HMI Controller Magelis XBTGC, XBTGT, XBTGK

Funciones y variables del sistema
Guía de la biblioteca XBT PLCSystem

11/2015

EIO0000000630.07

www.schneider-electric.com

Schneider Electric

La información que se ofrece en esta documentación contiene descripciones de carácter general y/o características técnicas sobre el rendimiento de los productos incluidos en ella. La presente documentación no tiene como objeto sustituir dichos productos para aplicaciones de usuario específicas, ni debe emplearse para determinar su idoneidad o fiabilidad. Los usuarios o integradores tienen la responsabilidad de llevar a cabo un análisis de riesgos adecuado y completo, así como la evaluación y las pruebas de los productos en relación con la aplicación o el uso de dichos productos en cuestión. Ni Schneider Electric ni ninguna de sus filiales o asociados asumirán responsabilidad alguna por el uso inapropiado de la información contenida en este documento. Si tiene sugerencias de mejoras o modificaciones o ha hallado errores en esta publicación, le rogamos que nos lo notifique.

No se podrá reproducir este documento de ninguna forma, ni en su totalidad ni en parte, ya sea por medios electrónicos o mecánicos, incluida la fotocopia, sin el permiso expreso y por escrito de Schneider Electric.

Al instalar y utilizar este producto es necesario tener en cuenta todas las regulaciones sobre seguridad correspondientes, ya sean regionales, locales o estatales. Por razones de seguridad y para garantizar que se siguen los consejos de la documentación del sistema, las reparaciones solo podrá realizarlas el fabricante.

Cuando se utilicen dispositivos para aplicaciones con requisitos técnicos de seguridad, siga las instrucciones pertinentes.

Si con nuestros productos de hardware no se utiliza el software de Schneider Electric u otro software aprobado, pueden producirse lesiones, daños o un funcionamiento incorrecto del equipo.

Si no se tiene en cuenta esta información, se pueden causar daños personales o en el equipo.

Tabla de materias

Información de seguridad

Información importante

AVISO

Lea atentamente estas instrucciones y observe el equipo para familiarizarse con el dispositivo antes de instalarlo, utilizarlo, revisarlo o realizar su mantenimiento. Los mensajes especiales que se ofrecen a continuación pueden aparecer a lo largo de la documentación o en el equipo para advertir de peligros potenciales, o para ofrecer información que aclara o simplifica los distintos procedimientos.

La inclusión de este icono en una etiqueta “Peligro” o “Advertencia” indica que existe un riesgo de descarga eléctrica, que puede provocar lesiones si no se siguen las instrucciones.

Éste es el icono de alerta de seguridad. Se utiliza para advertir de posibles riesgos de lesiones. Observe todos los mensajes que siguen a este icono para evitar posibles lesiones o incluso la muerte.

⚠ PELIGRO

PELIGRO indica una situación de peligro que, si no se evita, **provocará** lesiones graves o incluso la muerte.

⚠ ADVERTENCIA

ADVERTENCIA indica una situación de peligro que, si no se evita, **podría provocar** lesiones graves o incluso la muerte.

⚠ ATENCIÓN

ATENCIÓN indica una situación peligrosa que, si no se evita, **podría provocar** lesiones leves o moderadas.

AVISO

AVISO indica una situación potencialmente peligrosa que, si no se evita, **puede provocar** daños en el equipo.

TENGA EN CUENTA LO SIGUIENTE:

La instalación, el manejo, las revisiones y el mantenimiento de equipos eléctricos deberán ser realizados sólo por personal cualificado. Schneider Electric no se hace responsable de ninguna de las consecuencias del uso de este material.

Una persona cualificada es aquella que cuenta con capacidad y conocimientos relativos a la construcción, el funcionamiento y la instalación de equipos eléctricos, y que ha sido formada en materia de seguridad para reconocer y evitar los riesgos que conllevan tales equipos.

Acerca de este libro

Presentación

Objeto

Este documento permite familiarizarse con las funciones y las variables de sistema ofrecidas dentro de los controladores XBTGC, XBTGK y XBTGT. La biblioteca XBT PLCSystem contiene funciones y variables para obtener información y enviar comandos al sistema XBT.

En esta documentación también se describen los tipos de datos asociados a las funciones y las variables de la biblioteca XBT PLCSystem.

Se necesita el siguiente conocimiento básico:

- Información básica sobre la funcionalidad, la estructura y la configuración de XBTGC, XBTGT y XBT GK HMI Controller
- Programación en lenguaje FBD, LD, ST, IL, SFC o CFC
- Variables del sistema (variables globales)

Campo de aplicación

Este documento se ha actualizado para la publicación de SoMachine V4.1 SP2.

Información relativa al producto

⚠ ADVERTENCIA

PÉRDIDA DE CONTROL

- El diseñador del esquema de control debe tener en cuenta las posibles modalidades de fallo de rutas de control y, para ciertas funciones de control críticas, proporcionar los medios para lograr un estado seguro durante y después de un fallo de ruta. Funciones de control críticas son, por ejemplo, una parada de emergencia y una parada de sobrerrecorrido, un corte de alimentación y un reinicio.
- Para las funciones de control críticas deben proporcionarse rutas de control separadas o redundantes.
- Las rutas de control del sistema pueden incluir enlaces de comunicación. Deben tenerse en cuenta las implicaciones de los retrasos de transmisión no esperados o los fallos en el enlace.
- Tenga en cuenta todas las reglamentaciones para la prevención de accidentes y las directrices de seguridad locales.[1]
- Cada implementación de este equipo debe probarse de forma individual y exhaustiva antes de entrar en servicio.

El incumplimiento de estas instrucciones puede causar la muerte, lesiones serias o daño al equipo.

[1] Para obtener información adicional, consulte NEMA ICS 1.1 (última edición), "Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control" (Directrices de seguridad para la aplicación, la instalación y el mantenimiento del control de estado estático) y NEMA ICS 7.1 (última edición), "Safety Standards for Construction and Guide for Selection, Installation and Operation of Adjustable-Speed Drive Systems" (Estándares de seguridad para la construcción y guía para la selección, instalación y utilización de sistemas de unidades de velocidad ajustable) o su equivalente aplicable a la ubicación específica.

⚠ ADVERTENCIA

FUNCIONAMIENTO IMPREVISTO DEL EQUIPO

- Utilice solo software aprobado por Schneider Electric para este equipo.
- Actualice el programa de aplicación siempre que cambie la configuración de hardware física.

El incumplimiento de estas instrucciones puede causar la muerte, lesiones serias o daño al equipo.

Terminología derivada de los estándares

Los términos técnicos, símbolos y las descripciones correspondientes del presente manual o que aparecen en la parte interior o exterior de los propios productos se derivan, por lo general, de los términos y las definiciones de estándares internacionales.

En el área de los sistemas de seguridad funcional, unidades y automatización general se incluyen, pero sin limitarse a ellos, términos como *seguridad*, *función de seguridad*, *estado de seguridad*, *fallo*, *reinicio tras fallo*, *avería*, *funcionamiento incorrecto*, *error*, *mensaje de error*, *peligroso*, etc.

Estos estándares incluyen, entre otros:

Estándar	Descripción
EN 61131-2:2007	Controladores programables, parte 2: Requisitos y ensayos de los equipos.
ISO 13849-1:2008	Seguridad de la maquinaria: partes de seguridad de los sistemas de control. Principios generales del diseño.
EN 61496-1:2013	Seguridad de la maquinaria: equipo de protección electrosensible. Parte 1: Requisitos y ensayos generales.
ISO 12100:2010	Seguridad de las máquinas. Principios generales para el diseño. Evaluación del riesgo y reducción del riesgo
EN 60204-1:2006	Seguridad de las máquinas. Equipo eléctrico de las máquinas. Parte 1: Requisitos generales
EN 1088:2008 ISO 14119:2013	Seguridad de la maquinaria. Dispositivos de bloqueo asociados con protecciones: principios de diseño y selección
ISO 13850:2006	Seguridad de la maquinaria. Parada de emergencia: principios de diseño
EN/IEC 62061:2005	Seguridad de la maquinaria. Seguridad funcional de los sistemas de control programable de seguridad eléctrica y electrónica

Estándar	Descripción
IEC 61508-1:2010	Seguridad funcional de sistemas de seguridad programable eléctricos y electrónicos: requisitos generales.
IEC 61508-2:2010	Seguridad funcional de los sistemas de seguridad electrónicos programables eléctricos y electrónicos: requisitos de los sistemas de seguridad electrónicos programables eléctricos y electrónicos.
IEC 61508-3:2010	Seguridad funcional de los sistemas de seguridad electrónicos programables eléctricos y electrónicos: requisitos de software.
IEC 61784-3:2008	Comunicación digital de datos para la medición y control: buses de campo de seguridad funcional.
2006/42/EC	Directiva de maquinaria
2004/108/EC	Directiva de compatibilidad electromagnética
2006/95/EC	Directiva de baja tensión

Además, los términos utilizados en este documento se pueden usar de manera tangencial porque se obtienen de otros estándares como:

Estándar	Descripción
Serie IEC 60034	Máquinas eléctricas giratorias
Serie IEC 61800	Accionamientos eléctricos de potencia de velocidad variable
Serie IEC 61158	Comunicación digital de datos para la medición y control - Bus de campo para su uso en Sistemas de control

Por último, el término *zona de operación* se puede utilizar junto con la descripción de peligros específicos, y se define tal cual para una *zona de peligro* o *zona peligrosa* en la *Directiva de maquinaria EC* (*EC/2006/42*) y *ISO 12100:2010*.

NOTA: Los estándares mencionados anteriormente podrían o no aplicarse a los productos específicos citados en la presente documentación. Para obtener más información en relación con los diferentes estándares aplicables a los productos descritos en este documento, consulte las tablas de características de las referencias de dichos productos.

Capítulo 1
Variables de sistema XBTGC, XBTGT y XBTGK

Descripción general

En este capítulo:
- se ofrece una introducción a las variables del sistema *(véase página 12)*.
- se describen las variables de sistema *(véase página 18)* incluidas en la biblioteca XBT PLCSystem

Contenido de este capítulo

Este capítulo contiene las siguientes secciones:

Sección 1.1
Variables del sistema: definición y uso

Descripción general

En esta sección se definen las variables del sistema y cómo implementarlas en Magelis XBTGC HMI Controller.

Contenido de esta sección

Esta sección contiene los siguientes apartados:

Descripción de las variables de sistema

Introducción

En esta sección se describe cómo se implementan las variables de sistema para el controlador. Estas variables tienen los atributos siguientes:

- Las variables de sistema permiten acceder a información general del sistema, realizar diagnósticos del sistema y realizar acciones sencillas.
- Las variables de sistema son variables estructuradas que cumplen las definiciones y las normas de asignación de nombres de IEC 61131. Puede accederse a ellas con el nombre simbólico IEC `PLC_GVL`.
- Algunas de las variables de `PLC_GVL` son de sólo lectura (por ejemplo, `PLC_R`), y otras son de lectura y escritura (por ejemplo, `PLC_W`).
- Las variables de sistema se declaran automáticamente como variables globales. Tienen un alcance en todo el sistema y deben gestionarse con cuidado porque se puede acceder a ellas desde cualquier unidad de organización de programa (POU) en cualquier tarea.

Normas de asignación de nombres de las variables de sistema

Las variables de sistema se identifican mediante:

- un nombre de estructura que representa la categoría de la variable del sistema (por ejemplo, `PLC_R` representa un nombre de estructura de variables de sólo lectura utilizadas para el diagnóstico del controlador).
- un conjunto de nombres de componentes que identifica el objetivo de la variable (por ejemplo, `i_wVendorID` representa el ID de fabricante del controlador).

Puede acceder a las variables escribiendo el nombre de estructura de las variables seguido del nombre del componente.

Aquí tiene un ejemplo de implementación de variables de sistema:

```
VAR
    myCtr_Serial : DWORD;
    myCtr_ID : DWORD;
    myCtr_FramesRx : UDINT;
END_VAR

myCtr_Serial := PLC_R.i_dwSerialNumber;
myCtr_ID := PLC_R.i_wVendorID;
myCtr_FramesRx := SERIAL_R[0].i_udiFramesReceivedOK;
```

NOTA: el nombre completo de la variable de sistema del ejemplo anterior es `PLC_GVL.PLC_R.i_wVendorID`. `PLC_GVL` es implícito al declarar una variable con **Accesibilidad**, pero también puede especificarse por completo. Las buenas prácticas de programación suelen dictar la utilización de un nombre de variable completo en las declaraciones.

Ubicación de variables de sistema

Se definen dos tipos de variables de sistema para utilizar al programar el controlador:

- variables ubicadas
- variables no ubicadas

Las variables ubicadas:

- Tienen una ubicación fija en un área %MW estática:
  - De %MW60000 a %MW60199 para variables de sistema de sólo lectura
  - De %MW62000 a %MW62199 para variables de sistema de lectura/escritura
- se utilizan en programas de SoMachine según la convención `nombre_estructura.nombre_componente` explicada anteriormente (se puede acceder directamente a las direcciones %MW entre 0 y 59.999; SoMachine considera las direcciones mayores que estos valores fuera de rango, por lo que solo se puede acceder a ellas a través de la convención `nombre_estructura.nombre_componente`).

Las variables no ubicadas:

- no están ubicadas físicamente en el área %MW
- no son accesibles a través de ninguna petición de bus de campo ni de red
- se utilizan en programas de SoMachine según la convención `nombre_estructura.nombre_componente` explicada anteriormente.

Utilización de variables de sistema

Introducción

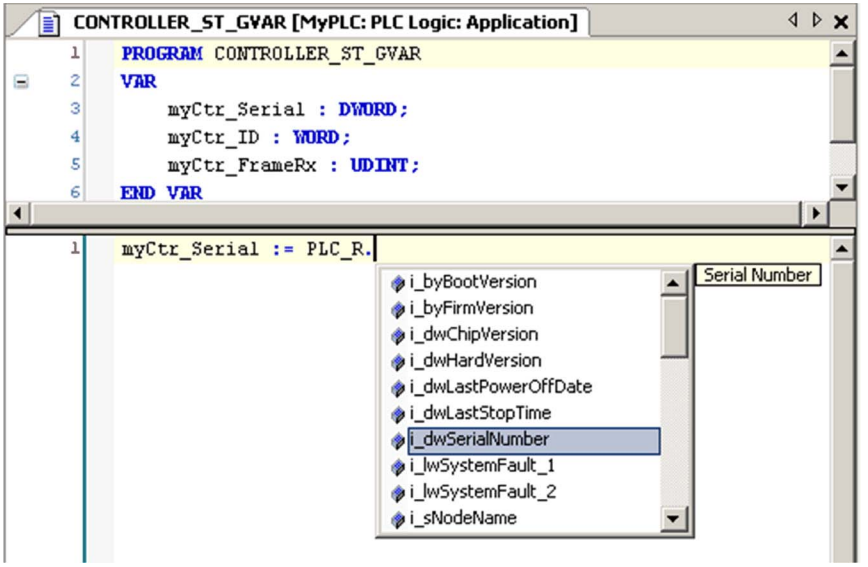
En este apartado se describen los pasos necesarios para programar y usar las variables de sistema en SoMachine.

Las variables de sistema son de ámbito global y pueden usarse en todas las unidades de organización del programa (POU) de la aplicación.

No es necesario declarar las variables de sistema en la Lista de variables globales (GVL). Se declaran automáticamente desde la biblioteca de sistema del controlador.

Utilización de variables de sistema en un POU

SoMachine tiene una función de autocompletado. En un **POU**, empiece por especificar el nombre de estructura de la variable de sistema (`PLC_R`, `PLC_W`...) seguido de un punto. Aparecerán las variables de sistema en **Accesibilidad**. Puede seleccionar la variable que desea o especificar el nombre completo manualmente.



NOTA: En el ejemplo anterior, tras introducir el nombre de estructura `PLC_R.`, SoMachine ofrece un menú desplegable de nombres/variables de componentes posibles.

Ejemplo

En el ejemplo siguiente se muestra el uso de algunas variables de sistema:

```
VAR
    myCtr_Serial : DWORD;
    myCtr_ID : WORD;
    myCtr_FramesRx : UDINT;
END_VAR
myCtr_Serial := PLC_R.i_dwSerialNumber;
myCtr_ID := PLC_R.i_wVendorID;
myCtr_FramesRx := SERIAL_R[0].i_udiFramesReceivedOK;
```

Sección 1.2
Estructuras PLC_R y PLC_W

Descripción general

En este apartado se describen las diferentes variables de sistema incluidas en las estructuras `PLC_R` y `PLC_W`.

Contenido de esta sección

Esta sección contiene los siguientes apartados:

PLC_R: Variables de sólo lectura del controlador

Estructura de variables

En la tabla siguiente se describen los parámetros de la variable del sistema `PLC_R` (tipo `PLC_R_STRUCT`):

Nombre de variable	Tipo	Comentario
`i_wVendorID`	`WORD`	ID de fabricante del controlador. 101A hex= Schneider Electric
`i_wProductID`	`WORD`	ID de referencia del controlador. **NOTA:** ID de fabricante e ID de referencia son los componentes del ID de destino del controlador mostrado en la vista Configuración de comunicación (ID del destino = 101A XXXX hex).
`i_byFirmVersion[0..3]`	`ARRAY[0..3] OF BYTE`	Controlador Firmware Versión `[aa.bb.cc.dd]`: ● i_byFirmVersion[0]= aa ● ... ● i_byFirmVersion[3]= dd
`i_byBootVersion[0..3]`	`ARRAY[0..3] OF BYTE`	Versión de inicio del controlador `[aa.bb.cc.dd]`: ● i_byBootVersion[0]= aa ● ... ● i_byBootVersion[3]= dd
`i_dwHardVersion`	`DWORD`	Versión de hardware del controlador.
`i_wStatus`	`PLC_R_STATUS` *(véase página 40)*	Estado del controlador.
`i_wBootProjectStatus`	`PLC_R_BOOT_PROJECT_STATUS` *(véase página 39)*	Devuelve información sobre la aplicación de inicio almacenada en la memoria FLASH.
`i_wLastStopCause`	`PLC_R_STOP_CAUSE` *(véase página 41)*	Causa de la última transición desde el estado EJECUTAR a otro estado.
`i_wLastApplicationError`	`PLC_R_APPLICATION_ERROR` *(véase página 37)*	Se ha detectado la causa de la última excepción del controlador.
`i_wIOStatus1`	`PLC_R_IO_STATUS` *(véase página 36)*	Estado de E/S incrustadas. **NOTA:** Esto es válido sólo para XBT GC.
`i_wIOStatus2`	`PLC_R_IO_STATUS` *(véase página 36)*	Estado de E/S de TM2. **NOTA:** Esto es válido sólo para XBT GC.

Nombre de variable	Tipo	Comentario
i_dwAppliSignature1	DWORD	Primera DWORD de la firma de 4 DWORD (16 bytes en total). El software genera la firma de la aplicación durante la compilación.
i_dwAppliSignature2	DWORD	Segunda DWORD de la firma de 4 DWORD (16 bytes en total). El software genera la firma de la aplicación durante la compilación.
i_dwAppliSignature3	DWORD	Tercera DWORD de la firma de 4 DWORD (16 bytes en total). El software genera la firma de la aplicación durante la compilación.
i_dwAppliSignature4	DWORD	Cuarta DWORD de la firma de 4 DWORD (16 bytes en total). El software genera la firma de la aplicación durante la compilación.

PLC_W: Variables de sistema de lectura/escritura del controlador

Estructura de variables

En la tabla siguiente se describen los parámetros que contiene la variable del sistema `PLC_W` (tipo `PLC_W_STRUCT`):

Nombre de variable	Tipo	Comentario
`q_uiOpenPLCControl`	`UINT`	Cuando el valor pasa de 0 a 6.699 se ejecuta el comando previamente escrito en el `PLC_W.q_wPLCControl` siguiente.
`q_wPLCControl`	`PLC_W_COMMAND` *(véase página 43)*	El comando EJECUTAR/DETENER del controlador se ha ejecutado cuando el valor de la variable de sistema `PLC_R.q_uiOpenPLCControl` ha pasado de 0 a 6.699.

Sección 1.3
Estructuras SERIAL_R y SERIAL_W

Descripción general

En este apartado se enumeran y describen las diversas variables del sistema incluidas en las estructuras `SERIAL_R` y `SERIAL_W`.

Contenido de esta sección

Esta sección contiene los siguientes apartados:

SERIAL_R[0..1]: Variables de sólo lectura de línea serie

Introducción

SERIAL_R es una matriz de 2 elementos del tipo SERIAL_R_STRUCT. Cada elemento de la matriz devuelve las **variables de sistema** de diagnóstico para la línea serie correspondiente:

- Serial_R[0] hace referencia al puerto COM1 (no corresponde a XBTGC1100).
- Serial_R[1] hace referencia al puerto COM2 (no corresponde a la serie XBTGC).

Estructura de variables

En la tabla siguiente se describen los parámetros de la variable de sistema de SERIAL_R[0..1]:

Nombre de variable	Tipo	Comentario
i_udiFramesTransmittedOK	UDINT	Número de tramas transmitidas correctamente.
i_udiFramesReceivedOK	UDINT	Número de tramas recibidas sin error detectado.
i_udiRX_MessagesError	UDINT	Número de tramas recibidas con errores detectados (suma de comprobación, paridad).

NOTA:
Los contadores de SERIAL_R se restablecen en las situaciones siguientes:

- Descarga
- Restablecimiento del controlador
- Comando SERIAL_W[x].q_wResetCounter
- Comando de restablecimiento por código de función de petición Modbus núm. 8.

SERIAL_W[0..1]: Variables de sistema de lectura/escritura de la línea serie

Introducción

SERIAL_W es una matriz de 2 elementos del tipo SERIAL_W_STRUCT. Cada elemento de la matriz fuerza las **variables del sistema** SERIAL_R para restablecer la línea serie correspondiente:

- Serial_W[0] hace referencia al puerto COM1 (no corresponde a la serie XBTGC1100).
- Serial_W[1] hace referencia al puerto COM2 (no corresponde a la serie XBTGC).

Estructura de variables

En la tabla siguiente se describen los parámetros de la variable de sistema de SERIAL_W[0..1]:

Nombre de variable	Tipo	Comentario
q_wResetCounter	WORD	La transición de 0 a 1 restablece todos los contadores de SERIAL_R[0..1]. Para volver a restablecer los contadores es necesario escribir este registro en 0 antes de que pueda llevarse a cabo otra transición de 0 a 1.

Capítulo 2
Funciones de sistema XBTGC, XBTGT y XBTGK

Sección 2.1
Funciones de lectura XBTGC, XBTGT y XBTGK

Descripción general

En este apartado se describen las funciones de lectura incluidas en la biblioteca PLCSystem de XBTGC.

Contenido de esta sección

Esta sección contiene los siguientes apartados:

HMI_GetRightBusStatus: Devuelve el estado del bus de ampliación

Descripción de la función

Esta función devuelve el estado del bus de ampliación de E/S en un campo de bit. A continuación del parámetro de entrada (`Mask`), esta función devuelve un diagnóstico de configuración en el bus de ampliación de E/S (discrepancia de configuración del bus de ampliación de E/S con los módulos conectados) o un diagnóstico detallado del módulo de E/S analógicas de ampliación solicitado.

NOTA: La función `HMI_GetRightBusStatus` solo es aplicable a XBT GC HMI Controller.

Representación gráfica

Representación IL y ST

Para ver la representación general en lenguaje IL o ST, consulte el capítulo *Representación de las funciones y los bloques de funciones* *(véase página 47)*.

Descripción de variables de E/S

En la tabla siguiente se describe el parámetro de entrada:

Entrada	Tipo	Comentario
`Máscara`	`RIGHTBUS_GET_STATUS` *(véase página 44)*	Define el tipo de diagnóstico del bus de ampliación de E/S devuelto por la función: configuración de bus o uno de los tres posibles módulos de ampliación.

En la tabla siguiente se describe la variable de salida:

Salida	Tipo	Comentario
`HMI_GetRightBusStatus`	`WORD`	Diagnóstico del bus de ampliación de E/S. Consulte los detalles a continuación.

Diagnóstico genérico del bus de ampliación de E/S

En la tabla siguiente se describe el campo de bits que devuelve la función `HMI_GetRight-BusStatus` cuando el parámetro de entrada (`Mask`) es `RIGHT_BUS_GET_GEN_STATUS` *(véase página 44)* para un diagnóstico de configuración del bus de ampliación (0000 hex si no se detecta ningún error):

Bit	Descripción
0	Reservado (siempre 0)
1	TRUE si la configuración del módulo 1 TM2 no coincide con el módulo conectado.
2	TRUE si la configuración del módulo 2 TM2 no coincide con el módulo conectado.
3	TRUE si la configuración del módulo 3 TM2 no coincide con el módulo conectado.
4...15	Reservado (siempre 0)

Diagnóstico de los módulos del bus de ampliación de E/S

En la tabla siguiente se describe el campo de bit que devuelve la función `HMI_GetRight-BusStatus` cuando el parámetro de entrada (`Mask`) es `RIGHTBUS_GET_STATUSx` *(véase página 44)* (donde x = 1 a 3) para un diagnóstico detallado del módulo de ampliación "x".

NOTA: El diagnóstico detallado sólo es válido para los módulos de E/S analógicas y el significado del campo de bit depende del tipo de módulo analógico correspondiente.

Cuando el módulo asociado es un módulo de E/S analógicas estándar (hasta 2 canales de entrada):

- TM2AMM3HT
- TM2ALM3LT
- TM2AMI2HT
- TM2AMI2LT
- TM2AVO2HT
- TM2AMO1HT

Bit	Descripción
0	Todos los canales analógicos se encuentran en estado normal.
1	Módulo en estado de inicialización
2	La fuente de alimentación no funciona correctamente.
3	Configuración incorrecta: es necesario realizar un análisis.
4	Proceso de conversión para el canal de entrada 0
5	Proceso de conversión para el canal de entrada 1
6	Parámetro no válido para el canal de entrada 0.
7	Parámetro no válido para el canal de entrada 1.

Bit	Descripción
8	No utilizado
9	No utilizado
10	Valor de desborde para el canal de entrada 0.
11	Valor de desborde para el canal de entrada 1.
12	Valor de transgresión por debajo del rango para el canal de entrada 0.
13	Valor de transgresión por debajo del rango para el canal de entrada 1.
14	No utilizado
15	Parámetro no válido para el canal de salida.

Cuando el módulo asociado es uno de los 4 u 8 módulos de canales analógicos de entrada:
- TM2ARI8HT
- TM2AMI8HT
- TM2ARI8LT
- TM2ARI8LRJ
- TM2AMI4LT
- TM2AMM6HT

Bit	Descripción	Significado
0, 1	Estado del canal 0	00: canal analógico en estado normal 01: parámetro no válido para el canal de entrada 10: valor de entrada no disponible (módulo en estado de inicialización, proceso de conversión) 11: valor no válido para el canal de entrada (valor de desborde o de subdesbordamiento)
2, 3	Estado del canal 1	consulte el bit 0, 1
4, 5	Estado del canal 2	consulte el bit 0, 1
6, 7	Estado del canal 3	consulte el bit 0, 1
8, 9	Estado del canal 4	consulte el bit 0, 1 (sólo para módulos de 8 canales de entrada)
10, 11	Estado del canal 5	consulte el bit 0, 1 (sólo para módulos de 8 canales de entrada)
12, 13	Estado del canal 6	consulte el bit 0, 1 (sólo para módulos de 8 canales de entrada)
14, 15	Estado del canal 7	consulte el bit 0, 1 (sólo para módulos de 8 canales de entrada)

NOTA: Cuando el módulo de ampliación de destino es un módulo de E/S digital, el diagnóstico que se devuelve no es válido (0000 hex).

Ejemplo

En el ejemplo siguiente se describe un método en el que se utiliza `HMI_GetRightBusStatus` para el diagnóstico de los módulos y el bus de ampliación de E/S:

```
VAR
(*Diagnóstico de config. de los módulos de 1 a 3 = MyRightBusStatus bits
de 1 a 3*)
MyRightBusStatus: WORD;
(*Códigos de diagnóstico de los módulos de 1 a 3*)
    ModuleError:Array [1..3] of WORD;
END_VAR

(*Diagnóstico de config. en el bus de ampliación*)
MyRightBusStatus:=HMI_GetRightBusStatus(RIGHTBUS_GET_GEN_STATUS);
IF MyRightBusStatus<>0 THEN
(*Discrepancia detectada en la config. => establecer una alarma,
comprobar los valores de bits...*)END_IF;
(*Obtener diagnóstico de módulos: error detectado si se realiza el
diagnóstico <> 0*)
(*Limitar la lista solo a los módulos analógicos configurados*)
ModuleError[1]:=HMI_GetRightBusStatus(RIGHTBUS_GET_STATUS1);
ModuleError[2]:=HMI_GetRightBusStatus(RIGHTBUS_GET_STATUS2);
ModuleError[3]:=HMI_GetRightBusStatus(RIGHTBUS_GET_STATUS3);
```

HMI_IsFirstMastColdCycle: Indica si el ciclo es el primer ciclo MAST del arranque en frío

Descripción de funciones

Esta función devuelve TRUE durante el primer ciclo MAST después de un arranque en frío (primer ciclo tras la descarga o el restablecimiento en frío).

Representación gráfica

Representación en IL y ST

Para ver la representación general en lenguaje IL o ST, consulte el capítulo *Representación de las funciones y los bloques de funciones* *(véase página 47)*.

Descripción de la variable de E/S

En la tabla siguiente se describe la variable de salida:

Salida	Tipo	Comentario
`HMI_IsFirstMastColdCycle`	`BOOL`	TRUE durante el primer ciclo de tarea MAST después de un arranque en frío.

Ejemplo

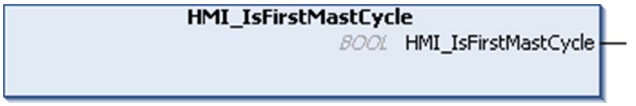
Consulte la función `HMI_IsFirstMastCycle` *(véase página 32)*.

HMI_IsFirstMastCycle: indica si el ciclo es el primer ciclo MAST

Descripción de funciones

Esta función devuelve TRUE durante el primer ciclo MAST después de un arranque.

Representación gráfica



Representación IL y ST

Para ver la representación general en lenguaje IL o ST, consulte el capítulo *Representación de las funciones y los bloques de funciones* *(véase página 47)*.

Descripción de la variable de E/S

Salida	Tipo	Comentario
`HMI_IsFirstMastCycle`	`BOOL`	TRUE durante el primer ciclo de tarea MAST después de un arranque.

Ejemplo

En este ejemplo se describen las tres funciones `HMI_IsFirstMastCycle`, `HMI_IsFirstMastColdCycle` y `HMI_IsFirstMastWarmCycle` utilizadas juntas:

NOTA: Las funciones se deben utilizar en la tarea MAST, pues de lo contrario las acciones de inicialización se pueden ejecutar más de una vez o nunca (una tarea adicional puede llamarse varias veces o no llamarse durante un ciclo de tarea MAST):

```
IF HMI_IsFirstMastWarmCycle() THEN
(*Este es el primer ciclo MAST después de un arranque en caliente: todas las variables se establecen en sus valores de inicialización excepto las variables retenidas*)
(*=> inicializa las variables necesarias para que la aplicación funcione de la forma esperada en este caso*)
END_IF;
IF HMI_IsFirstMastColdCycle() THEN
(*Este es el primer ciclo MAST después de un arranque en frío: todas las variables se establecen en sus valores de inicialización las variables de retención*)
 (*=> inicializa las variables necesarias para que la aplicación funcione de la forma esperada en este caso*)
END_IF;
IF HMI_IsFirstMastCycle() THEN
 (*Este es el primer ciclo MAST después de un arranque, en frío o en caliente, así como comandos DETENER/EJECUTAR*)
 (*=> inicializa las variables necesarias para que la aplicación funcione de la forma esperada en este caso*)
END_IF;]
```

HMI_IsFirstMastWarmCycle: indica si el ciclo es el primer ciclo MAST del arranque en caliente

Descripción de funciones

Esta función devuelve TRUE durante el primer ciclo MAST después de un arranque en caliente.

Representación gráfica

Representación IL y ST

Para ver la representación general en lenguaje IL o ST, consulte el capítulo *Representación de las funciones y los bloques de funciones* *(véase página 47)*.

Descripción de la variable de E/S

En la tabla siguiente se describe la variable de salida:

Salida	Tipo	Comentario
`HMI_IsFirstMastWarmCycle`	`BOOL`	TRUE durante el ciclo de tarea MAST después de un arranque en caliente.

Ejemplo

Consulte la función `HMI_IsFirstMastCycle` *(véase página 32)*.

Capítulo 3
Tipos de datos de la biblioteca XBT PLCSystem

Descripción general

En este capítulo se describen los **tipos de datos** de la biblioteca XBT PLCSystem.

Hay dos clases de **tipos de datos** disponibles:

- **Los tipos de datos de las variables del sistema** los utilizan las **variables del sistema** *(véase página 11)* de la biblioteca XBT PLCSystem (PLC_R, PLC_W,...).
- **Los tipos de datos de funciones del sistema** los utilizan las **funciones del sistema** *(véase página 25)* de lectura/escritura de la biblioteca XBT PLCSystem.

Contenido de este capítulo

Este capítulo contiene los siguiente apartados:

PLC_R_IO_STATUS: Códigos de estado de E/S

Descripción del tipo enumerado

El tipo de datos de enumeración `PLC_R_IO_STATUS` contiene los valores siguientes:

Enumerador	Valor	Comentario
`PLC_R_IO_OK`	FFFF hex	Las entradas/salidas están operativas.
`PLC_R_IO_NO_INIT`	0001 hex	Las entradas/salidas no se han inicializado.
`PLC_R_IO_CONF_FAULT`	0002 hex	Se han detectado parámetros de configuración de E/S no válidos.

PLC_R_APPLICATION_ERROR: Códigos de estado del error detectado de la aplicación

Descripción del tipo enumerado

El tipo de datos de enumeración `PLC_R_APPLICATION_ERROR` contiene los valores siguientes:

Enumerador	Valor	Comentario	Qué hacer
`PLC_R_APP_ERR_UNKNOWN`	FFFF hex	Error no definido detectado.	Póngase en contacto con su servicio de soporte técnico local.
`PLC_R_APP_ERR_NOEXCEPTION`	0000 hex	No se ha detectado ningún error.	–
`PLC_R_APP_ERR_WATCHDOG`	0010 hex	Ha caducado el watchdog de la tarea.	Compruebe la aplicación. Consulte del capítulo . Se necesita reiniciar para entrar en modalidad de ejecución.
`PLC_R_APP_ERR_HARDWAREWATCHDOG`	0011 hex	Ha caducado el watchdog de sistema.	Si se puede reproducir el problema, busque puertos de comunicación desconectados. Si el problema persiste, actualice el firmware. Si aún no se ha resuelto, póngase en contacto con su servicio de soporte técnico local.
`PLC_R_APP_ERR_IO_CONFIG_ ERROR`	0012 hex	Se han detectado parámetros de configuración de E/S incorrectos.	La aplicación puede estar dañada. Para resolver este problema, utilice uno de los siguientes métodos: **1. Compilar → Limpiar todo** **2.** Exporte/Importe la aplicación. **3.** Actualice SoMachine a la versión más reciente.
`PLC_R_APP_ERR_UNRESOLVED_ EXTREFS`	0018 hex	Funciones no definidas detectadas.	Elimine las funciones no resueltas de la aplicación.

Enumerador	Valor	Comentario	Qué hacer
PLC_R_APP_ERR_IEC_TASK_CONFIG_ERROR	0025 hex	Se han detectado parámetros de configuración de tareas incorrectos.	La aplicación puede estar dañada. Para resolver este problema, utilice uno de los siguientes métodos: **1. Compilar → Limpiar todo** **2.** Exporte/Importe la aplicación. **3.** Actualice SoMachine a la versión más reciente.
PLC_R_APP_ERR_ILLEGAL_INSTRUCTION	0050 hex	Instrucción no definida detectada.	Para resolver el problema, depure la aplicación.
PLC_R_APP_ERR_ACCESS_VIOLATION	0051 hex	Intento de acceso al área de memoria reservada.	Para resolver el problema, depure la aplicación.
PLC_R_APP_ERR_PROCESSORLOAD_WATCHDOG	0105 hex	Las tareas de la aplicación han sobrecargado el procesador.	Reduzca la carga de trabajo de la aplicación mejorando la arquitectura de la aplicación. Aumente la duración del ciclo de tarea. Reduzca la frecuencia de evento.

PLC_R_BOOT_PROJECT_STATUS: Códigos de estado del proyecto de inicio

Descripción del tipo enumerado

El tipo de datos de enumeración `PLC_R_BOOT_PROJECT_STATUS` contiene los valores siguientes:

Enumerador	Valor	Comentario
`PLC_R_NO_BOOT_PROJECT`	0000 hex	El proyecto de arranque no existe en la memoria flash.
`PLC_R_BOOT_PROJECT_CREATION_IN_PROGRESS`	0001 hex	Se va a crear el proyecto de arranque.
`PLC_R_DIFFERENT_BOOT_PROJECT`	0002 hex	El proyecto de arranque en la memoria flash es distinto del cargado en la RAM.
`PLC_R_VALID_BOOT_PROJECT`	FFFF hex	El proyecto de arranque en la memoria flash es el mismo que el cargado en la RAM.

PLC_R_STATUS: Códigos de estado del controlador

Descripción del tipo enumerado

El tipo de datos de la enumeración `PLC_R_STATUS` contiene los valores siguientes:

Enumerador	Valor	Comentario
`PLC_R_EMPTY`	0000 hex	El controlador no contiene ninguna aplicación.
`PLC_R_STOPPED`	0001 hex	El controlador se ha detenido.
`PLC_R_RUNNING`	0002 hex	El controlador está en ejecución.
`PLC_R_HALT`	0004 hex	El controlador está en estado HALT (Parado). (Consulte el diagrama del estado del controlador en la *guía de programación* del controlador).
`PLC_R_BREAKPOINT`	0008 hex	El controlador se ha detenido en un punto de interrupción.

PLC_R_STOP_CAUSE: Códigos RUN para causa de transición a otro estado

Descripción del tipo enumerado

El tipo de datos de enumeración `PLC_R_STOP_CAUSE` contiene los valores siguientes:

Enumerador	Valor	Comentario	Qué hacer
`PLC_R_STOP_REASON_ UNKNOWN`	0000 hex	El valor inicial o la causa de la detención es indefinido.	En caso de causa de detención no definida, póngase en contacto con su servicio de soporte técnico local.
`PLC_R_STOP_REASON_HW_ WATCHDOG`	0001 hex	Detenido tras un watchdog de hardware.	Póngase en contacto con su servicio de soporte técnico local.
`PLC_R_STOP_REASON_RESET`	0002 hex	Detenido tras resetear.	Consulte las posibilidades de reinicio en el capítulo . Se necesita reiniciar para entrar en modalidad de ejecución.
`PLC_R_STOP_REASON_ EXCEPTION`	0003 hex	Detenido tras detectar una excepción.	Compruebe la aplicación. Consulte del capítulo .
`PLC_R_STOP_REASON_USER`	0004 hex	Detenido tras una petición de usuario.	Consulte del capítulo .
`PLC_R_STOP_REASON_ IECPROGRAM`	0005 hex	Detenido tras una petición de comando de programa (p. ej.: comando de control con el parámetro `PLC_W.q_wPLCControl:= PLC_W_COMMAND.PLC_W_ STOP;`).	–
`PLC_R_STOP_REASON_DELETE`	0006 hex	Detenido tras un comando de eliminación de aplicación.	Consulte del capítulo .
`PLC_R_STOP_REASON_ DEBUGGING`	0007 hex	Detenido tras entrar en la modalidad de depuración.	–
`PLC_R_STOP_FROM_NETWORK_ REQUEST`	000A hex	Detenido tras una petición de la red (llave USB o comando PLC_W).	–
`PLC_R_STOP_FROM_INPUT`	000B hex	Detención requerida por una entrada de controlador.	–
`PLC_R_STOP_REASON_RETAIN_ MISMATCH`	000C hex	Enviada al estado STOPPED después del reinicio y de la detección de una discrepancia en la definición de las variables remanentes.	Las variables retentivas se han eliminado porque no están referenciadas en la aplicación. Si la aplicación establece las variables retentivas en sus valores de inicialización, está disponible.

Enumerador	Valor	Comentario	Qué hacer
PLC_R_STOP_REASON_BOOT_APPLI_MISMATCH	000D hex	Enviada al estado STOPPED después del reinicio y de la detección de una discrepancia en la aplicación de inicio.	Cree una aplicación de arranque válida.
PLC_R_STOP_REASON_POWERFAIL	000E hex	El controlador se ha detenido a causa de una interrupción de la alimentación.	Compruebe la fuente de alimentación.

PLC_W_COMMAND: Códigos de comando de control

Descripción del tipo enumerado

El tipo de datos de enumeración `PLC_W_COMMAND` contiene los valores siguientes:

Enumerador	Valor	Comentario
`PLC_W_STOP`	0001 hex	Comando para detener el controlador.
`PLC_W_RUN`	0002 hex	Comando para ejecutar el controlador.
`PLC_W_RESET_COLD`	0004 hex	Comando para iniciar un reinicio en frío del controlador.
`PLC_W_RESET_WARM`	0008 hex	Comando para iniciar un reinicio en caliente del controlador.

RIGHTBUS_GET_STATUS: `HMI_GetRightBusStatus` Códigos de parámetros de funciones

Descripciones de los tipos enumerados

El tipo de datos de la enumeración contiene los valores siguientes:

Enumerador	Valor	Descripción
`RIGHTBUS_GET_GEN_STATUS`	00 hex	Parámetro para un diagnóstico de configuración del bus de ampliación
`RIGHTBUS_GET_STATUS1`	01 hex	Parámetro para un diagnóstico del módulo 1 del bus de ampliación
`RIGHTBUS_GET_STATUS2`	02 hex	Parámetro para un diagnóstico del módulo 2 del bus de ampliación
`RIGHTBUS_GET_STATUS3`	03 hex	Parámetro para un diagnóstico del módulo 3 del bus de ampliación

NOTA: Para obtener más información acerca del uso del tipo de parámetro `RIGHTBUS_GET_STATUS`, consulte la función `HMI_GetRightBusStatus` *(véase página 27)*.

Apéndices

Apéndice A
Representación de funciones y de bloques de funciones

Descripción general

Cada función se puede representar en los lenguajes siguientes:
- IL: Lista de instrucciones
- ST: Texto estructurado
- LD: Diagrama de contactos
- FBD: Diagrama de bloques de funciones
- CFC: Diagrama de función continua

En este capítulo se proporcionan funciones y ejemplos de representación de bloques de funciones y se describe cómo utilizarlas en lenguajes IL y ST.

Contenido de este capítulo

Este capítulo contiene los siguiente apartados:

Diferencias entre una función y un bloque de funciones

Función

Una función:

- Es una POU (Unidad de organización de programa) que devuelve un resultado inmediato.
- Se le llama directamente por su nombre (y no a través de una instancia).
- No tiene un estado persistente desde una llamada hasta la otra.
- Se puede utilizar como un operando en otras expresiones.

Ejemplos: operadores booleanos (`AND`), cálculos, conversión (`BYTE_TO_INT`)

Bloque de funciones

Bloque de funciones

- Es una POU (Unidad de organización de programa) que devuelve una o más salidas.
- Debe llamarse a través de una instancia (copia del bloque de funciones con nombre y variables dedicados).
- Todas las instancias tienen un estado persistente (salidas y variables internas) de una llamada a otra desde un bloque de funciones o programa.

Ejemplos: temporizadores, contadores

En el ejemplo, `Timer_ON` es una instancia del bloque de funciones `TON`:

```
1  PROGRAM MyProgram_ST
2  VAR
3      Timer_ON: TON;  // Function Block Instance
4      Timer_RunCd: BOOL;
5      Timer_PresetValue: TIME := T#5S;
6      Timer_Output: BOOL;
7      Timer_ElapsedTime: TIME;
8  END_VAR
```

```
1  Timer_ON(
2      IN:=Timer_RunCd,
3      PT:=Timer_PresetValue,
4      Q=>Timer_Output,
5      ET=>Timer_ElapsedTime);
```

Cómo utilizar una función o un bloque de funciones en lenguaje IL

Información general

En esta sección se describe cómo implementar una función y un bloque de funciones en lenguaje IL.

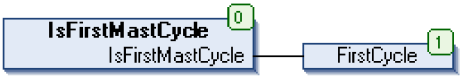
Las funciones `IsFirstMastCycle` y `SetRTCDrift` y el bloque de funciones `TON` se utilizan como ejemplos para mostrar implementaciones.

Uso de una función en lenguaje IL

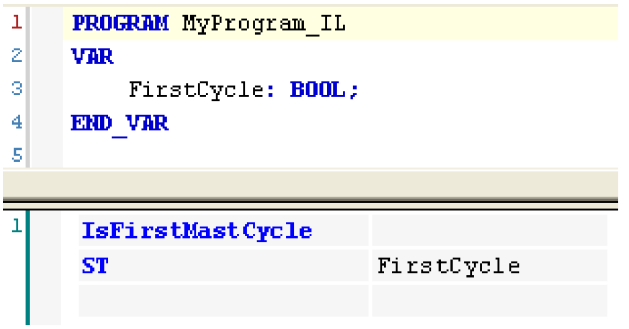
En este procedimiento se describe cómo insertar una función en lenguaje IL:

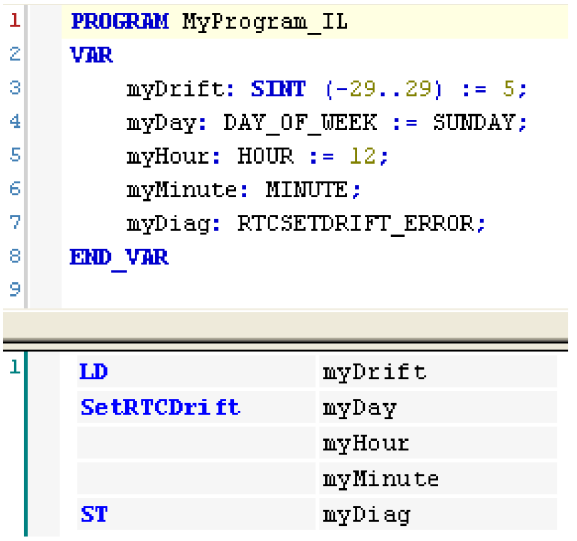
Paso	Acción
1	Abra o cree una nueva POU en el lenguaje de Lista de instrucciones (IL). **NOTA:** Aquí no se detalla el procedimiento para crear una POU. Para obtener más información, consulte Adding and Calling POUs *(véase SoMachine, Guía de programación)*.
2	Las variables de entrada son los parámetros de entrada requeridos por la función.
3	Si la función tiene 1 o más entradas, empiece a cargar la primera entrada utilizando la instrucción LD.
4	Inserte una nueva línea abajo y: ● escriba el nombre de la función en la columna de operadores (campo izquierdo); o ● Utilice la opción **Accesibilidad** para seleccionar la función (seleccione **Insertar llamada de módulo** en el menú contextual).
5	Si la función tiene más de una entrada y se utiliza Accesibilidad, se crea automáticamente el número necesario de líneas con `???` en los campos de la derecha. Sustituya los `???` por el valor o la variable adecuada que corresponda al orden de las entradas.
6	Inserte una nueva línea para almacenar el resultado de la función en la variable adecuada: Escriba la instrucción ST en la columna de operadores (campo de la izquierda) y un nombre de variable en el campo de la derecha.

Para ilustrar el procedimiento, considere las funciones `IsFirstMastCycle` (sin parámetro de entrada) y `SetRTCDrift` (con parámetros de entrada) que se representan gráficamente a continuación:

Función	Representación gráfica
sin parámetros de entrada: `IsFirstMastCycle`	
con parámetros de entrada: `SetRTCDrift`	

En lenguaje IL, el nombre de la función se utiliza directamente en la columna de operadores:

Función	Representación en el Editor POU IL de SoMachine
Ejemplo en IL de una función sin parámetros de entrada: `IsFirstMastCycle`	

Función	Representación en el Editor POU IL de SoMachine
Ejemplo en IL de una función con parámetros de entrada: `SetRTCDrift`	

```
PROGRAM MyProgram_IL
VAR
    myDrift: SINT (-29..29) := 5;
    myDay: DAY_OF_WEEK := SUNDAY;
    myHour: HOUR := 12;
    myMinute: MINUTE;
    myDiag: RTCSETDRIFT_ERROR;
END_VAR
```

```
LD          myDrift
SetRTCDrift myDay
            myHour
            myMinute
ST          myDiag
```

Uso de un bloque de funciones en lenguaje IL

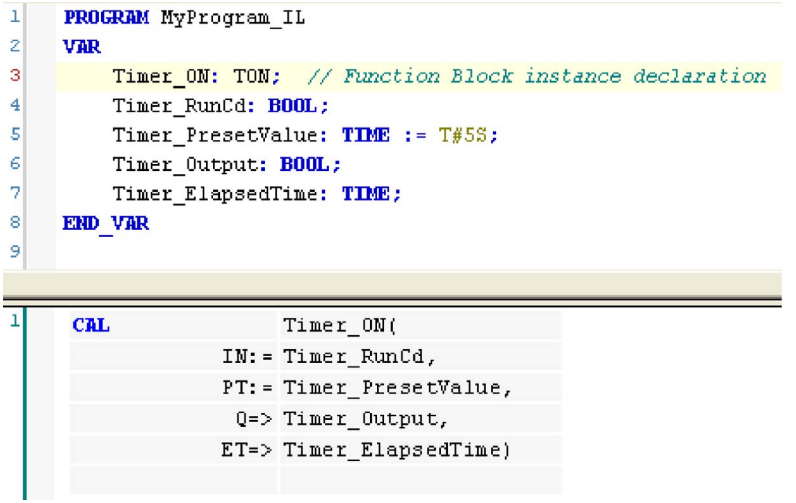
En este procedimiento se describe cómo insertar un bloque de funciones en lenguaje IL:

Paso	Acción
1	Abra o cree una POU nueva en el lenguaje de Lista de instrucciones (IL). **NOTA:** Aquí no se detalla el procedimiento para crear una POU. Para obtener más información, consulte Adding and Calling POUs *(véase SoMachine, Guía de programación)*.
2	Cree las variables que necesita el bloque de funciones, incluido el nombre de instancia.
3	Se llama a los bloques de funciones utilizando una instrucción `CAL`: ● Utilice la opción **Accesibilidad** para seleccionar el bloque de funciones (botón derecho del ratón y seleccionar **Insertar llamada de módulo** en el menú contextual). ● La instrucción `CAL` y la E/S necesaria se crean automáticamente. Cada parámetro (E/S) es una instrucción: ● Los valores de las entradas se establecen con ":=". ● Los valores de las salidas se establecen con "=>".
4	En el campo de la derecha `CAL`, sustituya `???` por el nombre de la instancia.
5	Sustituya otros `???` por una variable apropiada o un valor inmediato.

Para ilustrar el procedimiento, considere este ejemplo con el bloque de funciones `TON` que se representa gráficamente a continuación:

3	Representación gráfica
TON	

En lenguaje IL, el nombre del bloque de funciones se utiliza directamente en la columna de operadores:

Bloque de funciones	Representación en el Editor POU IL de SoMachine
TON	

Cómo utilizar una función o un bloque de funciones en lenguaje ST

Información general

En esta sección se describe el modo de implementar una función y un bloque de funciones en lenguaje ST.

La función `SetRTCDrift` y el bloque de funciones `TON` se utilizan como ejemplos para mostrar implementaciones.

Uso de una función en lenguaje ST

En este procedimiento se describe cómo insertar una función en lenguaje ST:

Paso	Acción
1	Abra o cree una POU nueva en el lenguaje de Texto estructurado (ST). **NOTA:** Aquí no se detalla el procedimiento para crear una POU. Para obtener más información, consulte Adding and Calling POUs *(véase SoMachine, Guía de programación)*.
2	Las variables de entrada son los parámetros de entrada requeridos por la función.
3	Utilice la sintaxis general en el **Editor POU ST** para el lenguaje ST de una función. La sintaxis general es: `FunctionResult:= FunctionName(VarInput1, VarInput2,.. VarInputx);`

Para ilustrar el procedimiento, considere la función `SetRTCDrift` que se representa gráficamente a continuación:

Función	Representación gráfica
`SetRTCDrift`	

El lenguaje ST de esta función es este:

Función	Representación en el Editor POU IL de SoMachine

`SetRTCDrift`	`PROGRAM MyProgram_ST`
	`VAR myDrift: SINT(-29..29) := 5;`
	`myDay: DAY_OF_WEEK := SUNDAY;`
	`myHour: HOUR := 12;`
	`myMinute: MINUTE;`
	`myRTCAdjust: RTCDRIFT_ERROR;`
	`END_VAR`
	`myRTCAdjust:= SetRTCDrift(myDrift, myDay, myHour, myMinute);`

Uso de un bloque de funciones en lenguaje ST

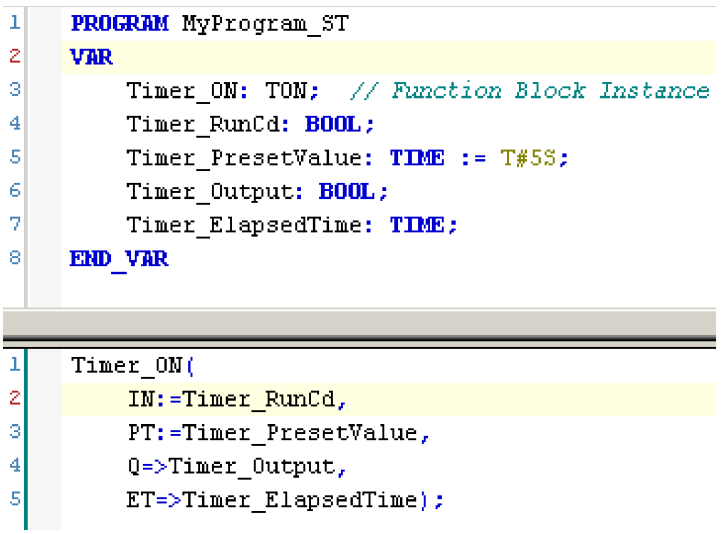
En este procedimiento se describe cómo insertar un bloque de funciones en lenguaje ST:

Paso	Acción
1	Abra o cree una POU nueva en el lenguaje de Texto estructurado (ST). **NOTA:** Aquí no se detalla el procedimiento para crear una POU. Para obtener más información sobre la adición, declaración y llamadas de POU, consulte la documentación *(véase SoMachine, Guía de programación)* relacionada.
2	Cree las variables de entrada y salida y la instancia requeridas para el bloque de funciones: ● Las variables de entrada son los parámetros de entrada requeridos por el bloque de funciones ● Las variables de salida reciben el valor devuelto por el bloque de funciones
3	Utilice la sintaxis general en el **Editor POU ST** para el lenguaje ST de un bloque de funciones. La sintaxis general es: `FunctionBlock_InstanceName(Input1:=VarInput1, Input2:=VarInput2,... Ouput1=>VarOutput1, Ouput2=>VarOutput2,...);`

Para ilustrar el procedimiento, considere este ejemplo con el bloque de funciones `TON` que se representa gráficamente a continuación:

3	Representación gráfica
`TON`	

En esta tabla se muestran ejemplos de una llamada de bloque de funciones en lenguaje ST:

Bloque de funciones	Representación en el Editor POU IL de SoMachine
TON	

```
PROGRAM MyProgram_ST
VAR
    Timer_ON: TON;  // Function Block Instance
    Timer_RunCd: BOOL;
    Timer_PresetValue: TIME := T#5S;
    Timer_Output: BOOL;
    Timer_ElapsedTime: TIME;
END_VAR
```

```
Timer_ON(
    IN:=Timer_RunCd,
    PT:=Timer_PresetValue,
    Q=>Timer_Output,
    ET=>Timer_ElapsedTime);
```

Glosario

!

%

Según el estándar IEC, % es un prefijo que identifica direcciones de memoria interna en el controlador lógico que se utilizan para almacenar el valor de las variables del programa, constantes, E/S, etc.

%MW

Según el estándar IEC, %MW representa un registro de palabra de memoria (por ejemplo, un objeto de lenguaje del tipo palabra de memoria).

A

aplicación

Un programa que incluye datos de configuración, símbolos y documentación.

Aplicación de arranque

(*aplicación de arranque*) El archivo binario que contiene la aplicación. Normalmente está guardada en el controlador y permite que este arranque en la aplicación generada por el usuario.

ARRAY

La disposición sistemática de objetos de datos de un solo tipo en forma de tabla definida en la memoria del controlador lógico. La sintaxis es la siguiente: `ARRAY [<dimensión>] OF <Tipo>`

Ejemplo 1: `ARRAY [1..2] OF BOOL` es una tabla de una dimensión compuesta por dos elementos de tipo `BOOL`.

Ejemplo 2: `ARRAY [1..10, 1..20] OF INT` es una tabla de dos dimensiones compuesta por 10 x 20 elementos de tipo `INT`.

B

bus de ampliación

Un bus de comunicación electrónico entre los módulos de E/S de ampliación y un controlador.

byte

Un tipo que está codificado en un formato de 8 bits que, en el formato hexadecimal, va de 00 hex a FF hex.

C

CFC

(*diagrama de función continua*) Un lenguaje de programación (una ampliación del estándar IEC 61131-3) basado en el lenguaje de diagrama de bloque de funciones (FBD) y que funciona como un diagrama de flujo. Sin embargo, no se utiliza ninguna red y es posible un posicionamiento libre de elementos gráficos, lo que permite bucles de realimentación. En cada bloque, las entradas se sitúan a la izquierda y las salidas, a la derecha. Las salidas del bloque se pueden conectar a las entradas de otros bloques para formar expresiones complejas.

configuración

Organización e interconexión de los componentes de hardware en un sistema y los parámetros del hardware y software que determina las características operativas del sistema.

controlador

Automatiza procesos industriales (también conocido como controlador lógico programable o controlador programable).

D

diagrama de bloques de funciones

Uno de los cinco lenguajes para lógica o control que cumplen con el estándar IEC 61131-3 para sistemas de control. El diagrama de bloques de funciones es un lenguaje de programación orientado gráficamente. Funciona con una lista de redes en la que cada red contiene una estructura gráfica de cuadros y líneas de conexión que representa una expresión lógica o aritmética, la llamada de un bloque de funciones, un salto o una instrucción de retorno.

DWORD

(*palabra doble*) Con codificación en formato de 32 bits.

E

E/S

(*entrada/salida*)

E/S digitales

(*entrada/salida digital*) Una conexión de circuito individual con el módulo que corresponde directamente a un bit de la tabla de datos. El bit de la tabla de datos contiene el valor de la señal en el circuito de E/S. Proporciona el acceso digital lógico de control a los valores de E/S.

ejecución

Un comando que hace que el controlador explore el programa de la aplicación, lea las entradas físicas y escriba en las salidas físicas según la solución de la lógica del programa.

elemento

El nombre abreviado de ARRAY.

entrada analógica

Convierte los niveles de tensión o corriente recibidos en valores numéricos. Puede almacenar y procesar estos valores en el controlador lógico.

F

FB

(*bloque de funciones*) Un práctico mecanismo de programación que consolida un grupo de instrucciones de programación para realizar una acción específica y normalizada, por ejemplo, el control de velocidad, el control de intervalo o el conteo. Un bloque de funciones se puede componer de datos de configuración, un conjunto de parámetros de funcionamiento internos o externos y, normalmente, una o diversas entradas y salidas de datos.

firmware

Representa el BIOS, los parámetros de datos y las instrucciones de programación que constituyen el sistema operativo en un controlador. El firmware se almacena en la memoria no volátil del controlador.

función

Una unidad de programación que dispone de una entrada y devuelve un resultado inmediato. No obstante, a diferencia de los FBs, se llama directamente por su nombre (y no mediante una instancia), no tiene un estado persistente desde una llamada hasta la siguiente y se puede utilizar como un operando en otras expresiones de programación.

Ejemplos: operadores booleanos (AND), cálculos, conversiones (BYTE_TO_INT)

G

GVL

(*lista de variables globales*) Gestiona las variables globales que se pueden transferir entre controladores en una red Ethernet TCP/IP Modbus.

H

hex

(*hexadecimal*)

HMI

(*interfaz hombre-máquina*) Una interfaz de operador (generalmente gráfica) para el control de equipos industriales por parte de personas.

I

ID

(*identificador/identificación*)

IEC

(*International Electrotechnical Commission*) Una organización de estándares internacional sin ánimo de lucro y no gubernamental que prepara y publica estándares internacionales para todas las tecnologías eléctricas, electrónicas y relacionadas.

IL

(*lista de instrucciones*) Un programa escrito en lenguaje que se compone de una serie de instrucciones basadas en texto y ejecutadas secuencialmente por el controlador. Cada instrucción incluye un número de línea, un código de instrucción y un operando (consulte IEC 61131-3).

INT

(*entero*) Un número entero con codificación de 16 bits.

L

LD

(*diagrama de contactos*) Una representación gráfica de instrucciones de un programa de controlador con símbolos para contactos, bobinas y bloques en una serie de escalones ejecutados de forma secuencial por un controlador (consulte IEC 61131-3).

M

MAST

Una tarea del procesador que se ejecuta en el software de programación. La tarea MAST consta de dos secciones:

- **IN:** las entradas se copian en la sección IN antes de ejecutar la tarea MAST.
- **OUT:** las salidas se copian en la sección OUT después de ejecutar la tarea MAST.

memoria Flash

Una memoria no volátil que se puede sobrescribir. Se almacena en una memoria EEPROM especial que se puede borrar y volver a programar.

Modbus

El protocolo de comunicaciones que permite las comunicaciones entre muchos dispositivos conectados a la misma red.

P

PLC

(*controlador lógico programable*) Un ordenador industrial que se usa para automatizar procesos industriales, de fabricación y otros procesos electromecánicos. Los PLCs se diferencian de los ordenadores comunes en que están diseñados de forma que tienen varias matrices de entrada y salida, y que disponen de especificaciones más sólidas contra los golpes, las vibraciones, la temperatura, las interferencias eléctricas, etc.

POU

(*unidad de organización de programas*) Una declaración variable en el código fuente y el conjunto de instrucciones correspondiente. Las POUs facilitan la reutilización modular de programas de software, funciones y bloques de funciones. Una vez declaradas, cada una de las POUs está disponible para las otras.

programa

El componente de una aplicación consistente en código fuente compilado capaz de poder ser instalado en la memoria de un controlador lógico.

R

red

Un sistema de dispositivos interconectados que comparten una ruta de datos común y un protocolo de comunicaciones.

S

ST

(*texto estructurado*) Un lenguaje que incluye instrucciones complejas y anidadas (por ejemplo, bucles de repetición, ejecuciones condicionales o funciones). ST cumple con IEC 61131-3.

STOP

Comando que hace que el controlador detenga la ejecución de un programa de aplicación.

T

tarea

Grupo de secciones y subrutinas ejecutadas cíclica o periódicamente si se trata de la tarea MAST, o periódicamente si se trata de la tarea FAST.

Una tarea siempre tiene un nivel de prioridad y tiene asociadas entradas y salidas del controlador. Estas E/S se actualizan en función de la tarea.

Un controlador puede tener diversas tareas.

V

variable
Una unidad de memoria direccionada y modificada por un programa.

variable no ubicada
Una variable que no tiene dirección (consulte *variable ubicada*).

variable ubicada
Consulte (*variable no ubicada*).

W

watchdog
Un watchdog es un cronómetro especial utilizado para garantizar que los programas no superen su tiempo de exploración asignado. El cronómetro watchdog suele configurarse con un valor superior al tiempo de exploración y se resetea a 0 cuando termina cada ciclo de exploración. Si el cronómetro watchdog alcanza el valor predeterminado, por ejemplo, porque el programa queda atrapado en un bucle infinito, se declara un error·y el programa se detiene.

X

XBT
Cualquier panel gráfico de Magelis XBT.

Índice