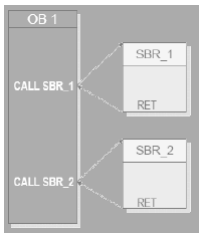
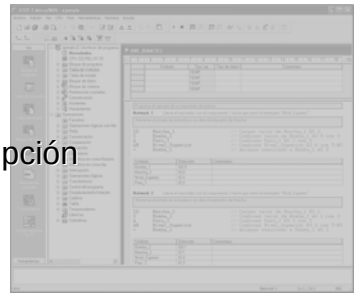
Programación de Autómatas



STEP 7
Subrutinas
Rutinas de Interrupción

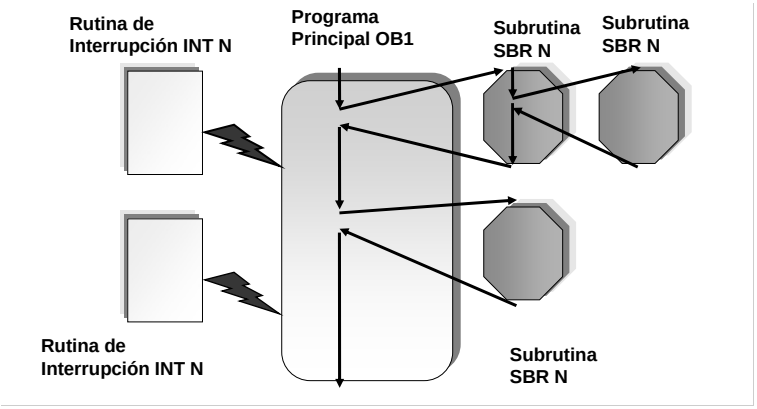


Indice

- Introducción
  - Subrutinas vs Rutinas de Interrupción
- Subrutinas
  - Tareas a realizar para utilizar una subrutina en el programa
  - Ejemplos de uso de Subrutinas
  - Parámetros y variables locales
- Rutinas de interrupción
- Ejemplos Rutinas de Interrupción
  - Ejemplo de interrupciones temporizadas
  - Ejemplo de tratamiento de Interrupciones de E/S

Introducción

- Subrutinas vs Rutinas de interrupción
  - Son bloques de organización del programa
  - Las rutinas de interrupción se ejecutan de forma independiente del ciclo del ejecución del autómata.



Subrutinas

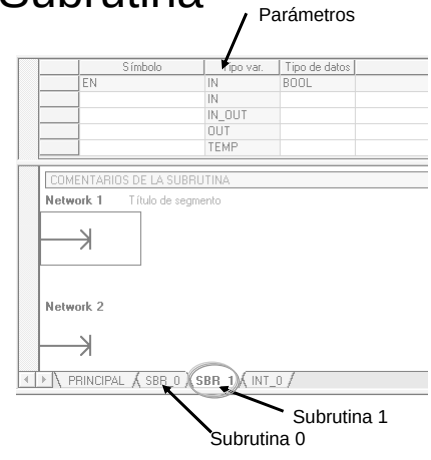
- Las subrutinas se utilizan para estructurar o dividir el programa en bloques más pequeños. Más fáciles de gestionar.
- Facilita las tareas de comprobación, eliminación de errores y mantenimiento del programa.
- La CPU también se puede utilizar más eficientemente, invocando el bloque sólo cuando se necesite, en vez de ejecutar todos los bloques en cada ciclo.
- Las subrutinas se pueden transportar si se hace referencia únicamente a sus parámetros y a su memoria local.
  - Para que una subrutina se pueda transportar, se debe evitar la utilización de variables/símbolos globales (direcciones absolutas en las áreas de memoria I, Q, M, SM, AI, AQ, V, T, C, S, AC).
  - Si la subrutina no tiene parámetros de llamada (IN, OUT ó IN_OUT), o si utiliza únicamente variables locales en la memoria L, la subrutina se puede exportar a e importar de un proyecto diferente.

Uso de Subrutinas

- Tareas a realizar para utilizar una subrutina en el programa
  - Crear la subrutina.
  - Definir los *parámetros* (en caso necesario) en la tabla de variables locales de la subrutina.
  - Llamar a la subrutina desde la unidad de organización del programa en cuestión (p.ej., desde el programa principal OB1 o desde otra subrutina).
    - CALL: realiza una llamada a la subrutina
    - RET (Return): termina la ejecución de la subrutina y devuelve el control (*el Editor KOP lo inserta automáticamente al final del esquema de la subrutina*)

Crear una Subrutina

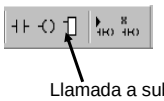
- En el menú Edición, elegir los comandos *Insertar* > *Subrutina*
- En la ventana del editor de programas, hacer clic con el botón derecho del ratón y elegir el comando *Insertar* > *Subrutina* del menú emergente.



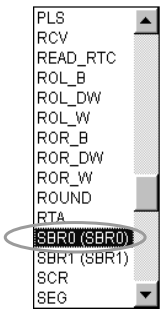
- El editor de programas cambia de la anterior unidad de organización del programa visualizada a la nueva subrutina. En el borde inferior del editor de programas aparece una nueva ficha correspondiente a la nueva subrutina

Llamar a una Subrutina

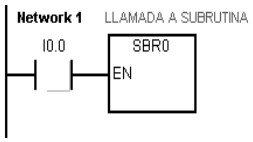
- Seleccionar insertar cuadro en el programa principal.



Llamada a subrutina SBR0



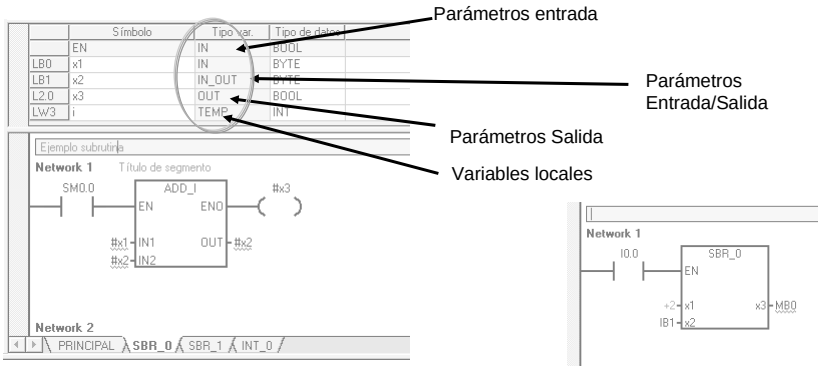
- Se insertara la siguiente línea en el programa



- La subrutina se ejecuta cuan la entrada EN está a nivel alto.

Subrutinas: Parámetros

- Parámetros de Entrada y Salida
  - EN señal de activación de la subrutina (cuando está a nivel alto se ejecuta el código de la subrutina)
  - Adicionalmente podemos definir parámetros de entrada y salida con su tipo en la región **de** memoria local (**L**) (*máximo 16 bytes*)

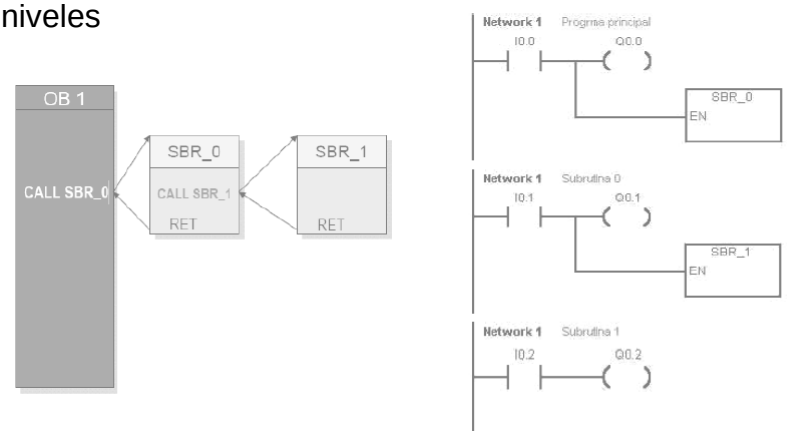


Subrutinas: Parámetros

- Parámetros de Entrada (IN)
  - Los parámetros se transfieren a la subrutina
- Parámetros de Salida (OUT)
  - El valor resultante de la subrutina se devuelve a la dirección del parámetro indicado
- Parámetros de Entrada/Salida (IN_OUT)
  - El valor de la dirección del parámetro indicado se transfiere a la subrutina y el valor resultante de la subrutina se devuelve luego a la misma dirección
- Variables Locales (TEMP)
  - Cualquier memoria local que no se utilice para la transferencia de parámetros se puede emplear para el almacenamiento temporal dentro de la subrutina.

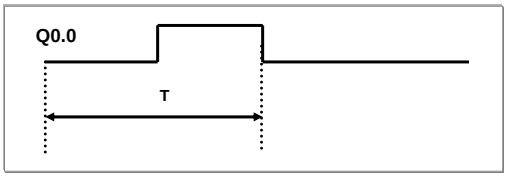
Subrutinas Anidadas

- Una Subrutina puede llamar a otra
- Se permite una profundidad de anidamiento de 8 niveles



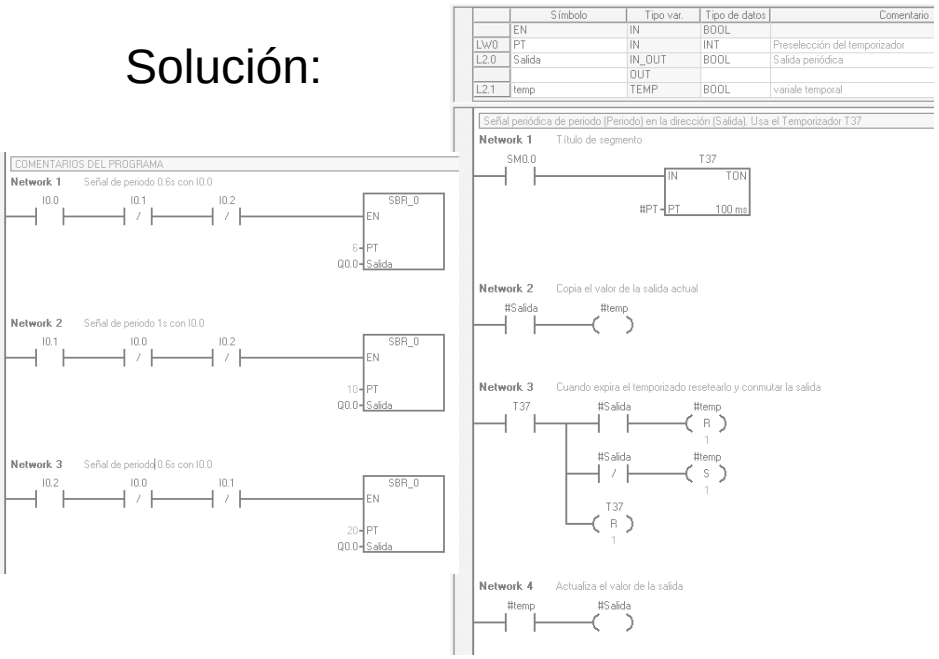
Ejemplo Subrutinas

- Mediante la preselección de tres interruptores se pretende conseguir una señal de periodo variable



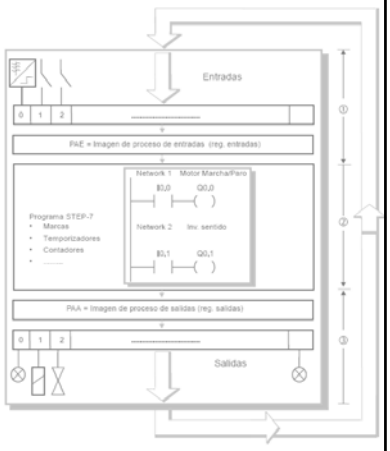
- Se desea obtener una señal de los siguientes periodos:
  - Si esta activa la entrada I0.0: 0.6 seg
  - Si esta activa la entrada I0.1: 1 seg
  - Si esta activa la entrada I0.2: 2 seg
- En el caso de que no estén activas ninguna entrada la salida debe de anularse

Solución:

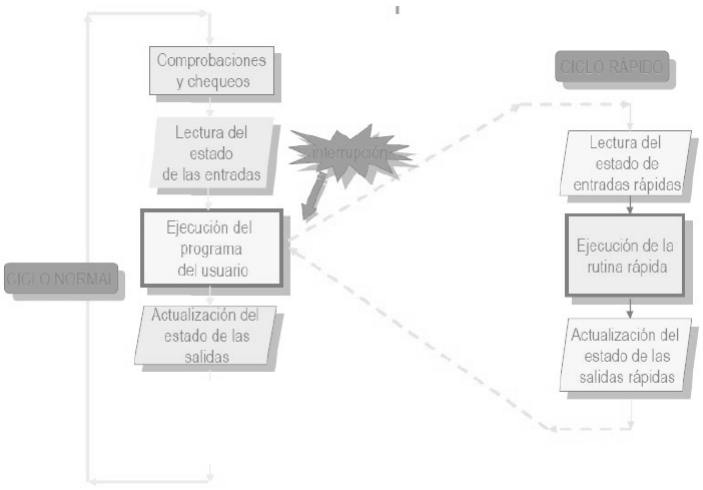


Rutinas de interrupción

- Permite ejecutar una serie de instrucciones ante eventos de interrupción
- El ciclo de ejecución del autómata (OB1) realiza la lectura de las entradas al principio y luego congela su valor hasta que termina (3-10ms)
- Cualquier evento que se produzca en este intervalo se perdería
- Las rutinas de interrupción permite tratar eventos asíncronos asociados a entradas y temporizadores
- Los eventos y las entradas están limitadas según el autómata utilizado.



Rutinas de Interrupción



Rutinas de interrupción

- Antes de poder llamar a una rutina de interrupción es preciso establecer un enlace entre el evento de interrupción y la parte del programa que se desee ejecutar cuando se presente el evento (RUTINA DE INTERRUPCIÓN)
- La operación Asociar interrupción (ATCH) sirve para asignar el evento de interrupción (*indicado por el número de evento*) a una parte del programa (*indicada por el número de la rutina de interrupción*).
- También es posible asociar varios eventos de interrupción a una única rutina de interrupción. Por el contrario, no se puede asociar un solo evento a distintas rutinas.

Rutinas de interrupción

- Eventos de Interrupción:

Nº de evento	Descripción de la interrupción	CPU 221	CPU 222	CPU 224
0	Flanco positivo, I0.0	Sí	Sí	Sí
1	Flanco negativo, I0.0	Sí	Sí	Sí
2	Flanco positivo, I0.1	Sí	Sí	Sí
3	Flanco negativo, I0.1	Sí	Sí	Sí
4	Flanco positivo, I0.2	Sí	Sí	Sí
5	Flanco negativo, I0.2	Sí	Sí	Sí
6	Flanco positivo, I0.3	Sí	Sí	Sí
7	Flanco negativo, I0.3	Sí	Sí	Sí
8	Puerto 0: Recibir carácter	Sí	Sí	Sí
9	Puerto 0: Transmisión finalizada	Sí	Sí	Sí
10	Interrupción temporizada 0, SMB34	Sí	Sí	Sí

Rutinas de interrupción

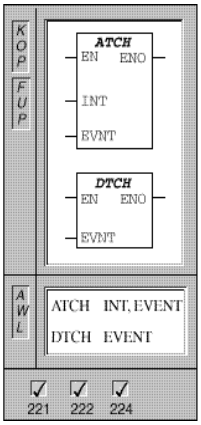
- Cuando se asocia un evento a una rutina de interrupción, se habilita automáticamente el evento. Si se inhiben todos los eventos de interrupción, entonces cada vez que se presente la interrupción, se pondrá en cola de espera hasta que las interrupciones se habiliten de nuevo, utilizando para ello la operación Habilitar todos los eventos de interrupción.
  - ENI: Habilita todos los eventos de interrupción.
  - DISI: Inhibe todos los eventos de interrupción
- También es posible inhibir ciertos eventos de interrupción, eliminando la asociación entre el evento y la correspondiente rutina mediante la operación DTCH (Desasociar interrupción). Esta operación retorna la interrupción a un estado inactivo o ignorado.

Rutinas de interrupción

- Crear una rutina de interrupción (Similar a crear subrutinas)

- Notas
  - En un programa se permiten 128 rutinas de interrupción como máximo.
  - La CPU procesa las interrupciones según su prioridad y después en el orden que aparecen.
  - Sólo se ejecuta una rutina de interrupción a la vez.
  - Las interrupciones que se presenten mientras se está ejecutando otra interrupción se ponen en cola de espera para ser procesadas posteriormente.

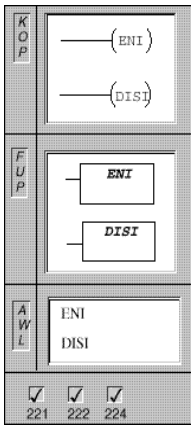
Rutinas de interrupción



- Asociar interrupción, Desasociar interrupción
  - La operación ATCH (Asociar interrupción) asocia el número de una rutina de interrupción (INT) a un evento de interrupción (EVNT), habilitando así este último.
  - La operación DTCH (Desasociar interrupción) desasocia un evento de interrupción (EVNT) de todas las rutinas de interrupción, deshabilitando así el evento.

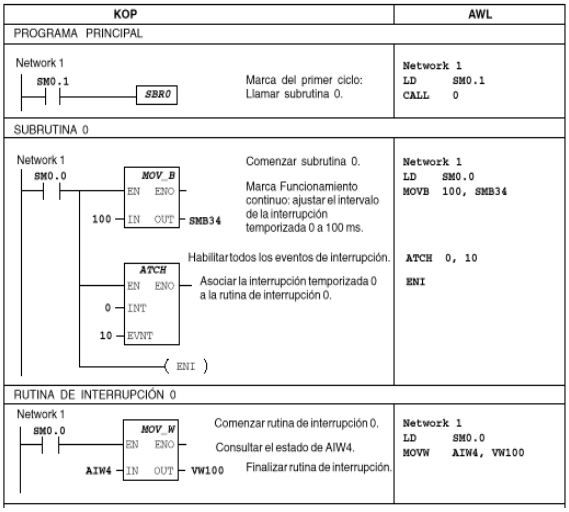
Rutinas de interrupción

- Habilitar todos los eventos de interrupción, Inhibir todos los eventos de interrupción



  - La operación ENI habilita todos los eventos de interrupción habilita la ejecución de todos los eventos asociados.
  - La operación DISI inhibe todos los eventos de interrupción inhibe la ejecución de todos los eventos asociados.
  - Cuando la CPU pasa a modo RUN, las interrupciones se inhiben. Estando en modo RUN, se pueden habilitar todos los eventos de interrupción con la operación global ENI. La operación DISI permite poner las interrupciones en cola de espera, pero no llamar a ninguna rutina de interrupción.

Ejemplo: Rutinas de interrupción



Rutinas de interrupción

- Tipos de interrupciones
  - Interrupciones de comunicación
    - El puerto serie. La comunicación a través de dicho puerto se denomina modo Freeport (comunicación programable por el usuario). En modo Freeport, el programa define la velocidad de transferencia, los bits por carácter, la paridad y el protocolo.
    - Las interrupciones de transmisión y recepción permiten controlar la comunicación mediante el programa.
  - Interrupciones E/S
    - Las interrupciones E/S abarcan interrupciones en flancos positivos y negativos, interrupciones de los contadores rápidos, así como interrupciones de la salida de impulsos.

Interrupciones E/S	CPU S7-200
Entradas y salidas	I0.0 a I0.3

Rutinas de interrupción

- Tipos de interrupciones
  - Interrupciones temporizadas (0, 1)
    - Las interrupciones temporizadas se utilizan para indicar tareas que deban ejecutarse cíclicamente
    - Las interrupciones temporizadas incluyen también las de los temporizadores T32/T96.
    - El tiempo de ciclo se incrementa en intervalos de 1 ms, abarcando desde 1 ms hasta 255 ms.
    - El tiempo de ciclo de la interrupción temporizada 0 se debe escribir en SMB34, y el de la interrupción temporizada 1, en SMB35.
    - Típicamente, las interrupciones temporizadas se utilizan para controlar el muestreo de las entradas analógicas en intervalos regulares o para ejecutar un bucle de control PID.

Rutinas de interrupción

- Prioridades de las interrupciones y colas de espera
  - La prioridad de las interrupciones es la siguiente:
    - Interrupciones de comunicación (prioridad más alta)
    - Interrupciones E/S
    - Interrupciones temporizadas (prioridad más baja)
  - La CPU procesa las interrupciones según su prioridad y después en el orden en que aparecen. Sólo se ejecuta una rutina de interrupción en cada caso.
  - Las interrupciones que aparezcan mientras se esté ejecutando otra interrupción se ponen en cola de espera para ser procesadas posteriormente.

Rutinas de interrupción

- Reglas para el buen uso de las interrupciones
  - El procesamiento de interrupciones permite reaccionar rápidamente ante determinados eventos internos o externos. Las rutinas de interrupción se deben estructurar de forma que, una vez ejecutadas determinadas tareas, devuelvan el control al programa principal
  - Para ello es conveniente crear rutinas de interrupción cortas con indicaciones precisas, de manera que se puedan ejecutar rápidamente sin interrumpir otros procesos durante períodos demasiado largos.
  - Si no se observan estas medidas, es posible que se produzcan estados imprevistos que pueden afectar a la instalación controlada por el programa principal. Al utilizar interrupciones, conviene atenerse al lema de ”cuanto más breve, mejor”.

Ejemplos:

Rutinas de Interrupción

Ejemplo de interrupciones temporizadas

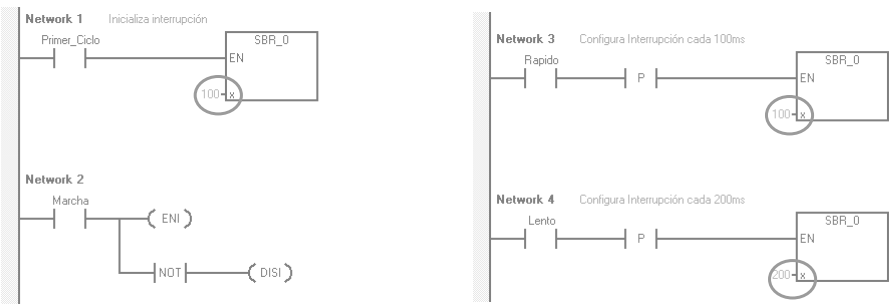
- Objetivo
  - Utilizar las interrupciones temporizadas para generar una secuencia de destellos.
    - La activación de la entrada I0.1 reduce la frecuencia de destellos a la mitad de la mencionada secuencia.
    - La activación de la entrada I0.0 restablece la frecuencia original de destellos. (semiperiodo 100ms)
    - I0.2 activa/desactiva la secuencia de destellos
- Este ejemplo explica el tratamiento general de las interrupciones temporizadas así como la modificación de la base de tiempo.

Ejemplo de interrupciones temporizadas

- Tabla de símbolos

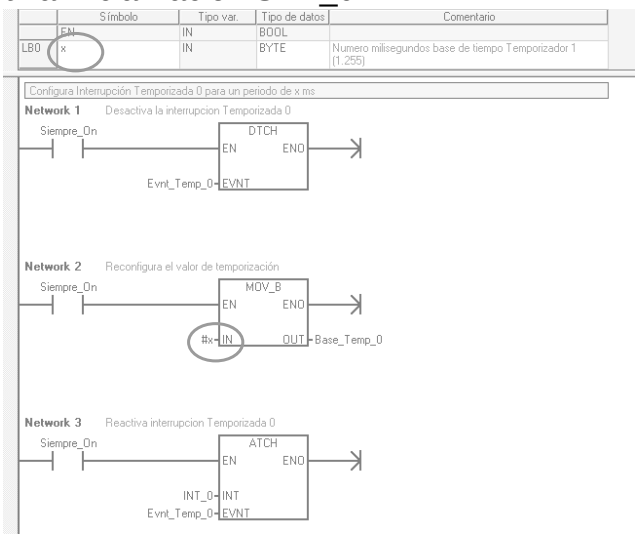
Símbolo	Dirección	Comentario
Siempre_On	SM0.0	
Primer_Ciclo	SM0.1	
Evnt_Temp_0	10	Evento interrupción temporizada 0
Base_Temp_0	SMB34	Base de tiempo (1ms-255ms) interurpción temporizada 0
Evnt_Temp_1	11	Evento interrupción temporizada 1
Base_Temp_1	SMB35	Base de tiempo (1ms-255ms) interurpción temporizada 1
Marcha	I0.2	Activa interrupciones
Lento	I0.1	Pulso lento
Rapido	I0.0	Pulso rápido
Salida	Q0.0	Salida

- PROGRAMA PRINCIPAL OB1:



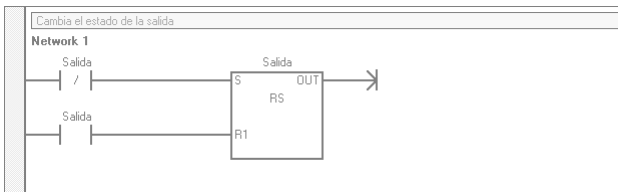
Ejemplo de interrupciones temporizadas

- Subrutina inicialización SBR_0



Ejemplo de interrupciones temporizadas

- RUTINA INTERRUPCIÓN INT 0

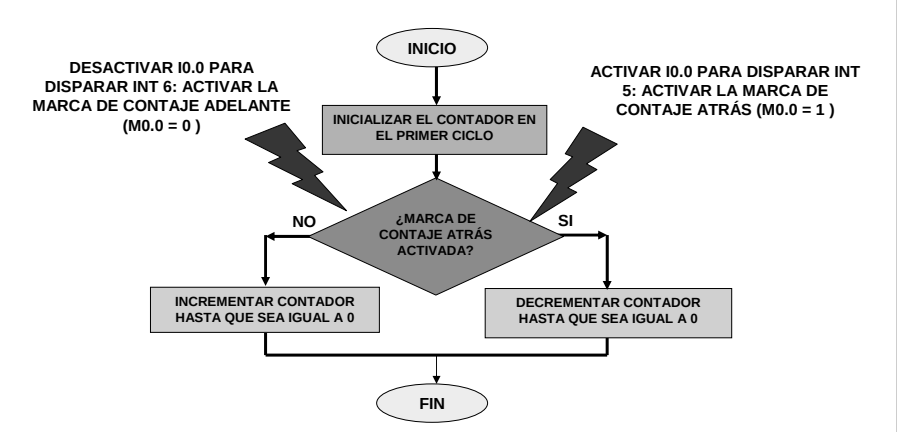


Ejemplo de Interrupciones de E/S

- Objetivo
  - Realizar un programa que cuente de 0 hasta 255, en función de la entrada I0.0. Si está activada la entrada I0.0, el programa cuenta hacia atrás. Si no está activada la entrada I0.0, el programa cuenta hacia adelante.
  - Si se conmuta la entrada, se dispara una rutina de interrupción de entrada/salida (E/S). Esta rutina de interrupción activa o desactiva la marca M0.0 de contaje atrás.
  - La activación de la entrada I0.1 provoca el reseteo del contador.
  - Utilizar la marca SM0.5.

Ejemplo de Interrupciones de E/S

- Flujograma



Ejemplo de Interrupciones de E/S

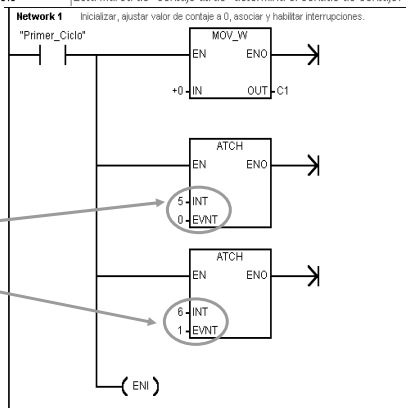
- Tabla de símbolos

	Nombre	Dirección	Comentario
1	Siempre_On	SM0.0	Este bit siempre está activado.
2	Primer_Ciclo	SM0.1	Este bit se activa sólo en el primer ciclo.
3	Reloj_1s	SM0.5	Reloj activado 0,5 s, desactivado 0,5 s, durante un tiempo de ciclo de 1 s.
4	Marca_Atrás	M0.0	Esta marca de "contaje atrás" determina el sentido de contaje.

- Programa principal OB1

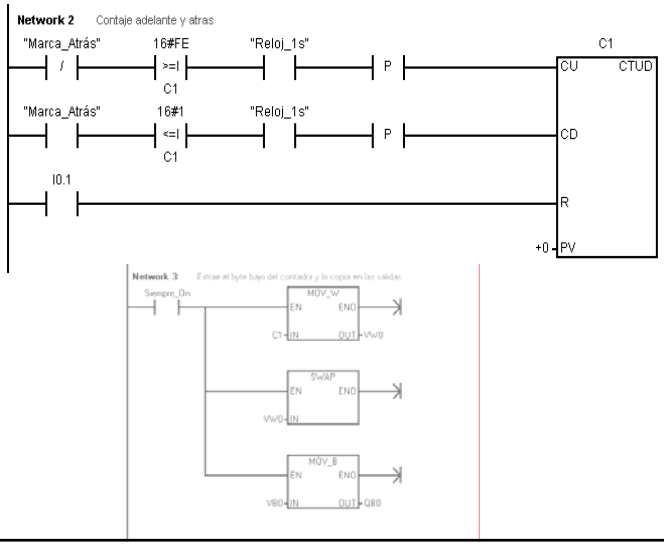
Evento de interrupción

Nº de evento	Descripción de la interrupción
0	Flanco positivo, I0.0
1	Flanco negativo, I0.0



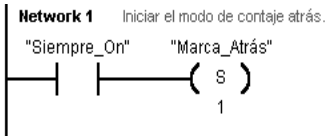
Ejemplo de Interrupciones de E/S

- Programa principal OB1



Ejemplo de Interrupciones de E/S

- Rutina de interrupción 5



- Rutina de interrupción 6