

Curso: “SIMATIC Step7 S7-300 Nivel 1”

Unidad 5

“Semana 5”

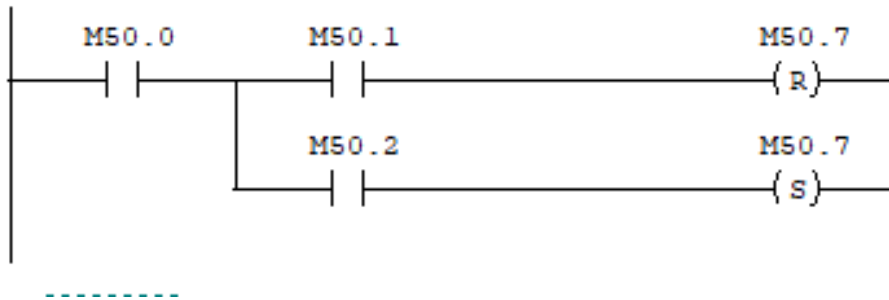
Instructor: D.Zarate Guillermo



Set-Reset

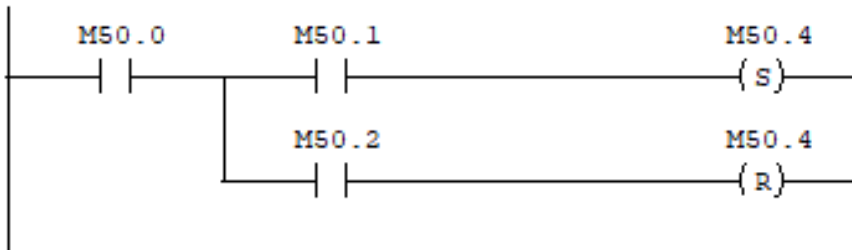
Con Bobina de SET/RESET

Segm. 4 : Título:



Este es Reset/Set. La prioridad es SET

Segm. 5 : Título:

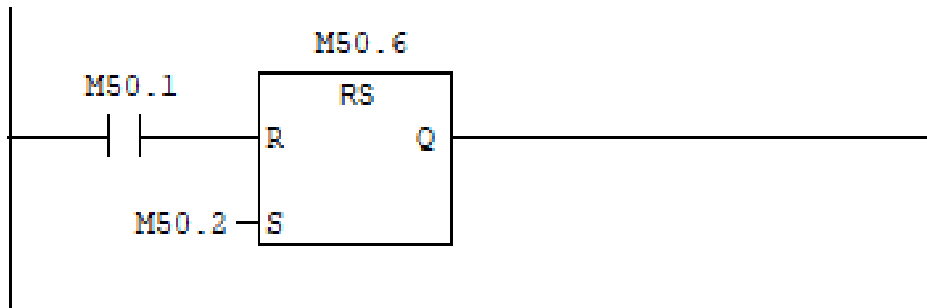


Este es Set/Reset. La prioridad es RESET

Set-Reset

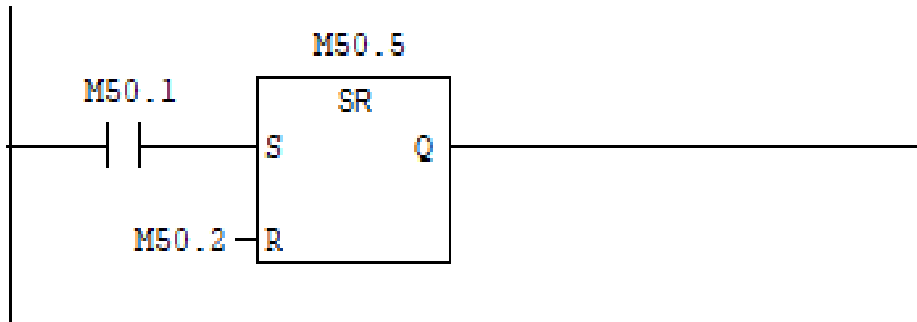
Con bloque de SET/RESET

Segm. 6 : Título:



Este es Reset/Set. La prioridad es SET

Segm. 7 : Título:



Este es Set/Reset. La prioridad es RESET

Bloque de Datos

Los bloques de datos (DB) se crean en el programa de usuario para almacenar los datos de los bloques lógicos. Todos los bloques del programa de usuario pueden acceder a los datos en un DB global. En cambio, un DB de instancia almacena los datos de un bloque de función (FB) específico.

El programa de usuario puede almacenar los datos en las distintas áreas de memoria de la CPU, p. ej. en las áreas de entradas (I), salidas (Q) y marcas (M). Además, es posible utilizar un bloque de datos (DB) para acceder rápidamente a los datos almacenados en el programa.

Los datos almacenados en un DB no se borran cuando se cierra el bloque de datos o cuando finaliza la ejecución del bloque lógico asociado. Hay dos tipos de DBs, a saber:

- Un DB global almacena los datos de los bloques lógicos en el programa. Cualquier OB, FB o FC puede acceder a los datos de un DB global.
- Un DB de instancia almacena los datos de un FB específico. La estructura de los datos en un DB de instancia refleja los parámetros (Input, Output e InOut) y los datos estáticos del FB. La memoria temporal del FB no se almacena en el DB de instancia.

Aunque el DB de instancia refleja los datos de un FB específico, cualquier bloque lógico puede acceder a los datos de un DB de instancia.

Bloque de Datos

Los DB en Step 7 son unas herramientas muy potentes para el almacenamiento y tratamiento de datos, de ahí que tengas que aprender una serie de conceptos que te serán útiles a la hora de realizar tus proyectos.

Los DB (Data Block) pueden ser:

- Globales
- De instancia

El acceso a los DB globales está pensado para que puedan ser usados desde cualquier parte del programa (desde un OB, FC o FB).

En cambio, los DB de instancia están asociados a los Bloques de función FB y almacenan los valores de las variables estáticas usadas en estos FB

Bloque de Datos

Los tipos de datos que nos podemos encontrar y definir en los DB en STEP son de dos tipos:

- *Simples
- *Compuestos

Los datos simples son los siguientes (con un ejemplo en su formato):

Bool: True/False

Byte: B#16#A

Word: W#16#432

DWord: DW#16#12300

INT: 124

DINT: L#130000

REAL: 0.45e+1

S5TIME: S5T#2s

TIME: T#1D5H

DATE: D#2012-12-24

TIME_OF_DAY:TOD#11:34:15

CHAR: 'B'

Hay que decir que la asignación de memoria se hace por palabras. Esto implica que si definimos 3 bools se ocuparan 2 bytes. Si usamos un byte se ocuparan igualmente 2 bytes. Por tanto, si se van a definir bools es conveniente definir hasta completar la palabra dejando los que no vayamos a usar como libres para futuros posibles usos.

Bloque de Datos

Los datos compuestos son:

DATE_AND_TIME: DT#12-12-24-11:34:1.0

STRING: 'Hola mundo'

ARRAY: Array[1..20]

STRUCT

UDT: UDT1

El uso de los datos simples es inmediato, pero en el caso de los compuestos, hay que hacer alguna consideración:

En el caso de los Strings habrá que definir su longitud de tal forma que la memoria reservada será mayor o menor según definamos cómo queremos que sea de largo.

Los arrays se usarán definiendo a posteriori de decir la longitud qué tipo de datos albergarán. Pueden ser datos simples o compuestos como por ejemplo otras estructuras.

Las estructuras no son un tipo propiamente ya que no almacenan información. Simplemente se indica la forma en la que se van a almacenar. Por ejemplo podremos crear un array de 10 posiciones de STRUCT y dentro de esta, almacenar un entero y un String[18] por poner un ejemplo de forma que cada posición del array tenga un número y un string.

Se pueden insertar UDT ya definidos con las precauciones que ya hemos comentado arriba. En la parte de definición al introducir un UDT sólo veremos que en el DB se reserva la memoria que ocupa el UDT. Si queremos ver las variables deberemos ponernos en modo visualizar datos.

Bloque de Datos

Los datos compuestos son:

DATE_AND_TIME: DT#12-12-24-11:34:1.0

STRING: 'Hola mundo'

ARRAY: Array[1..20]

STRUCT

UDT: UDT1

El uso de los datos simples es inmediato, pero en el caso de los compuestos, hay que hacer alguna consideración:

En el caso de los Strings habrá que definir su longitud de tal forma que la memoria reservada será mayor o menor según definamos cómo queremos que sea de largo.

Los arrays se usarán definiendo a posteriori de decir la longitud qué tipo de datos albergarán. Pueden ser datos simples o compuestos como por ejemplo otras estructuras.

Las estructuras no son un tipo propiamente ya que no almacenan información. Simplemente se indica la forma en la que se van a almacenar. Por ejemplo podremos crear un array de 10 posiciones de STRUCT y dentro de esta, almacenar un entero y un String[18] por poner un ejemplo de forma que cada posición del array tenga un número y un string.

Se pueden insertar UDT ya definidos con las precauciones que ya hemos comentado arriba. En la parte de definición al introducir un UDT sólo veremos que en el DB se reserva la memoria que ocupa el UDT. Si queremos ver las variables deberemos ponernos en modo visualizar datos.

Bloque de Datos

Propiedades - Bloque de datos

General - 1ª parte | General - 2ª parte | Llamadas | Atributos

Nombre y tipo: DB2 DB global

Nombre simbólico:

Comentario del símbolo:

Lenguaje: DB

Ruta del proyecto:

Ubicación del proyecto: C:\Program Files (x86)\Siemens\Step7\s7proj\Examen

Fecha de creación: 05/04/2020 09:50:58

Última modificación: 05/04/2020 09:50:58

Comentario:

Aceptar Cancelar Ayuda

DB1 -- "Estado del motor" -- Examen\Examen\CPU 315-2 PN/DP\...\DB1

Dirección	Nombre	Tipo	Valor inicial	Comentario
0.0		STRUCT		
+0.0	DB_VAR	INT	0	Variable comodín provisional
=2.0		END_STRUCT		

Función FB

Un bloque de función (FB) es como una subrutina con memoria. Un FB es un bloque lógico cuyas llamadas pueden programarse mediante parámetros del bloque. El FB almacena los parámetros de entrada (IN), salida (OUT), y entrada/salida (IN_OUT) en una memoria variable integrada en un bloque de datos (DB), o en un DB "instancia". El DB de instancia ofrece un bloque de memoria asociado a esa instancia (o llamada) del FB y almacena datos una vez que haya finalizado el FB.

Por lo general, los FBs se utilizan para controlar tareas o dispositivos cuya operación no finaliza dentro de un ciclo. Para almacenar los parámetros operativos de manera que sea posible acceder rápidamente a ellos de un ciclo a otro, todo FB del programa de usuario tiene uno o más DBs instancia. Cuando se llama un FB, se abre también un DB de instancia que almacena los valores de los parámetros del bloque y los datos locales estáticos de esa llamada o "instancia" del FB. Estos valores se almacenan en el DB de instancia una vez que finaliza el FB.

Los valores de arranque se asignan a los parámetros en la interfaz del FB. Estos valores se transfieren al DB de instancia asociado. Si no se asignan parámetros, se utilizan los valores almacenados actualmente en el DB de instancia. En algunos casos es necesario asignar parámetros.

Es posible asociar distintos DBs de instancia a diferentes llamadas del FB. Los DBs instancia permiten utilizar un FB genérico para controlar varios dispositivos. El programa se estructura de manera que un bloque lógico llame un FB y un DB de instancia. La CPU ejecuta entonces la lógica del programa en ese FB y almacena los parámetros del bloque y los datos locales estáticos en el DB de instancia. Cuando finaliza la ejecución del FB, la CPU regresa al bloque lógico que ha llamado el FB. El DB de instancia conserva los valores de esa instancia del FB. Si el FB se diseña para realizar tareas de control genéricas, es posible reutilizarlo para varios dispositivos, seleccionando diferentes DB de instancia para las distintas llamadas del FB.

Función FB

Un bloque **FB (Function Block)** es una subrutina la cual también puede contener una secuencia u operaciones dentro de él, y que puede ser llamado también desde otro bloque como OB, FC o FB. A diferencia del FC, éste bloque tiene asociado por default un DB de Instancia, en el cual los valores procedentes del FB son almacenados, a éstos datos se les llama ***ESTÁTICOS***, y solo los tienen los FBs.

Por tanto, al ser llamado el FB y ejecutada su lógica previamente programada, sus datos se almacenan en el DB de instancia asociado a este FB, por tanto, en la siguiente llamada del bloque, los datos inmediatos anteriores se vuelven a cargar, ya que están contenidos en el DB.

Función FB

Propiedades - Bloque de función

General - 1ª parte | General - 2ª parte | Llamadas | Atributos

Nombre: ☒ Apto para multiinstancia

Nombre simbólico:

Comentario del símbolo:

Lenguaje: ▼





Ruta del proyecto:

Ubicación del proyecto:

	Código	Interface
Fecha de creación:	05/04/2020 10:13:03	
Última modificación:	05/04/2020 10:13:03	05/04/2020 10:13:03

Comentario:

Aceptar Cancelar Ayuda

 Datos de sistema	---	---	---	SDB
 OB1	Principal	KOP	566	Bloque de organizaci...
 FB1	Motor	KOP	46	Bloque de función
 DB1	Estado del motor	DB	38	Bloque de datos

Función FB

Interface

IN

entrada

OUT

salida

IN_OUT

STAT

TEMP

Contenido de: 'Entorno\Interface'

Nombre
IN
OUT
IN_OUT
STAT
TEMP

FB1 : Título:

Comentario:

Segm. 1: Título:

#entrada

#entrada

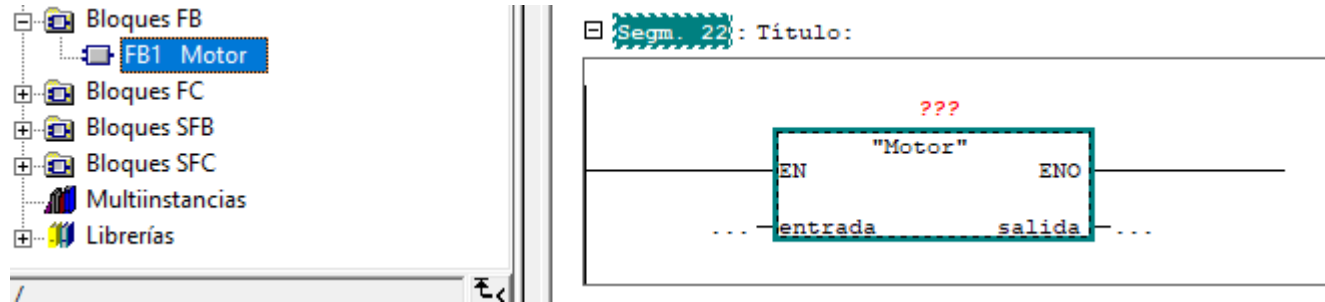
#salida

#salida

()

Se debe declarar entradas, salidas, variables locales. Luego programar lo que necesitamos

Función FB



Para utilizar un FB, se debe buscar en las opciones “Bloques FB” y buscar el FB que necesitamos. Al arrastrarlo al segmento de bloque que estemos (OB-FC-FB) nos va a solicitar las entradas y salidas de FB. Como así también un DB que tenemos que crear

Función FB

Propiedades - Bloque de datos

General - 1ª parte | General - 2ª parte | Llamadas | Atributos

Nombre y tipo: DB2 DB global

Nombre simbólico: DB global
DB de instancia

Comentario del símbolo:

Lenguaje: DB

Ruta del proyecto:

Ubicación del proyecto: C:\Program Files (x86)\Siemens\Step 7\s7proj\Examen

	Código	Interface
Fecha de creación:	05/04/2020 10:23:58	
Última modificación:	05/04/2020 10:23:58	05/04/2020 10:23:58

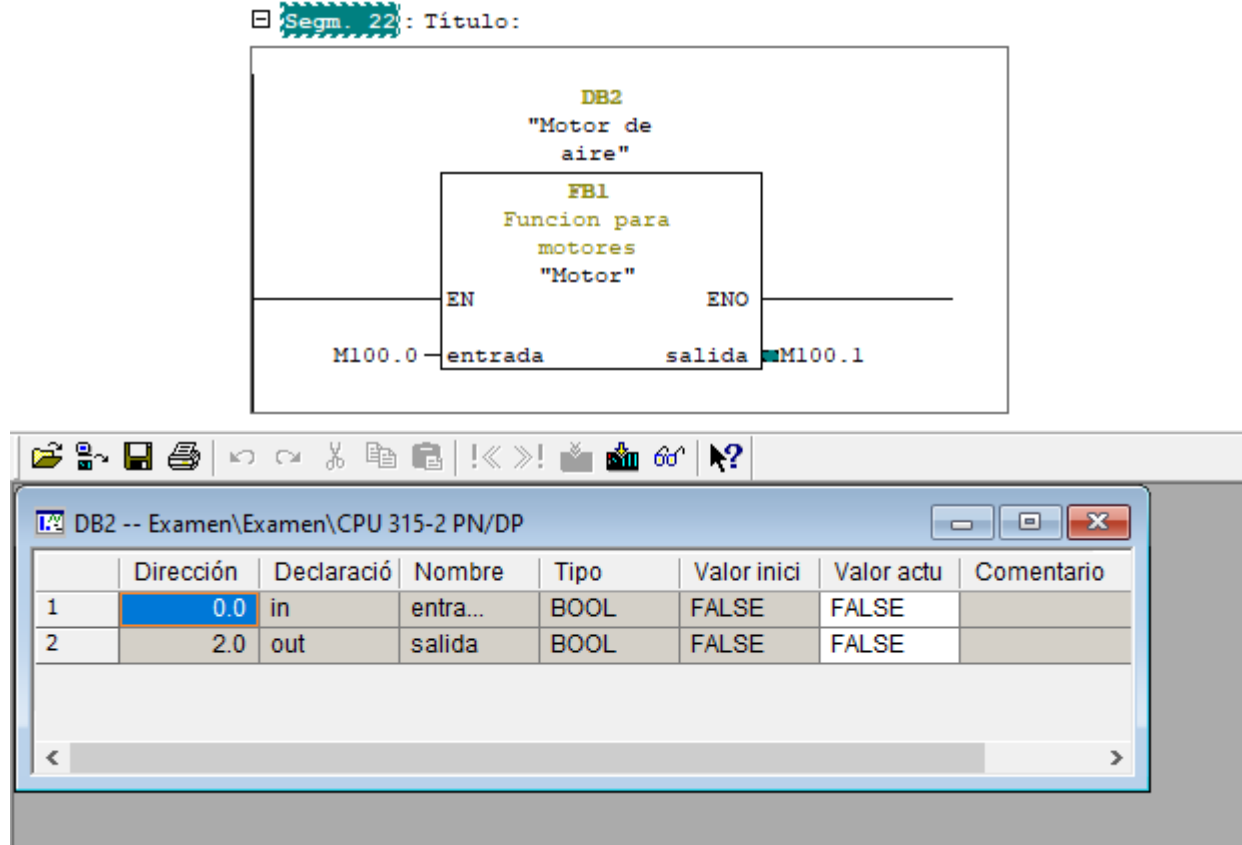
Comentario:

Aceptar Cancelar Ayuda

Para los FB se deben crear bloques de datos de instancia y asociarla a un FB

Nombre y tipo: DB2 DB de instancia FB1

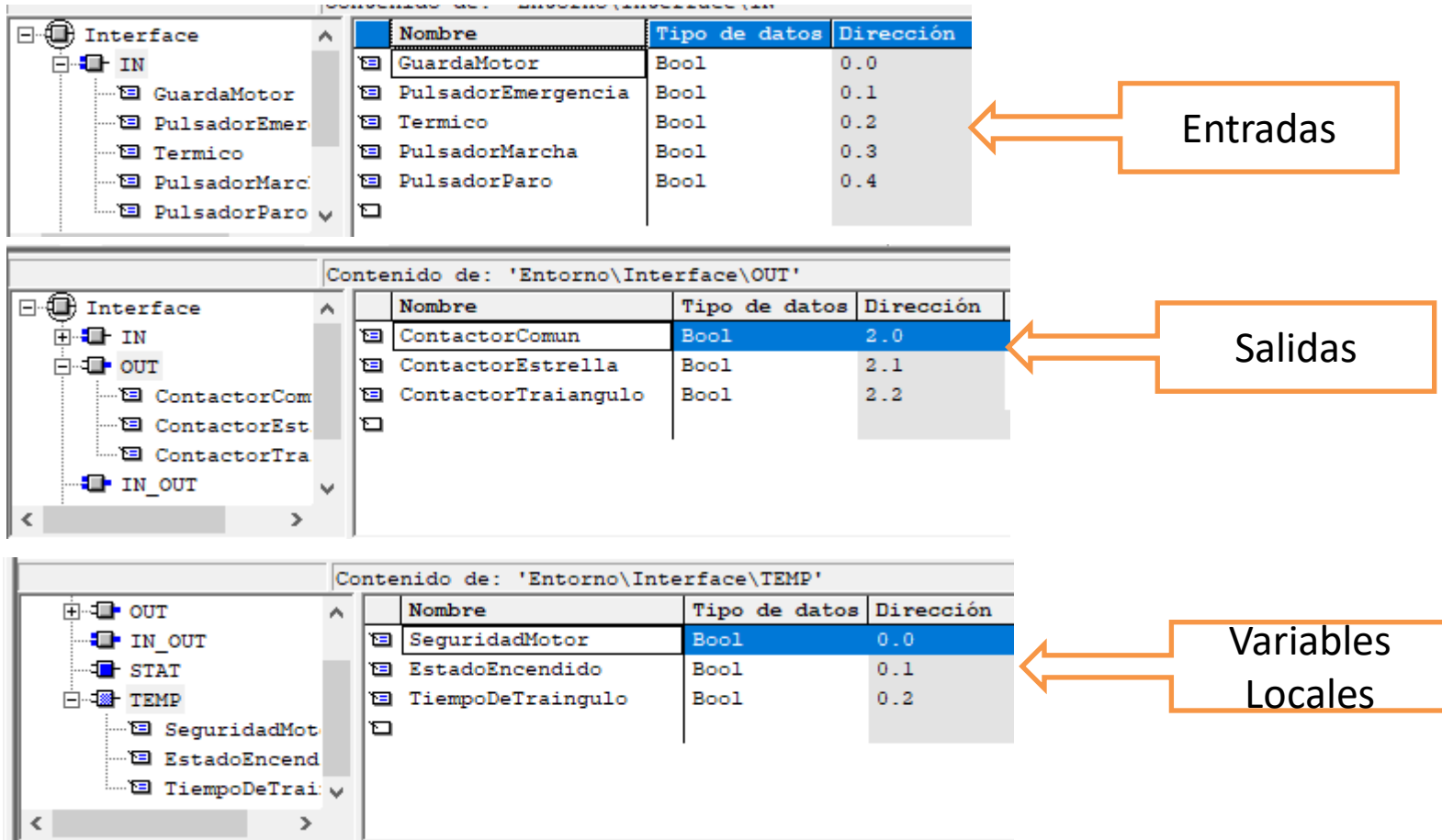
Función FB



De esta manera se ve el FB utilizado y el DB que se genero

Ejemplo de FB

Primero Diseñar FB



Entradas

Nombre	Tipo de datos	Dirección
GuardaMotor	Bool	0.0
PulsadorEmergencia	Bool	0.1
Termico	Bool	0.2
PulsadorMarcha	Bool	0.3
PulsadorParo	Bool	0.4

Salidas

Contenido de: 'Entorno\Interface\OUT'

Nombre	Tipo de datos	Dirección
ContactorComun	Bool	2.0
ContactorEstrella	Bool	2.1
ContactorTraiangulo	Bool	2.2

Variables Locales

Contenido de: 'Entorno\Interface\TEMP'

Nombre	Tipo de datos	Dirección
SeguridadMotor	Bool	0.0
EstadoEncendido	Bool	0.1
TiempoDeTraiangulo	Bool	0.2

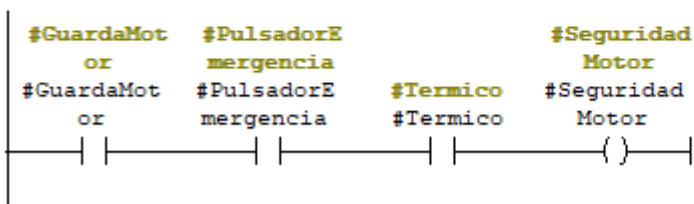
Ejemplo de FB

Primero Diseñar FB

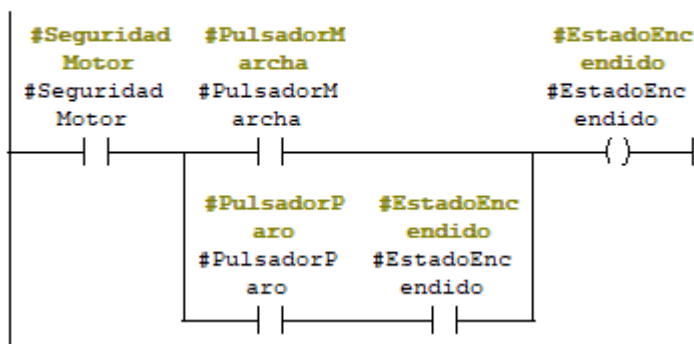
FB1 : Configuracion Motor Estrella/Traianguulo

En este bloque se se activa en un determinado tiempo la conexion estrella-trianguulo

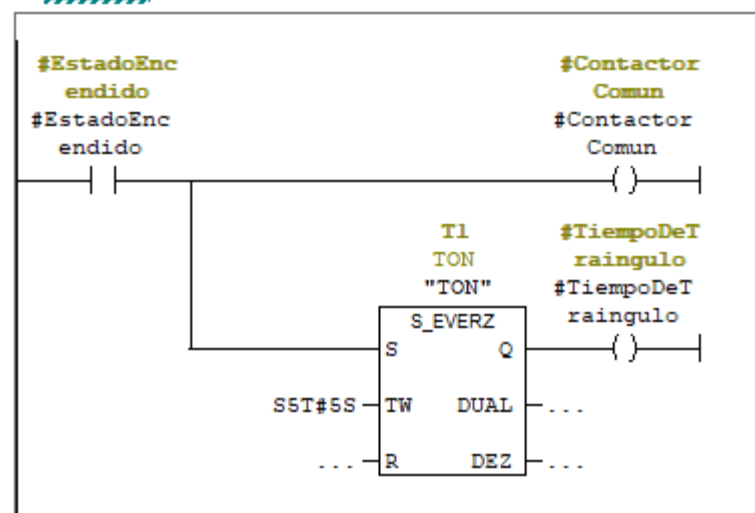
Segm. 1 : Seguridad del motor



Segm. 2 : Marcha o parada



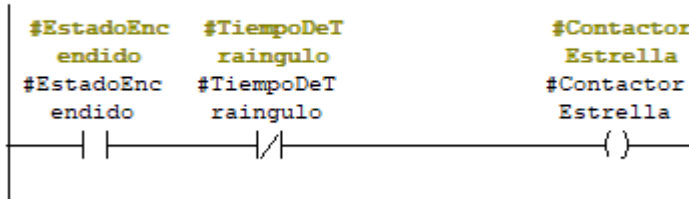
Segm. 3 : Contactor Comun



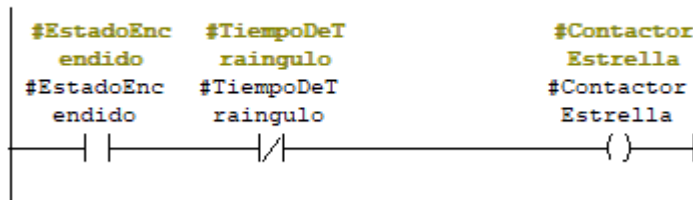
Ejemplo de FB

Primero Diseñar FB

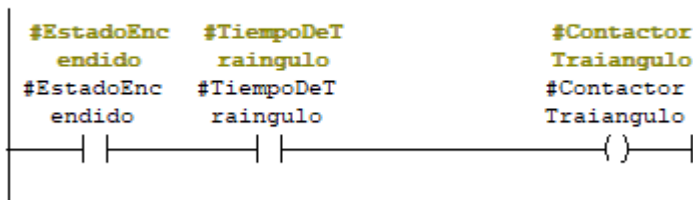
Segm. 4 : Contactor estrella



Segm. 4 : Contactor estrella



Segm. 5 : Contactor triangulo

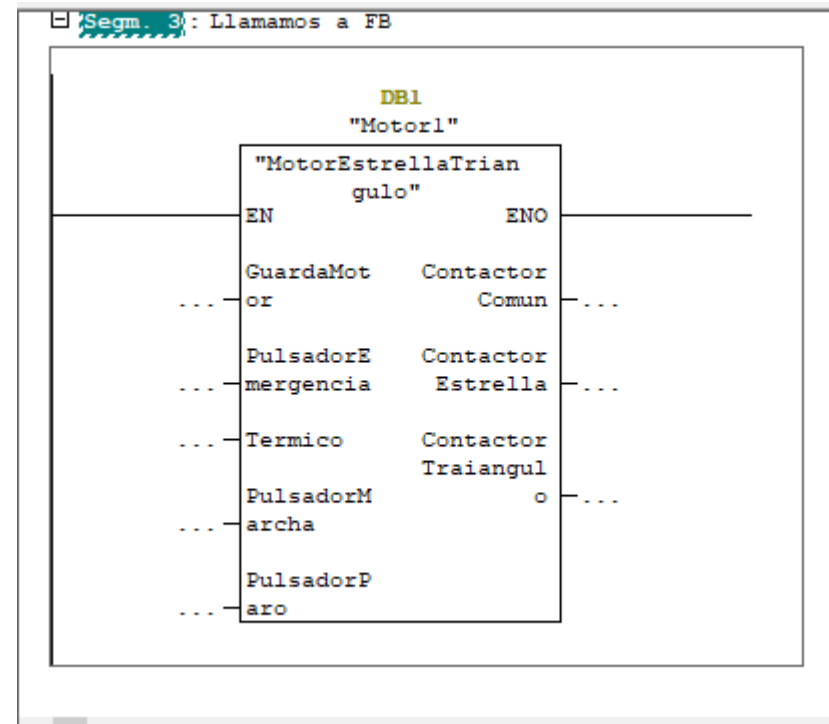


Ejemplo de FB

Creamos un DB de instancia

Nombre del objeto	Nombre simbólico	Lenguaje	Tamaño en la memor...	Tipo
OB1		KOP	174	Bloque de organización
FB1	MotorEstrellaTriangulo	KOP	144	Bloque de función
FC1	Triangulo estrella	KOP	116	Función
FC2	Giro de un motor	KOP	130	Función
DB1	Motor1	DB	40	DB de instancia del FB 1

Lo llamamos de OB1



Ejemplo de FB

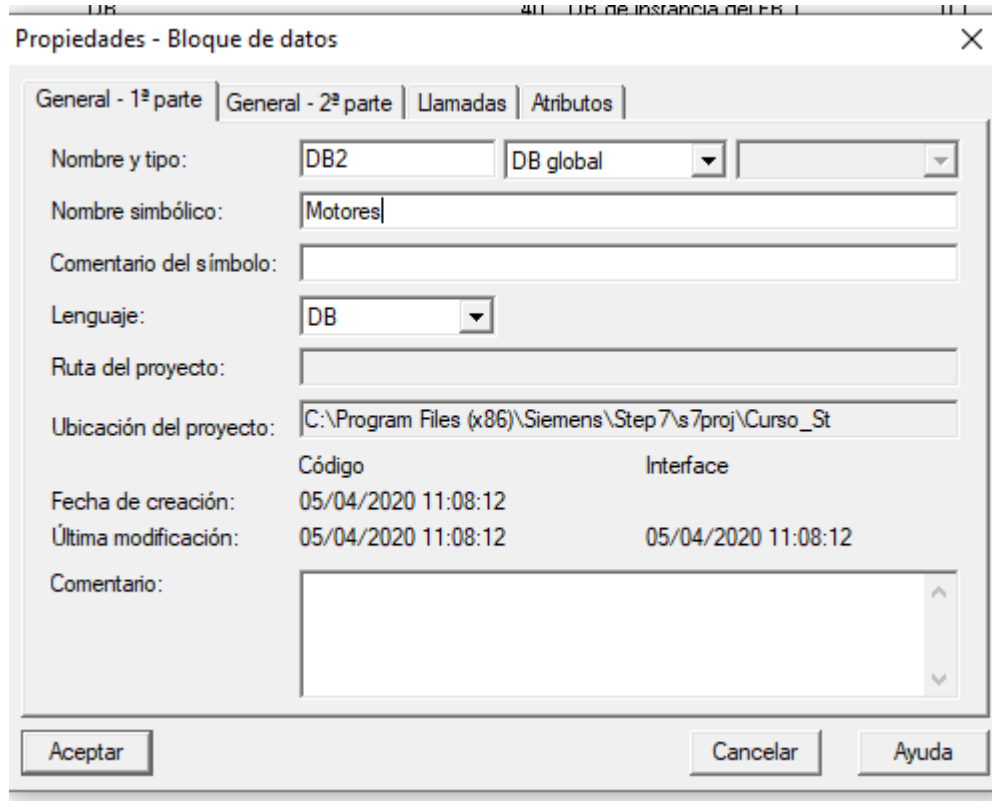
El DB quedaría de esta manera

DB1 -- Curso_Step7\STEP7\CPU 315-2 PN/DP

	Dirección	Declaració	Nombre	Tipo	Valor inici	Valor actu	Comentario
1	0.0	in	Guar...	BOOL	FALSE	FALSE	
2	0.1	in	Puls...	BOOL	FALSE	FALSE	
3	0.2	in	Term...	BOOL	FALSE	FALSE	
4	0.3	in	Puls...	BOOL	FALSE	FALSE	
5	0.4	in	Puls...	BOOL	FALSE	FALSE	
6	2.0	out	Cont...	BOOL	FALSE	FALSE	
7	2.1	out	Cont...	BOOL	FALSE	FALSE	
8	2.2	out	Cont...	BOOL	FALSE	FALSE	

Ejemplo de DB

Creamos un DB de Global



Propiedades - Bloque de datos

General - 1ª parte | General - 2ª parte | Llamadas | Atributos

Nombre y tipo: DB2 DB global

Nombre simbólico: Motores

Comentario del símbolo:

Lenguaje: DB

Ruta del proyecto:

Ubicación del proyecto: C:\Program Files (x86)\Siemens\Step7\s7proj\Curso_St

	Código	Interface
Fecha de creación:	05/04/2020 11:08:12	
Última modificación:	05/04/2020 11:08:12	05/04/2020 11:08:12

Comentario:

Aceptar Cancelar Ayuda

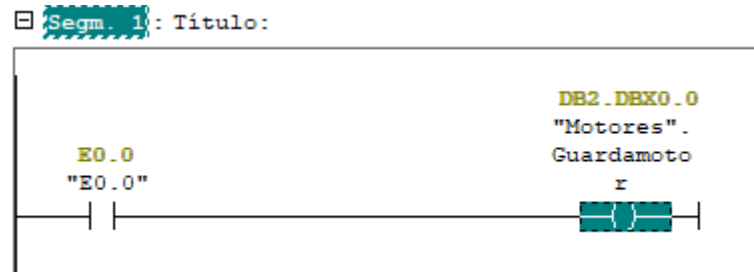
Ejemplo de DB

Creamos un DB de Global

Dirección	Nombre	Tipo	Valor inicial
0.0		STRUCT	
+0.0	Guardamotor	BOOL	FALSE
+0.1	PulsadorEmergencia	BOOL	FALSE
+0.2	TermicodelMotor	BOOL	FALSE
+0.3	PulsadorMarcha	BOOL	FALSE
+0.4	PulsadorParo	BOOL	FALSE
+0.5	ContactorComun	BOOL	FALSE
+0.6	ContactorEstrella	BOOL	FALSE
+0.7	ContactorTraingulo	BOOL	FALSE
=2.0		END_STRUCT	

Ejemplo de DB

Usar DB en un FC



DB2.DBX0.0

Formato en bit

DB2.DBB0

Formato en byte

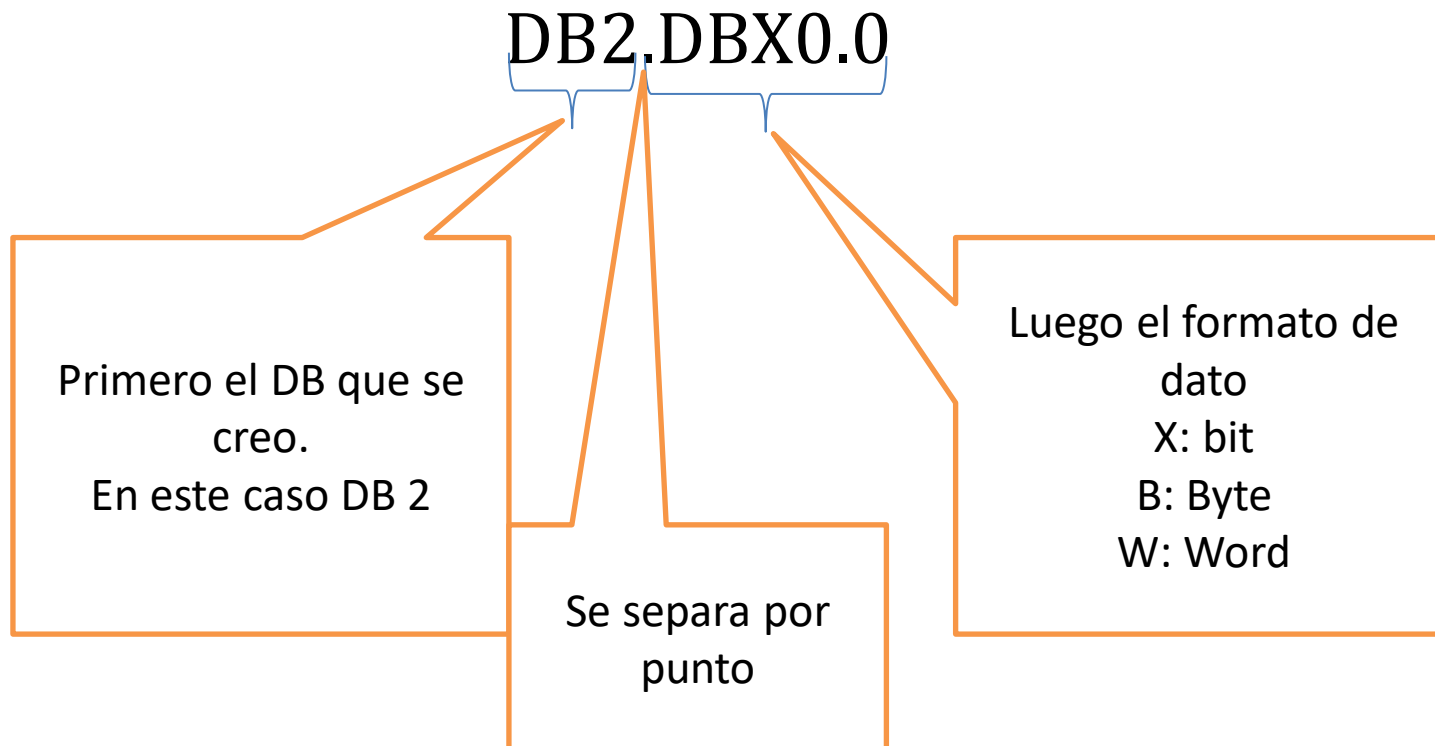
DB2.DBW0

Formato en Word

DB2.DBD0

Formato en doble Word

Ejemplo de DB



IMPORTANTE

Los DB se utilizan igual que un NA/NC (en caso de ser bit).
Para que cambien el valor se deben declarar una bobina de salida para que
tomen valor 0-1.

En caso de usar otro formato, podemos usar la sentencia MOVE.