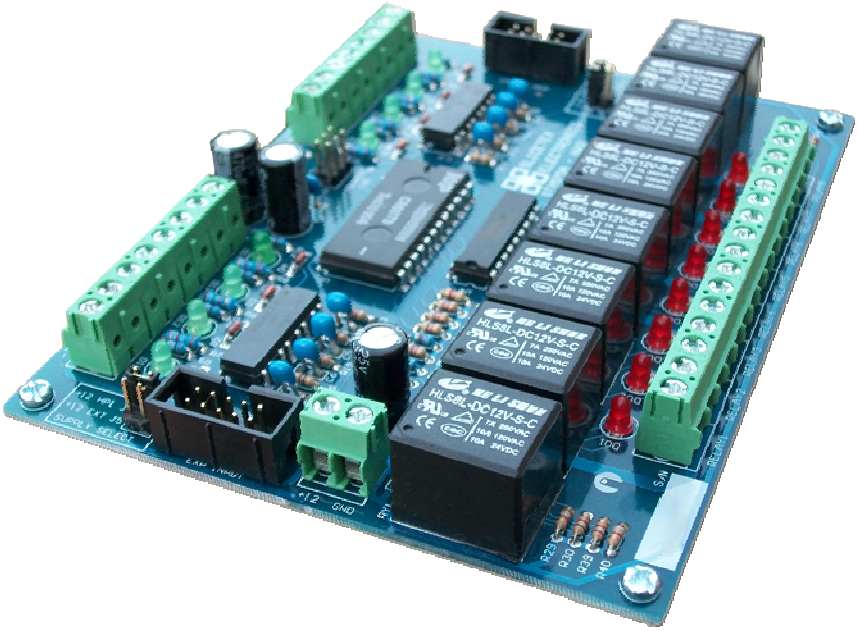
# *STX570*
# *Módulo de Expansión de E/S Discretas*

# Manual de Usuario

*Autor: Ing. Boris Estudiez*

| Modelos Aplicables | A, B y C |
|---|---|



## *1 Descripción General*

*La placa STX570 fue diseñada para expandir las entradas y salidas discretas del PLC. El presente documento explica cómo utilizar la placa **STX570** conectada al **PLC**, en lenguaje Ladder y lenguaje Pawn.*

## 2 Lecturas Recomendadas

Antes de leer este documento, recomendamos que se familiarice con la placa STX570 y el PLC. Para ello recomendamos leer los siguientes documentos, en el orden detallado a continuación:

1. **STX570A-DS**: Hoja de Datos del modelo STX570A.
2. Hoja de datos técnicos del PLC.
3. Manual de Usuario del software **StxLadder**.
4. Manual de Programación Pawn del PLC (si utiliza lenguaje Pawn)

Este manual asume que usted maneja correctamente el PLC al cual debe conectar la placa STX570. Para mayores detalles, consulte la documentación específica del modelo de PLC.

Para soporte utilice nuestro foro de soporte técnico en: www.slicetex.com/foro

Mayor documentación puede encontrar en la pagina del producto: www.slicetex.com.

## 3 Requerimientos

Para este manual de usuario, debe tener instalado en su computadora el entorno de Programación **StxLadder** (Slicetex Ladder) versión 1.6.5 o superior, y utilizar la última versión de firmware en su PLC con soporte para la STX570.

# 4 Lenguaje Ladder

Para utilizar la STX570 en lenguaje Ladder hay dos métodos:

- Administración automática: Muy simple y fácil de utilizar.
- Administración manual: Consiste en utilizar componentes específicos para acceder a la placa.

## 4.1 Administración Automática

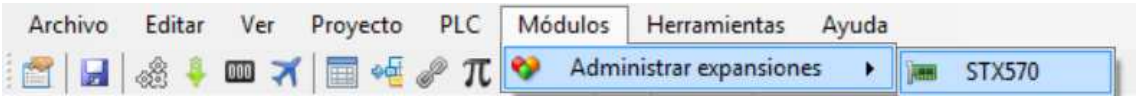
Este método consiste en declarar la cantidad de módulos **STX570** conectados al PLC en el proyecto, y luego acceder a las entradas/salidas de cada modulo con variables de periféricos tipo **Bool** con nombres EDINx (entradas) y EDOUTx (salidas). Donde x es el numero de entrada de cada modulo.

El requerimiento es que cada modulo, tenga configurado por hardware una dirección I2C consecutiva, comenzando por 0. Es decir si hay 3 módulos conectados, un modulo debe tener la dirección 0, el siguiente la dirección 1 y el ultimo la dirección 2.

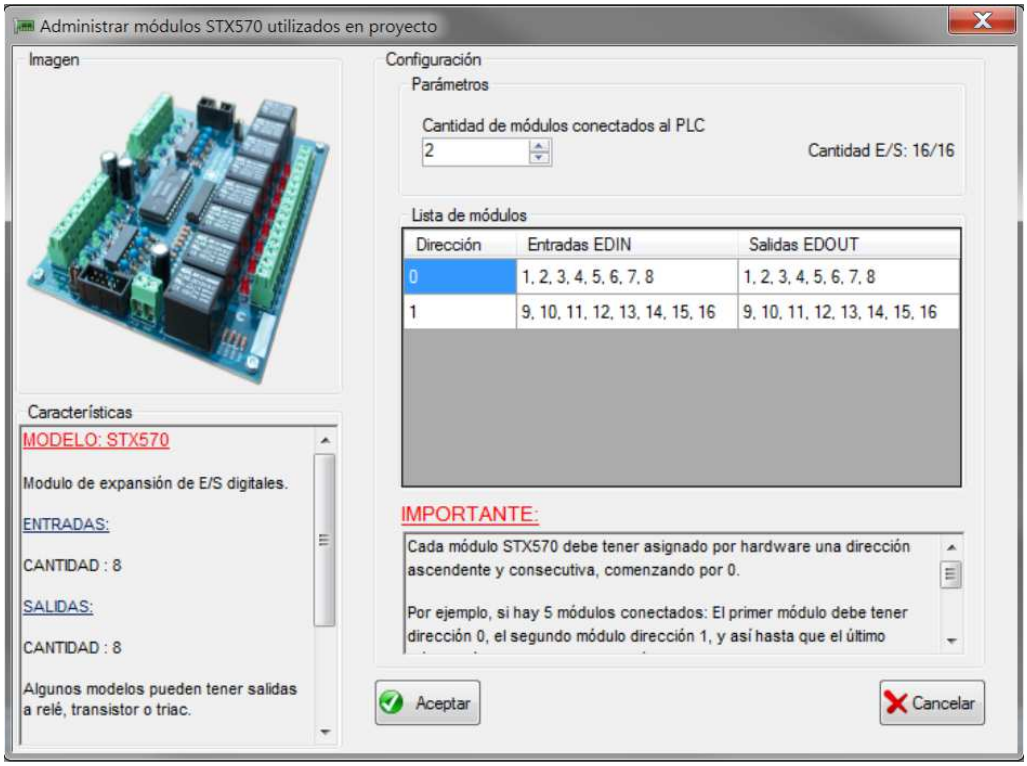
### 4.1.1 Ejemplo

En el siguiente ejemplo conectaremos 2 módulos STX570 al PLC y accederemos a sus entradas y salidas.

Primero declaramos los módulos desde StxLadder, para ello ir al menú "**Módulos > Administrar expansiones > STX570**", como muestra la siguiente imagen:



Aparecerá la siguiente ventana:



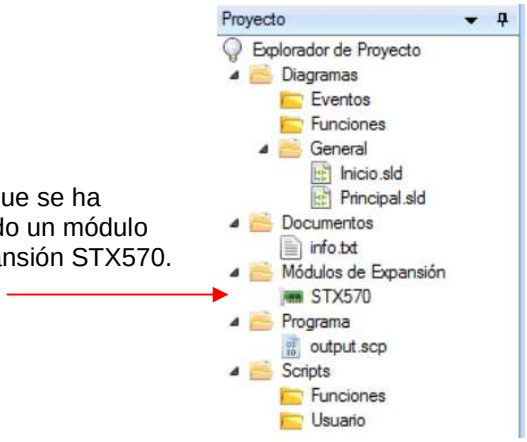

Seleccione en "**Cantidad de módulos conectados al PLC**" el valor "2". Esto es porque conectaremos 2 módulos al PLC.

En la misma ventana puede ver la numeración de las entradas/salidas EDIN / EDOUT correspondiente a cada dirección de cada módulo.

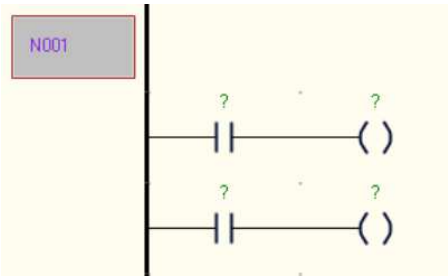
Presione el botón **Aceptar** para volver al proyecto y utilizar los módulos.

Una vez que retorne al proyecto, podrá ver el módulo agregado en el **Explorador de Proyecto**:

Indica que se ha agregado un módulo de expansión STX570.

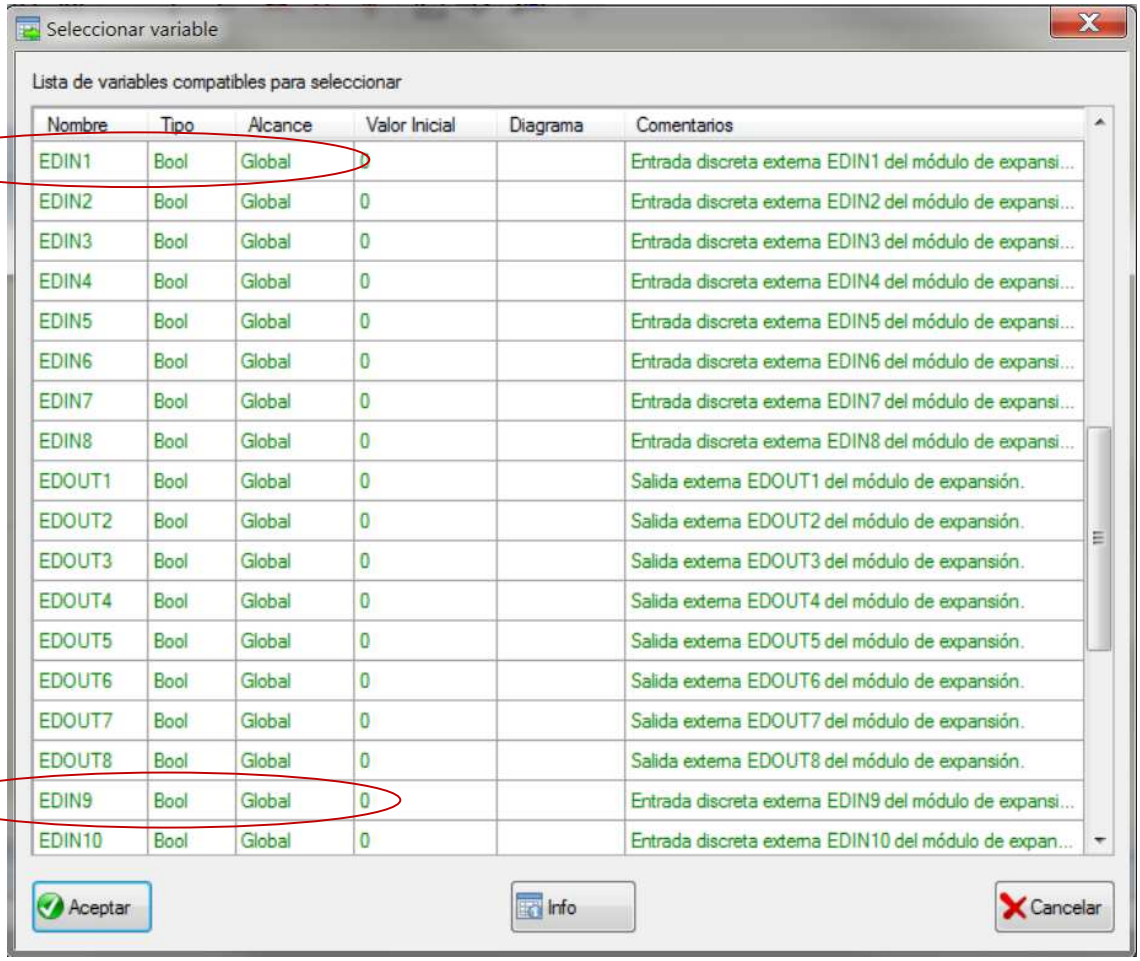


Ahora en el proyecto, diríjase al diagrama "**Principal.sld**" y agregue los siguientes componentes en la Network **N001**:




En los componentes de contactos normal abierto seleccionaremos la variable EDIN1 y EDIN9 respectivamente.

Para ello desde los componentes, acceder a la lista de variables del proyecto:

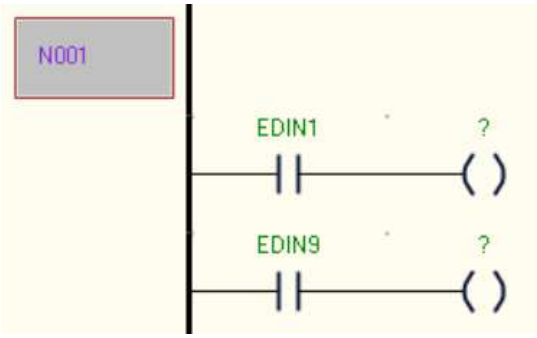



| Nombre | Tipo | Alcance | Valor Inicial | Diagrama | Comentarios |
|---|---|---|---|---|---|
| EDIN1 | Bool | Global | 0 | | Entrada discreta externa EDIN1 del módulo de expansi... |
| EDIN2 | Bool | Global | 0 | | Entrada discreta externa EDIN2 del módulo de expansi... |
| EDIN3 | Bool | Global | 0 | | Entrada discreta externa EDIN3 del módulo de expansi... |
| EDIN4 | Bool | Global | 0 | | Entrada discreta externa EDIN4 del módulo de expansi... |
| EDIN5 | Bool | Global | 0 | | Entrada discreta externa EDIN5 del módulo de expansi... |
| EDIN6 | Bool | Global | 0 | | Entrada discreta externa EDIN6 del módulo de expansi... |
| EDIN7 | Bool | Global | 0 | | Entrada discreta externa EDIN7 del módulo de expansi... |
| EDIN8 | Bool | Global | 0 | | Entrada discreta externa EDIN8 del módulo de expansi... |
| EDOUT1 | Bool | Global | 0 | | Salida externa EDOUT1 del módulo de expansión. |
| EDOUT2 | Bool | Global | 0 | | Salida externa EDOUT2 del módulo de expansión. |
| EDOUT3 | Bool | Global | 0 | | Salida externa EDOUT3 del módulo de expansión. |
| EDOUT4 | Bool | Global | 0 | | Salida externa EDOUT4 del módulo de expansión. |
| EDOUT5 | Bool | Global | 0 | | Salida externa EDOUT5 del módulo de expansión. |
| EDOUT6 | Bool | Global | 0 | | Salida externa EDOUT6 del módulo de expansión. |
| EDOUT7 | Bool | Global | 0 | | Salida externa EDOUT7 del módulo de expansión. |
| EDOUT8 | Bool | Global | 0 | | Salida externa EDOUT8 del módulo de expansión. |
| EDIN9 | Bool | Global | 0 | | Entrada discreta externa EDIN9 del módulo de expansi... |
| EDIN10 | Bool | Global | 0 | | Entrada discreta externa EDIN10 del módulo de expan... |

Notar que el proyecto contiene las variables **EDIN1**, **EDIN2**, **EDIN3**, etc. Que corresponden a las entradas de los módulos STX570. Son variables tipo Bool (0 o 1) de tipo periférico (porque corresponden a un valor físico).

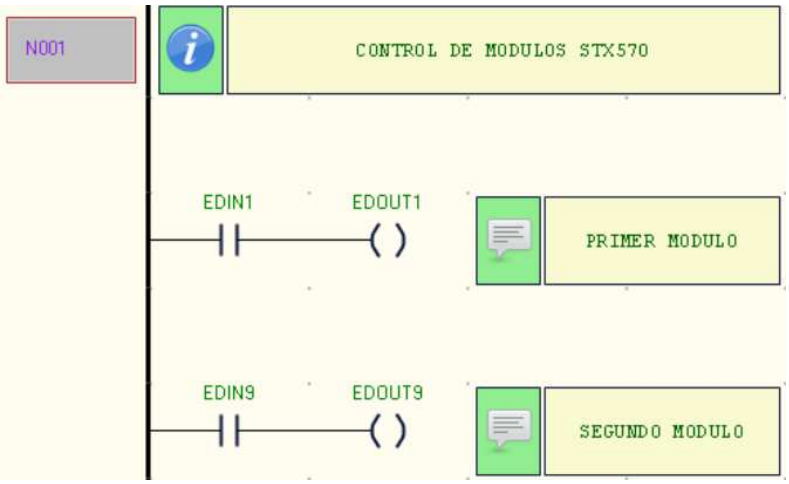
Lo mismo para las variables **EDOUT1**, **EDOUT2**, **EDOUT3**, etc. Son variables de periféricos que corresponden a las salidas de los módulos STX570.

Una vez seleccionadas las variables, resulta lo siguiente:




Ahora hacemos lo mismo, pero para las bobinas. Seleccionamos **EDOUT1** y **EDOUT9**.

Resultando el siguiente circuito final en el diagrama principal:



### *4.1.2 Prueba del Ejemplo*

Recuerde declarar el modulo en el PLC desde "**PLC > Configurar PLC**", pestaña "**Expansión**" como indica la hoja de datos de la placa STX570.

Luego conecte ambos módulos al PLC (desconectando energía eléctrica), el primero con dirección I2C igual a 0, y el segundo con dirección 1.

Compile el ejemplo y cárguelo al PLC.

Pruebe energizar la entrada 1 del primer modulo, esto debería activar la salida 1 del módulo.
Lo mismo con el segundo modulo, energice su entrada 1 para que el mismo active la salida 1.

Esto demuestra que el PLC controla mediante Ladder la lógica de cada modulo conectado.

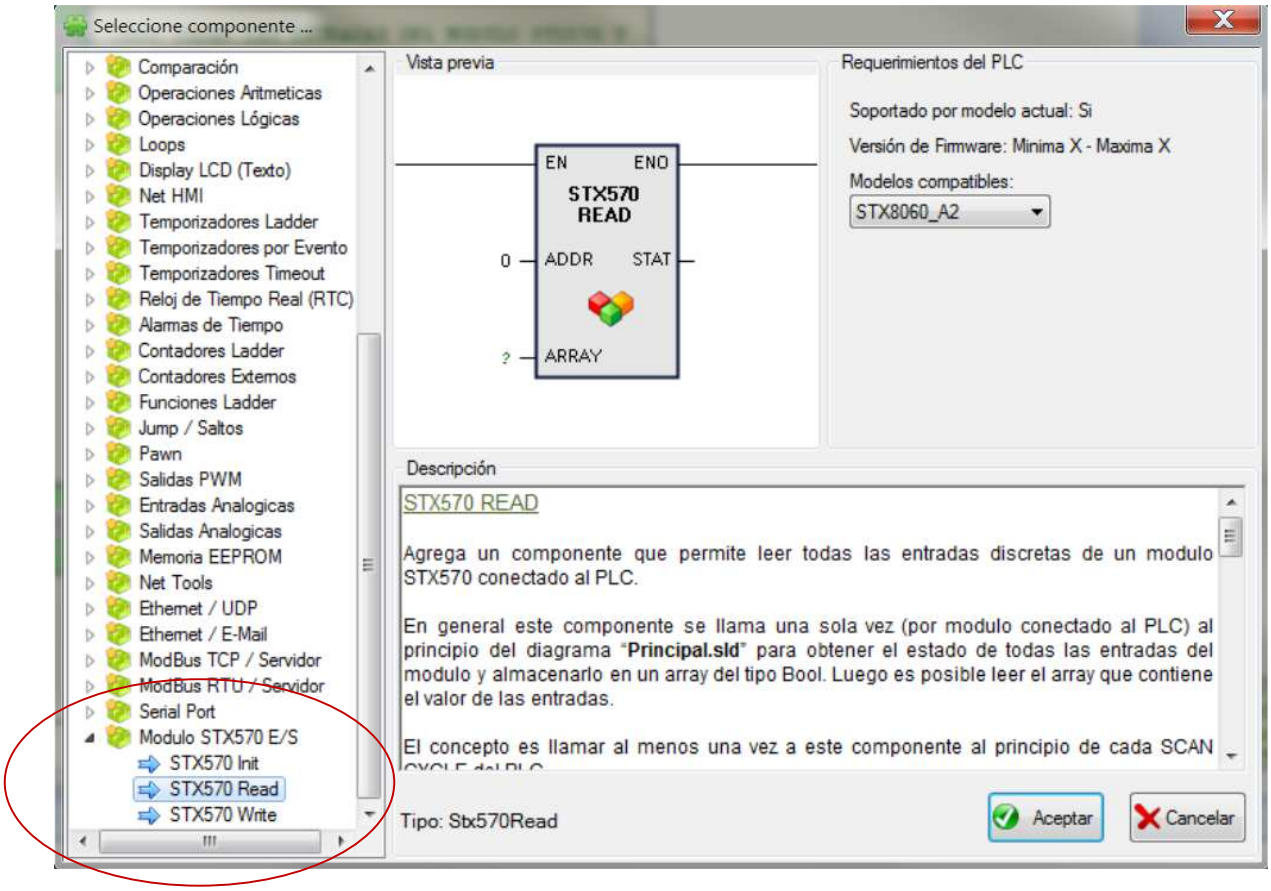
Notar que el PLC actualiza en memoria los estados de las variables **EDIN** y **EDOUT** en cada SCAN-CYLE del PLC.

## 4.2 Administración Manual

La administración totalmente manual de los módulos STX570 es una alternativa para no declarar en el proyecto la cantidad de placas a utilizar. Pero esto requiere leer y escribir las salidas en las placas de forma manual y coordinada. Ver proyecto de ejemplo **Stx570Manual1** y **Stx570Manual2** disponibles en nuestra página Web o con los archivos adjuntos de este manual.

Otra función de la administración manual (cuando se combina con la administración automática) es la de obtener un acceso directo a las salidas y entradas de los módulos. Es decir, actuar de forma inmediata sin esperar un SCAN-CYCLE.

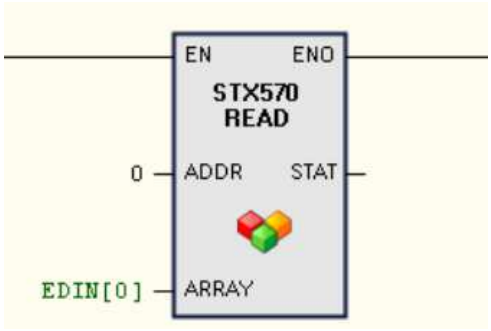
Para ello existen dos componentes muy útiles, **Stx570Read** y **Stx570Write**, disponibles en el selector de componentes, bajo el grupo "**Modulo STX570 E/S**", como muestra la siguiente ventana:




**Nota**: no hace falta utilizar la administración manual si usted está utilizando la administración automática, a menos que requiera acceso inmediato a las salidas o entradas de cada modulo.

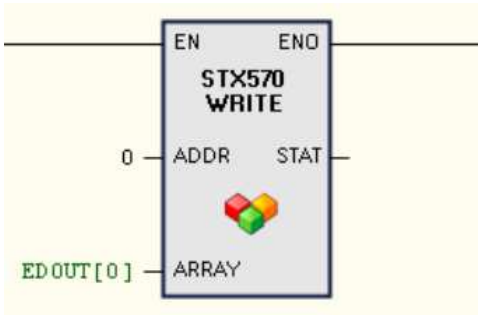
Los componentes están documentados dentro del entrono **StxLadder**, pero los mostraremos brevemente a continuación:

**Componente Stx570Read:**




Lee inmediatamente las entradas del modulo localizado en la dirección **ADDR** y devuelve su estado en un **ARRAY** del tipo **Bool_Array** de al menos 9 elementos. Cuando el componente se ejecute, se retornará sobre cada elemento del array el estado de cada entrada discreta del modulo STX570.

**Componente Stx570Write:**




Escribe inmediatamente las salidas del modulo localizado en la dirección **ADDR** y en la entrada "**ARRAY**" debe seleccionarse un array del tipo **Bool_Array** de al menos 9 elementos. Cuando el componente se ejecute, se escribirá el valor de cada elemento del array a cada salida discreta del modulo STX570.

# 5 Lenguaje Pawn

A partir de esta sección se explica de forma completa y con ejemplos, todas las funciones nativas disponibles en su PLC para comandar la placa STX570. Ejemplos de uso también se suministran.

Se recomienda actualizar el firmware de su PLC a la última versión, consulte "Requerimientos" en la hoja de datos de la placa STX570.

**Las funciones están agrupadas por características, consulte el índice en página 36 para una búsqueda simple.**

Recordar que una función nativa, es aquella que el PLC contiene internamente y puede ser ejecutada desde su script. Una función nativa, la ejecuta el procesador en código nativo, por ello, siempre son más rápidas, que cualquiera que usted pueda crear para realizar la misma operación.

## 5.1 Como leer las Funciones

En este documento, las funciones se presentan en el siguiente formato:

| Función(Arg1, Arg2): Función de prueba. | | |
|---|---|---|
| **Argumentos** | **Tipo** | **Descripción** |
| Arg1 | E | Argumento de entrada. |
| Arg2 | S | Argumento de Salida. Se retorna un valor en la variable Arg2. |
| **Retorno** | **Tipo** | **Descripción** |
| 0 | S | Operación exitosa. |

Donde:

- **Función(Arg1, Arg2)**: Prototipo de la función, que incluye nombre de argumentos.
- **Arg1 y Arg2**: Argumentos o parámetros de la función.
- **Tipo**: Si es "E" significa entrada. "S" significa salida (en caso de ser un argumento, se entiende que la función retornara un valor en la variable). Si es "E/S" el argumento es de entrada y salida.
- **Retorno**: El retorno de la función es su resultado, puede ser un constante, en ese caso se listan todas las constantes posibles, de lo contrario se le da nombre al retorno con fines descriptivos.

# 6 Funciones para ExPort (Puerto de Expansión)

En el puerto de expansión ExPort (HP2 / HP3) del PLC, se conectan los módulos externos, por ejemplo la placa STX570. A continuación de describen funciones que permiten configurar o controlar dicho puerto, y se utilizan en conjunto con las funciones nativas de la placa STX570.

## 6.1 Funciones para Manejar Eventos

La placa STX570 genera un evento "@OnExPort" cuando alguna de su entradas discreta cambia de estado. El uso de eventos para leer las entradas es opcional, también puede leer las mismas utilizando el método polling (ver pag. 24) que consiste leer dichas entradas en intervalos fijos de tiempo.

El concepto de eventos, se explica en el manual de su PLC. A continuación describimos como configurar el PLC para recibir eventos del puerto expansión, es decir de la placa STX570 conectada al PLC.

| **ExPortSetEvent():** Activa el evento @OnExPort() que será generado cuando alguna entrada discreta (DIN1 a DIN8) de la placa STX570 cambie de estado respecto del último valor leído. | | |
|---|---|---|
| **Argumentos** | **Tipo** | **Descripción** |
| - | - | |
| **Retorno** | **Tipo** | **Descripción** |
| 0 | S | Operación exitosa, evento creado. |
| -1 | S | Error, el evento no pudo ser creado (leer nota 1). |
| **Notas** | | **Descripción** |
| 1 | | Asegúrese de crear la función pública @OnExPort() antes de llamar a esta función. |

Ejemplo:

```
PlcMain()
{
      …

      // Activar evento @OnExPort()
      ExPortSetEvent()

      …
}

@OnExPort()
{
      // Código para el evento...
      // Por ejemplo: leer entradas de placa STX570.
}
```

| ExPortClrEvent(): Desactiva el evento @OnExPort(). | | |
|---|---|---|
| **Argumentos** | **Tipo** | **Descripción** |
| - | - | |
| **Retorno** | **Tipo** | **Descripción** |
| 0 | S | Operación exitosa, evento desactivado. |
| **Notas** | | **Descripción** |
| 1 | | Utilice esta función cuando no necesite comprobar más las entradas. Esto disminuye la sobrecarga en el procesador de la placa. |

Ejemplo:

```
ExPortClrEvent()
```

## *6.2 Eventos*

| @OnExPort(): Es llamado cuando hay un evento en el puerto de expansión. Si la placa STX570 está conectada, significa que el estado de alguna entrada discreta difiere del último valor leído. | | |
|---|---|---|
| **Argumentos** | **Tipo** | **Descripción** |
| - | - | |
| **Retorno** | **Tipo** | **Descripción** |
| - | - | |
| **Notas** | | **Descripción** |
| - | | |

Ejemplo:

Ver Ejemplo completo 4 – Lectura de entradas por Eventos, pagina 28.

# 7 Funciones para placa STX570

A continuación se listan las funciones y constantes para comandar la placa STX570 conectada al puerto de expansión.

Recuerde leer la hoja de datos de la placa STX570 para el conexionado eléctrico y configuración de su dirección I2C (determinada por jumper J1, J2 y J3).

Todas las funciones relativas a la placa STX570, tienen el prefijo “Stx570”.

## 7.1 Constantes

| EDINx: Nombres de entradas discretas de placa STX570 utilizadas por funciones | | | |
|---|---|---|---|
| **Nombre** | **Tipo** | **Valor (bin)** | **Descripción** |
| EDIN1 | - | 00000001 | Representa a la entrada discreta numero 1. |
| EDIN2 | - | 00000010 | Representa a la entrada discreta numero 2. |
| EDIN3 | - | 00000100 | Representa a la entrada discreta numero 3. |
| EDIN4 | - | 00001000 | Representa a la entrada discreta numero 4. |
| EDIN5 | - | 00010000 | Representa a la entrada discreta numero 5. |
| EDIN6 | - | 00100000 | Representa a la entrada discreta numero 6. |
| EDIN7 | - | 01000000 | Representa a la entrada discreta numero 7. |
| EDIN8 | - | 10000000 | Representa a la entrada discreta numero 8. |

| EDOUTx: Nombres de salidas discretas de placa STX570 utilizadas por funciones | | | |
|---|---|---|---|
| **Nombre** | **Tipo** | **Valor (bin)** | **Descripción** |
| EDOUT1 | - | 00000001 | Representa a la salida discreta numero 1. |
| EDOUT2 | - | 00000010 | Representa a la salida discreta numero 2. |
| EDOUT3 | - | 00000100 | Representa a la salida discreta numero 3. |
| EDOUT4 | - | 00001000 | Representa a la salida discreta numero 4. |
| EDOUT5 | - | 00010000 | Representa a la salida discreta numero 5. |
| EDOUT6 | - | 00100000 | Representa a la salida discreta numero 6. |
| EDOUT7 | - | 01000000 | Representa a la salida discreta numero 7. |
| EDOUT8 | - | 10000000 | Representa a la salida discreta numero 8. |
| EDOUT_ALL | - | 11111111 | Representa todas las salidas discretas. |

## *7.2 Función para Inicializar la Placa STX570*

Antes de llamar a cualquier función para controlar la placa STX570, debe inicializarla. Se debe proceder a inicializar cada placa STX570 conectada al puerto de expansión (en caso de conexión de modo cascada).

| **Stx570_Init**(Addr): Inicializa la placa STX570 para operación. Debe ser la primera función en llamar antes de cualquier otra relacionada a la placa STX570. | | |
|---|---|---|
| **Argumentos** | **Tipo** | **Descripción** |
| Addr | E | Dirección de la placa STX570. Valor entre 0 y 7. Ver hoja de datos de placa. |
| **Retorno** | **Tipo** | **Descripción** |
| 0 | S | Operación exitosa. Placa inicializada correctamente. |
| -1 a -19 | S | Error. Código de error del bus I2C. Verifique hardware y/o conflicto de recursos. |
| -500 | S | La STX570 no fue declarada desde en el PLC, ver hoja de datos. |
| **Notas** | | **Descripción** |
| 1 | | Las salidas discretas son puestas a su estado inicial. |

Ejemplo 1:

```
PlcMain()
{
      …

      // Inicializar placa STX570, dirección 0,
      // conectada al puerto de expansión.
      Stx570_Init(0)

      …
}
```

Ejemplo 2:

Si la placa tiene configurada con los jumpers J1 a J3, la dirección 5, la función de inicialización se debería llamar como sigue:

```
PlcMain()
{
      …

      // Inicializar placa STX570, dirección 5,
      // conectada al puerto de expansión.
      Stx570_Init(5)

      …
}
```

## *7.3 Funciones para controlar Salidas Discretas*

Las siguientes funciones controlan las salidas discretas de la placa STX570, denominadas EDOUT1 a EDOUT8.

Físicamente, una salida discreta en la placa STX570 puede consistir en un RELAY, en un Transistor o en un TRIAC. El tipo de salida depende del modelo de la placa adquirido, consulte con la hoja de datos. Las funciones de control son idénticas independientemente del tipo de salida.

| **Stx570_DoutSet**(Addr, Dout): Activa una o varias salidas discretas de la placa STX570. | | |
|---|---|---|
| **Argumentos** | **Tipo** | **Descripción** |
| Addr | E | Dirección de la placa STX570. Valor entre 0 y 7. Ver hoja de datos de placa. |
| Dout | E | Salida o salidas discretas para activar. |
| **Retorno** | **Tipo** | **Descripción** |
| 0 | S | Operación exitosa. |
| -1 a -19 | S | Error. Código de error del bus I2C. Verifique hardware y/o conflicto de recursos. |
| -500 | S | La STX570 no fue declarada en el PLC, ver hoja de datos. |
| **Notas** | | **Descripción** |
| 1 | | Si quiere asegurarse que la función fue ejecutada correctamente, compruebe que el valor de retorno sea 0. |

Ejemplo 1:

```
// Activar salida 1 de la placa de expansión.
Stx570_DoutSet(0, EDOUT1)
```

Ejemplo 2:

```
// Activar salida 1, 4 y 7 de la placa de expansión al mismo tiempo.
Stx570_DoutSet(0, EDOUT1 | EDOUT4 | EDOUT7)
```

Ejemplo 3:

```
// Activar todas las salidas al mismo tiempo.
Stx570_DoutSet(0, EDOUT_ALL)
```

Ejemplo 4:

```
// Comprobar qué la función se ejecutó correctamente.
if(Stx570_DoutSet(0, EDOUT1) != 0)
{
      // Error.
}
```

| **Stx570_DoutClr**(Addr, Dout): Desactiva una o varias salidas discretas de la placa STX570. | | |
|---|---|---|
| **Argumentos** | **Tipo** | **Descripción** |
| Addr | E | Dirección de la placa STX570. Valor entre 0 y 7. Ver hoja de datos de placa. |
| Dout | E | Salida o salidas discretas para desactivar. |
| **Retorno** | **Tipo** | **Descripción** |
| 0 | S | Operación exitosa. |
| -1 a -19 | S | Error. Código de error del bus I2C. Verifique hardware y/o conflicto de recursos. |
| -500 | S | La STX570 no fue declarada en el PLC, ver hoja de datos. |
| **Notas** | | **Descripción** |
| 1 | | Si quiere asegurarse que la función fue ejecutada correctamente, compruebe que el valor de retorno sea 0. |

Ejemplo 1:

```
// Desactivar salida 1 de la placa de expansión.
Stx570_DoutClr(0, EDOUT1)
```

Ejemplo 2:

```
// Desactivar salida 1, 4 y 7 de la placa de expansión al mismo tiempo.
Stx570_DoutClr(0, EDOUT1 | EDOUT4 | EDOUT7)
```

Ejemplo 3:

```
// Desactivar todas las salidas al mismo tiempo.
Stx570_DoutClr(0, EDOUT_ALL)
```

Ejemplo 4:

```
// Comprobar qué la función se ejecutó correctamente.
if(Stx570_DoutClr(0, EDOUT1) != 0)
{
      // Error.
}
```

| **Stx570_DoutToggle**(Addr, Dout): Conmuta una o varias salidas discretas de la placa STX570. La salida invierte su estado. | | |
|---|---|---|
| **Argumentos** | **Tipo** | **Descripción** |
| Addr | E | Dirección de la placa STX570. Valor entre 0 y 7. Ver hoja de datos de placa. |
| Dout | E | Salida o salidas discretas para conmutar. |
| **Retorno** | **Tipo** | **Descripción** |
| 0 | S | Operación exitosa. |
| -1 a -19 | S | Error. Código de error del bus I2C. Verifique hardware y/o conflicto de recursos. |
| -500 | S | La STX570 no fue declarada en el PLC, ver hoja de datos. |
| **Notas** | | **Descripción** |
| 1 | | Si quiere asegurarse que la función fue ejecutada correctamente, compruebe que el valor de retorno sea 0. |

Ejemplo 1:

```
// Conmutar salida 1 de la placa de expansión.
Stx570_DoutToggle(0, EDOUT1)
```

Ejemplo 2:

```
// Conmutar salida 1, 4 y 7 de la placa de expansión al mismo tiempo.
Stx570_DoutToggle(0, EDOUT1 | EDOUT4 | EDOUT7)
```

Ejemplo 3:

```
// Conmutar todas las salidas al mismo tiempo.
Stx570_DoutToggle(0, EDOUT_ALL)
```

Ejemplo 4:

```
// Conmutar qué la función se ejecutó correctamente.
if(Stx570_DoutToggle(0, EDOUT1) != 0)
{
      // Error.
}
```

| **Stx570_DoutReadAll**(Addr, DoutState): Retorna el estado de todas las salidas discretas de la placa en la variable “DoutState”. | | |
|---|---|---|
| **Argumentos** | **Tipo** | **Descripción** |
| Addr | E | Dirección de la placa STX570. Valor entre 0 y 7. Ver hoja de datos de placa. |
| DoutState | S | Retorno del estado de todas las salidas. Cada bit representa una salida.<br>Bit 0 representa salida 1 y bit 7 representa la salida numero 8. Ver tabla en pag. 13. |
| **Retorno** | **Tipo** | **Descripción** |
| 0 | S | Operación exitosa. |
| -1 a -19 | S | Error. Código de error del bus I2C. Verifique hardware y/o conflicto de recursos. |
| -500 | S | La STX570 no fue declarada en el PLC, ver hoja de datos. |
| **Notas** | | **Descripción** |
| 1 | | Si quiere asegurarse que la función fue ejecutada correctamente, compruebe que el valor de retorno sea 0. |

Ejemplo 1:

```
// En esta variable almacenamos en cada uno de sus bits, el estado de las
// salidas discretas de la placa STX570 conectada.
new Stx570_DoutState = 0

// Inicializar placa STX570, dirección 0,
// conectada al puerto de expansión.
Stx570_Init(0)

// Leer valores de salidas desde placa STX570 y almacenar
// su estado en la variable Stx570_DoutState.
Stx570_DoutReadAll(0, Stx570_DoutState)
```

| **Stx570_DoutWrite**(Addr, DoutValue): Escribe un valor en cada salida discreta. | | |
|---|---|---|
| **Argumentos** | **Tipo** | **Descripción** |
| Addr | E | Dirección de la placa STX570. Valor entre 0 y 7. Ver hoja de datos de placa. |
| DoutValue | E | Valor a escribir en cada salida discreta de la placa. El bit-0 representa la salida 1 y el bit-7 representa la salida 8 de la placa. |
| **Retorno** | **Tipo** | **Descripción** |
| 0 | S | Operación exitosa. |
| -1 a -19 | S | Error. Código de error del bus I2C. Verifique hardware y/o conflicto de recursos. |
| -500 | S | La STX570 no fue declarada en el PLC, ver hoja de datos. |
| **Notas** | | **Descripción** |
| 1 | | Si quiere asegurarse que la función fue ejecutada correctamente, compruebe que el valor de retorno sea 0. |

Ejemplo 1:

```
// Inicializar placa STX570, dirección 0,
// conectada al puerto de expansión.
Stx570_Init(0)

// Escribir el valor 7, que en binario es: 00000111
// Esto implica que las salidas 1 a 3 se activaran y las salidas
// 4 a 8 se desactivaran.

Stx570_DoutWrite(0, 7)
```

## *7.4 Ejemplo completo 1 - Salidas Discretas*

El siguiente ejemplo pertenece al PLC de la familia STX8081-CX, pero puede utilizarse en cualquier PLC con lenguaje PAWN de Slicetex. Este documento adjunta varios scripts de prueba como referencia.

El script inicializa la placa STX570 cuya dirección I2C es cero, luego activa y desactiva la salida 1 cada 2 segundos. **Notar** que no se utilizan eventos en este script, ya que no usamos las entradas discretas.

```
// ****************************************************************************
// PlcMain(): Funcion principal. Punto de entrada al script.
//
// ENTRADAS:
//
//    - Ninguna.
//
// SALIDAS:
//
//    - Ninguna.
//
// ****************************************************************************

PlcMain()
{
      // Limpiar LCD.
      LcdClear()

      // Mensaje de Bienvenida.
      LcdPrintf(0, 0, "STX570 Test 1")

      // Inicializar placa STX570, dirección 0,
      // conectada al puerto de expansión.
      Stx570_Init(0)

   while(true)
   {
      // Pausa.
      DelayS(2)

      // Activar salida 1 de la placa de expansión.
      Stx570_DoutSet(0, EDOUT1)

      // Pausa.
      DelayS(2)

      // Desactivar salida 1 de la placa de expansión.
      Stx570_DoutClr(0, EDOUT1)
   }
}
```

## 7.5 Funciones para leer Entradas Discretas

Las funciones descriptas a continuación, se pueden utilizar para leer las entradas discretas de la placa STX570, denominadas EDIN1 a EDIN8.

Debe notar que existen dos métodos para leer las entradas discretas, que se aplican a criterio del usuario, dependiendo del proyecto:

1. El primer método consiste en **leer** las entradas **por polling**, es decir interrogar a la placa STX570 cada un intervalo especifico. Esto puede ser útil, si solo queremos leer entradas que cambian su estado lentamente o en periodos prolongados. También para aprovechar todas las entradas en un sistemas con placas en cascadas. Ver ejemplo en pag. 24.

2. El segundo método, consiste en **crear un evento**, que nos indique en el momento justo en que se produce el cambio de una entrada discreta. Esto puede ser útil con entradas que desconocemos en qué momento pueden cambiar su estado y necesitamos un tiempo de respuesta elevado. Ver ejemplo en pag. 28.

---

**Stx570_DinReadAll**(Addr, DinState): Retorna el estado de todas las entradas discretas de la placa en la variable “DinState”.

| **Argumentos** | **Tipo** | **Descripción** |
|---|---|---|
| Addr | E | Dirección de la placa STX570. Valor entre 0 y 7. Ver hoja de datos de placa. |
| DinState | S | Retorno del estado de todas las entradas. Cada bit representa una entrada.<br>Bit 0 representa entrada 1 y bit 7 representa la entrada numero 8. Ver tabla en pag. 13. |
| **Retorno** | **Tipo** | **Descripción** |
| 0 | S | Operación exitosa. |
| -1 a -19 | S | Error. Código de error del bus I2C. Verifique hardware y/o conflicto de recursos. |
| -500 | S | La STX570 no fue declarada en el PLC, ver hoja de datos. |
| **Notas** | | **Descripción** |
| 1 | | Si quiere asegurarse que la función fue ejecutada correctamente, compruebe que el valor de retorno sea 0. |

Ejemplo 1:

```
// En esta variable almacenamos en cada uno de sus bits, el estado de las
// entradas discretas de la placa STX570 conectada.
new Stx570_DinState = 0

// Inicializar placa STX570, dirección 0,
// conectada al puerto de expansión.
Stx570_Init(0)

// Leer valores de entradas desde placa STX570 y almacenar
// su estado en la variable Stx570_DinState.
Stx570_DinReadAll(0, Stx570_DinState)
```

| **Stx570_DinRead**(DinState, Din): Lee el estado de una entrada discreta desde variable “DinState” y retorna su estado. | | |
|---|---|---|
| **Argumentos** | **Tipo** | **Descripción** |
| DinState | E | Variable con el estado de todas las entradas. Esta variable se obtiene con la función **Stx570_DinReadAll(),** ver pag. 21. |
| Din | E | Entrada discreta para comprobar. Pueden comprobarse varias al mismo tiempo. Utilizar constantes EDIN definidas en pag. 13. |
| **Retorno** | **Tipo** | **Descripción** |
| 0 | S | Entrada discreta en nivel bajo (o **todas** las entradas seleccionadas en bajo nivel). |
| 1 | S | Entrada discreta en nivel alto (o **alguna** entrada seleccionada en nivel alto). |
| -500 | S | La STX570 no fue declarada en el PLC, ver hoja de datos. |
| **Notas** | | **Descripción** |
| 1 | | Esta función no se conecta mediante bus I2C a la placa STX570 a diferencia del resto de las funciones, por lo tanto es muy útil y rápida para usar en conjunto con eventos. |

Ejemplo 1:

```
// En esta variable almacenamos en cada uno de sus bits, el estado de las
// entradas discretas de la placa STX570 conectada.
new Stx570_DinState = 0

// Inicializar placa STX570, dirección 0,
// conectada al puerto de expansión.
Stx570_Init(0)

// Leer valores de entradas desde placa STX570 y almacenar
// su estado en la variable Stx570_DinState.
Stx570_DinReadAll(0, Stx570_DinState)

// …

// Comprobar si la entrada discreta 1 está en nivel alto.
// Recordar que solo leemos el valor de variable “Stx570_DinState“.
if(Stx570_DinRead(Stx570_DinState, EDIN1) == 1)
{
   // Si, está en nivel alto.
}

// Comprobar si la entradas 3 y 6 están en nivel bajo.
// Recordar que solo leemos el valor de variable “Stx570_DinState“.
if(Stx570_DinRead(Stx570_DinState, EDIN3 | EDIN6) == 0)
{
   // Si, ambas entradas están en nivel bajo.
}
```

| **Stx570_DinGetValue**(Addr, Din): Se conecta a la placa STX570 y devuelve el estado de la entrada seleccionada. | | |
|---|---|---|
| **Argumentos** | **Tipo** | **Descripción** |
| Addr | E | Dirección de la placa STX570. Valor entre 0 y 7. Ver hoja de datos de placa. |
| Din | E | Entrada discreta para comprobar. Pueden comprobarse varias al mismo tiempo. Utilizar constantes EDIN definidas en pag. 13. |
| **Retorno** | **Tipo** | **Descripción** |
| 0 | S | Entrada discreta en nivel bajo (o **todas** las entradas seleccionadas en bajo nivel). |
| 1 | S | Entrada discreta en nivel alto (o **alguna** entrada seleccionada en nivel alto). |
| -500 | S | La STX570 no fue declarada en el PLC, ver hoja de datos. |
| **Notas** | | **Descripción** |
| 1 | | Esta función no comprueba errores del bus I2C, por lo tanto en aplicaciones donde necesitamos verificar errores no se aconseja su uso. La función es práctica para aplicaciones simples donde deseamos evitar llamar a la función Stx570_DinReadAll() y Stx570_DinRead() al mismo tiempo. |

Ejemplo 1:

```
// Inicializar placa STX570, dirección 0,
// conectada al puerto de expansión.
Stx570_Init(0)


// …

// Comprobar si la entrada discreta 1 está en nivel alto.
// Notar que leemos la entrada directamente, sin variables intermedias.
if(Stx570_DinGetValue(0, EDIN1) == 1)
{
   // Si, está en nivel alto.
}

// Comprobar si la entradas 3 y 6 están en nivel bajo.
if(Stx570_DinGetValue(0, EDIN3 | EDIN6) == 0)
{
   // Si, ambas entradas están en nivel bajo.
}
```

### *7.5.1 Ejemplo completo 2 – Lectura de entradas por Polling*

Este script muestra el valor de las entradas discretas de la placa de expansión STX570 en el display LCD.

Si la entrada DIN1 esta polarizada, en el display se mostrará: "1 0 0 0 0 0 0 0".

Si la entradas DIN1 y DIN4 están polarizadas, se mostrará: "1 0 0 1 0 0 0 0".

Cada carácter, representa el valor de una entrada: "1" o "0".

Este ejemplo utiliza el método de polling para saber en qué momento el valor alguna entrada ha cambiado su valor.

Constantemente el programa principal requiere consultar a la placa STX570 el estado de sus entradas. Su ventaja consiste en que no hay eventos programados y en un sistema en cascada (varias placas conectadas en serie) se aprovechan todas las entradas, pero como desventaja, debemos hacer comprobaciones constantemente y no aprovechamos ese tiempo del procesador para realizar otras tareas útiles.

```
// *****************************************************************************
// PlcMain(): Función principal. Punto de entrada al script.
//
// ENTRADAS:
//
//    - Ninguna.
//
// SALIDAS:
//
//    - Ninguna.
//
// *****************************************************************************

PlcMain()
{
   // Limpiar LCD.
   LcdClear()

   // Mensaje de Bienvenida.
   LcdPrintf(0, 0, "STX570 Test 6")

   // Inicializar placa STX570, dirección 0,
   // conectada al puerto de expansión.
   Stx570_Init(0)

   while(true)
   {
      // Comprobar Estado de Entradas Discretas de placa STX570.
      //
      // Notar que se interroga a la placa STX570 para conocer
      // el estado de sus entradas constantemente.
      //
      // El uso de la función Stx570_DinGetValue() tiene como ventaja
```

```
      // su simple uso, pero no comprueba errores de conexión entre
      // la placa STX8081 y la STX570, por lo tanto si existe alguno,
      // no podremos saberlo. Se recomienda utilizar para operaciones
      // simples.
      //
      // Notar que cada vez que es llamada la función, debe conectarse
      // a la placa STX570, por lo tanto es ineficiente en cuanto a velocidad
      // comparado a la función Stx570_DinReadAll() que lee todas
      // las entradas al mismo tiempo.

      // Imprimir Valor de Entrada 1.
      LcdPrintf(0, 1, "%d",  Stx570_DinGetValue(0, EDIN1))

      // Imprimir Valor de Entrada 2.
      LcdPrintf(2, 1, "%d", Stx570_DinGetValue(0, EDIN2))

      // Imprimir Valor de Entrada 3.
      LcdPrintf(4, 1, "%d", Stx570_DinGetValue(0, EDIN3))

      // Imprimir Valor de Entrada 4.
      LcdPrintf(6, 1, "%d", Stx570_DinGetValue(0, EDIN4))

      // Imprimir Valor de Entrada 5.
      LcdPrintf(8, 1, "%d", Stx570_DinGetValue(0, EDIN5))

      // Imprimir Valor de Entrada 6.
      LcdPrintf(10, 1, "%d", Stx570_DinGetValue(0, EDIN6))

      // Imprimir Valor de Entrada 7.
      LcdPrintf(12, 1, "%d", Stx570_DinGetValue(0, EDIN7))

      // Imprimir Valor de Entrada 8.
      LcdPrintf(14, 1, "%d", Stx570_DinGetValue(0, EDIN8))

      // Pausa 80 mS.
      DelayMS(80)

      // Conmutar led debug del PLC.
      LedToggle()
   }
}
```

### *7.5.2 Ejemplo completo 3 – Lectura de entradas por Polling 2*

Este script es idéntico a “Ejemplo completo 2”, solo que utiliza la función “Stx570_DinReadAll()” para leer todas las entradas al mismo tiempo y así ser más eficiente en cuanto a velocidad.

El script muestra el valor de las entradas discretas de la placa de expansión STX570 en el display LCD.

---

```
// *****************************************************************************
// PlcMain(): Función principal. Punto de entrada al script.
//
// ENTRADAS:
//
//    - Ninguna.
//
// SALIDAS:
//
//    - Ninguna.
//
// *****************************************************************************

PlcMain()
{
   // En esta variable almacenamos en cada uno de sus bits, el estado de las
   // entradas discretas de la placa STX570 conectada.
   new Stx570_DinState = 0

   // Limpiar LCD.
   LcdClear()

   // Mensaje de Bienvenida.
   LcdPrintf(0, 0, "STX570 Test 4")

   // Inicializar placa STX570, dirección 0,
   // conectada al puerto de expansion.
   Stx570_Init(0)

   while(true)
   {
      // Comprobar Estado de Entradas Discretas de placa STX570.
      //
      // Notar que se interroga a la placa STX570 para conocer
      // el estado de sus entradas constantemente.

      // Leer valores de entradas desde placa STX570 y almacenar
      // su estado en la variable Stx570_DinState.
      Stx570_DinReadAll(0, Stx570_DinState)

      // Imprimir Valor de Entrada 1.
      LcdPrintf(0, 1, "%d",  Stx570_DinRead(Stx570_DinState, EDIN1))

      // Imprimir Valor de Entrada 2.
      LcdPrintf(2, 1, "%d", Stx570_DinRead(Stx570_DinState, EDIN2))
```

```
      // Imprimir Valor de Entrada 3.
      LcdPrintf(4, 1, "%d", Stx570_DinRead(Stx570_DinState, EDIN3))

      // Imprimir Valor de Entrada 4.
      LcdPrintf(6, 1, "%d", Stx570_DinRead(Stx570_DinState, EDIN4))

      // Imprimir Valor de Entrada 5.
      LcdPrintf(8, 1, "%d", Stx570_DinRead(Stx570_DinState, EDIN5))

      // Imprimir Valor de Entrada 6.
      LcdPrintf(10, 1, "%d", Stx570_DinRead(Stx570_DinState, EDIN6))

      // Imprimir Valor de Entrada 7.
      LcdPrintf(12, 1, "%d", Stx570_DinRead(Stx570_DinState, EDIN7))

      // Imprimir Valor de Entrada 8.
      LcdPrintf(14, 1, "%d", Stx570_DinRead(Stx570_DinState, EDIN8))

      // Pausa 80 mS.
      DelayMS(80)

      // Conmutar led debug del PLC.
      LedToggle()
   }
}
```

### 7.5.3 Ejemplo completo 4 – Lectura de entradas por Eventos

Este script muestra el valor de las entradas discretas de la placa de expansión STX570 en el display LCD.

Si la entrada DIN1 esta polarizada, en el display se mostrará: "1 0 0 0 0 0 0 0".

Si la entradas DIN1 y DIN4 están polarizadas, se mostrará: "1 0 0 1 0 0 0 0".

Cada carácter, representa el valor de una entrada: "1" o "0".

Este ejemplo utiliza eventos para saber en qué momento el valor alguna entrada ha cambiado su valor. Muy útil en aplicaciones donde necesitamos un tiempo de respuesta optimo.

Su ventaja respecto a los ejemplos anteriores, consiste en que el programa principal puede hacer otra actividad útil y no estar constantemente interrogando a la placa STX570.

```
// ----------------------------------------------------------------------------
// VARIABLES GLOBALES
// ----------------------------------------------------------------------------

// En esta variable global almacenamos en cada uno de sus bits,
// el estado de las entradas discretas de la placa STX570 conectada.
new Stx570_DinState = 0

// ----------------------------------------------------------------------------
// FUNCIONES
// ----------------------------------------------------------------------------

// ****************************************************************************
// PlcMain(): Función principal. Punto de entrada al script.
//
// ENTRADAS:
//
//    - Ninguna.
//
// SALIDAS:
//
//    - Ninguna.
//
// ****************************************************************************

PlcMain()
{
   // Limpiar LCD.
   LcdClear()

   // Mensaje de Bienvenida.
   LcdPrintf(0, 0, "STX570 Test 5")
```

```
   // Inicializar placa STX570, dirección 0,
   // conectada al puerto de expansión.
   Stx570_Init(0)

   // Leer valores de entradas desde placa STX570 y almacenar
   // su estado en la variable global Stx570_DinState.
   Stx570_DinReadAll(0, Stx570_DinState)

   // Permitir eventos en Puerto de Expansión.
   ExPortSetEvent()

   while(true)
   {
      // Comprobar Estado de Entradas Discretas de placa STX570.
      //
      // Notar que no se interroga a la placa STX570 para conocer
      // el estado de sus entradas, sino que solo se verifica
      // el la variable Stx570_DinState que es actualizada
      // cuando alguna entrada cambia desde el evento @OnExPort().

      // Imprimir Valor de Entrada 1.
      LcdPrintf(0, 1, "%d",  Stx570_DinRead(Stx570_DinState, EDIN1))

      // Imprimir Valor de Entrada 2.
      LcdPrintf(2, 1, "%d", Stx570_DinRead(Stx570_DinState, EDIN2))

      // Imprimir Valor de Entrada 3.
      LcdPrintf(4, 1, "%d", Stx570_DinRead(Stx570_DinState, EDIN3))

      // Imprimir Valor de Entrada 4.
      LcdPrintf(6, 1, "%d", Stx570_DinRead(Stx570_DinState, EDIN4))

      // Imprimir Valor de Entrada 5.
      LcdPrintf(8, 1, "%d", Stx570_DinRead(Stx570_DinState, EDIN5))

      // Imprimir Valor de Entrada 6.
      LcdPrintf(10, 1, "%d", Stx570_DinRead(Stx570_DinState, EDIN6))

      // Imprimir Valor de Entrada 7.
      LcdPrintf(12, 1, "%d", Stx570_DinRead(Stx570_DinState, EDIN7))

      // Imprimir Valor de Entrada 8.
      LcdPrintf(14, 1, "%d", Stx570_DinRead(Stx570_DinState, EDIN8))

      // Pausa 50 mS.
      DelayMS(50)
   }
}
```

```
// ***************************************************************************
// @OnExPort(): Manejador del evento @OnExPort.
//
// Cuando este evento es llamado, significa que hay un evento que atender en
// el puerto de expansión.
//
// Si hay una placa STX570 conectada al puerto, significa que el ESTADO
// de alguna entrada discreta de la placa cambio de estado.
//
// Esto nos indica que podemos leer el nuevo estado de las entradas ya
// que ocurrió un cambio en ellas.
//
// Cuando
//
// ENTRADAS:
//
//    - Ninguna.
//
// SALIDAS:
//
//    - Ninguna.
//
//
***************************************************************************

@OnExPort()
{
   // Alguna entrada discreta cambio su estado.
   // Leer valores desde placa STX570 y almacenar estado
   // en la variable global Stx570_DinState.
   Stx570_DinReadAll(0, Stx570_DinState)

   // Conmutar LED debug del PLC.
   LedToggle()
}
```

### *7.5.4 Caso Especial – Placas Conectadas en Cascada - Eventos*

En el caso de conectar varias placas en cascada, como se describe en la hoja de datos, será necesario tener en cuenta lo siguiente:

1. La entrada discreta numero 8 de cada placa conectada, indica que alguna placa conectada en cascada tiene alguna entrada que cambio de estado. Esto no se aplica en la última placa, donde la entrada discreta 8 puede utilizarse normalmente.
2. Cuando un evento nos informe el cambio de estado de una entrada, recomendamos leer con Stx570_ReadAll() todas las placas conectadas, dentro de la función @OnExPort().

**Alternativa**: Si no usa eventos, puede leer las entradas de todas las placas por polling (ver pag. 24) y aprovechar todas las entradas de cada placa.

Ejemplo:

Se tienen dos placas conectadas en cascada.

```
// Variables globales para almacenar valor entradas de ambas placas.

new Stx570_DinState_0 = 0
new Stx570_DinState_1 = 0

PlcMain()
{
   …

   // Inicializar primera STX570, dirección 0.
   Stx570_Init(0)

   // Inicializar segunda STX570, dirección 1.
   Stx570_Init(1)

   // Activar evento @OnExPort()
   ExPortSetEvent()

   …

   while(true)
   {
      // Leer entrada 2 de primera placa.
      if(Stx570_DinRead(Stx570_DinState_0, EDIN2) == 1)
      {
         // Esta en nivel alto.
      }

      // Leer entrada 6 de segunda placa.
      if(Stx570_DinRead(Stx570_DinState_1, EDIN6) == 0)
      {
         // Esta en nivel bajo.
      }
   }
}
```

```
@OnExPort()
{
   // Alguna entrada discreta cambio su estado.

   // Leer valores de primera placa STX570 y almacenar estado
   // en la variable global Stx570_DinState_0.
   Stx570_DinReadAll(0, Stx570_DinState_0)

   // Leer valores de segunda placa STX570 y almacenar estado
   // en la variable global Stx570_DinState_1.
   Stx570_DinReadAll(1, Stx570_DinState_1)

   // Toggle OnBoard Led.
   LedToggle()
}
```

Notar que en el ejemplo, usamos dos variables globales para mantener el último valor leído de las entradas discretas de ambas placas. Luego desde la función del evento, las actualizamos llamando a la función Stx570_DinReadAll() para cada placa.

Desde el programa principal, utilizamos Stx570_DinRead() para comprobar entradas en ambas placas, a través de las variables globales actualizadas.

Finalmente, si tenemos 3 o más placas conectadas en cascada, el proceso es similar, pero teniendo en cuenta un mayor número de placas.

# 8 Abreviaciones y Términos Empleados

- **PLC**: Programable Logic Controller (Controlador Lógico Programable).
- **I2C**: Bus de datos llamado Inter-Integraterd Circuit.
- **DIN:** Entrada Discreta, también conocida como DI.
- **DOUT:** Salida Discreta, también conocida como DO. Puede consistir en relés.
- **HP2**: Es el nombre del puerto de expansión “ExPort” de un PLC. Se refiere a un puerto de salida.
- **EXT**: Bornera que contiene entradas +12 y GND para fuente 12Vdc externa.
- **Cascada**: Conexión de varias placas en serie.

# 9 Historial de Revisiones

**Tabla: Historia de Revisiones del Documento**

| Revisión | Cambios | Descripción | Estado |
|---|---|---|---|
| 03<br>15/May/2015 | 1 | 1. Documentación para lenguaje Ladder agregada. | Preliminar |
| 02<br>16/SEP/2012 | 1 | 2. Documentación adaptada a StxLadder. | Preliminar |
| 01<br>06/FEB/2011 | 1 | 1. Versión preliminar liberada. | Preliminar |

# 10 Referencias

Ninguna.

# 11 Información Legal

## 11.1 Aviso de exención de responsabilidad

**General:** La información de este documento se da en buena fe, y se considera precisa y confiable. Sin embargo, Slicetex Electronics no da ninguna representación ni garantía, expresa o implícita, en cuanto a la exactitud o integridad de dicha información y no tendrá ninguna responsabilidad por las consecuencias del uso de la información proporcionada.

**El derecho a realizar cambios:** Slicetex Electronics se reserva el derecho de hacer cambios en la información publicada en este documento, incluyendo, especificaciones y descripciones de los productos, en cualquier momento y sin previo aviso. Este documento anula y sustituye toda la información proporcionada con anterioridad a la publicación de este documento.

**Idoneidad para el uso:** Los productos de Slicetex Electronics no están diseñados, autorizados o garantizados para su uso en aeronaves, área médica, entorno militar, entorno espacial o equipo de apoyo de vida, ni en las aplicaciones donde el fallo o mal funcionamiento de un producto de Slicetex Electronics pueda resultar en lesiones personales, muerte o daños materiales o ambientales graves. Slicetex Electronics no acepta ninguna responsabilidad por la inclusión y / o el uso de productos de Slicetex Electronics en tales equipos o aplicaciones (mencionados con anterioridad) y por lo tanto dicha inclusión y / o uso es exclusiva responsabilidad del cliente.

**Aplicaciones:** Las aplicaciones que aquí se describen o por cualquiera de estos productos son para fines ilustrativos. Slicetex Electronics no ofrece representación o garantía de que dichas aplicaciones serán adecuadas para el uso especificado, sin haber realizado más pruebas o modificaciones.

**Los valores límites o máximos:** Estrés por encima de uno o más valores límites (como se define en los valores absolutos máximos de la norma IEC 60134) puede causar daño permanente al dispositivo. Los valores límite son calificaciones de estrés solamente y el funcionamiento del dispositivo en esta o cualquier otra condición por encima de las indicadas en las secciones de Características de este documento, no está previsto ni garantizado. La exposición a los valores limitantes por períodos prolongados puede afectar la fiabilidad del dispositivo.

**Documento**: Prohibida la modificación de este documento en cualquier medio electrónico o impreso, sin autorización previa de Slicetex Electronics por escrito.

## 12 Información de Contacto

Para mayor información, visítenos en www.slicetex.com

Para información general y ventas, envíe un mail a: info@slicetex.com

Para soporte técnico, ingrese a nuestro foro: www.slicetex.com/foro

Ing. Boris Estudiez

Slicetex Electronics
Córdoba, Argentina

# 13 Contenido

www.slicetex.com