Con la segunda operación de carga, primero se traspaasa el primer operando del **ACCU 1** al **ACCU 2** y a continuación se registra el segundo operando en el **ACCU 1**.
- Posteriormente compara el contenido del **ACCU 1** con el contenido del **ACCU 2**.

### 8.6 Funciones aritméticas

Estas operaciones permiten tratar el contenido de los acumuladores como números decimales con punto fijo, y operar con ellos aritméticamente. El resultado es depositado en el **ACCU 1**.

Las operaciones son las siguientes:

Operación Aritmética	Descripción
+ F	Se suma el contenido de los dos <b>ACCU</b>
- F	El <b>ACCU 2</b> se resta del <b>ACCU 1</b>

Tabla 8.1 Funciones aritméticas

Antes de ejecutar las operaciones aritméticas es necesario cargar en los **ACCU** los valores de los números con los que se desee realizar la operación.

Las operaciones aritméticas se ejecutan con independencia del **RLO** (Resultado Lógico de la Operación) y no lo afectan. El resultado está disponible en el **ACCU 1** para seguir utilizándolo en el programa. No se modifica el contenido del **ACCU 2**.

### 8.7 Operaciones de llamada de bloque

Estas operaciones permiten fijar a secuencia de un programa estructurado. Las operaciones de llamada de bloque, también conocidas como operaciones de salto, son dos:

- **JUMP UNCONDITIONAL (JU)** Salto absoluto incondicional. La ejecución del programa continúa en otro bloque, con la independencia total del **RLO** (Resultado Lógico de la Operación).
- **JUMP CONDITIONAL (JC)** Salto condicionado. Con **RLO** (Resultado Lógico de la Operación) = 1 se salta a otro bloque. De no ser así, el programa se sigue ejecutando en el mismo bloque, y el **RLO** (Resultado Lógico de la Operación) se pone en 1. Con un salto condicional, no solo se puede pasar a otro bloque, sino también a otra parte del mismo programa (solo en **FB's**)

### 8.8 Operaciones de terminación de bloque

Estas instrucciones tienen la función de terminar la ejecución de un bloque. A continuación se resume las operaciones de terminación de bloque.

TESIS CON  
FALLA DE ORIGEN

Operación	Significado
<b>BE</b>	<b>Terminar bloque (fin de bloque)</b> BE es siempre la última instrucción de un bloque. Se ejecuta con independencia del RLO (Resultado Lógico de la Operación)
<b>BEU</b>	<b>Terminar bloque de forma absoluta (Incondicional)</b> Con independencia del RLO (Resultado Lógico de la Operación) se finaliza el bloque actual
<b>BEC</b>	<b>Terminar bloque de forma condicional</b> <ul style="list-style-type: none"> <li>▪ Con RLO (Resultado Lógico de la Operación) = 1 se finaliza el bloque actual. Al cambiar de bloque no varía el RLO, sigue siendo 1.</li> <li>▪ Con RLO (Resultado Lógico de la Operación) = 0 no se ejecuta la operación. El RLO se pone a 1 y el programa sigue ejecutándose linealmente.</li> </ul>

Tabla 8.2 Operaciones de terminación de bloque

**NOTA:** Estas operaciones sólo pueden programarse en lista de instrucciones (STL).

### 8.9 Operación Stop "STP"

En el momento que esta operación se ejecute, el autómatas pasa a "STOP". La operación "STOP" se ejecuta con independencia del RLO (Resultado Lógico de la Operación).

TESIS CON  
FALLA DE ORIGEN

## CAPÍTULO 9

### Bloques funcionales (FB).

Un bloque de funciones es un bloque de programación **STEP 5** semejante al **OB** o al **FB**. Mientras que éstos solamente contienen operaciones básicas de las instrucciones **STEP 5**, los bloques funcionales **FB** contienen:

- Operaciones básicas
- Operaciones complementarias
- Operaciones del sistema

En un bloque funcional se dispone de un conjunto de operaciones ampliado. Cada **FB** se encuentra una única vez en la memoria del **PC** pudiendo ser llamado una o varias veces por otros bloques. La llamada de un **FB** puede ser en forma absoluta (**JU**) o condicionada (**JC**)

**Nota:** Los **FB's** sólo pueden programarse en **Lista de Instrucciones (STL)**.

#### 9.1 Estructura de un bloque de funciones (FB)

Un bloque funcional se compone de tres partes:

- Cabecera de bloque (**FV**)
- Cabeza de bloque (al igual que todos los bloques)
- Cuerpo del bloque

##### 9.1.1 Cabecera de bloque

La cabecera de bloque contiene las indicaciones de marcas de salto que han sido introducidas en ese bloque. La cabecera del bloque es generada automáticamente en la captación del bloque en el archivo de programa (previamente ajustado) como **FV**.

Para cada **FB**, se genera un **FV**. Al borrar un **FB** se borra también la correspondiente cabecera de bloque (**FV**).

Si al transferir un **FB** de la memoria del **PC** a disco, no existe la cabecera del bloque en el archivo de programa previamente ajustado, aparece un aviso que nos informa que no existe cabecera de bloque.

##### 9.1.2 Cabeza de bloque

En la cabeza del bloque se depositan:

- Tipo y número de bloque
- Número de biblioteca
- Tipo de operando de los operandos formales
- Longitud del bloque

**TESIS CON  
FALLA DE ORIGEN**

### 9.1.3 Cuerpo del bloque

En el cuerpo del bloque se incluye una lista del nombre del **FB** y los nombres de los operandos formales de todos los segmentos del bloque.

En el cuerpo del bloque está también el programa **STEP 5** del **FB** que se ha introducido en Lista de instrucciones (**STL**). Las designaciones (operandos formales) se introducen en el cuerpo del bloque con el signo igual por delante.

### 9.2 Creación de un FB

A diferencia de los otros bloques, un **F** es parametrizable. Los operandos de todos los segmentos del **FB** deben de estar depositados en la lista de operandos.

Al realizar la programación de un **FB** se determina la función del bloque, con la que los operandos introducidos son llamados **operandos formales**, los cuales realizan sustitución.

Al realizar la llamada de un **FB**, los operandos formales son sustituidos por los operandos actuales, es decir, la función del bloque es **parametrizada**.

Para la parametrización es obligatorio programar los siguientes datos relativos a los operandos del bloque:

**NOMBRE (NAME):**-El nombre del **FB** es arbitrario, el cual sólo admite 8 caracteres, debiendo ser el primer carácter una letra.

**DESIGNACIÓN DE OPERANDOS FORMALES (DECL):**- Cada operando del bloque recibe una designación (**DECL**) bajo la cual el operando formal es sustituido por el operando actual al llamar el **FB**.

El nombre del operando formal puede tener una longitud de 4 caracteres y debe comenzar con una letra. En cada bloque funcional es posible programar hasta 40 operandos formales.

Una vez introducido el nombre del operando formal se debe indicar el tipo de operando. Son posibles los siguientes tipos de operandos:

**I / Q / D / B / T / C:**

Donde:

**I** Operando de entrada  
**Q** Operando de salida  
**D** Datos  
**B** Bloques  
**T** Temporizadores  
**C** Contadores

En la representación gráfica, los operandos de salida se visualizan a la derecha del símbolo del **FB**. Los restantes operandos se encuentran a la izquierda.

TESIS CON  
 FALLA DE ORIGEN

Una vez seleccionado el tipo de operando, se debe indicar el formato del operando. Se puede seleccionar entre los siguientes formatos:

Donde:

**BI / BY / W / D:**

**BI** para operandos "bit"

**BY** para operandos "byte"

**W** para operandos "word" (palabra)

**D** para operandos "double word" (doble palabra)

**Nota:** Para finalizar la lista de operandos solo se debe accionar una vez más la tecla **enter**. El cursor salta a la primera línea del cuerpo del bloque.

Una vez terminada la lista de operandos formales se introducen las instrucciones **STEP 5** en Lista de instrucciones (STL). Después de la entrada de la última instrucción se puede memorizar el bloque accionando la tecla de aceptación ( $\overline{\text{ins}}$ ). Otra forma de finalizar y memorizar el bloque se logra introduciendo los caracteres "BE" y accionar a continuación la tecla **enter**.

### 9.3 Programación de un bloque de funciones (FB)

**Nota:** **FB 0** no se puede programar debido a que es propio del sistema.

Se va a programar el bloque de funciones **FB 1** con un segmento.

1.- Llamar a la función **INPUT / BLOCK** y rellenar la línea de comando:

- A partir de la máscara de "SELECT FUNCTION" se presiona **F1 (INPUT)** y enseguida **F1 (BLOCK)**. Aparece la siguiente línea de comandos:

**INPUT DEVICE:**

**BLOCK:**

- Se rellena la línea de comandos con los siguientes caracteres:

**FD:** Se refiere al archivo de programa ajustado previamente

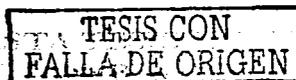
**FB1:** Ésta es la designación del bloque

**INPUT DEVICE: FD**

**BLOCK: FB1**

2.- Se acciona la tecla de aceptación (la tecla  $\overline{\text{ins}}$  de la zona numérica del teclado)

- En la pantalla aparece el editor de bloques de funciones:





Para finalizar la lista de operandos accione la tecla *enter*.

El cursor salta a la primera línea del cuerpo del bloque. El cursor está en la primera línea de instrucción. Introducir las siguientes instrucciones:

```
:A =OL
:A =PARO
:A(
:O =ARRA
:O =MOTO
:)
:= =MOTO
:BE
```

El bloque ha sido terminado y memorizado en disco.

Se visualiza la siguiente pantalla:

```
FB 1                                B:FUNCTBST.S5D                                LEN=0
SEGMENTO 1                            INPUT
NAME:PRUEBA
DECL:OL      I/Q/D/B/T/C:I      BI/BY/W/D:BI
DECL:PARO    I/Q/D/B/T/C:I      BI/BY/W/D:BI
DECL:ARRA    I/Q/D/B/T/C:I      BI/BY/W/D:BI
DECL:MOTO    I/Q/D/B/T/C:Q      BI/BY/W/D:BI

:A =OL
:A =PARO
:A(
:O =ARRA
:O =MOTO
:)
:= =MOTO
:BE

INPUT DEVICE: FD      BLOCK: FB 25      3% ( )
```

Fig. 9.2 Programación del FB 25

#### 9.4 Llamada de un bloque funcional (FB)

Los bloques funcionales se almacenan, como los otros bloques, en la memoria del programador (PC) bajo un determinado número (por ejemplo FB 25). Los números 246 al 255 están reservados para los FB's integrados.

TESIS CON  
FALLA DE ORIGEN

En casi todos los bloques, es posible programar llamadas de **FB's**. Una llamada se compone de:

- Instrucción de llamada (**JU** o **JC**)
- Lista de operandos formales (sólo necesario si en el **FB** han sido definidos operandos formales)

**Nota:** Sólo es posible llamar bloques funcionales que han sido ya programados.

Tan pronto como se haya programado una instrucción de llamada, se presenta en la pantalla la lista de operandos, siempre y cuando se hayan definido operandos en el **FB**.

La lista de operandos se compone de los nombres de los operandos formales, cada uno de ellos seguido de dos puntos (:).

Ahora se asigna a los operandos formales los denominados operandos actuales. Al llamar el **FB**, éstos sustituyen a los operandos formales, es decir, el **FB** trabaja con los operandos actuales.

### 9.5 Ejemplo de llamada y parametrización de un **FB**

El bloque de funciones **FB 1** es llamado en el **PB 10** y son introducidos los operandos actuales.

- **Punto de partida:** El bloque de funciones **FB 1** debe estar ya programado y en el mismo **Archivo de programa** donde se programe el bloque de programa **PB 10**.
- Programar en el bloque de programa **PB 10** las siguientes instrucciones:

```

PB 10                                FUNCTBST.SSD                        LEN=12
SEGMENT 1                             INPUT
:JU FB 1
NAME:PRUEBA
OL : I 32.0
PARO : I 32.1
ARRA : I 32.2
MOTO : Q 32.0
      :A I 32.3
      :A Q 32.0
      :BE

INPUT DEVICE: FD                      BLOCK: FB 1

```

Fig. 9.3 Parametrización del **FB 1**

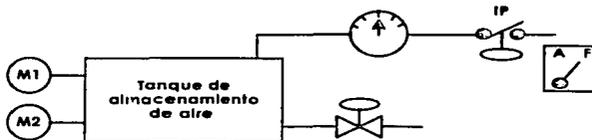
TESIS CON  
FALLA DE ORIGEN

## CAPÍTULO 10

### Algunos ejemplos de programación

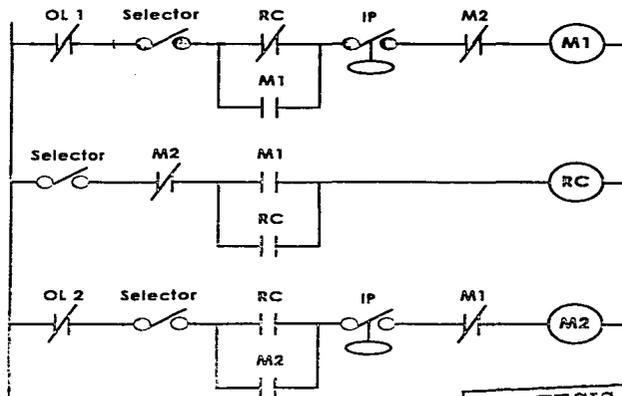
#### Control de dos compresores que suministran aire a un sistema neumático.

Diseñar el control para el siguiente sistema. Se tienen dos compresores que se encargan de suministrar aire a un sistema neumático.



- Los compresores no deben funcionar al mismo tiempo, su funcionamiento debe ser alternado.
- Se cuenta con un interruptor de presión para la conexión y desconexión automática de los compresores.
- Se cuenta con un selector para (A) y fuera (F).

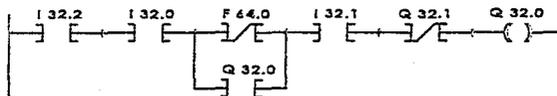
#### Diagrama de control convencional



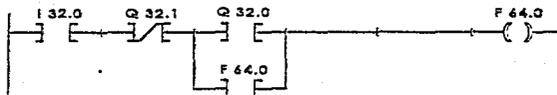
TESIS CON  
FALLA DE ORIGEN

## Programación en escalera

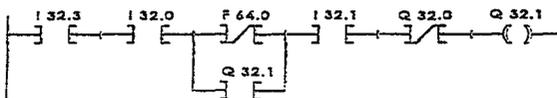
Segment 1



Segment 2

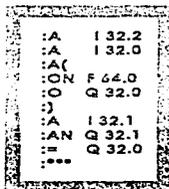


Segment 3

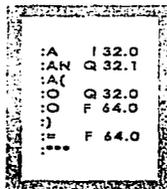


## Programación en lista de instrucciones

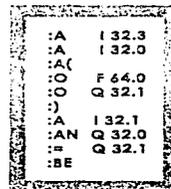
Segment 1



Segment 2



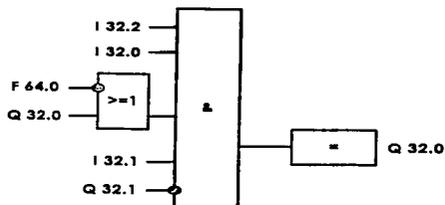
Segment 3



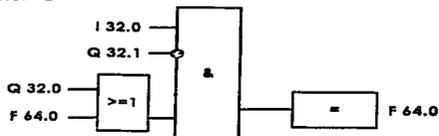
TESIS CON  
FALLA DE ORIGEN

## Programación en funciones lógicas

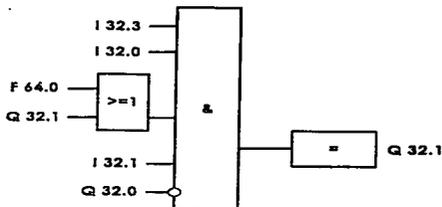
Segment 1



Segment 2

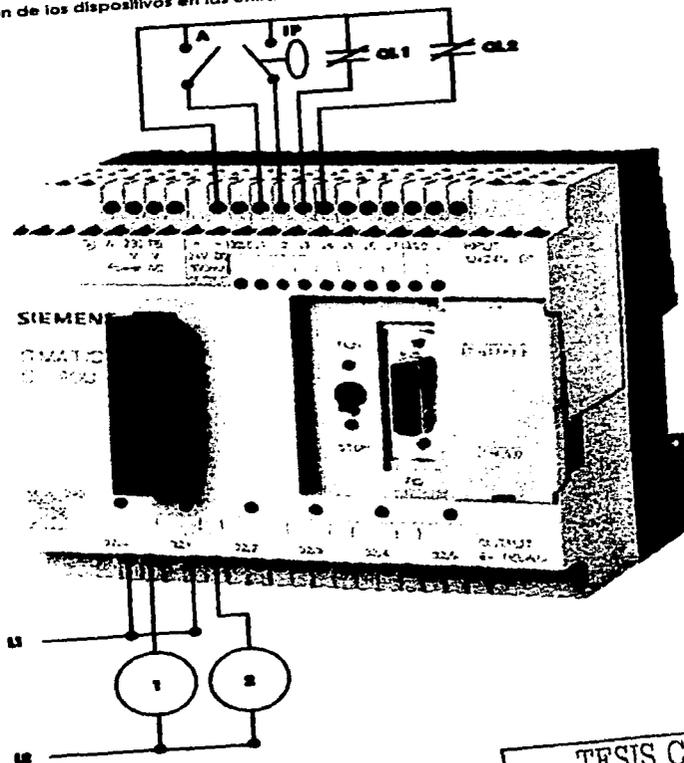


Segment 3



TESIS CON  
FALLA DE ORIGEN

Conexión de los dispositivos en las entradas y salidas



TESIS CON  
FALLA DE ORIGEN

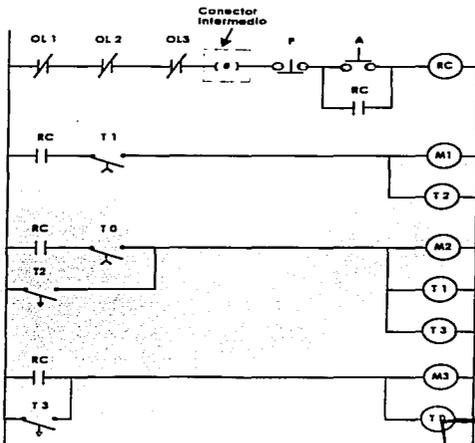
### Control para un sistema formado por tres transportadores.

Diseñar el programa de control para un sistema formado por tres transportadores.



- El sistema deberá comenzar a funcionar al pulsar el botón de arranque. Se comienza con la banda 3, cinco después la 2 y cinco segundos después la 1. Cada banda cuenta con su propio motor impulsor.
- El paro del sistema se inicia al pulsar el botón de paro. Se detiene primero la banda 1, cinco después la 2 y cinco segundos después la 3.
- En caso de sobrecarga en cualquiera de las 3 bandas, se debe detener el sistema.

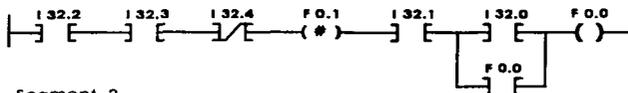
### Diagrama de control convencional



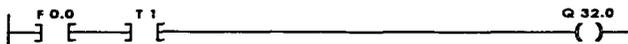
TESIS CON  
FALLA DE ORIGEN

### Programación en escalera

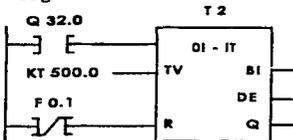
Segment 1



Segment 2



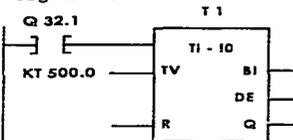
Segment 3



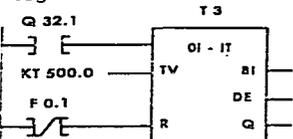
Segment 4



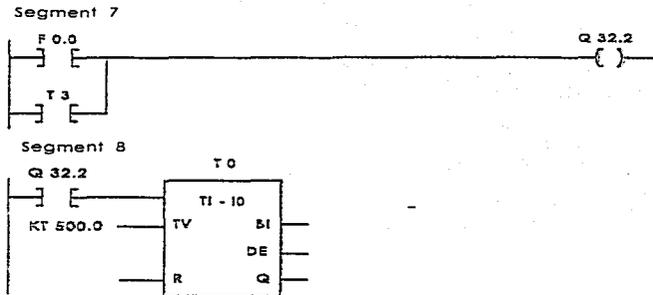
Segment 5



Segment 6



TESIS CON  
FALLA DE ORIGEN



### Programación en lista de instrucciones

Segment 1

```

:A 132.2
:A 132.3
:A 132.4
:T F 0.1
:A F 0.1
:A 132.1
:A(
:O 132.0
:O F 0.0
)
: F 0.0
***

```

Segment 2

```

:A F 0.0
:A T 1
:R Q 32.0
***

```

Segment 3

```

:A Q 32.0
:L KT 500.0
:SF T 2
:AN F 0.1
:R T 2
:NOP 0
:NOP 0
:NOP 0
***

```

Segment 4

```

:A F 0.0
:A T 0
:O T 2
:R Q 32.1
***

```

Segment 5

```

:A Q 32.1
:L KT 500.0
:SD T 1
:NOP 0
:NOP 0
:NOP 0
:NOP 0
***

```

Segment 6

```

:A Q 32.1
:L KT 500.0
:SF T 3
:AN F 0.1
:R T 3
:NOP 0
:NOP 0
:NOP 0
***

```

Segment 7

```

:O F 0.0
:O T 3
:R Q 32.2
***

```

Segment 8

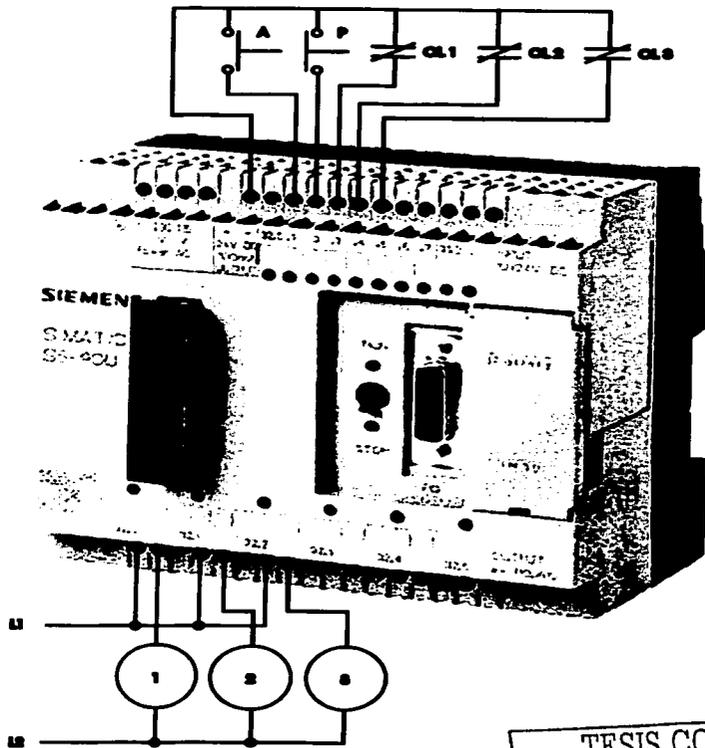
```

:A Q 32.2
:L KT 500.0
:SD T 0
:NOP 0
:NOP 0
:NOP 0
:NOP 0
:SE

```

**TESIS CON  
FALLA DE ORIGEN**

Conexión física de los dispositivos sensoriales y de salida.



TESIS CON  
FALLA DE ORIGEN

### Sincronización de dos semáforos.

Diseñar el programa de control para la sincronización de dos semáforos.

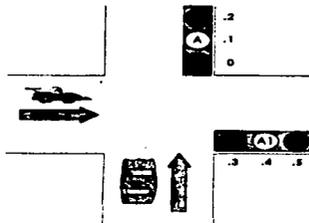
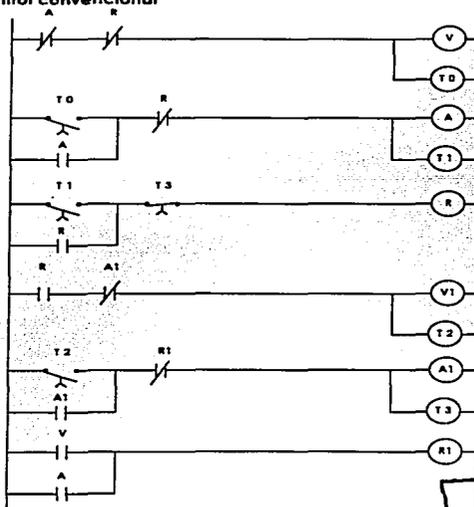


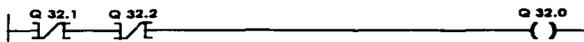
Diagrama de control convencional



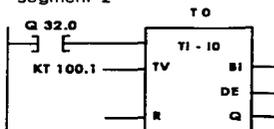
TESIS CON  
FALLA DE ORIGEN

### Programación en escalera

Segment 1



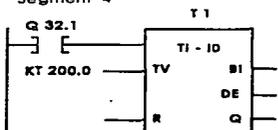
Segment 2



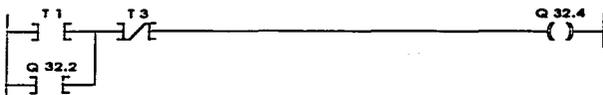
Segment 3



Segment 4



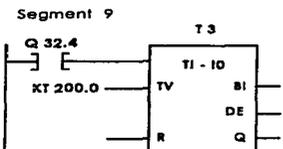
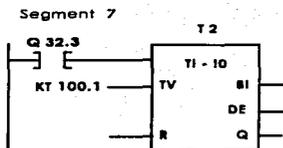
Segment 5



Segment 6



TESIS CON  
FALLA DE ORIGEN



TESIS CON  
FALLA DE ORIGEN

## Programación en lista de Instrucciones

Segment 1

```

:A Q 32.1
:L RT 200.0
:SD T 0
:NOP 0
:Q 32.0
:***

```

Segment 2

```

:A Q 32.0
:L RT 100.1
:SD T 0
:NOP 0
:NOP 0
:NOP 0
:***

```

Segment 3

```

:(
:O T 0
:O Q 32.1
)
:AN Q 32.2
:= Q 32.1
:***

```

Segment 4

```

:A Q 32.1
:L RT 200.0
:SD T 1
:NOP 0
:NOP 0
:NOP 0
:***

```

Segment 5

```

:(
:O T 1
:O Q 32.2
)
:ANT 3
:= Q 32.2
:***

```

Segment 6

```

:A Q 32.2
:AN Q 32.4
:= Q 32.3
:***

```

Segment 7

```

:A Q 32.3
:L RT 100.1
:SD T 2
:NOP 0
:NOP 0
:NOP 0
:***

```

Segment 8

```

:(
:O T 2
:O Q 32.4
)
:AN Q 32.5
:= Q 32.4
:***

```

Segment 9

```

:A Q 32.4
:L RT 200.0
:SD T 3
:NOP 0
:NOP 0
:NOP 0
:***

```

Segment 10

```

:OO 000 32.0
:EE 000 32.1
:EE 000 32.5

```

TESIS CON  
FALLA DE ORIGEN

## Conclusiones

---

---

El objetivo que se planteo en este trabajo se concluyó satisfactoriamente, dado que se dio una visión amplia de la programación en STEP 5, además de conocer las características de los diferentes PLC's de la familia SIMATIC S5.

Como se pudo observar la programación del STEP 5 permite tener las bases para operar otros softwares de programación; si se conocen los principios básicos de operación de los PLC's, así como también las características propias del PLC que se requiera utilizar en la automatización, se podrán ahorrar costos, además de llevar a cabo una eficiencia y rapidez en los sistemas de control.

Los sistemas de control automático son sistemas dinámicos y un conocimiento de la teoría de control como se vio proporcionan una base para entender el comportamiento de dichos sistemas. El STEP 5 constituye el medio de entendimiento entre el usuario y el sistema de automatización.

Si algún usuario está familiarizado con otro tipo de software de programación de PLC, que no sea el STEP 5, esta investigación le ayudará rápidamente a programar PLC's de la familia SIMATIC S5.

Debido a que la mayoría de las empresas han integrando la automatización para hacer cada vez más eficiente la producción en la industria, es importante que los egresados de ingeniería de la FES Cuautlián, tengan los conocimientos básicos para programar un PLC, porque como se observó al conocer el lenguaje de programación de STEP 5 será más fácil programar otros PLC's de otros fabricantes, debido a que se tienen los mismos principios básicos.

TESIS CON  
FALLA DE ORIGEN

### Abreviaturas

PLC	Controlador Lógico Programable (Programmable Logic Controller)
CPU	Unidad Central de Proceso
LAD	Diagrama de Contactos
CSF	Bloques de funciones
STL	Lista de Instrucciones
PB	Bloque de programa
FB	Bloques de funciones
OB	Bloques de Organización
SB	Bloque de Secuencia
DB	Bloque de Datos

TESIS CON  
FALLA DE ORIGEN

## Bibliografía

---

---

Balcells Josep Romeral, José Luis. *Autómatas Programables*, Alfa omega, México, 1998.

Martínez Sánchez, Victoriano A. *Automatización con Autómatas Programables*. ra-ma, España, 1991.

Mayol i Badía, Albert, *Autómatas Programables*. Marcombo, España, 1992.

Simón, Andre, *Autómatas Programables. Programación. Automatismo y Lógica programada*, Paraninfo, España, 1988

### Revistas:

Waller, Siegfried, "Estrategias para automatizar la producción" en revista Siemens, Año 47, Abril/Junio, República Federal de Alemania, 1987, pp. 25-28.

Manual de Siemens

### Internet:

"[www.siemens.com](http://www.siemens.com)"

"[www.altavista.com.mx/web/results?q=SIMATIC+S5-155U&kgs=0&kls=0&avkw=aoapt](http://www.altavista.com.mx/web/results?q=SIMATIC+S5-155U&kgs=0&kls=0&avkw=aoapt)"

"[www.digidome.nl/siemens.htm](http://www.digidome.nl/siemens.htm)"

"[www.sigmaautomation.net/Simatic%20S5.htm](http://www.sigmaautomation.net/Simatic%20S5.htm)"

TESIS CON  
FALLA DE ORIGEN