SHINYX 2.0 SP2

USER MANUAL

Conversion tool for SFSIEETH, SFNETLINK and MODBUS TCP/IP driver

Trademark Declarations

Automa, SUPER-FLASH, SFW, WRUNFILE, SHINYX are registered trademarks of Automa srl.
SIEMENS, S7 are registered trademarks of SIEMENS.
Modbus, MODBUS-TCP are registered trademarks of Schneider Automation.

All other trademarks not explicitly stated are the property of their respective companies.

INDEX

1 Introduction

SHINYX is a tool that facilitates the conversion of the communication mode with some types of peripheral devices, automating some of the operations necessary in the case of migration to SFSIEETH, SFNETLINK and MODBUS TCP/IP drivers within applications already developed.

The conversions supported by SHINYX are listed in the table below:

<table>
<tr><th>Source Peripheral</th><th>Destination Peripheral</th></tr>
<tr><td>Siemens S7-H1 (Applicom)
Siemens S7-MPI (Applicom)

Communication via Applicom cards or software.</td><td rowspan="3">SFNETLINK (Link Driver)
SFSIEETH (Link Driver)
Note 1</td></tr>
<tr><td>S7-MHI ADAPTER (Link Driver)

Communication via Siemens or Helmholz HMI Adapter.</td></tr>
<tr><td>Siemens Progr.
Siemens Progr. CP945
Siemens P3964 with RK512
Siemens P3964R with RK512

RS232/RS485 serial communication.</td></tr>
<tr><td>Siemens S7-200

RS232/RS485 serial communication.</td><td rowspan="2">SFSIEETH (Link Driver)</td></tr>
<tr><td>SFNETLINK (Link Driver)

Communication via Hilscher NetLINK-MPI Ethernet Gateway.</td></tr>
<tr><td>MODBUS

RS232/RS485 serial communication.</td><td rowspan="2">MODBUS TCP/IP (Link Driver)</td></tr>
<tr><td>Ethernet communication via peripherals:

MODBUSTCP 1.0 (MicroC Driver)

Ethernet communication.</td></tr>
</table>

Notes
1) SFNETLINK and SFSIEETH Link Driver use the same addressing mode; the only difference is that SFNETLINK does not support bit-type resources.

2 Procedure for communication conversion

To change the communication mode by switching to the use of Link Drivers such as SFSIEETH, SFNETLINK or MODBUS TCP/IP, the operations to be performed by the programmer are:

1. Install the driver on the PC
2. Install the driver in the existing application
3. Insert the driver in the Link Driver archive
4. Convert the variable addresses
5. Prepare the driver configuration .INI file
6. Verify use of MICROC change of address functions
7. Remove previously used software/hardware (optional)

SHINYX works only on points 3 and 4.

The other operations must be performed by the programmer and must be done in strict sequence; in short:

1. **Install the driver on the PC**

   The driver installer software copies the files into a folder on the PC.

2. **Install the driver in the existing application**

   Following the instructions in the driver's manual, the programmer copies the driver files into the application's folder.

3. **Insert the driver in the Link Driver archive**
4. **Convert the variable addresses**

   The two operations are carried out by SHINYX according to the operating mode described in the section **Using and operating SHINYX** on page **4**

   In the case of variables with addresses that are not compatible or not supported by the destination peripheral, the conversion will not be carried out and SHINYX, in this situation, will simply report how many and which variables have not been converted; the programmer will then have to modify them manually in the development system.

   In some cases the conversion may have different solutions based on factors that cannot be determined automatically; in this situation SHINYX performs the conversion by adopting the most standard solution but will signal to the programmer the need for verification.

   If the outcome of SHINYX was positive, it is necessary to open the project with the Development System, save it and proceed with the remaining conversion operations.

5. **Prepare the driver configuration .INI file**
   The .INI file (SFSIEETH.INI, NETLINK.INI or MODBUS.INI) must be prepared as documented in the driver's operating manual; each peripheral must be defined as equipment identified by a number. During conversion SHINYX automatically assigns an equipment number using the node number (where present) or the channel number used in the initial device. In the case of multi-node configurations on multiple channels and with coincident node numbers, it will be necessary to reassign the equipment number.

   For further information please refer to the **Reference tables for address conversion** section on page **6**.

6. **Verify use of MicroC change of address functions**
   Check the use of vmc_set_address() and set_address_var() functions in MICROC programs and change the address composition.

   Refer to the **Reference tables for address conversion** section on page **6**.

7. **Remove previously used sofware/hardware**

   The removal operation changes according to the device previously used; it is an optional operation not mandatory for the application to work with the new driver:

   - Applicom cards or software: if present, the Applicom software, the Applicom card and any Sentinel driver must be uninstalled from the PC.
   - Siemens or Helmholz HMI Adapter: The S7-HMI ADAPTER driver must be removed in the application configuration and the S7HMI.DLL and S7HMI.PAR files must be deleted.
   - NetLINK-MPI Hilscher Ethernet gateway: the SFNETLINK driver must be removed in the application configuration and the SFNETLNK.DLL and NETLINK.INI files must be deleted.
   - MODBUSTCP (MicroC driver): the 16-bit MODBUSTCP driver(s) and 32-bit MODBUSTCP driver(s) must be removed in the application configuration.

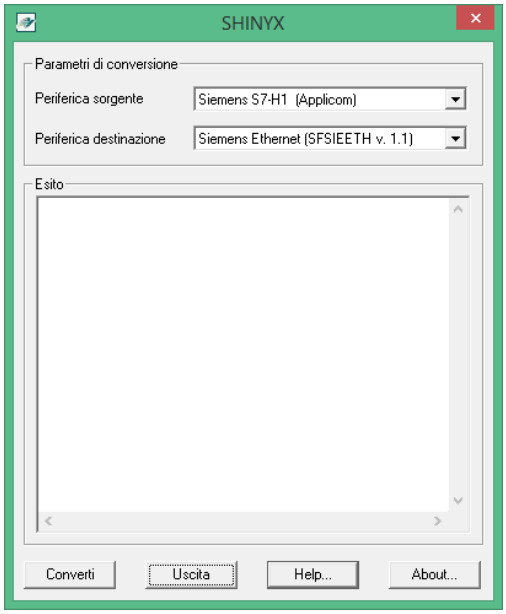
3 Using and operating SHINYX

To use SHINYX you need to:

1. Copy SHINYX.exe in the application directory
2. Close the project if it is open in the Development System

The tool can be used with SUPER-FLASH applications developed with version 3.2 and higher; running SHINYX does not cause the application to be updated, so it maintains the same version with which it was originally developed.

Launching the tool, you will see a simple interface window in which you have to choose the "Periferica sorgente" and the "Periferica destinazione" and then you have to start the conversion with the "Converti" Button.



Starting the conversion, the operations that SHINYX performs are:

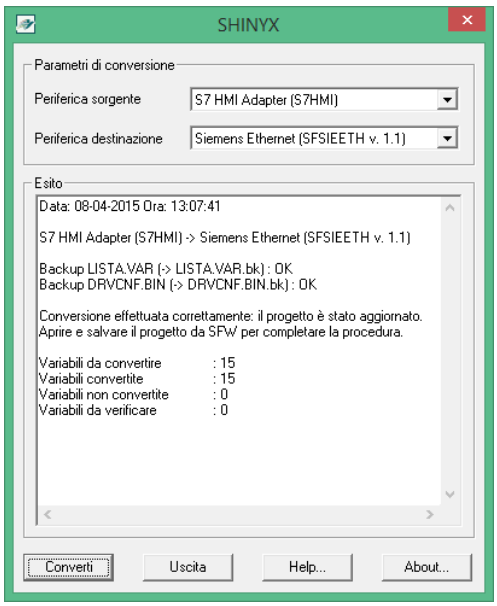
1. creates a backup of the Variables and Link Driver archive files (respectively LISTA.VAR.bk and DRVCNF.BIN.bk). If necessary, it is then possible to restore the original situation by renaming the files manually. This operation takes place only the first time the tool is started, before performing the conversion operations
2. inserts the driver in the Link Driver archive, verifying that it is not already present
3. looks for all the variables that have the source device as reference, if the address is compatible it replaces the device and converts the address

At the end of the conversion, it is possible to see the result of the operation in the specific window where are displayed:

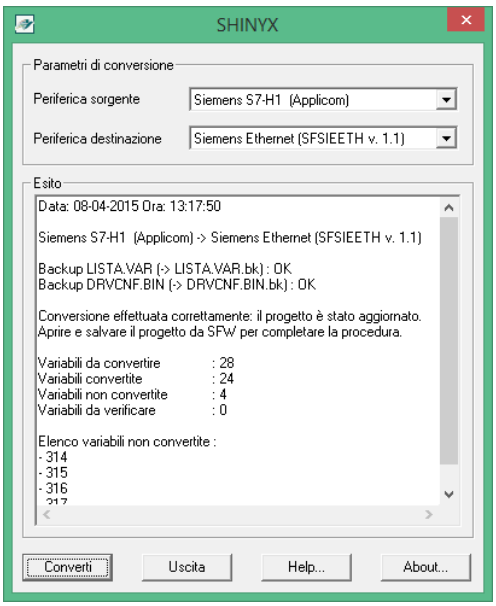
- the total number of detected variables to be converted (using the source device)
- the number of successfully converted variables
- the number of unconverted variables
- the code of the not converted variables, if any: the programmer will have to manually modify the variables
- the number of converted variables to verify
- the code of the variables to verify, if any: the programmer will have to verify and decide whether to maintain or modify the conversion made
- a message reporting whether the project has been updated or not

In the case of unconverted variables or variables to be verified, the programmer can refer to the indications contained in the conversion reference table for the concerned protocols.

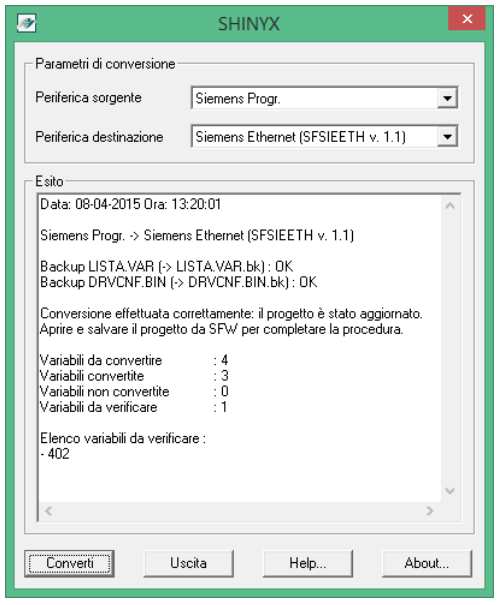
The following images indicate the possible results of the conversion operation:



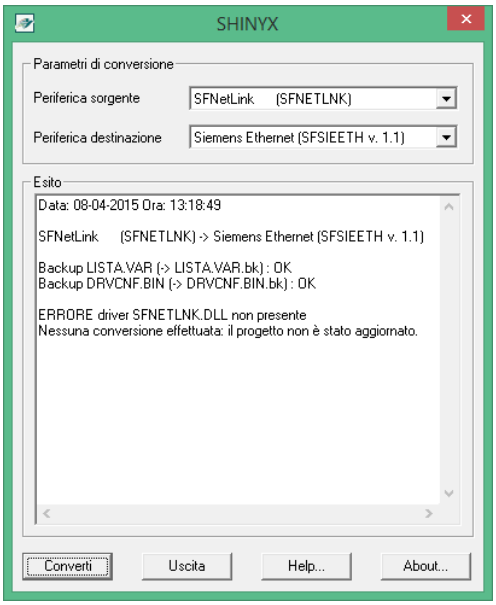
Conversion completed successfully: all variables have been converted



Successful partial conversion: 24 of 28 variables have been converted



Successful conversion with a variable to be verified.



Conversion failed: no variable to convert was found

The result of the last conversion is saved in a text file named SHINYX.log; the logger file SHINYX_HISTORY.log stores all the conversions eventually done in the application.

If the result has been positive, meaning that the project has been updated, it is necessary to open the project with the Development System, save it and proceed with the remaining conversion operations listed in the previous paragraph.

4 Reference tables for address conversion

To compose the new address refer to the conversion tables below.
In some cases the address conversion is also influenced by the size of the variable; where indicated, for brevity, the following identifiers are used:

DIM16 : Single word (16 bit), Date, Message
DIM32 : Double word (32 bit), Time
BOOL : Boolean
ALPHA : Alphanumeric
BLOCK : Block

4.1 Siemens S7 H1/MPI(Applicom) to SFSIEETH

Siemens S7-H1 (Applicom) **Siemens S7-MPI (Applicom)**	**SFSIEETH**
Old address	**New address**
cc\eee\MM\y.z	M\0\eee\0\My.z (1)
cc\eee\DM\x\y.z	M\0\eee\0\DBx.y.z (1)
cc\eee\IM\y.z	M\0\eee\0\Iy.z (1)
cc\eee\OM\y.z	M\0\eee\0\Oy.z (1)
cc\eee\MB\y	B\0\eee\0\My
cc\eee\DB\x\y	B\0\eee\0\DBx.y
cc\eee\IB\y	B\0\eee\0\Iy
cc\eee\OB\y	B\0\eee\0\Oy
cc\eee\MW\y	W\0\eee\0\My
cc\eee\DW\x\y	W\0\eee\0\DBx.y
cc\eee\IW\y	W\0\eee\0\Iy
cc\eee\OW\y	W\0\eee\0\Oy
cc\eee\MD\y	D\0\eee\0\My
cc\eee\DD\x\y	D\0\eee\0\DBx.y
(1) The SFSIEETH v. 1.1 driver only supports single bits; therefore the conversion is not performed for variables with block or alphanumeric size.	

4.2 Siemens S7 H1/MPI(Applicom) to SFNETLINK

Siemens S7-H1 (Applicom) **Siemens S7-MPI (Applicom)**	**SFNETLINK**
Old address	**New address** (2)
cc\eee\MM\y.z	Not convertible (1)
cc\eee\DM\x\y.z	Not convertible (1)
cc\eee\IM\y.z	Not convertible (1)
cc\eee\OM\y.z	Not convertible (1)
cc\eee\MB\y	B\0\eee\0\My
cc\eee\DB\x\y	B\0\eee\0\DBx.y
cc\eee\IB\y	B\0\eee\0\Iy
cc\eee\OB\y	B\0\eee\0\Oy
cc\eee\MW\y	W\0\eee\0\My
cc\eee\DW\x\y	W\0\eee\0\DBx.y
cc\eee\IW\y	W\0\eee\0\Iy
cc\eee\OW\y	W\0\eee\0\Oy
cc\eee\MD\y	D\0\eee\0\My
cc\eee\DD\x\y	D\0\eee\0\DBx.y

(1) SFNETLINK driver does not support bit resource management.
(2) The equipment number eee is assigned with the same number used in Applicom except in the case of 0 (zero), because the SFNETLINK driver does not support it; in that case it will be assigned with the first free number.

4.3 S7- HMI Adapter to SFSIEETH/SFNETLINK

S7- HMI Adapter	**SFSIEETH** **SFNETLINK**
Old address	**New address** (1)(2)
W\n\y\x\D	W\0\n\0\DBx.y
W\n\y\0\M	W\0\n\0\My
W\n\y\0\O	W\0\n\0\Oy
W\n\y\0\I	W\0\n\0\Iy
D\n\y\x\D	D\0\n\0\DBx.y
D\n\y\0\M	D\0\n\0\My
D\n\y\0\O	D\0\n\0\Oy
D\n\y\0\I	D\0\n\0\Iy

(1) The equipment number is assigned as the node number n of the old address except in the case of 0 (zero), because the SFNETLINK driver does not support it; in that case it will be assigned with the first free number.
(2) The SFSIEETH e SFNETLINK drivers manage byte addressing: if the ADDBYTE macro is not present in the S7HMI.PAR file (word addressing was used) the new address must be calculated by multiplying the previous one by two (New y = Old y*2)
Ex. W\1\6\10\D W\0\1\0\DB10.12

4.4 Siemens Progr/P3964 to SFSIEETH/SFNETLINK

Siemens Progr. **Siemens Progr. CP945** **Siemens P3964 con RK512** **Siemens P3964R con RK512**	**SFSIEETH** **SFNETLINK**
Old address	**New address** (1)(2)
D\x:y (BOOL, DIM16) D\x:y (DIM32) D\x:y (BLOCK) D\x:y (ALPHA)	W\0\n\0\DBx.y D\0\n\0\DBx.y (3) W\0\n\0\DBx.y (4) W\0\n\0\DBx.y o B\0\n\0\DBx.y (5)
(1) The equipment number n is assigned as the channel number configured in the variable. (2) The SFSIEETH e SFNETLINK drivers manage byte addressing: the new address must be calculated by multiplying the old one by two (New y = Old y *2). Ex. D\10:6 W\0\1\0\DB10.12 (3) The address is converted if the "Swap 32 bit" parameter is set to NO. (4) The address is converted with the W type: but if the variable is treated to derive double words, the address must be modified with the D type. (5) If the "Swap alfanumerica " parameter is set to SI the W resource type must be used, otherwise B.	

4.5 Siemens S7 200 to SFSIEETH/SFNETLINK

Siemens S7 200	**SFSIEETH** **SFNETLINK**
Old address	**New address** (1)(2)
n\V\y (DIM16,BOOL) n\V\y (DIM32) n\V\y (BLOCK) n\V\y (ALPHA)	W\0\n\0\DB1.y D\0\n\0\DBx.y (3) W\0\n\0\DBx.y (4) W\0\n\0\DBx.y o B\0\n\0\DBx.y (5)
n\M\y (DIM16,BOOL) n\M\y (DIM32) n\M\y (BLOCK) n\M\y (ALPHA)	W\0\n\0\M.y D\0\n\0\My (3) W\0\n\0\My (4) W\0\n\0\My o B\0\n\0\My (5)
n\I\y (DIM16,BOOL) n\I\y (DIM32) n\I\y (BLOCK) n\I\y (ALPHA)	W\0\n\0\I.y D\0\n\0\Iy (3) W\0\n\0\Iy (4) W\0\n\0\Iy o B\0\n\0\Iy (5)
n\O\y (DIM16,BOOL) n\O\y (DIM32) n\O\y (BLOCK) n\O\y (ALPHA)	W\0\n\0\O.y D\0\n\0\Oy (3) W\0\n\0\Oy (4) W\0\n\0\Oy o B\0\n\0\Oy (5)

(1) The equipment number n is assigned as the node number of the old address except in the case of 0 (zero), since the SFNETLINK driver does not support it; in that case it will be assigned with the first free number. In case of multi-peripheral situation on multiple channels with shared node numbers the parameter will be reassigned according to the configuration made in the driver configuration file.
(2) The SFSIEETH e SFNetLink drivers manage byte addressing: the new address must be calculated by multiplying the old one by two (New y = Old y *2).
Ex. D\10:6 W\0\1\0\DB10.12
(3) The address is converted if the "Swap 32 bit" parameter is set to NO
(4) The address is converted with the W type: but if the variable is treated to derive double words, the address must be modified with the D type.
(5) If the "Swap alfanumerica " parameter is set to SI the W resource type must be used, otherwise B.

4.6 MODBUS to MODBUS TCP/IP

MODBUS	**MODBUS TCP/IP**
Old address	**New address** (1)
n\W:y o n\R:y (DIM16,ALPHA) n\W:y o n\R:y (BLOCK) n\W:y o n\R:y (DIM32) n\A:y (DIM16,BLOCK,ALPHA) n\A:y (DIM32) n\B:y o n\F:y (DIM16,BOOL) n\D:y (DIM16, BOOL) n\B:y o n\F:y (BLOCK) n\D:y (BLOCK)	W\0\n\0\Oy W\0\n\0\Oy(3) D\0\n\0\Oy W\0\n\0\Iy D\0\n\0\Iy M\0\n\0\Oy M\0\n\0\Iy Not convertible (2) Not convertible (2)
(1) The equipment number n is assigned as the node number of the old address. In case of multi-peripheral situation on multiple channels with shared node numbers the parameter will be reassigned according to the configuration made in the driver configuration file. (2) Blocks of bits cannot be converted. The MODBUS driver puts each bit into separate words while the MODBUS TCP/IP driver uses packed mode which involves merging the bits. (3) The address is converted with type W assuming that only 16 bit words are derived from the block. If double words are derived from the block, the address must be modified with type D; finally, if the block is used in mixed mode to derive both words and double words, further verifications are required.	

4.7 MODBUSTCP (MicroC Driver) to MODBUS TCP/IP

MODBUSTCP (Driver MicroC)	**MODBUS TCP/IP**
Old address	**New address** (1)
c\n\OM\0\y (BOOL, DIM16, DIM32)	M\0\n\0\Oy
c\n\SM\0\y (BOOL, DIM16, DIM32)	M\0\n\0\Oy (3)
c\n\IM\0\y (BOOL, DIM16, DIM32)	M\0\n\0\Iy
c\n\OM\0\y (BLOCK)	Not convertible (2)
c\n\SM\0\y (BLOCK)	Not convertible (2)
c\n\IM\0\y (BLOCK)	Not convertible (2)
c\n\OW\0\y	W\0\n\0\Oy
c\n\SW\0\y (DIM16, BLOCK, ALPHA)	W\0\n\0\Oy (3)
c\n\IW\0\y (DIM16, BLOCK, ALPHA)	W\0\n\0\Iy
c\n\OD\0\y (DIM32, BLOCK)	D\0\n\0\Oy (4)
c\n\SD\0\y (DIM32, BLOCK)	D\0\n\0\Oy (3)(4)
c\n\ID\0\y (DIM32, BLOCK)	D\0\n\0\Iy (4)

(1) The equipment number is assigned like the node number. In case of multi-peripheral situation on several channels with shared node numbers the parameter will be reassigned according to the configuration made in the driver configuration file.
(2) Blocks of bits cannot be converted. The MODBUSTCP driver puts each bit in separate words, while the MODBUS TCP/IP driver uses packed mode, which involves packing bits into bytes.
(3) If the resource type of the old address is SM/SW/SD, you may need to specify the SingleWrite parameter in the equipment configuration to set the single write mode.
(4) If the ADDRESS32 parameter of the MicroC driver configuration profile is set to YES the new address must be calculated by multiplying the old one by two (New y = Old y *2).

5 Notes to the versions

5.1 Version 2.0 SP1 (April 2017)

Solved problems:

- In conversions to the SFNETLINK driver corrected the equipment number assignment to avoid using the number 0 (zero) (not managed by the driver)
- In the case of variables all of size "Block" some conversions (e.g. MODBUS to MODBUS TCP/IP) are not performed and the result is "No conversion performed".

5.2 Version 2.0 SP2 (July 2021)

Solved problems:

- The tool does not work in Super-Flash projects developed with version 4.6.