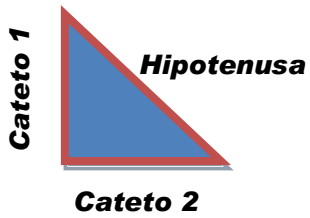
Control FPWIN Pro
Software Según el Estándar IEC 61131-3

Título	Funciones y Bloques de Funciones.
Versión	1.000

Introducción

En el siguiente documento se explica cómo crear una función y un bloque de función, diferencias entre ambas, cuándo se debe utilizar cada una, así como sus ventajas e inconvenientes.

Para ello se partirá del siguiente ejemplo: Cálculo de la hipotenusa en un triángulo rectángulo.



$$Hipotenusa = \sqrt{Cateto\ 1^2 + Cateto\ 2^2}$$

El uso de las funciones y bloques de funciones es aconsejable cuando se prevé que el código que se va a desarrollar se va a poder reutilizar en futuros programas o aplicaciones.

En ambos casos, la creación de una función o bloque de función consiste “en encapsular” un determinado código de programa para su reutilización en el mismo o diferente proyecto.

Procedimiento para la Creación de Funciones y Bloques de Función

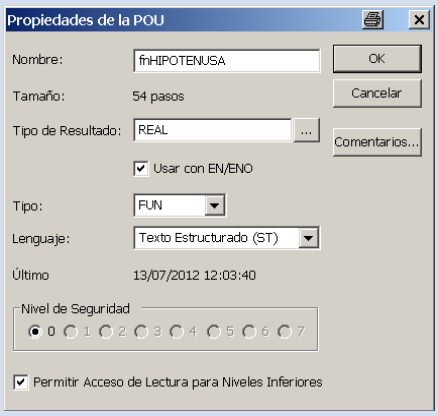
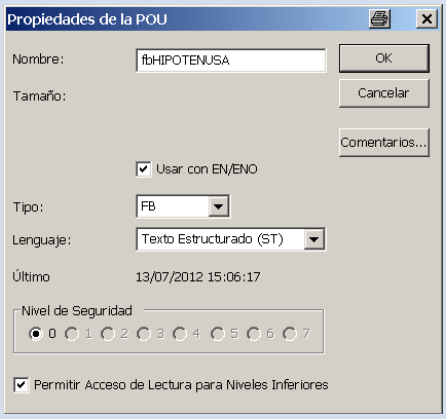
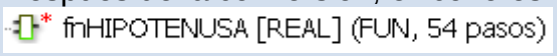
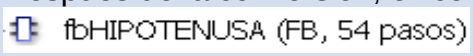
1.- Para crear una función o bloque de función, primero se ha de crear un programa estándar con el código necesario y comprobar su funcionamiento.

```
rHipotenusa:=SQRT(rCateto_1*rCateto_1 + rCateto_2**2);
```

2.- Una vez comprobado que funciona adecuadamente, se cierra la zona de edición del programa para poder modificar sus propiedades y convertirlo a FN (función) o FB (bloque de función).

3.- Se selecciona la POU a convertir y haciendo un clic con el botón derecho del ratón seleccionamos Propiedades de la POU.

4.- Convertir la POU de programa a FN o a FB tal y cómo se indica a continuación:

Función	Bloque de Función
Al convertir la POU a FUN, se tiene: 	Al convertir la POU a FB, se tiene: 
Se puede seleccionar una variable (en este caso REAL), cómo resultado de la función.	No es posible seleccionar una variable cómo resultado de la función.
Después de la conversión, el icono es: 	Después de la conversión, el icono es: 

Nota: Es muy importante no cambiar el lenguaje de programación. El FPWIN Pro **no convierte** los lenguajes. Si se cambia en propiedades el tipo de lenguaje se borra todo el código de programa, tras un mensaje de aviso previo.

5.- Se seleccionan las variables de entrada, salida o entrada-salida.

Se hace doble clic sobre el icono de FUN o FB para editar el bloque de función y se configuran las variables necesarias como variables de entrada VAR_INPUT, variables de salida VAR_OUTPUT o variables de entrada-salida VAR_IN_OUT.

- Variables de entrada: Aquellas variables que serán transferidas desde el flujo normal del programa al interior de la función y que son las necesarias para poder realizar las acciones propias de la función. Al finalizar la ejecución de la función, su valor sigue siendo el mismo (Cateto_1 y Cateto_2).
- Variables de salida: Aquellas variables calculadas en el interior de la función. Al finalizar la ejecución de la función, su valor se verá sobreescrito (hipotenusa).
- Variables de entrada-salida: Aquellas variables que serán transferidas desde el flujo normal del programa al interior de la función y que son las necesarias para poder realizar las acciones propias de la función. Además al finalizar la ejecución de la función o del bloque de función, su valor se sobreescribe.

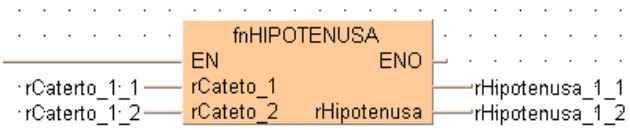
Nota: No todas las variables locales de una FN o FB han de ser de las anteriormente citadas. También pueden existir variables locales estandar e incluso variables globales.

	Clase	Identifica...	Tipo	I...
0	VAR_OUTPUT	rHipotenusa	REAL	0.0
1	VAR_INPUT	rCateto_1	REAL	0.0
2	VAR_INPUT	rCateto_2	REAL	0.0

Llamamiento a las funciones y bloques de función

Funciones

Ahora se puede usar la fnHIPOTENUSA de la lista estandar de funciones como cualquier otra con la siguiente apariencia:



Se observa que la función actual dispone de 2 salidas. Esto se debe a que el propio nombre de la función es una VAR_OUTPUT de la misma.

El motivo de asignar una salida a la función es porque se asume que las funciones se utilizan en tareas sencillas y se entiende que su propio nombre define su acción (ejemplo, sumar, restar, hipotenusa).

En este ejemplo, al convetir la POU de programa a función, se ha definido una salida del tipo REAL y se ha definido una variable **`rHipotenusa`** como VAR_OUTPUT.

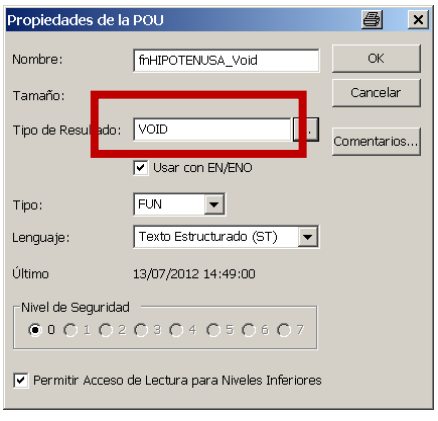
El interior actual de la función es:

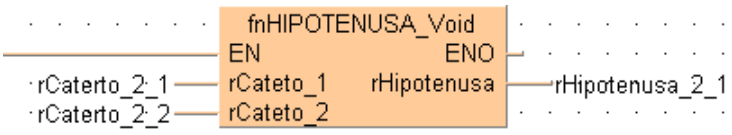
```
rHipotenusa:=SQRT(rCateto_1*rCateto_1 + rCateto_2**2);
```

Dado que en el ejemplo anterior no se ha asignado ningún valor a la variable **`fnHIPOTENUSA`** dentro de la función**, `rHipotenusa_1_1`** valdrá siempre 0 mientras que **`rHipotenusa_1_2`** será el resultado esperado.

La función Hipotenusa correctamente realizada se puede obtener de 2 formas:

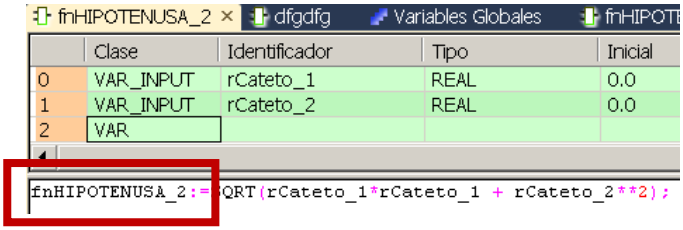
1.- Seleccionar VOID (vacío) como tipo de variable de salida a la hora de crear la función:

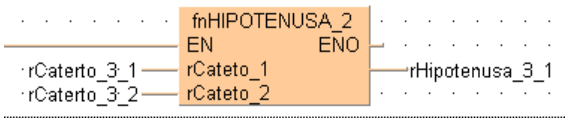




En este caso se visualiza el nombre de la salida de la función.

2.- Seleccionar el tipo de salida (REAL en este caso) como tipo de variable de salida a la hora de crear la función o convertir la POU a función y modificar el código de programa usando el nombre de la función como VAR_OUTPUT.



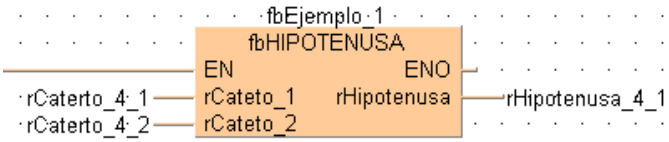


En este caso, se entiende que la salida es la hipotenusa porque así se llama la función

Tanto el punto 1 como el punto 2 dará el mismo resultado.

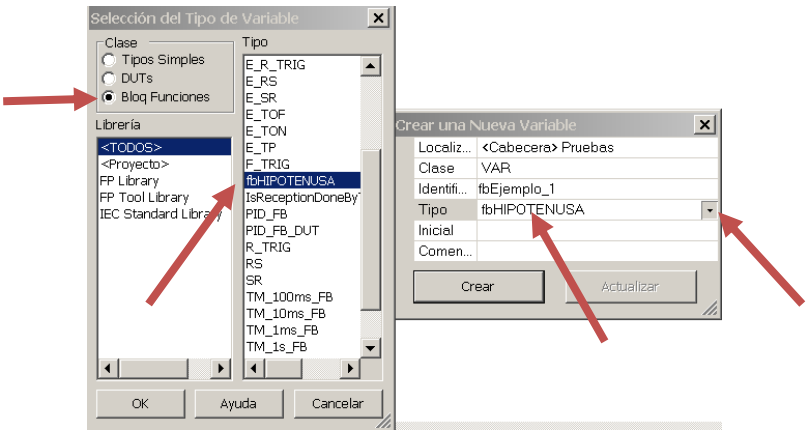
Bloques de Función

Si se ha creado un bloque de función, se puede usar la función fbHIPOTENUSA como cualquier otra del FPWInPro, teniendo la siguiente aparecencia:



Se observa que las FB requieren ser definidas (fbEJEMPLO_1) como variables normalmente locales. El tipo de una variable FB es del mismo nombre con el que se ha llamada al bloque de función, así para el caso de la hipotenusa se definirá del tipo **fbHIPOTENUSA**.

FB→Nombre del bloque de función (fbHIPOTENUSA)



Diferencias. Pros y Contras entre FN y FB

Características de la Función

1.- Una función es como ***UN SALTO A SUBRUTINA****. Al ser llamada a lo largo de un programa, es como si se tratase de una interrupción. Interrumpe la ejecución del programa, salta a donde se encuentra el código de la función, lo ejecuta y vuelve donde estaba.

2.- Se utiliza para ahorrar código de programa. Si se llama 100 veces a una función el código de programa ocupado será del tamaño de una única función.

3.- Todas sus variables internas se ven sobrescritas cada vez que es llamada, es por ello que no se pueden utilizar dentro de ellas otras funciones como temporizadores, contadores ni cualquier otro bloque de funciones (PID, etc). Si se llama varias veces a una función con las mismas variables de entrada, se obtiene siempre el mismo resultado.

4.- Normalmente se utiliza para códigos sencillos

**** SUBRUTINA:*** grupo de instrucciones escritas por separado del programa principal para realizar una función que puede ser usada repetidamente por el programa principal.

Características de los Bloques de Función

1.- Una bloque de función es como ***COPIAR Y PEGAR*** todo el código de la FB cuando es llamada. Las variables internas **no se ven sobrescritas** cuando llamamos varias veces a un bloque de función dado que la variable real es:

Nombre_de_función.Variable_del_Bloque ***fbEjemplo_1.rCateto_1***

Si se llama varias veces a un bloque de función con las mismas variables de entrada, no se tiene porque obtener siempre el mismo resultado.

2.- No sirve para ahorrar código. Se utiliza principalmente en los siguientes casos

2.1 Cuando se necesita introducir un bloque de función dentro de otra función (un caso muy típico es el uso de temporizadores dentro de una función).

2.3 En aquellos proyectos en los que existan muchos elementos idénticos de control. Ejemplo: una bodega de vinos en las que hay 50 cubas de fermentación. El proceso de fermentación es idéntico para todos ellos si bien varían simplemente las condiciones actuales de cada una y sus consignas. En el caso de ampliar la funcionalidad de la cuba de fermentación, bastará con cambiar el bloque de función para que afecte a todas las cubas.

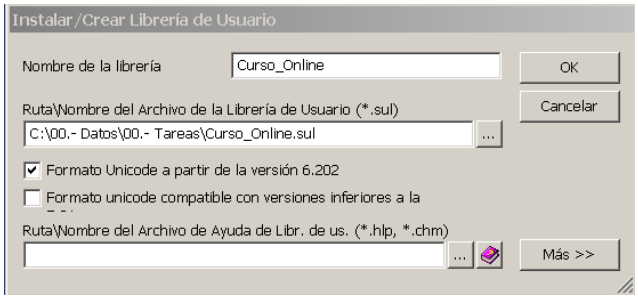
3.- Se suele utilizar para códigos de programa de longitud y complicación media/alta para clarificar su funcionamiento de un programa.

Librerías

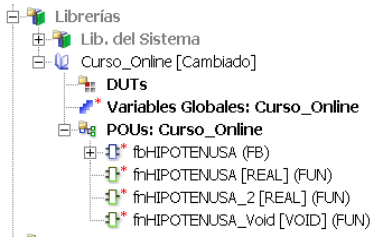
Se entiende por librería a la agrupación de códigos de programa, funciones y bloque de funciones para reutilización en el mismo o diferente proyecto.
Otra finalidad de la librería es la protección del "Know How" de un programa, mediante el uso de contraseñas de apertura de código.

Para crear una nueva librería:

Navegador del proyecto-→ Botón derecho sobre librería→ Librería→ Instalar/Crear una librería



Seleccionar la ruta de acceso y asignarle un nombre a la librería (Curso_Online) y de su fichero de ayuda (si existiese).
Se generará un fichero con extensión SUL en el que se podrá introducir las funciones cortándolas y pegándolas del proyecto normal.



Por último, se puede proteger con una contraseña

Navegador del proyecto-→ Botón derecho sobre Curso_Online→ Librería→ Cambiar Contraseña

No olvide cerrar la librería

Navegador del proyecto-→ Botón derecho sobre Curso_Online→ Librería→ Cerrar

Sobre Este Documento

Este documento no tiene carácter oficial ni se podrá responsabilizar a Panasonic Electric Works España por las erratas o información errónea contenida en el mismo, declinando toda responsabilidad por su utilización.