

**Manual de formación
para soluciones generales en automatización
Totally Integrated Automation (T I A)**

MÓDULO B5

**Programación estructurada con
bloques de función**

Este documento fue suministrado por SIEMENS Siemens A&D SCE (Tecnología en Automatización y Accionamientos, Siemens A&D, coopera con la Educación) para formación. Siemens no hace ningún tipo de garantía con respecto a su contenido.

El préstamo o copia de este documento, incluyendo el uso e informe de su contenido, sólo se permite dentro de los centros de formación.

En caso de excepciones se requiere el permiso por escrito de Siemens A&D SCE (Mr. Knust: E-Mail: michael.knust@hvr.siemens.de). Cualquier incumplimiento de estas normas estará sujeto al pago de los posibles perjuicios causados. Todos los derechos quedan reservados para la traducción y posibilidad de patente.

Agradecemos al Ingeniero Michael Dziallas, a los tutores de las escuelas de formación profesional, así como a todas aquellas personas que nos han prestado su colaboración para la elaboración de este documento.

PÁGINA:

1.	Introducción.....	4
2.	Notas sobre la Programación Estructurada con FCs y FBs.....	6
3.	Generando Bloques de Función con Declaración de Variables.....	8

Los símbolos siguientes acceden a los módulos especificados:



Información



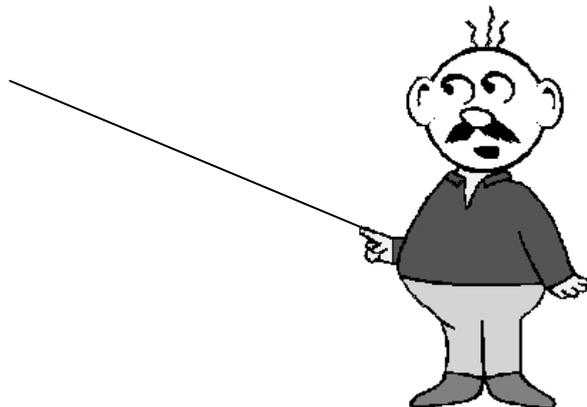
Programación



Ejercicio Ejemplo

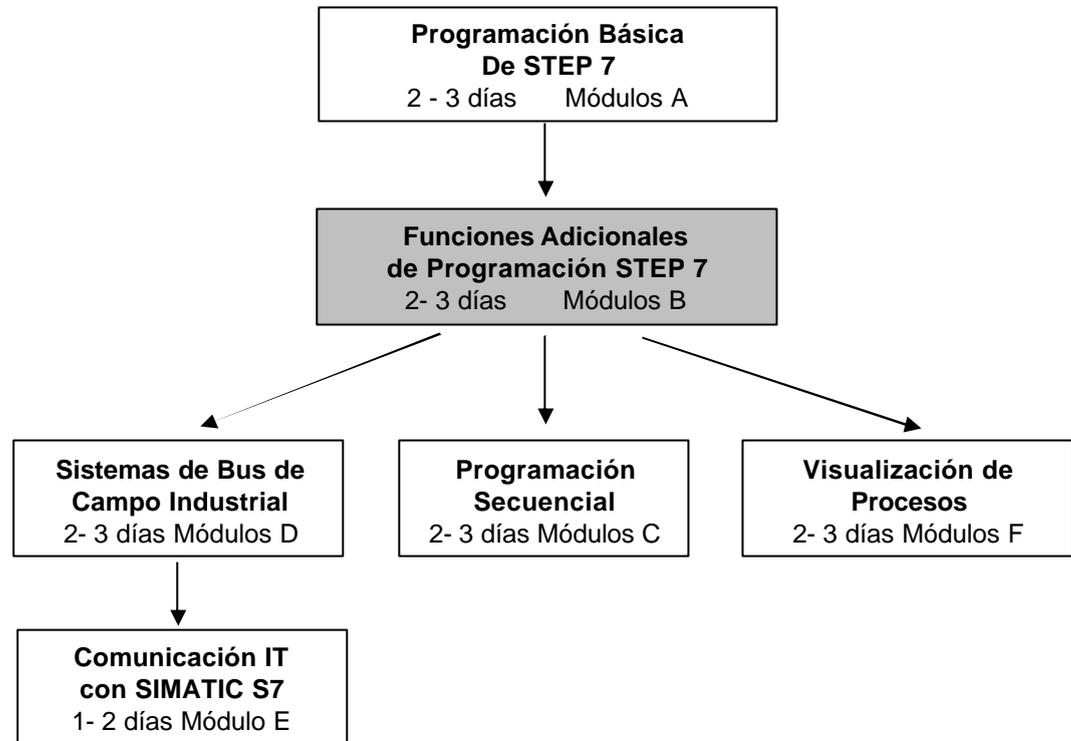


Notas



1. INTRODUCCIÓN

El módulo B5 pertenece al contenido de las **Funciones Adicionales de Programación STEP 7**.



Finalidad del Aprendizaje:

En este módulo, el lector aprenderá sobre como generar un bloque de función con definición de variables y su posterior aplicación en la programación estructurada.

- Generando un bloque de función
- Definición de variables internas
- Programando variables internas en un bloque de función
- Llamando y parametrizando un bloque de función en el OB1

Requisitos:

Para el correcto aprovechamiento de este módulo, se requieren los siguientes conocimientos:

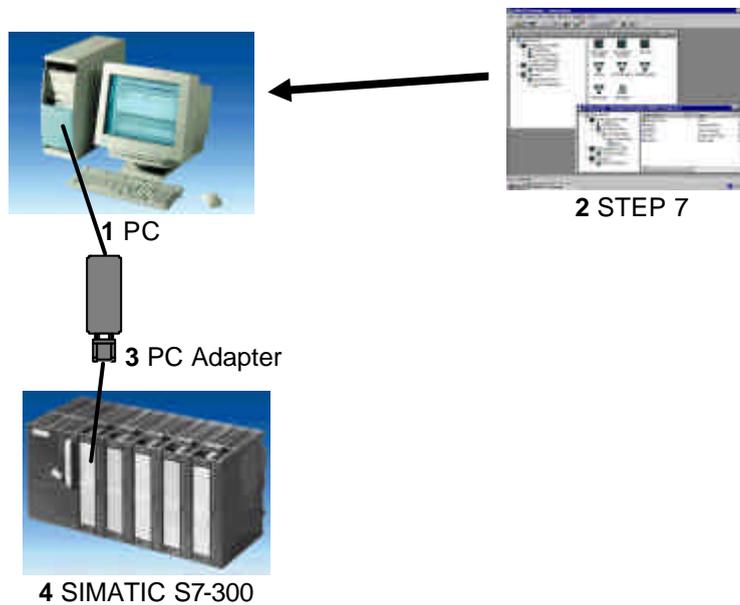
- Conocimientos de uso de Windows 95/98/2000/ME/NT4.0
- Programación Básica de PLC con STEP 7 (Módulo A3 - 'Puesta en Marcha' programando PLC con STEP 7)
- Conocimientos Básicos de programación estructurada (Anexo I – Programación Básica de PLC – Programando con un SIMATIC S7-300)

Hardware y software Necesarios

- 1 PC, Sistema Operativo Windows 95/98/2000/ME/NT4.0 con
 - Mínimo: 133MHz y 64MB RAM, aprox. 65 MB de espacio libre en disco duro
 - Óptimo: 500MHz y 128MB RAM, aprox. 65 MB de espacio libre en disco duro
- 2 Software STEP 7 V 5.x
- 3 Interfase MPI para PC (p.e. PC- Adapter)
- 4 PLC SIMATIC S7-300 con al menos un módulo de entradas/salidas

Ejemplo de configuración:

- Fuente de Alimentación: PS 307 2A
- CPU: CPU 314
- Entradas Digitales: DI 16x DC24V
- Salidas Digitales: DO 16x DC24V / 0.5 A



2. NOTAS SOBRE LA PROGRAMACIÓN ESTRUCTURADA CON FCS Y FBS



La ejecución del programa es escrita en bloques de STEP 7. El bloque de organización OB1 ya se encuentra disponible.

El programa describe el interfase con el sistema operativo de la CPU y es ejecutado automáticamente desde el OB1 de manera cíclica.

A través de extensivas tareas de control, se puede 'cortar' el programa en pequeños, manejables y ordenados bloques o funciones.

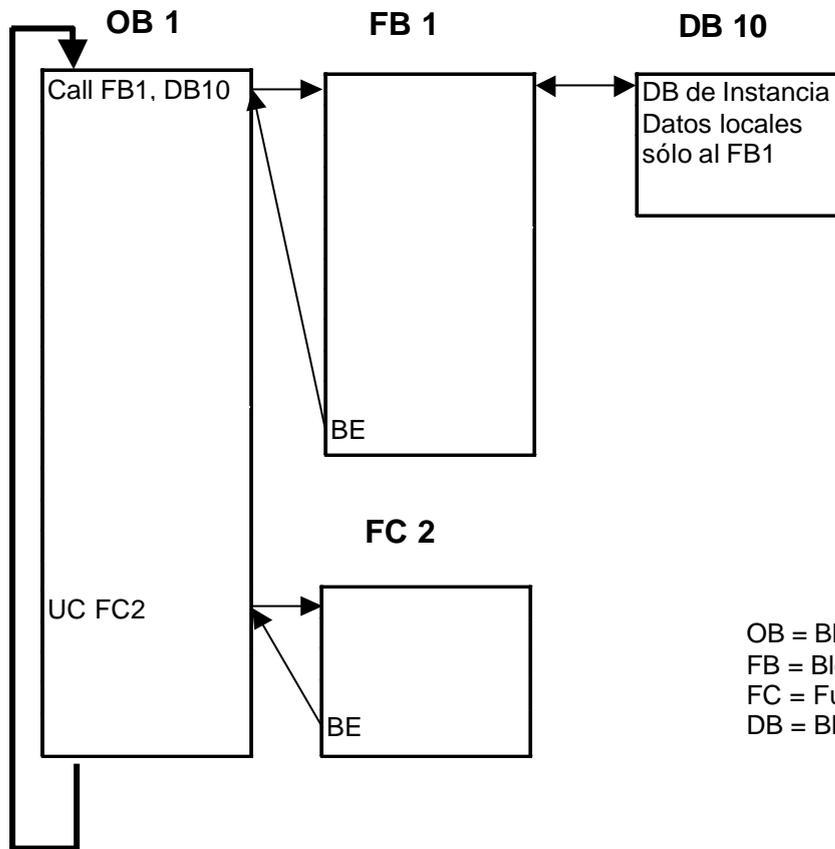
Esos bloques son entonces llamados desde el bloque de organización a través de instrucciones de llamada a bloque (Call xx / UC xx / CC xx). Una vez finalizado la ejecución del bloque llamado, el programa continúa su ejecución en el módulo llamante.

STEP 7 ofrece las siguientes herramientas para la estructuración de programas:

- **FB (Bloque de Función):**
El FB tiene asignada una área de almacenamiento. Si un FB es llamado, puede tener un bloque de datos asignado (DB). Desde los datos de esta instancia, el DB puede ser accedido a través de una llamada al FB. Un FB puede tener asignados diferentes DBs. Tanto los FBs como los FCs pueden ser llamados desde otro FB.
- **FC (Función):**
Un FC no posee un área de datos asignada. Los datos locales de una función se pierden tras finalizar la ejecución de la función. Tanto los FBs como los FCs pueden ser llamados desde otro FB.



La estructura de un programa puede parecerse a lo siguiente:



OB = Bloque de Organización
 FB = Bloque de Función
 FC = Función
 DB = Bloque de Datos



Nota: Para poder utilizar los bloques, primero deberán ser generados. Existe la posibilidad de generar esos FCs y FBs en forma de bloques estándar bajo el uso de variables internas. Cualquier FC puede llamarse tantas veces como se quiera, mientras que cada llamada a un FB exige de un DB de instancia diferente.

3. GENERANDO UN BLOQUE DE FUNCION CON DECLARACION DE VARIABLES



Cuando se genera este tipo de bloques con STEP 7, la llamada es una 'Caja-Negra' con una serie de variables asignadas. En esos bloques no se utiliza direccionamiento absoluto de Entradas/Salidas, marcas, temporizadores, contadores, etc....., sino que se asignan variables y constantes a través de parámetros.



En el ejemplo siguiente, se suministra un bloque de función con declaración de variables, el cual contiene un control de cinta transportadora y un contador de ciclos.

El motor de la cinta transportadora se activa con el botón 'S0' y se desactiva con el botón 'S1'. Los ciclos del programa de transporte se encuentran almacenados en una doble palabra de marcas.

El ejemplo corresponde a las siguientes direcciones:

Entradas:

- Botón ON S0 = E 0.0
- Botón OFF S1 = E 0.1

Salidas:

- Motor de la Cinta = A 4.0

Marcas:

- Contador de Ciclos = MD20



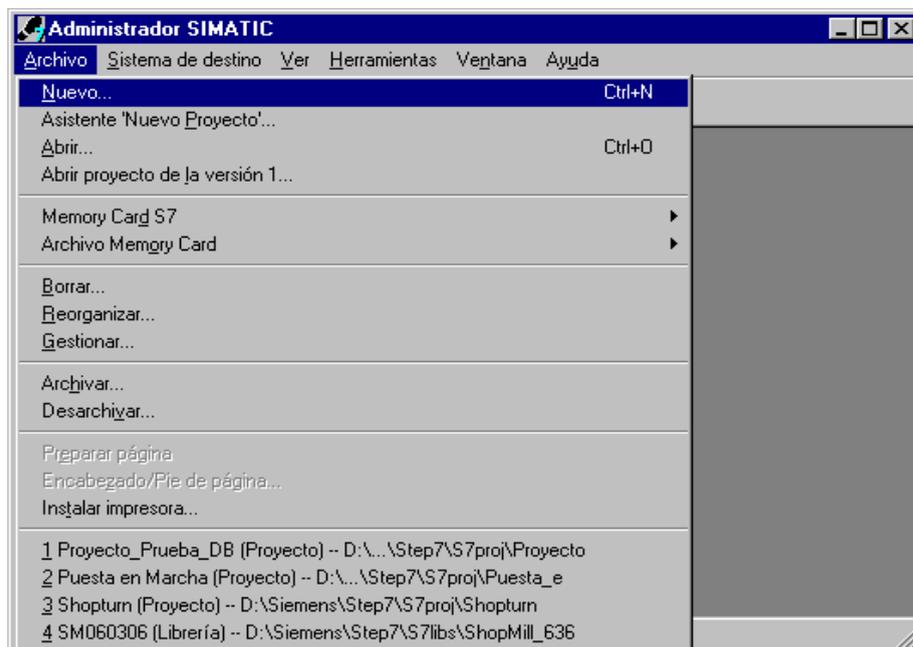
Para poder generar este ejemplo del programa, deben de los seguirse los siguientes pasos (Por eso el programa se distribuye con la creación de la configuración del hardware):

1. Llamar al **Administrador SIMATIC** con un doble click (→ Administrador SIMATIC)



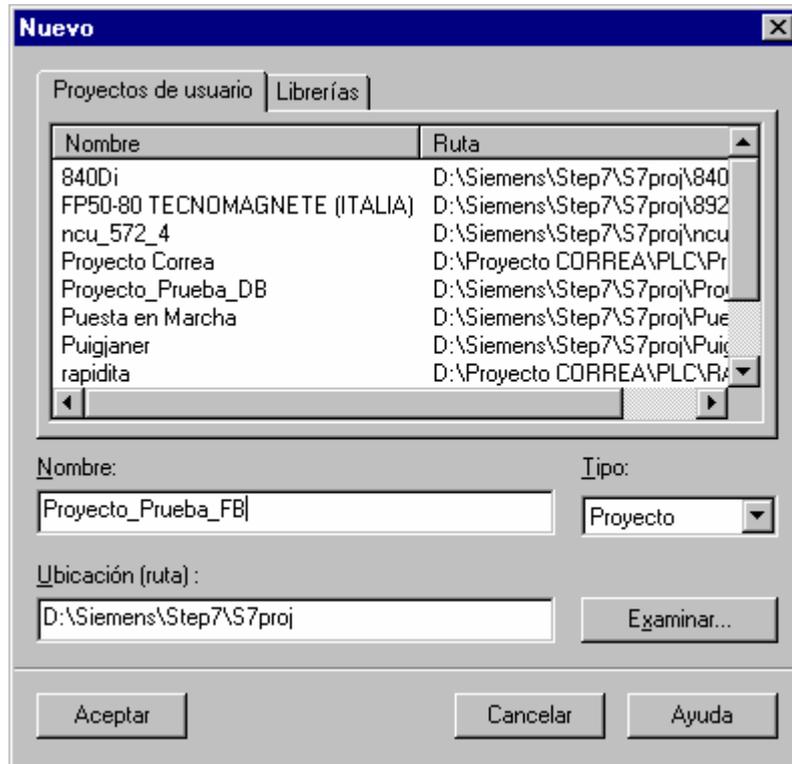
Administrador SIMATIC.Ink

2. Crear un proyecto nuevo (→ Archivo → Nuevo)

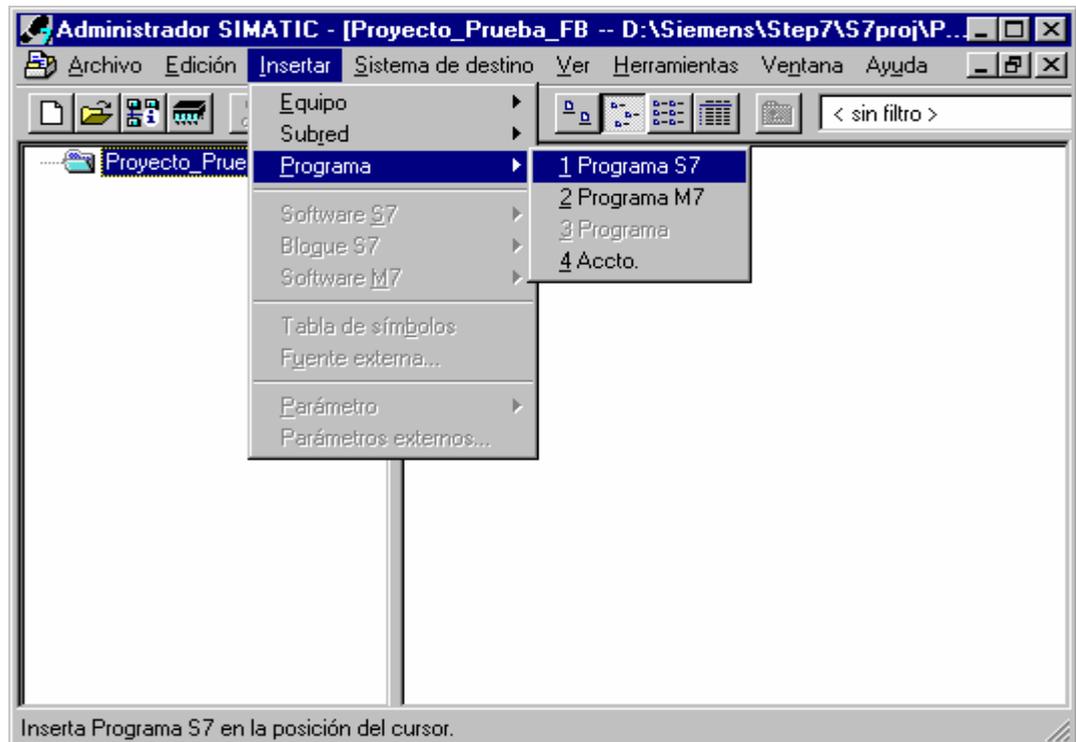




- En el campo Nombre, asignamos el nombre del proyecto **Proyecto_Prueba_FB**.
(→ 'Proyecto_Prueba_FB' → OK)

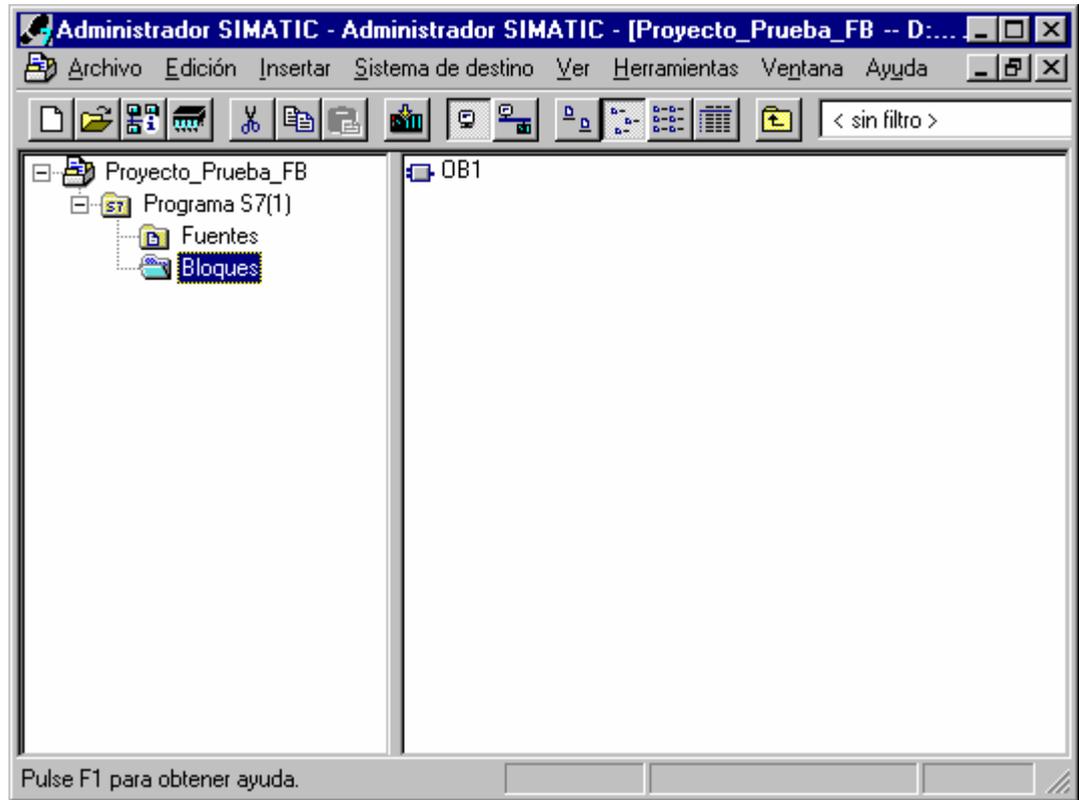


- Insertar un nuevo **Programa S7** (→ Insertar → Programa → Programa S7).

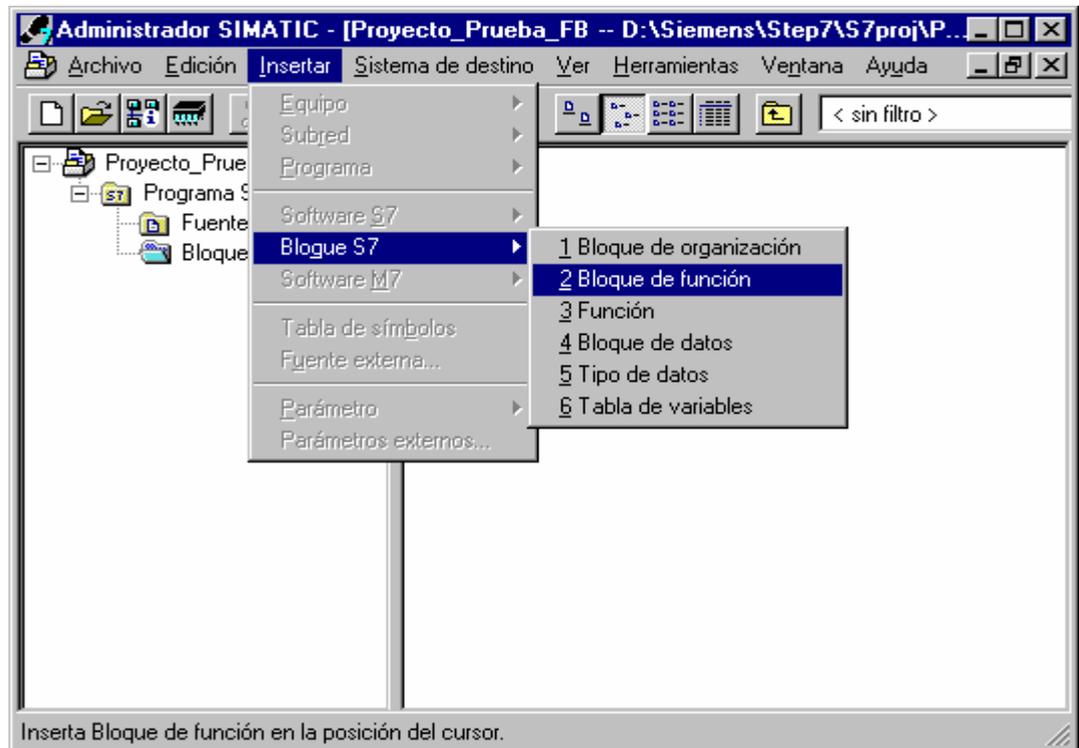




5. Seleccionar la carpeta **Bloques**. (→ Bloques)

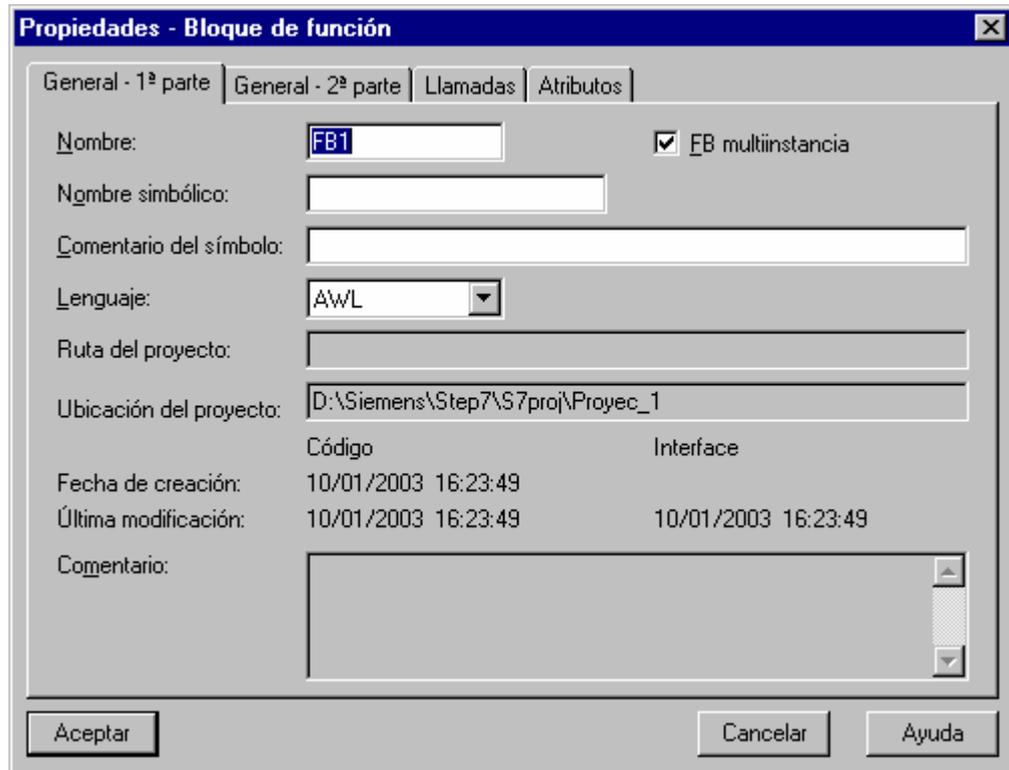


6. Insertar un **Bloque de Función** (→ Insertar → Bloque S7 → Bloque de Función).

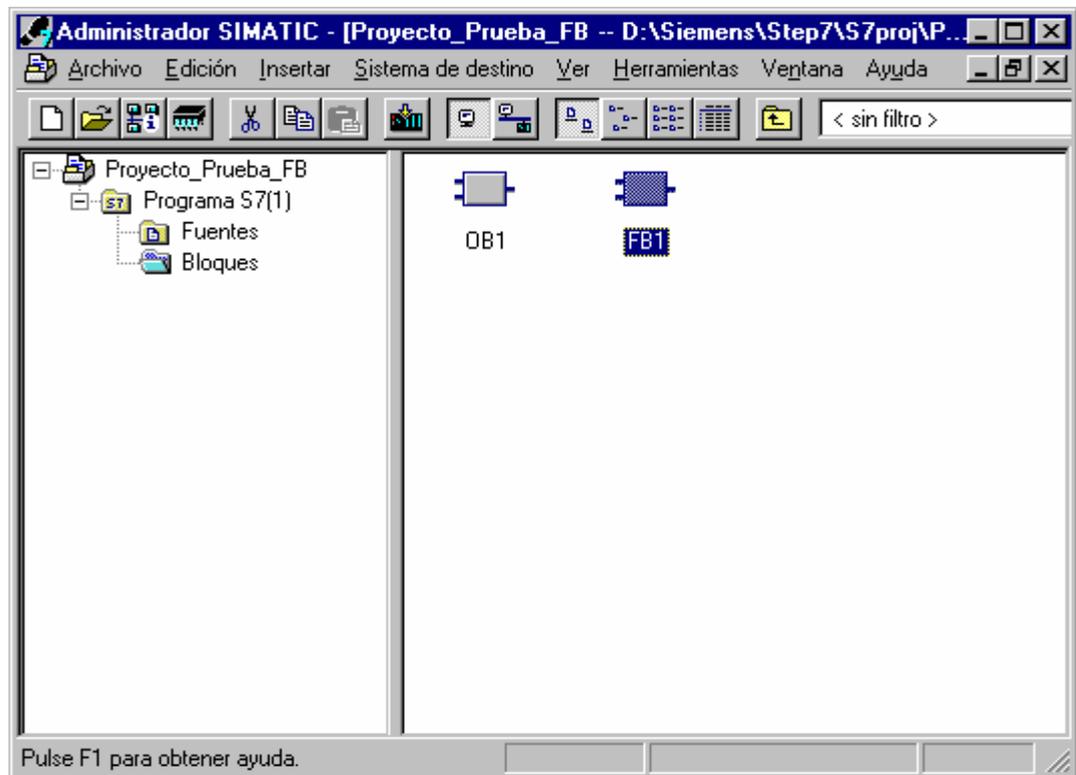




7. Introducir el nombre **FB1** y aceptar con **OK** (→ FB1 → OK)..



8. Abrir el Bloque de Función **FB1** con un doble click. (→ FB1)





9. Con el **Editor KOP/ AWL//FUP: Programar Bloques**, se dispone de una herramienta para poder editar funciones.
Además las variables deberían ser definidas en la tabla de definición de variables, la cual se encuentra en el FB1.
Esas variables pueden ser: 'Entrada', 'Salida', 'Entrada_Salida', 'Stat' y 'Temp'.

Parámetros de Entrada (IN) solo en FBs, FCs, SFBs y SFCs

Con ayuda de los parámetros de entrada se pueden asignar datos necesarios para el procesamiento del bloque.

Parámetros de Salida (OUT) solo en FBs, FCs, SFBs y SFCs

En los parámetros de salida, los resultados del procesamiento del bloque son depositados aquí.

Parámetros de Entrada/Salida (IN_OUT) solo en FBs, FCs, SFBs y SFCs

En los parámetros de Entrada/Salida, los contenidos de estos parámetros y el resultado del procesamiento del bloque depositado en ellos mismos.

Datos Estáticos (STAT) sólo en FBs y SFBs

Los datos estáticos son los datos locales a un bloque de función, los cuales son almacenados en un bloque de datos de instancia y por tanto preservados hasta el siguiente procesamiento del bloque.

Datos Temporales (TEMP) en todos los bloques

Los datos estáticos son los datos locales a un bloque que almacenan valores durante el proceso de dicho bloque en una pila de datos locales (L-Stack) y, una vez ha terminado de procesarse el bloque, el contenido de estas variables se pierde.



Nota: Hay que establecer una diferencia entre FB/SFB y FC/SFC. En un FC no existen variables estáticas (**stat**) por no existir memoria donde almacenar el contenido de dichas variables, una vez finalizado el procesamiento del PLC. En el FB, esas variables estáticas son almacenadas en su correspondiente DB de instancia hasta el siguiente procesamiento del FB.
Desde este punto de vista, sólo los FBs se hayan preparados para programas en los que datos, como por ejemplo bits de datos, deben guardar su contenido para ciclos posteriores.



La tabla de definiciones se compone de un nombre, tipo y, como opción, un valor inicial y un comentario. . Un ejemplo de Tabla de definiciones es el siguiente:

Dirección	Declaración	Nombre	Tipo	Valor inicial	Comentario
0.0	in	On	BOOL	FALSE	Motor Activado
0.1	in	Off	BOOL	FALSE	Motor Desactivado
2.0	out	Motor	BOOL	FALSE	Cinta
4.0	in_out	Ciclo	DINT	L#0	Contador de Ciclos
8.0	stat	Contador	WORD	W#16#0	Numero de Ciclos
0.0	temp	Error	BOOL		Error de Contaje

Columna de Declaración-
Determina el tipo de parámetro

Valor Inicial en formato compatible con el tipo de datos

Comentario de la documentación (opcional)

La dirección absoluta es generada por STEP 7 automáticamente.
El formato de la dirección es **BYTE.BIT**

Nombre simbólico asociado a la dirección absoluta. A través de esta dirección se puede acceder al parámetro

Tipo de datos seleccionado (ver tabla de tipos abajo).



Nota:

En la declaración se muestra cada uno de los tipos de variables estáticas. Se muestra en los FCs variables del tipo 'Entrada', 'Salida', 'Entrada_Salida' y 'Temp' y en los FBs, variables del tipo 'Entrada', 'Salida', 'Entrada_Salida', 'Stat' y 'Temp'. Si se necesita otra variable de un tipo en particular, se posicionará el cursor en el campo 'Comment' y se pulsará <Enter>. Aparecerá entonces una nueva fila vacía de declaración del mismo tipo que la anterior.



Los datos en un bloque de función deben de llevar asignado un tipo de datos.
Los tipos de datos estándar STEP 7 se definen en la tabla mostrada abajo :

Tipo y descripción	Tamaño en Bits	Formato-Opciones	Rango y notación numérica (Valores máximo y mínimo)	Ejemplo
BOOL (Bit)	1	Texto Booleano	TRUE/FALSE	TRUE
BYTE (Byte)	8	Número Hexadecimal	B#16#0 a B#16#FF	B#16#10
WORD (Palabra)	16	Número Binario	2#0 a 2#1111_1111_1111_1111	2#0001_0000_0000_0000
		Número Hexadecimal	W#16#0 a W#16#FFFF	W#16#1000
		BCD	C#0 a C#999	C#998
		Número Decimal sin signo	B#(0,0) a B#(255,255)	B#(10,20)
DWORD (Doble Palabra)	32	Número Binario	2#0 a 2#1111_1111_1111_1111_1111_1111_1111_1111	2#1000_0001_0001_1000_1011_1011_0111_1111
		Número Hexadecimal	DW#16#0000_0000 a DW#16#FFFF_FFFF	DW#16#00A2_1234
		Número Decimal sin signo	B#(0,0,0,0) a B#(255,255,255,255)	B#(1,14,100,120)
INT (Entero)	16	Número Decimal con signo	-32768 a 32767	1
DINT (Int,32 bit)	32	Número Decimal con signo	L#-2147483648 a L#2147483647	L#1
REAL (Número en coma flotante)	32	Número en coma flotante IEEE	Máximo: +/-3.402823e+38 Mínimo: +/-1.175495e-38	1.234567e+13
S5TIME (Tiempo Simatic)	16	Tiempo S7 en pasos de 10 ms	S5T#0H_0M_0S_10MS a S5T#2H_46M_30S_0MS and S5T#0H_0M_0S_0MS	S5T#0H_1M_0S_0MS S5TIME#1H_1M_0S_0MS
TIME (Tiempo IEC)	32	Tiempo IEC en pasos desde 1ms, entero con signo	-T#24D_20H_31M_23S_648MS a T#24D_20H_31M_23S_647MS	T#0D_1H_1M_0S_0MS TIME#0D_1H_1M_0S_0MS
DATE (Fecha IEC)	16	Fecha IEC en pasos de 1 día	D#1990-1-1 a D#2168-12-31	DATE#1994-3-15
TIME_OF_DAY (Fecha y Hora)	32	Tiempo en pasos de 1ms	TOD#0:0:0.0 a TOD#23:59:59.999	TIME_OF_DAY#1:10:3.3
CHAR (Carácter)	8	Caracteres ASCII	'A', 'B' etc.	'B'



10. Ahora el bloque puede programarse utilizando nombres simbólicos de variables (estos nombres se identifican por ir precedidos del símbolo '#'). Estas variables se muestran en el ejemplo AWL siguiente. El bloque de función FB1 debería de ser almacenado en el disco duro  y cargado en la CPU . El selector de modos de la CPU debe estar en STOP! (→  → )

FB1 : Control de Cinta (conectada con contador de ciclos)

Comentario:

Segm. 1 : Activar/Desactivar cinta

Comentario:

```

U   #Qa
S   #Motor
U   #Q1f
R   #Motor
    
```

Segm. 2 : Contador de ciclos

Comentario:

```

L   #Ciclo
L   1
+D
T   #Ciclo
    
```

Segm. 3 : Hay un error en el conteo

Comentario:

```

L   #Ciclo
L   -1
--D
EEE
    
```

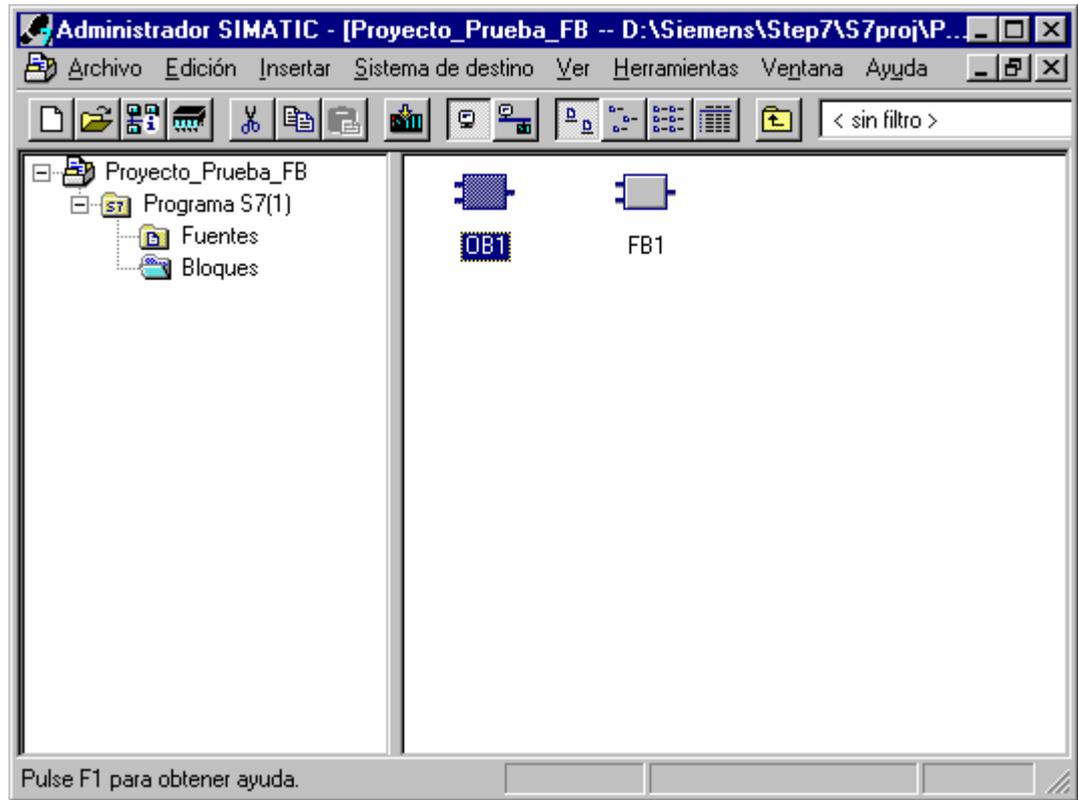
En el programa , cada una de las variables de la tabla son accedidas a través de su nombre simbólico, precedido de un '#'

1: Error 2: Info

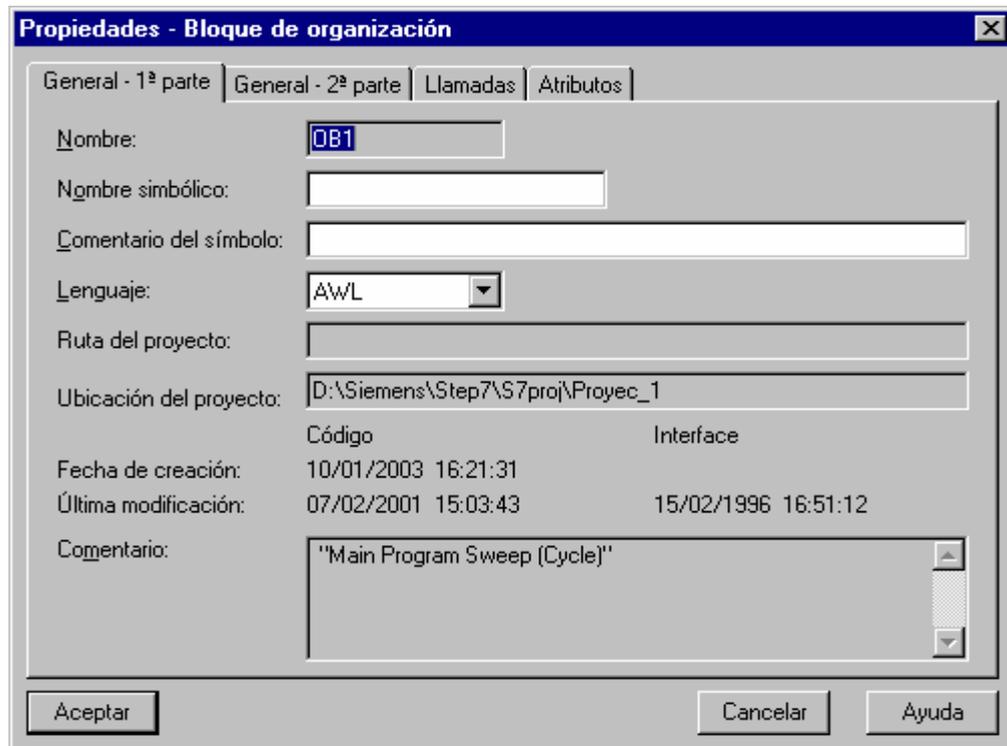
Pulse F1 para obtener ayuda. offline



- En el **Administrado SIMATIC**, sólo es necesario abrir el **OB1** para llamar al FB1 (→ OB1).



- Aceptar la ventana haciendo click en **OK** (→ OK).

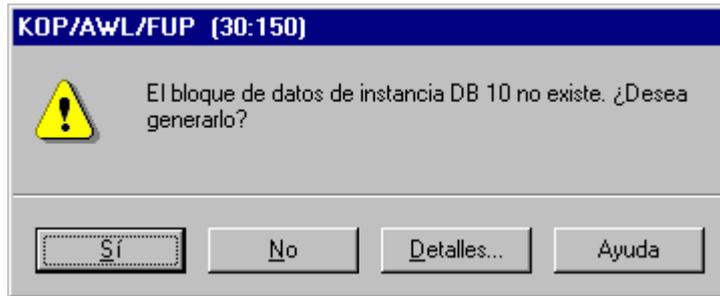




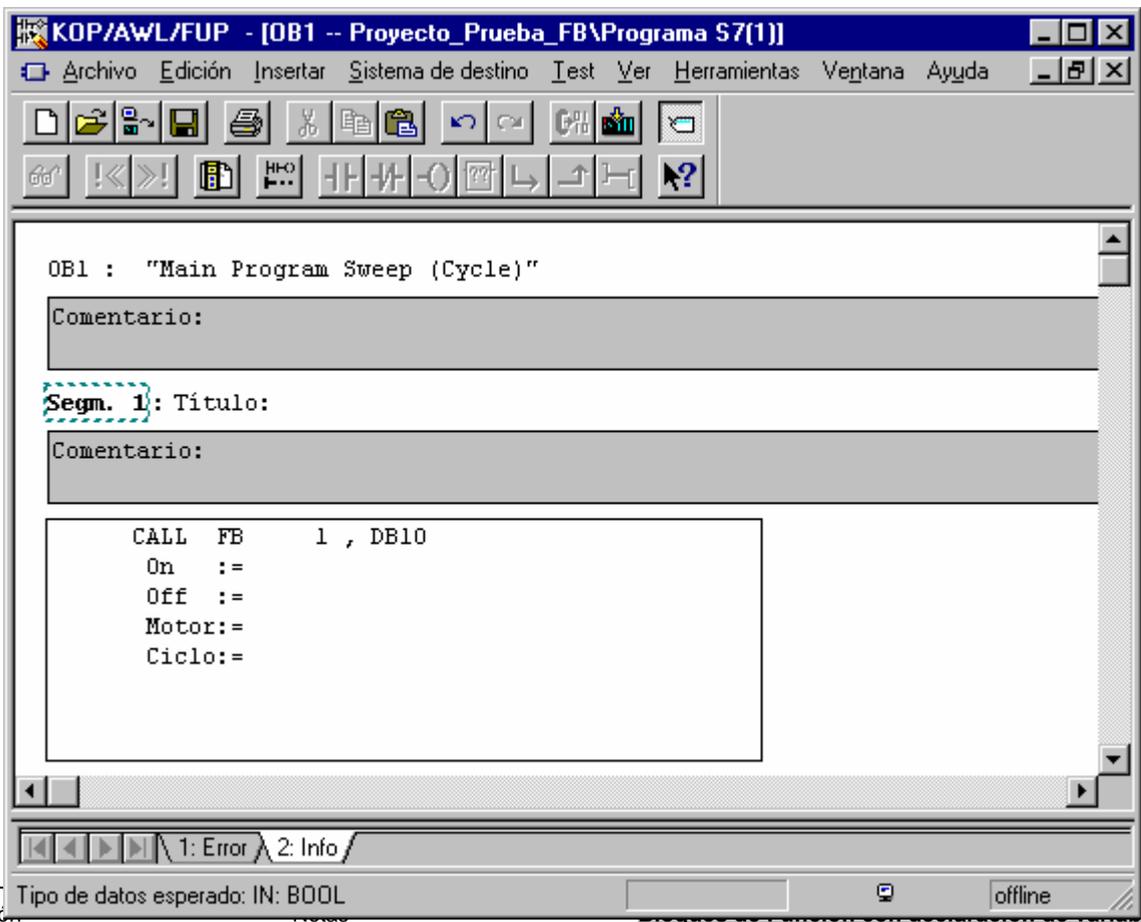
13. Con el **Editor KOP/ AWL//FUP: Programar Bloques**, se dispone de una herramienta para editar el OB1. El FB1 debería ser llamado junto con su DB de instancia asociado (también llamado DB local) a través de la siguiente línea de comando:

CALL FB1,DB10 <Enter>

El DB de instancia (DB10) será entonces generado automáticamente con tan solo contestar a la pregunta que aparecerá con un **Sí** (→ Call FB1,DB10 → Sí).

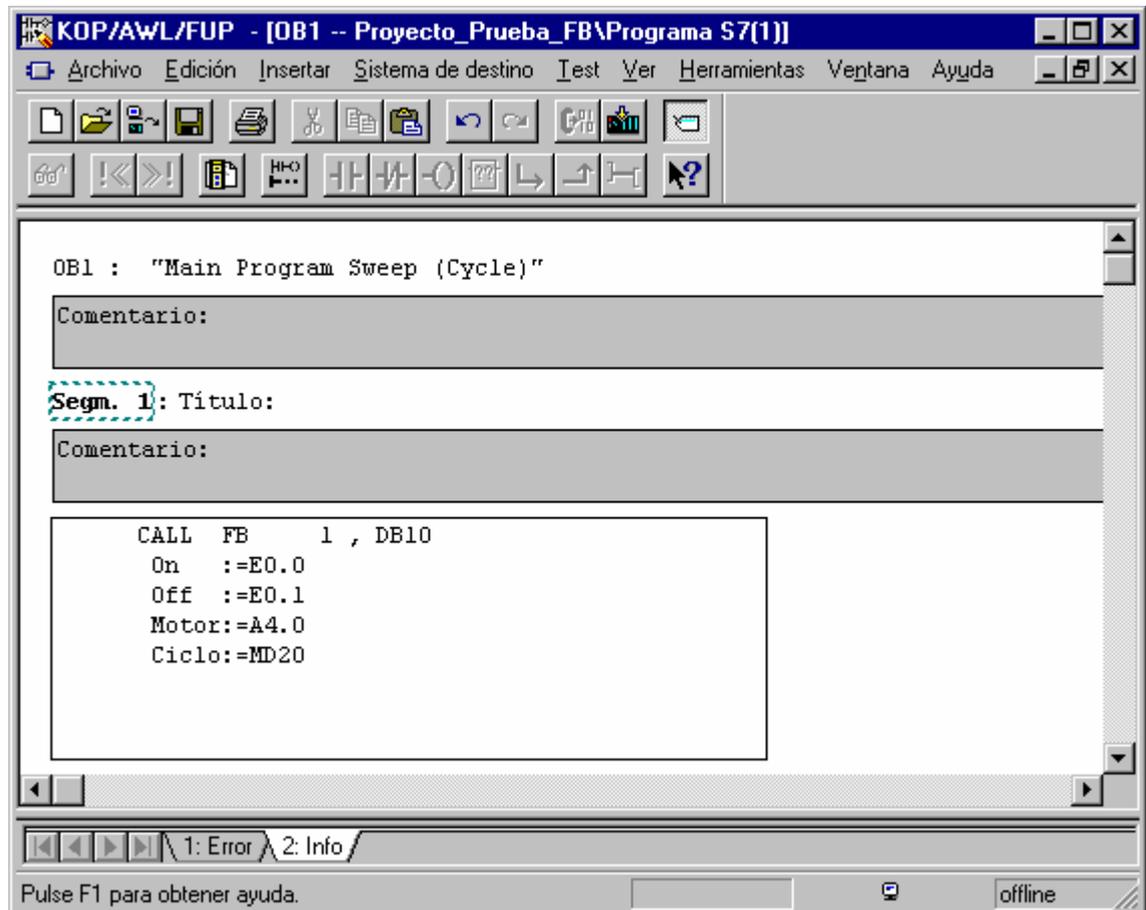


14. Seguidamente, se mostrarán todos los parámetros definidos en el FB del tipo **'Entrada'**, **'Salida y/o 'Entrada_Salida'**, para que puedan ser asociadas a direcciones de memoria de autómatas (p.e.: E100.0, MW2 etc ...).





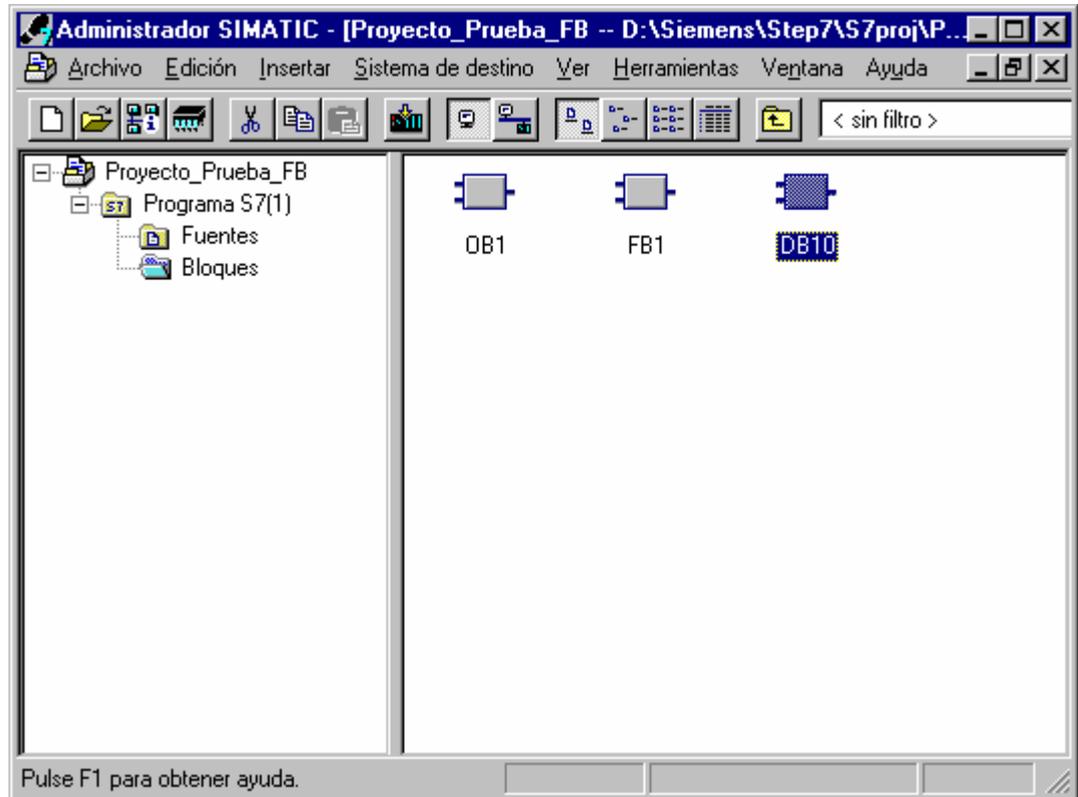
15. En nuestro ejemplo hemos llenado los campos según las condiciones del enunciado. El OB1 debe ser almacenado en el disco duro  y cargado en el PLC . El selector de modos del PLC debe estar en STOP! (→  → )



Nota: Bajo esta filosofía, el FB1 puede ser llamado varias veces, siempre que se haga con diferentes bloques de instancia y direcciones entrada/salida en cada una de las llamadas. Cada llamada representa a un bloque estándar diferente en este tipo especial de configuración de tareas.



16. Ahora en 'Administrador SIMATIC', el DB de instancia (ó DB local) 'DB10' se selecciona y carga en la CPU . El selector de modos de la CPU debe estar en STOP!(→ DB10 → )



17. Al poner el selector de modos en RUN, comienza la ejecución del programa. El motor arranca cuando se active la entrada E 0.0. Es parado al activar la entrada E 0.1. En la doble palabra de marcas MD20, se incrementa su valor cada vez que el fb1 es llamado desde el OB1. El contenido del MD20 se modificará a una velocidad muy alta, dado que un tiempo de ciclo es muy pequeño (milisegundos).