PROGRAMACIÓN DEL AUTÓMATA

Tiene una serie de pasos:

- Determinar qué debe hacer el sistema de control y en qué orden
- Identificar entradas y salidas al autómata
- Representar mediante un modelo el sistema de control, funciones, relaciones entre ellas, y secuencia que deben seguir
- Asignar direcciones de entrada, salida o internas a cada componente que aparece en el sistema
- Llevar la representación anterior a un lenguaje de autómata programable
- Depurar, simular y transferir a la memoria del autómata el programa

REPRESENTACIÓN DEL SISTEMA DE CONTROL

La complejidad de los automatismos y la necesidad de especificar con precisión las tareas => **útiles simbólicos de representación**

Deben ser:
Comunes para emisor y receptor (comprensibles por ambos)
De empleo coherente (reglas de sintaxis establecidas)

Permiten formar un modelo intermedio del sistema para análisis de funcionamiento y síntesis de la solución

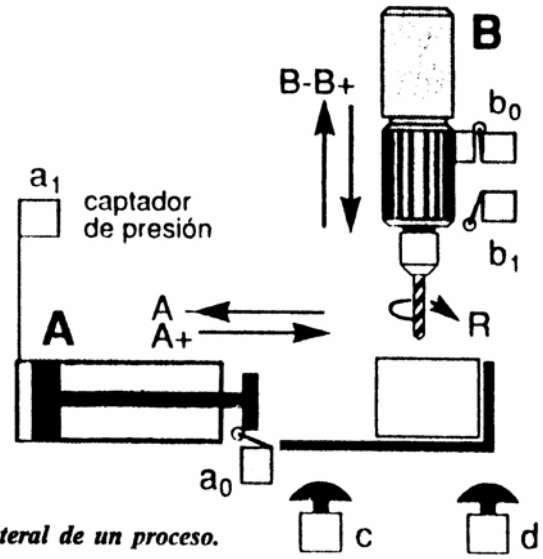
Clasificación del modelo según los símbolos utilizados:
Proposicional: *Descripciones literales*
Algebraico: *Funciones booleanas y algebraicas*
Gráfico: *Esquemas de relés, diagramas lógicos, de flujo y técnicas de Grafcet.-*

Descripciones literales

La descripción del proceso y el automatismo se hace por enumeración literal de las acciones, expuestas secuencialmente y con las condiciones de habilitación en cada caso
Fácil de realizar (se requiere poca calificación)
Poca precisión (suelen faltar especificaciones del proceso, variables e interacción entre ellas)

El operador pulsa los mandos manuales c y d y se reanuda el ciclo:

- apriete de la pieza por medio del cilindro A;
- verificación de este apriete por la presión comprobada por el manómetro a_1;
- bajada de la cabeza de taladrado B y rotación R del taladro;
- fin de la perforación controlada por el tope b_1;
- subida de B y parada de R;
- cuando se acciona b_0, afloje de la pieza.



Descripción literal de un proceso.

Del ejemplo de la figura se ve:
Es incompleto para efectuar el sistema de control
Una descripción más exhaustiva lo hará poco legible

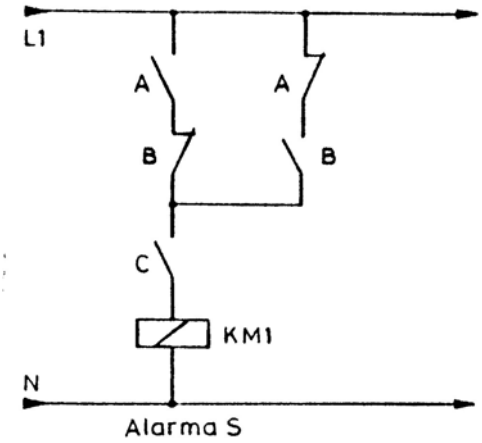
Funciones algebraicas

Acción de mando resultado de una función algebraica a partir de:
Especificaciones del cliente
Métodos de síntesis del Álgebra de Boole

Ejemplo: Alarma **S** activa cuando **C** esté cerrado y los contactos **A** y **B** estén en estados opuestos
$S = (A \bar{B} + \bar{A} B) C$
Se expande definiendo operaciones entre variables de varios bits (operaciones aritméticas, de comparación, etc)
Representan sistemas combinacionales y secuenciales
Uso limitado en sistemas secuenciales (análisis y síntesis de difícil ejecución)

Esquemas de relés

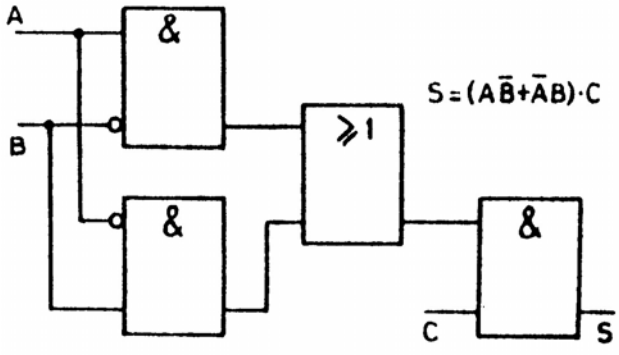
Representación gráfica mediante símbolos de contactos abierto – cerrado
La función de control depende de la conexión entre los distintos contactos
(Ver ejemplo anterior en figura)



Origen en las tecnologías electromecánicas de los sistemas de control
Son deficientes para describir sistemas secuenciales complejos y señales digitales de varios bits
Uso difundido en sistemas combinacionales y secuenciales sencillos por su familiaridad (electricistas)

Diagramas lógicos

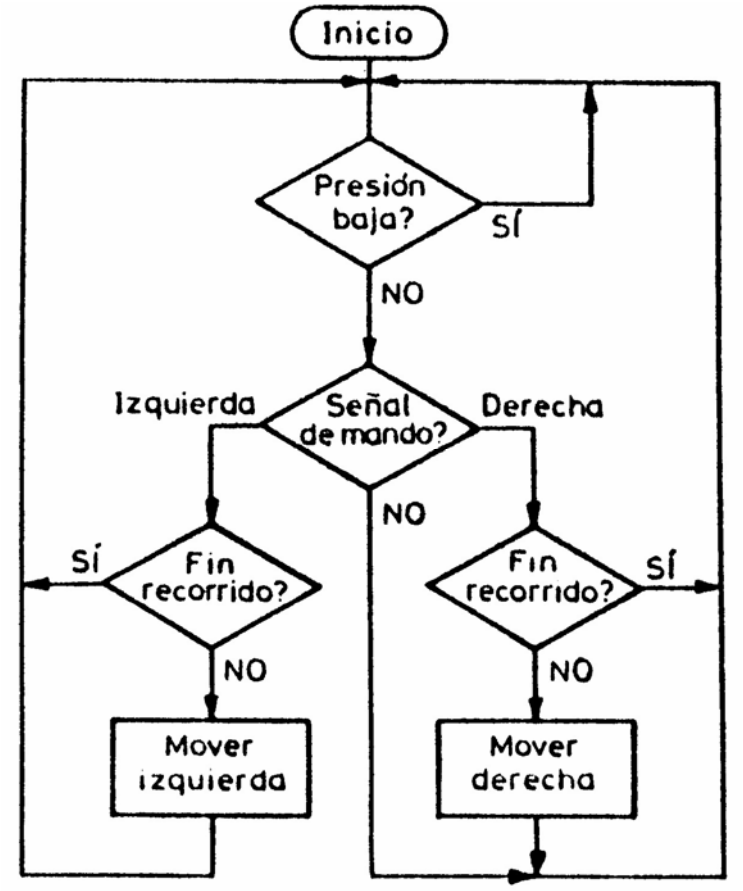
Representación gráficas mediante símbolos normalizados que representan funciones directas del Álgebra de Boole (or, and, etc)
Ejemplo de la alarma



Representación compacta e independiente de la tecnología de implementación del sistema de control (eléctrica, neumática, etc)

Diagramas de flujo

Representación gráfica útil para describir secuencias de evolución y toma de decisiones
Muy utilizado en fases iniciales del diseño

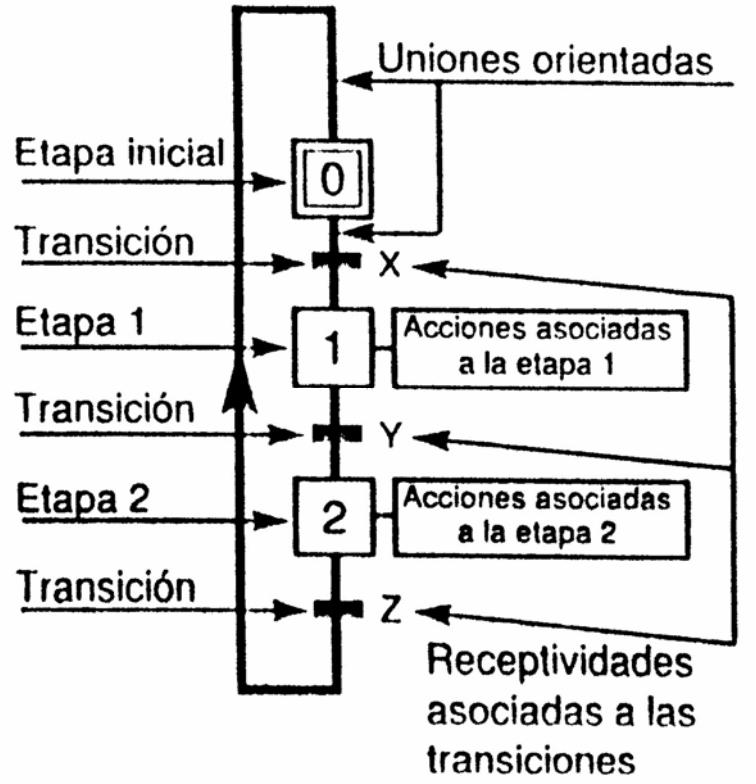


Ejemplo en de un posicionamiento hidráulico manual

Son claros para describir el funcionamiento general no para representar las variables señales que intervienen y las relaciones entre ellas a no ser que el modelo se complete con expresiones algebraicas que restan claridad inicial.

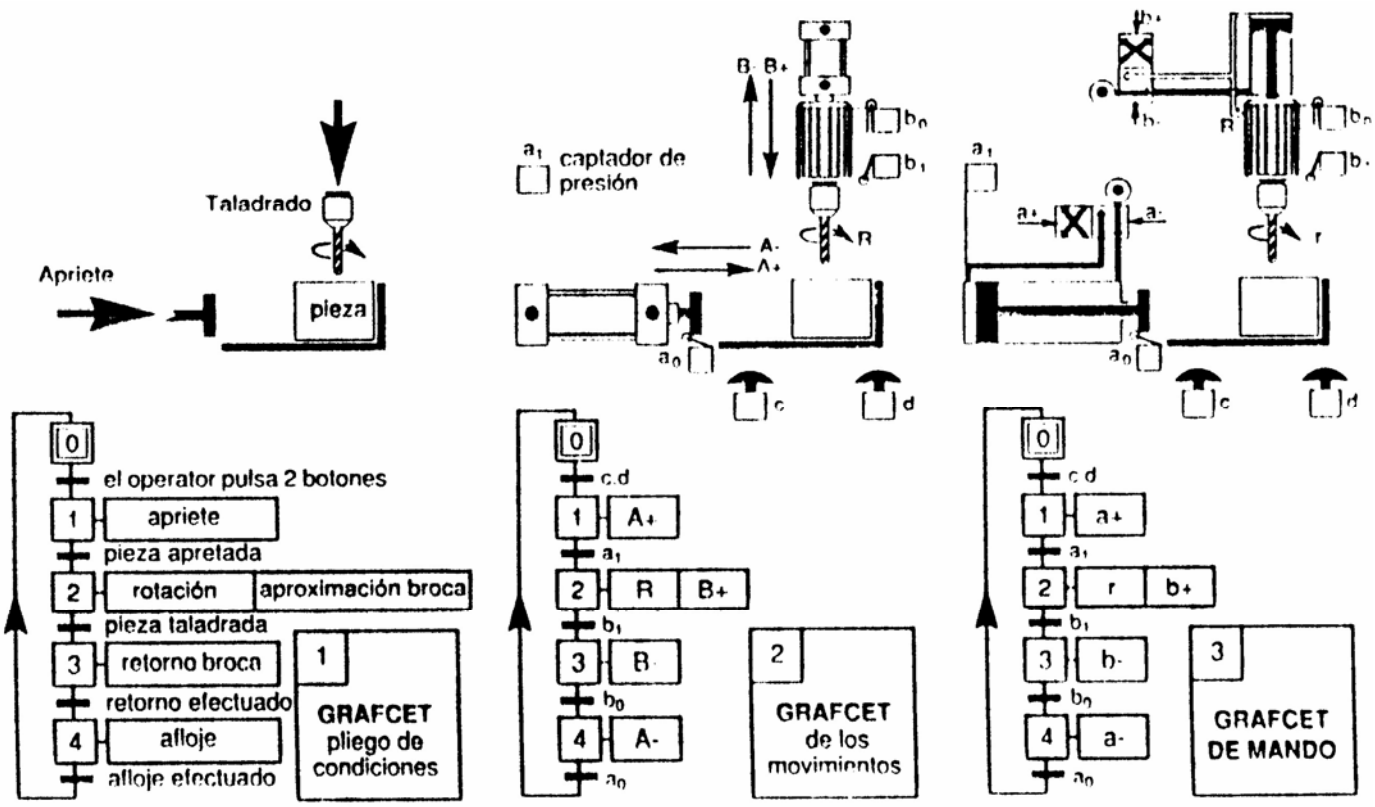
Representación Grafcet (Grafico de **C**omando **E**tapa **T**ransición)

Combina las ventajas de la representación secuencial gráfica con los modelos preexistentes
Normalizado por **IEC 848** (Internacional Electrotechnical Commission)
Representa las **etapas** de un proceso productivo, con las **transiciones** (condiciones) para pasar de una a otra



En una etapa activa el control:
Ejecuta la función de mando correspondiente
Consulta las condiciones de transición

Suele utilizarse en **todas las fases** del diseño:
Especificaciones de funcionamiento o GRAFCET del pliego de condiciones
Especificaciones tecnológicas o GRAFCET de movimientos
Programación del autómata o GRAFCET de mando



IDENTIFICACIÓN DE VARIABLES Y ASIGNACIÓN DE DIRECCIONES

Con la descripción funcional del sistema => Se determinan las señales de entrada y salida, que están involucradas.
Identificación y referenciación de **entradas / salidas** y variables internas

	ELEMENTO	REFERENCIA	DIRECCIÓN
SALIDAS	Motor 1	MOT 1	0101
	Motor 2	MOT 2	0102
	Electroválvula 1	EV 1	0103
	Piloto	PLT	0104
	Consigna de velocidad	CVEL	CH3 | 0300 / 0315 (16 puntos)
ENTRADAS	Pulsador marcha	MARCHA	0001
	Pulsador paro	PARO	0002
	Fotocélula A	FOTO A	0003
	Detector 1	INDU 1	0004
	Báscula 1	PESO 1	CH2 | 0200 / 0215 (16 puntos)
INTERNAS	Condición tiempo	MTM 1	1000
	Temporizador 1	TIM 1	TIM 00
	Preselección TIM 1	PTIM 1	DM00
	Alarma 1	ALM 1	1001
	Reloj interno	CLK	1902

Tabla 10.4. Ejemplo de asignación de direcciones.

Las variables internas que según la función serán:

Variables de usuario: accesibles de la unidad de programación representan parámetros que necesita el programa

Variables de cálculo intermedio o memoria

Variables de consulta de estados: variables auxiliares definidas por el fabricante que reflejan estados internos del autómata
Relés internos (1 bit)
Temporizadores, contadores (varios bits)

Identificadas las **entradas / salidas** y las **variables internas** => **asignación de direcciones** (Borneras de E/S y direcciones de memoria interna)

Direcciones absolutas: Siempre la misma ubicación (E/S autómatas compactos y vbles. internas)
Dirección en un solo campo: posición de la bornera o memoria

Direcciones relativas: Según la ubicación del módulo que la contiene (E/S autómatas modulares)
Dirección dos campos:
Dirección del módulo sobre el bastidor
Dirección el borne en el módulo
Ej.: - 5.7 punto 7 dentro del módulo 5
007 punto 7 dentro del módulo 0 (TI305)

LENGUAJES DE PROGRAMACIÓN

Son dependientes del autómata empleado
Tipo de unidad o software de programación: Literal o gráfica

Son similares a los modelos de representación (facilidad en la trascripción)

Clasificación:

Algebraicos:
Lenguajes Booleanos
Lista de instrucciones
Lenguajes de alto nivel

Gráficos:
Diagrama de contactos
Diagrama de funciones / bloques
Intérprete GRAFCET

LENGUAJES BOOLEANOS Y LISTA DE INSTRUCCIONES

Formato de las instrucciones

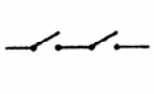
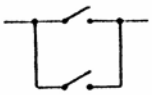
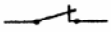
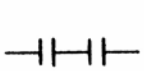
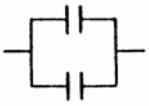
Campo operación	Operando

Instrucciones típicas básicas

OR AND NOT	Booleanas
LD OUT SET RST	Carga y asignación
TIM CNT	Contador y temporizador
ADD SBB MUL DIV	Aritméticas
CMP SHIFT MOV	Manejo de datos
END JMP MCS	Gestión de programa

DIAGRAMAS DE CONTACTOS (Ladder Diagram)

El lenguaje de contactos expresa las relaciones entre señales binarias como una sucesión de contactos en serie y en paralelo

FUNCIÓN LÓGICA	Operación producto lógico	Operación suma lógica	Operación negación	Asignación de valor
INSTRUCCIONES BOOLEANAS	AND	OR	NOT	OUT
ESQUEMAS DE RELÉS (DIN 40713-16)				
DIAGRAMAS DE CONTACTOS (NEMA/DIN 19239)				

Adoptado por muchos fabricantes de autómatas (norteamericanos y japoneses) como lenguaje base de programación

Contactos de relés => componentes de dos estados (0: contacto abierto y 1: contacto cerrado)

Álgebra de Boole con contactos: cualquier función lógica puede ser transcrita directa e inmediatamente a diagrama de contactos y viceversa

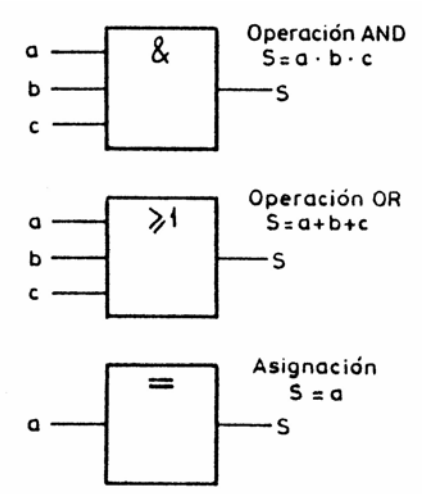
Incluyen **bloques funcionales**: temporizadores y contadores

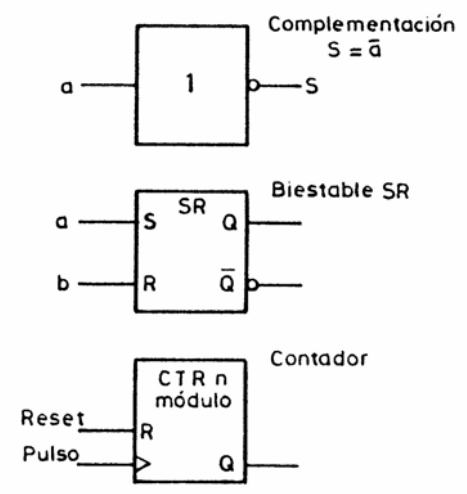
Incluyen **bloques funcionales complejos**: para la manipulación de datos y variables digitales de varios bits

Como en las extensiones al lenguaje booleano no todos los modelos acceden a la totalidad de extensiones del lenguaje

DIAGRAMA DE FUNCIONES

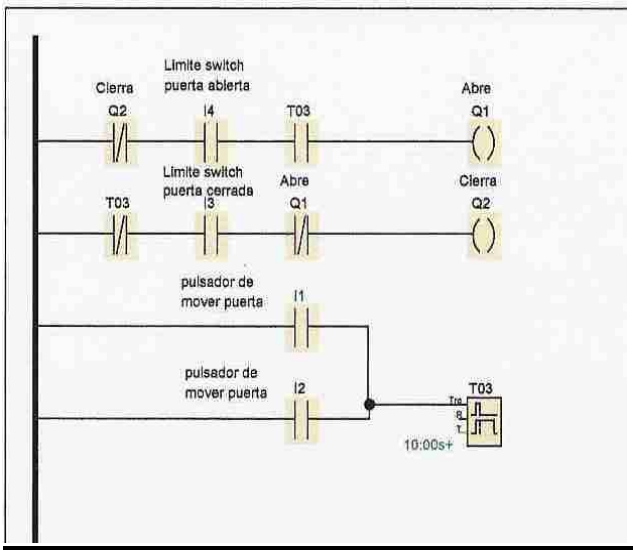
El diagrama lógico o de funciones es la representación de las tareas de automatización utilizando los símbolos contenidos en las normas DIN 40700 y DIN 40719

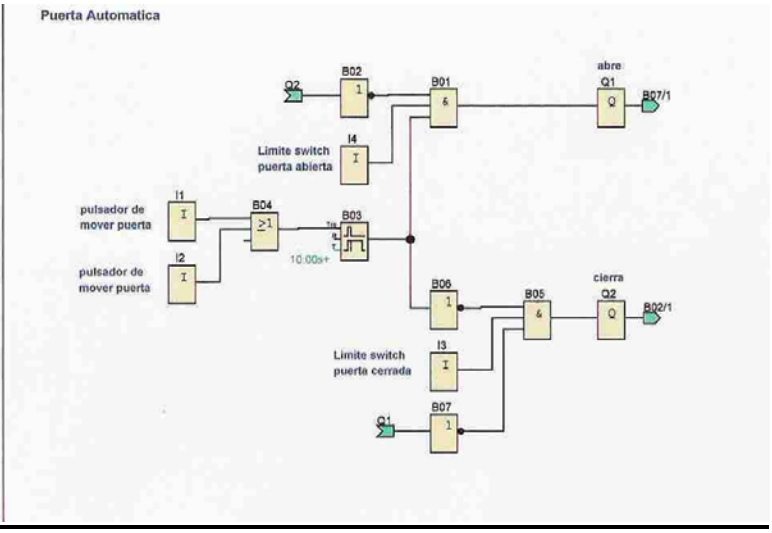




Incluye como bloques normalizados funciones secuenciales típicas y algunos bloques de tratamientos numéricos

Ejemplo: Puerta corrediza con dos pulsadores que sirven para cerrarla o abrirla en forma indistinta. Desde la posición de abierta hasta la posición de cerrada el mecanismo demora 10 segundos.





Ventajas e inconvenientes de los lenguajes vistos

Listado de instrucciones

Para PLC de funciones limitadas, las instrucciones son mnemónicos fáciles de entender.
La carga se realiza mediante teclados básicos que pueden estar en el PLC.
No requiere una interfaz gráfica.
Hay dificultad para interpretar la función de un listado de instrucciones.
Hay dificultad para realizar análisis y modificaciones.

Diagramas de contactos

Es de fácil aprendizaje para quien tiene conocimientos básicos de electricidad.
La interpretación y modificación del diagrama es sencilla.
Permite la incorporación de comentarios y referencias.
Requiere un recurso que pueda presentar gráficos.
Para la carga se requiere un software para convertir los diagramas en lenguaje de máquina.

Diagramas de funciones

La interpretación y modificación del diagrama es sencilla en programas chicos
Permite la incorporación de comentarios y referencias.
Permite la incorporación de comentarios y referencias.
Requiere un recurso que pueda presentar gráficos.
Para la carga se requiere un software para convertir los diagramas en lenguaje de máquina.

LENGUAJES DE ALTO NIVEL

Los autómatas de gamas altas realizan aplicaciones reservadas a las computadoras industriales
Los lenguajes tradicionales son insuficientes para estas aplicaciones => se utilizan leguajes informáticos tradicionales convenientemente adaptados
Basic (el más difundido), C, etc.
Disponen de *instrucciones de manipulación de cadenas de caracteres*
Constituyen posibilidades adicionales pero no el lenguaje básico (lista de instrucciones y diagramas de contactos)

Estructuras de programación

Introducción

Tarea: Se define como el conjunto de instrucciones ejecutables que describen un tratamiento limitado y completo sobre variables de proceso

Estructura monotarea: aplicación desarrollada sobre una tarea única, que contiene el total del programa con todas sus variables de entrada / salida y sentencias de operación

Estructura multitarea: aplicación que divide el programa en subconjuntos, independientes o no, que forman tareas aisladas, normalmente en correspondencia con tratamientos particulares de la aplicación.

Programa
- **Monotarea:** Un ciclo de operación
- **Multitarea:** Múltiples ciclos de operación, uno por tarea, de desarrollo simultaneo.Tareas periódicas o no

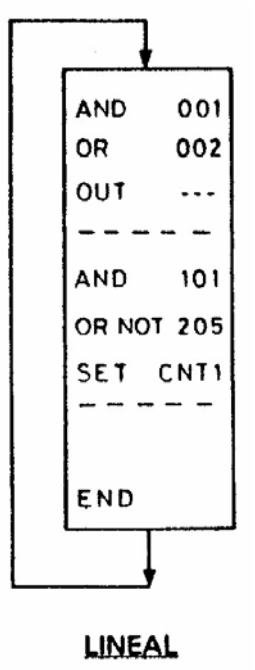
Tarea
- **Programación Lineal**
- **Programación estructurada**

Programación lineal

El problema de control se realiza escribiendo las instrucciones según una secuencia lineal, una tras otra desde la primera a la última.

La secuencia de ejecución se puede alterar mediante el empleo de instrucciones de modificación de ciclo.

El programa lineal se divide en partes o bloques de ejecución condicionada.



LINEAL

Dos tipos de instrucciones:

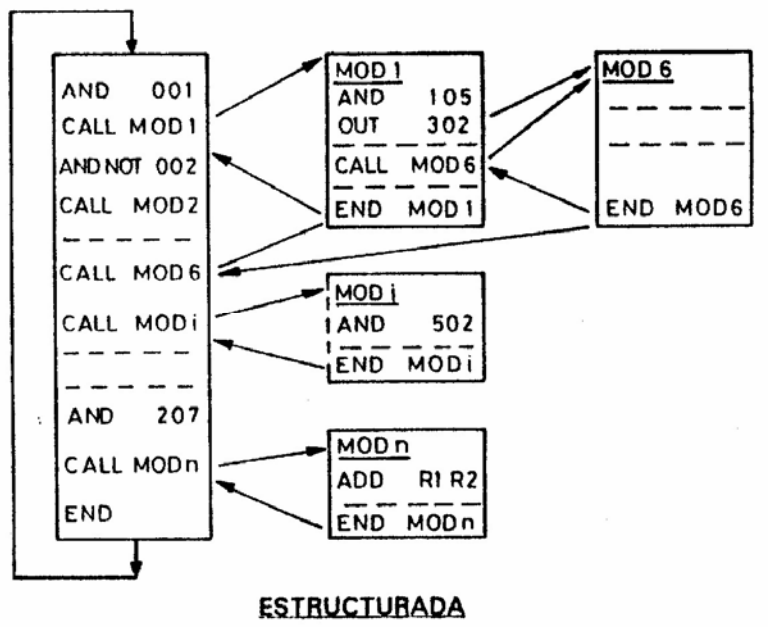
Saltos: incluyendo las sentencias de alto nivel IF... THEN... ELSE, GOTO...WHILE, etc.
Habilitación de bloques: Master Control Set / Master Control Reset, Block Program Pause, etc.

Programación estructurada

Se divide la tarea a programar en subprogramas o módulos, que corresponden a tratamientos parciales, y son llamados durante el escrutinio desde un programa raíz.

Estos bloques están diferenciados por el tipo de tratamiento que realizan.

El agrupamiento por tipo de instrucciones permiten optimizar el tiempo de ejecución, si los bloques son ejecutados sobre un sistema de coprocesadores o multiprocesadores especializados.

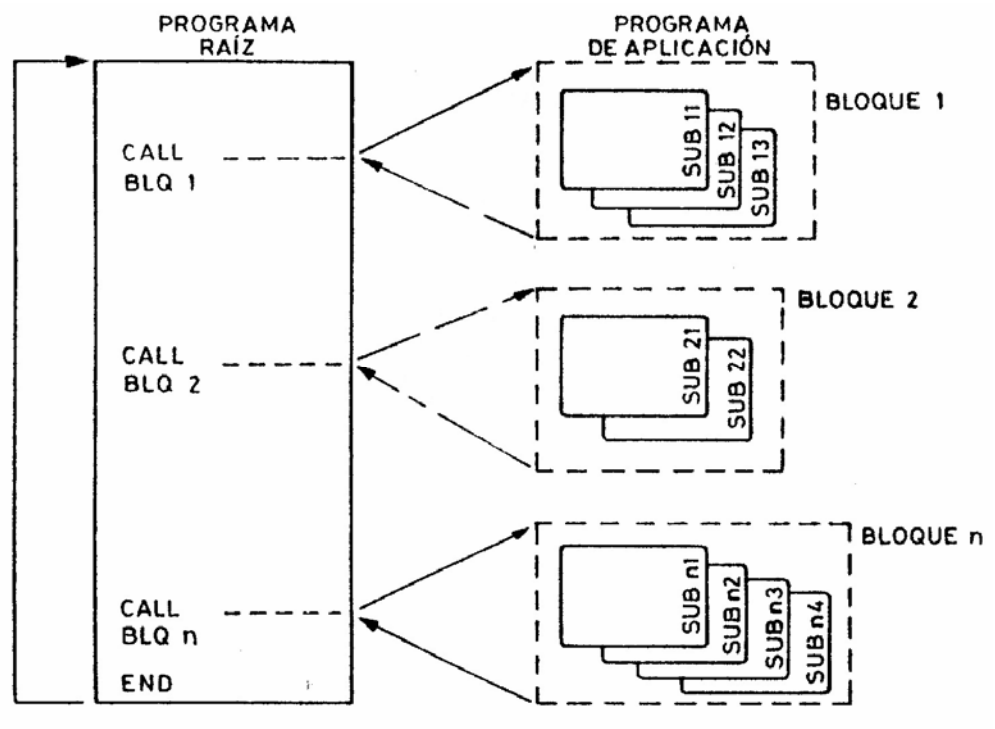


ESTRUCTURADA

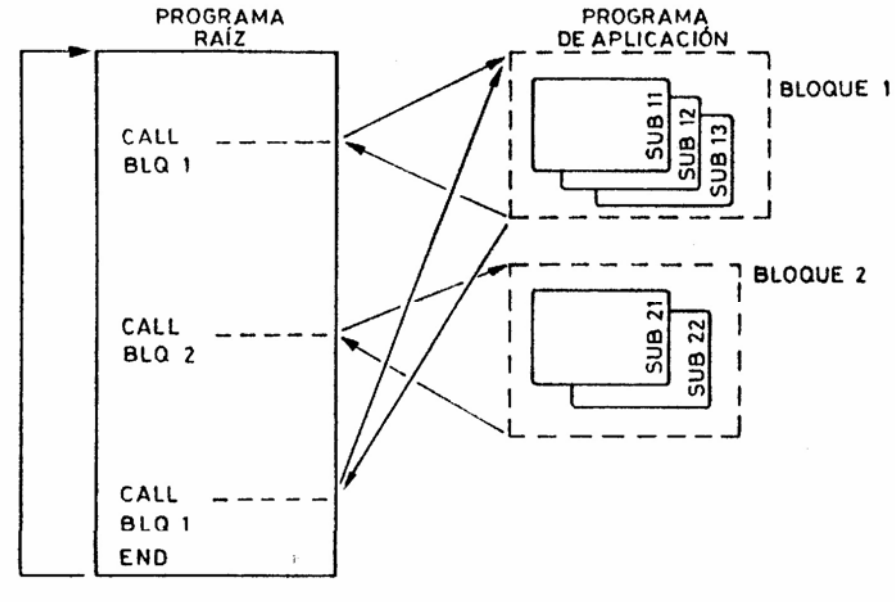
Programación estructurada → Modular

→ Subrutinas

Programación modular: La tarea está contenida en módulos independientes entre sí, cuya ejecución está organizada desde un módulo raíz, que básicamente contiene las llamadas, condicionales o no, a los módulos de programa.

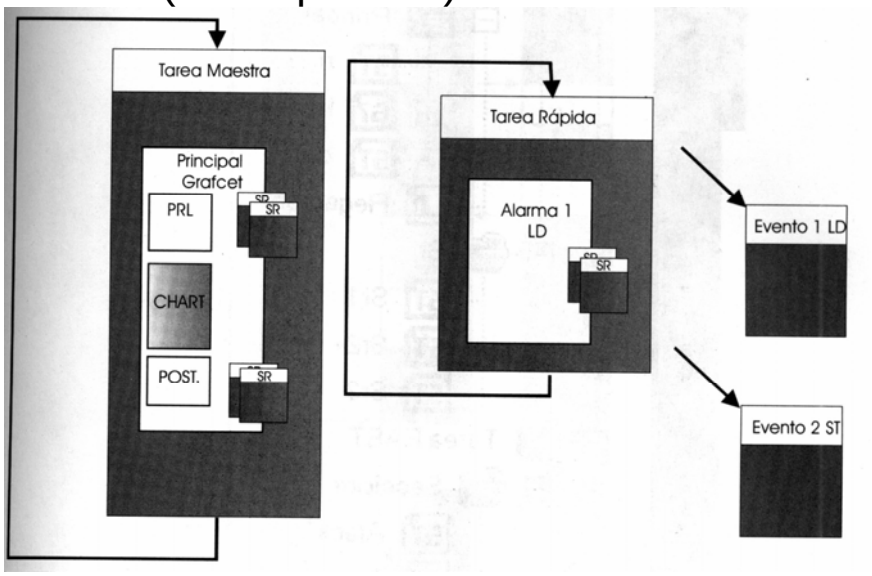


Uso de subrutinas: Bloques de programa de uso reiterado dentro de la ejecución; que son llamadas desde diferentes puntos del programa principal.



Estructuras multitarea

Tarea maestra: Lectura / escritura de E/S y programa de aplicación
Tarea rápida: De ejecución opcional, es periódica y permite la ejecución de programas muy cortos
Tareas de eventos: Tratamiento de eventos producidos por módulos de entrada (interrupciones)



Prioridades: La tarea maestra está siempre activa, la tarea rápida interrumpe a la maestra y la tarea de evento interrumpe a las anteriores.

Equipos y Software de Programación

Permiten realizar la programación y proveen utilitarios y funciones como:

Escribir y editar un programa en los lenguajes permitidos por el autómata

Simular el funcionamiento fuera del autómata o “en línea” pudiendo incluso forzar valores de las variables.

Brindan utilidades como leer programas, cambio de parámetros, etc.

Permiten presentaciones gráficas y colocación de comentarios para una mejor interpretación.