

## Driver Manual

# FS-8700-40 SBT-FSI

(Siemens Building Technologies – Foreign Systems Interface)

### APPLICABILITY & EFFECTIVITY

Effective for all systems manufactured after October 2023.



Driver Revision: 1.14  
Document Revision: 13.C



**fieldserver**

MSA Safety  
1000 Cranberry Woods Drive  
Cranberry Township, PA 16066 USA  
Website: [www.MSAsafety.com](http://www.MSAsafety.com)

U.S. Support Information:

+1 408 964-4443  
+1 800 727-4377  
Email: [smc-support@msasafety.com](mailto:smc-support@msasafety.com)

EMEA Support Information:

+31 33 808 0590  
Email: [smc-support.emea@msasafety.com](mailto:smc-support.emea@msasafety.com)

## Contents

<b>1</b>	<b>Description</b>	<b>4</b>
<b>2</b>	<b>Driver Scope of Supply</b>	<b>4</b>
2.1	Provided by the Supplier of 3 <sup>rd</sup> Party Equipment	4
2.1.1	Hardware	4
2.1.2	Required 3 <sup>rd</sup> Party Configuration	4
<b>3</b>	<b>Hardware Connections</b>	<b>5</b>
3.1	FieldServer RS-485 Connection to the MXL Panel	5
3.2	FieldServer RS-232 Connection to the XLS Panel	6
3.3	FSI Connection Notes	7
<b>4</b>	<b>Data Array Parameters</b>	<b>8</b>
<b>5</b>	<b>Client Side Configuration</b>	<b>9</b>
5.1	Client Side Connection Parameters	11
5.2	Client Side Node Parameters	12
5.3	Client Side Map Descriptor Parameters	12
5.3.1	FieldServer Specific Map Descriptor Parameters	12
5.3.2	Driver Related Map Descriptor Parameters	13
5.3.3	Timing Parameters	13
5.4	Map Descriptor Examples	14
5.4.1	Monitoring an MXL/XLS Device for Events	14
5.4.2	Monitoring Analog Loop Devices	16
5.4.3	Enumerated Message Type	17
<b>6</b>	<b>Server Side Configuration</b>	<b>18</b>
6.1	Server Side Connection Parameters	19
6.2	Server Side Node Parameters	19
6.3	Server Side Map Descriptor Parameters	20
6.3.1	FieldServer Specific Map Descriptor Parameters	20
6.3.2	Driver Related Map Descriptor Parameters	20
6.4	Map Descriptor Examples	21
6.4.1	Emulating a MXL/XLS Device with Events	21
6.4.2	Emulating an MXL/XLS Device with Analogs	22
<b>7</b>	<b>Useful Features</b>	<b>23</b>
7.1	Specifying a Poller Map Descriptor Length	23
7.2	Interpreting Panel Error Messages	24
7.3	Obtaining Point Addresses from the Panel Report	24
7.4	Activating and De-activating Entities	25
7.5	Arming and De-Arming Entities	26
7.6	Request ASD or Sensitivity	28
7.7	Request % Compensation Used / % Dirty	28
7.8	Opcode 45	29
7.9	Node Status Function	29
7.10	XLS Addressing	30
<b>8</b>	<b>Troubleshooting</b>	<b>31</b>
8.1	Checksum Errors	31
8.2	Recording Active Events – XLS Panel	31
8.3	Using Trouble Codes	31
8.4	Setting the Panel Language to French	34

## 1 Description

The SBT-FSI (Siemens Building Technologies – Foreign Systems Interface) driver allows the FieldServer to transfer data to and from Siemens MXL/XLS devices (NIM-1R<sup>1</sup> or NIM-1W or RPM<sup>2</sup>) over either RS-232 or RS-485 using MXL/XLS to foreign system protocol. The FieldServer can emulate either a Server or a Client.

**NOTE: The Driver does not support MXL and XLS panels networked together.**

### Max Nodes Supported

FieldServer Mode	Nodes	Comments
Client	1-10 (one Client per physical FieldServer serial port) depends on FieldServer model and configuration	Only 1 Client node allowed on multidrop systems
Server	99	Panel numbers from 1 to 99 may be emulated

## 2 Driver Scope of Supply

### 2.1 Provided by the Supplier of 3<sup>rd</sup> Party Equipment

#### 2.1.1 Hardware

Part #	Description
	Siemens MXL/XLS System
	Siemens NIM-1R Network Interface Module (for MXL usage)
	Siemens RPM Network Interface Module (for XLS usage)
500-699103	Siemens NIM-1W Network Interface Module (for MXL usage)
	Siemens Cable P/N 555-193085 (for RS-232 connection)

#### 2.1.2 Required 3<sup>rd</sup> Party Configuration

- The SBT-FSI System must be set up for remote monitoring with the MXL or XLS to foreign systems protocol.
- The ports on the RPM must be configured by the XLS panel. The RPM can be configured for RS-232 or RS-485, but not both together. Connection to the RS-485 port has not been tested and is not supported.

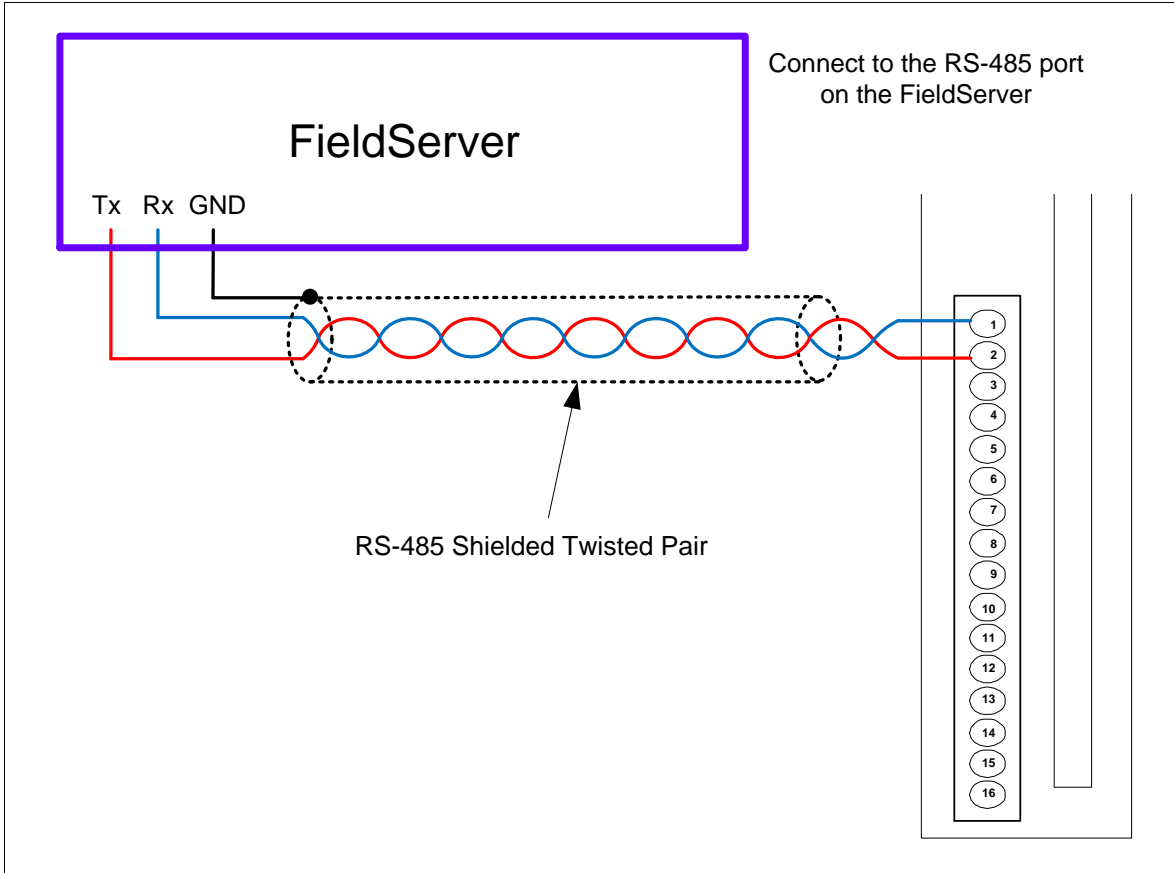
<sup>1</sup> NIM-1R is no longer supported by Siemens and is therefore considered to be legacy while NIM-1W is common.

<sup>2</sup> The ports on the RPM need to be configured by the XLS panel. Configuration can be set up for either RS-232 or RS-485, but not both together. Connection to the RS-485 port of the RPM has not been tested and is not supported.

## 3 Hardware Connections

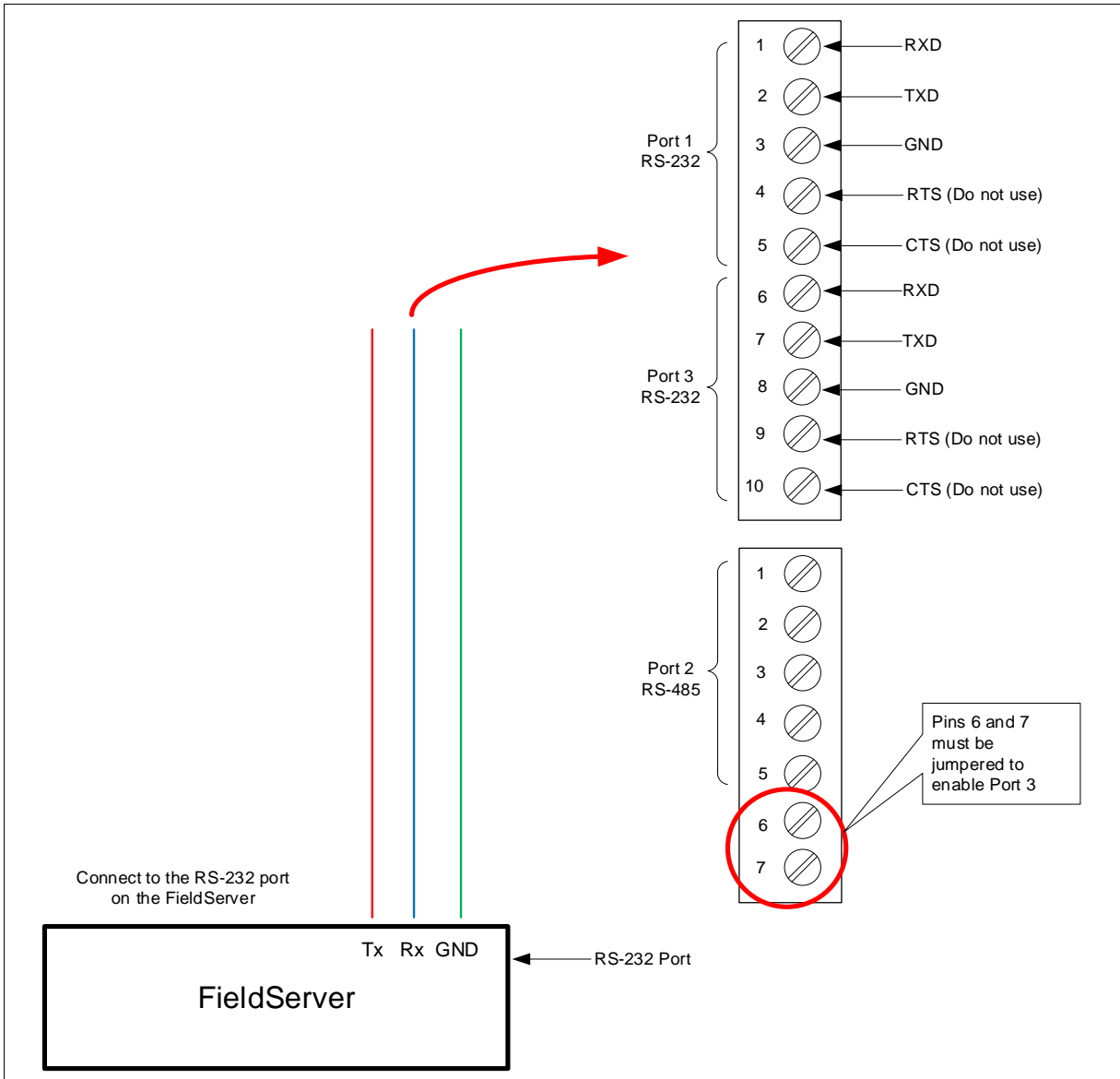
### 3.1 FieldServer RS-485 Connection to the MXL Panel

Connect the NIM-W to the RS-485 port on the FieldServer as depicted in the diagram below.



## 3.2 FieldServer RS-232 Connection to the XLS Panel

1. Configure the NIM-1R<sup>3</sup> or NIM-1W for Foreign System Interface (FSI) by setting all of the switches in SW2 to open (or OFF).
2. The ports on the RPM must be configured for RS-232 only.
3. The FieldServer can be connected to either Port 1 or Port 3 on the XLS Panel as shown in the connection drawing. Pins 6 and 7 on Port 2 must be jumpered to enable RS-232 on Port 3.



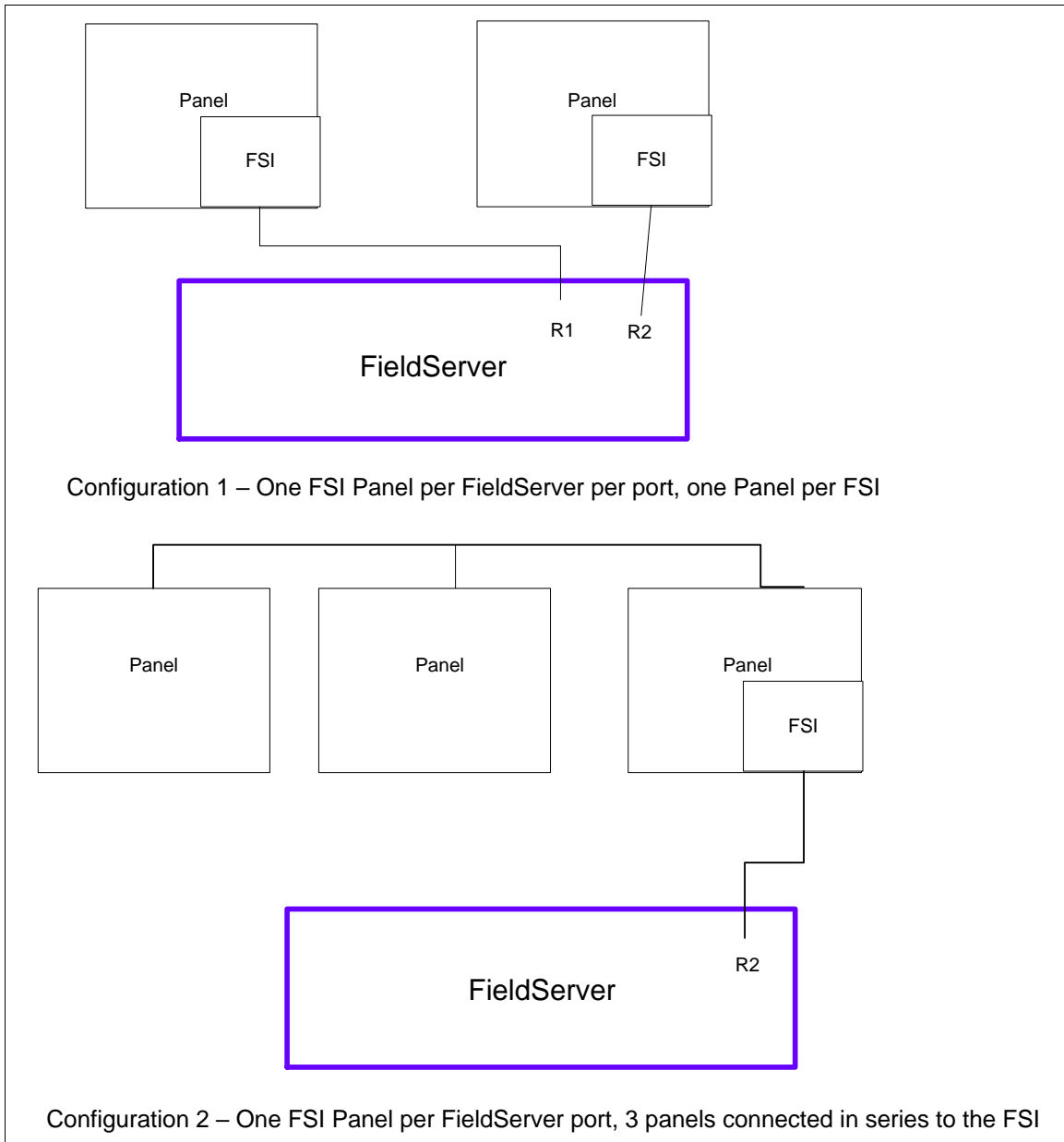
### Connector Pinouts for RS-232 Ports

FieldServer	XLS Panel Port 1 (3)	
Pin	Signal	Pin
Rx	TXD	2 (7)
Tx	RXD	1 (6)
GND	GND	3 (8)

<sup>3</sup> NIM-1R is no longer supported by Siemens and is therefore considered to be legacy while NIM-1W is common.

## 3.3 FSI Connection Notes

1. It is possible to connect an MXL device to any RS-232 or RS-485 port. The XLS device can only be connected to a RS-232 port.<sup>4</sup> These ports simply need to be configured for MXL/XLS in the configuration file.
2. The driver does not support MXL and XLS panels networked together.
3. One FSI panel may be connected per FieldServer port, but multiple panels may be connected per FSI. The following diagram illustrates two possible connection configurations.
4. Make sure the Event Message -> FSI Ack State is not selected in the Siemens port/panel setup program to ensure correct communication with the FieldServer.



<sup>4</sup> Connecting the XLS to a RS-485 port will cause the panel to go into reboot mode and continuously send events from 1^0^0^0^0

## 4 Data Array Parameters

Data Arrays are “protocol neutral” data buffers for storage of data to be passed between protocols. It is necessary to declare the data format of each of the Data Arrays to facilitate correct storage of the relevant data.

Section Title		
Data_Arrays		
Column Title	Function	Legal Values
Data_Array_Name	Provide name for Data Array.	Up to 15 alphanumeric characters
Data_Array_Format	Provide data format. Each Data Array can only take on one format.	Float, Bit, Byte, Uint16, Uint32, Sint16, Sint32
Data_Array_Length	Number of Data Objects. Must be larger than the data storage area required by the Map Descriptors for the data being placed in this array.	1-10000
Data_Array_Function	Special function for Data Array – Refer to <b>Section 7.9</b> .	Node_Status

### Example

```
// Data Arrays
Data_Arrays
Data_Array_Name , Data_Array_Format , Data_Array_Length
Event_Counts , Uint16 , 19
Fire , Bit , 2048
Fire_UnAck , Bit , 2048
```



## 5 Client Side Configuration

For detailed information on FieldServer configuration, refer to the FieldServer Configuration Manual. The information that follows describes how to expand upon the factory defaults provided in the configuration files included with the FieldServer (see “.csv” sample files provided with the FieldServer).

This section documents and describes the parameters necessary for configuring the FieldServer to communicate with an SBT-FSI Server.

In FSI mode, the NIM-1R or NIM-1W or RPM allows the FieldServer to gather data from up to 63 MXL/XLS Panels connected via an MXL/XLS network. The default configuration shipped with the FieldServer will monitor two panels (1 and 2) with 8 modules (1 to 8) each.

When the FieldServer is configured as a Client, two sets of data are collected by a “poller” Map Descriptor.

The first is a collection of 19 counters per panel. Each 16-bit counter is incremented by the poller Map Descriptor whenever the corresponding event occurs. An external Client can read these counters to quickly determine whether a new event has been reported to the Server. The counters are mapped into the Data Array specified for the poller Map Descriptor. Counter offsets into the Data Array per event type are indicated below.

Event	Counter Offset
Fire Alarm In	0
Fire Alarm Out	1
Fire Alarm Acknowledge	2
Trouble In	3
Trouble Out	4
Trouble Acknowledge	5
Supervisory In	6
Supervisory Out	7
Supervisory Acknowledge	8
Security Alarm In	9

Event	Counter Offset
Security Alarm Out	10
Security Alarm Acknowledge	11
Status In	12
Status Out	13
Test In	14
Test Out	15
Audible Silenced	16
Audible Unsilenced	17
System Reset	18

The second set of data is a collection of bit maps that can be queried to identify the device that has reported the event. When an alarm event arrives from a device, the corresponding bit is set in two separate arrays. The bit in one array indicates the alarm has occurred, and the bit in the other array indicates that the alarm has not yet been acknowledged. When an “alarm clear” event arrives, the bit in the alarm array is cleared. If an “alarm acknowledge” event arrives, the bit in the un-acknowledged array is cleared. All event arrays in a Client driver configuration are optional. To enable an event array, a Map Descriptor must be defined in the configuration using a Map Descriptor message type that corresponds to the array as shown in the following table

Array	Msg_Type
Fire Alarm	Fire
Fire Alarm Un-Acknowledged	Fire_Ack
Trouble Alarm	Trouble
Trouble Un-Acknowledged	Trouble_Ack
Supervisory Alarm	Super
Supervisory Un-Acknowledged	Super_Ack
Security Alarm	Secur

Array	Msg_Type
Security Alarm Un-Acknowledged	Secur_Ack
Status	Status
Test	Test
Audible (Un)Silenced	Audible
System Reset	Reset

### NOTES:

1. Map Descriptors are used to determine which bit to set in the Data Array. When an event arrives from a panel, the FieldServer will scan for a Map Descriptor with a Node\_ID matching the panel number reported in the event. This Map Descriptor will point to the Data Array to be modified. The location can be further refined if a starting module within the panel is specified in the Block\_Number field. Each module represents 256 bits that can be set or reset by an event. A Map Descriptor that is 512 bits long with a starting module (Block\_Number) of 3 represents the events for Modules 3 and 4. A Map Descriptor that is 10 bits long will only map the first 10 devices for that module. The combination of Block\_Number and Data\_Array\_Location can be used to map multiple panels with various numbers of modules into a packed array to be read by the Server side of the FieldServer.
2. When the customer receives a Communication FieldServer with the default driver installed, the configuration is set to poll 2 MXL/XLS devices once per second.
3. In addition to the two sets of event counters, alarm bit arrays are set up for 8 modules (1 to 8) per device. If the required configuration falls within this size, only the Server side of the FieldServer needs to be configured. To connect to a device using Modbus RTU protocol, configure the interface as if the FieldServer were another Modbus PLC and create a Map Descriptor to assign a Modbus address (such as 40001) to the Counts Data Array so that it can be read by the Modbus bus master.
4. If the customer has different panel numbers, the Nodes section of the configuration file will need to be changed. To add more panels, add Node entries and Map Descriptors by copying the original ones and modifying the Panel number entries and Data Array offsets. To access panels with more modules, first increase the Data Array lengths. If the extra modules are contiguous, the Map Descriptors for that panel can have their lengths increased also. If there are large gaps in the numbering of the module, it will be more efficient to add additional Map Descriptors by copying the ones that are given and modifying the Block\_Number to start at the next existing module number.
5. While the creation of passive Client-side Map Descriptors is optional, error messages will appear when events arrive for which Map Descriptors cannot be found. It is therefore recommended that Map Descriptors (especially ack mapdescs) be allocated even if only as dummies to prevent nuisance error messages.

## Configuring the FieldServer

The configuration file tells the FieldServer about its interfaces, and the routing of data required. In order to enable the FieldServer for SBT-FSI communications, the driver independent FieldServer buffers need to be declared in the “Data Arrays” section, the destination device addresses need to be declared in the “Client Side Nodes” section, and the data required from the Servers needs to be mapped in the “Client Side Map Descriptors” section. Details on how to do this can be found below.

**NOTE:** In the tables below, \* indicates an optional parameter, with the bold legal value being the default.

### 5.1 Client Side Connection Parameters

Section Title		
Connections		
Column Title	Function	Legal Values
Port	Specify which port the device is connected to the FieldServer.	P1-P2, R1-R2 <sup>5</sup>
Protocol	Specify protocol used.	Cerberus, SBT-FSI
Enhanced_Protocol	Enables legacy or enhanced protocol. MXL panels do not currently support Enhanced_Protocol, so this parameter must be set to “No” for MXL. The parameter is normally set to “Yes” for XLS but may be set to “No” if problems are being experienced.	Yes (XLS) <b>No</b> (MXL)
System_Type	Specifies Siemens Panel type.	MXL, XLS
Block_Acks <sup>6</sup>	Specifies if the driver uses block or individual command acks.	Yes, No
Fire_Alarms_Hold*	Specifies if the driver should hold fire alarms until the system reset, or clear on fire alarms going out.	Yes, <b>No</b>
Baud*	Specify baud rate.	<b>19200</b> (Vendor limitation)
Parity*	Specify parity.	<b>Even</b> (Vendor limitation)
Data_Bits*	Specify data bits.	<b>7</b> (Vendor limitation)
Stop_Bits*	Specify stop bits.	<b>1</b> (Vendor limitation)
Poll_Delay*	Time between internal polls.	0-32000 seconds, <b>1 second</b>
Language_Pack*	Set this parameter to yes if the panel is configured for a language other than English.	Yes, <b>No</b>

#### Example

```
// Client Side Connections
Connections
Port , Protocol , Baud , Parity , Data_Bits , Poll_Delay
P1 , SBT-FSI , 19200 , Even , 7 , 0.100s
```

<sup>5</sup> Not all ports shown are necessarily supported by the hardware. Consult the appropriate Instruction manual for details of the ports available on specific hardware.

<sup>6</sup> Refer to the Advanced Driver Manual for usage. This function not enabled on standard driver.

## 5.2 Client Side Node Parameters

Section Title		
Nodes		
Column Title	Function	Legal Values
Node_Name	Provide name for Node.	Up to 32 alphanumeric characters
Node_ID	Panel number.	1-999
Protocol	Specify protocol used.	Cerberus, SBT-FSI
Port	Specify on which port the device is connected to the FieldServer.	P1-P2, R1-R2 <sup>1</sup>

### Example

```
// Client Side Nodes

Nodes
Node_Name , Node_ID , Protocol , Port
Panel1 , 1 , SBT-FSI , P1
```

## 5.3 Client Side Map Descriptor Parameters

### 5.3.1 FieldServer Specific Map Descriptor Parameters

Column Title	Function	Legal Values
Map_Descriptor_Name	Name of this Map Descriptor.	Up to 32 alphanumeric characters
Data_Array_Name	Name of Data Array where data is to be stored in the FieldServer.	One of the Data Array names from <b>Section 4</b>
Data_Array_Offset	Starting location in Data Array.	0 to (Data_Array_Length -1) as specified in <b>Section 4</b>
Function	Function of Client Map Descriptor.	<p><b>Rdbc for Msg_Type</b> Poller, Analog_Volts, Analog_Sense, Analog_Thres</p> <p><b>Passive for Msg_Type</b> Fire_Alarm, Fire_Ack, Trouble, Trouble_Ack, Secur, Secur_Ack, Super, Super_Ack, Status, Test, Audible, Reset</p> <p><b>Awt for Msg_Type</b> Entity_Arm_Disarm; Entity_Activate_Deact</p>

## 5.3.2 Driver Related Map Descriptor Parameters

Column Title	Function	Legal Values
Node_Name	Name of Node to fetch data from.	One of the Node names specified in <b>Section 5.2</b>
Block_Number	Starting module number within panel. There are 256 points per module.	1 – 256
Sub_Module	Sub module number for XLS systems only.	0 – 99
Component_Address	Component address for XLS systems only.	0 – 99 <sup>7</sup>
Length	Length of Map Descriptor specifies number of points. There are 256 points per module.	1 – 10000
Msg_Type	Indicates a “poller” Map Descriptor or identifies the type of event message this Map Descriptor will handle. If this is configured, the driver will store a value that indicates the point type of the event if the Data Array used on the Map Descriptor is not of a bit type.	Poller; Fire_Alarm; Fire_Ack; Trouble; Trouble_Ack; Super; Super_Ack; Secur; Secur_Ack; Status; Test <sup>8</sup> ; Audible; Reset; Analog_Volts; Analog_Sense; Analog_Thres; Entity_Arm_Disarm <sup>9</sup> ; Entity_Activate_Deact <sup>10</sup> , Enumerated <sup>11</sup>
Analog_Device	Indicates range of device data to be requested by analogs.	0: devices 1-60 1: devices 61-121 2: devices 122-182 3: devices 183-243 4: devices 244-252
Block_Offset	This field allows you to set the start address of a MXL or XLS point in a Map Descriptor.	0 - 255
Entity_Types	Used to indicate the type of Entity on the panel. Refer to <b>Section 7.5</b> for more information.	Inputs, Outputs, (if all elements are Input or Output) OR specify each element of the entity individually (e.g. IOIOIOO)
Entity_Energizations	Used to control the entity on the panel. Refer to <b>Section 7.5</b> for more information.	De-energize, Energize, (if all elements are Energize or De-energize) OR specify each element of the entity individually (e.g. EDED---DED)

## 5.3.3 Timing Parameters

Column Title	Function	Legal Values
Scan_Interval*	Rate at which data is polled.	≥0.001s, <b>1.0s</b>

<sup>7</sup> Refer to **Section 7.10**.

<sup>8</sup> Test points are only used for testing purposes in a WalkTest mode and should not be used in a live system.

<sup>9</sup> Refer to **Section 7.5**.

<sup>10</sup> Refer to **Section 7.4**.

<sup>11</sup> Refer to **Section 5.4.3**.

## 5.4 Map Descriptor Examples

### 5.4.1 Monitoring an MXL/XLS Device for Events

To monitor an MXL/XLS device we need only one “poller” Map Descriptor as shown below. The function of this Map Descriptor’s is to continuously check the MXL device for events from panels. Only one poller Map Descriptor per FieldServer port is required when monitoring more than one MXL/XLS device. The Node\_Name will indicate on which port this Map Descriptor resides.

```
// Client Side Map Descriptors
Map_Descriptors
Map_Descriptor_Name , Data_Array_Name , Data_Array_Offset , Function , Node_Name , Block_Number , Length , Msg_Type
CD_Poller1 , Event_Counts1 , 0 , RDBC , Panel1 , 1 , 19 , Poller
```

In the above example:

- **Data\_Array\_Name** – Specifies Data Array that will hold event counters. Data Array length must be equal or greater than Data Array offset plus Map Descriptor length.
- **Data\_Array\_Offset** – Specifies offset into event counters Data Array where event counts will be stored.
- **Node\_Name** – Choose a Node\_Name from the passive Map Descriptors in the next section to specify the port on which this poller will be active.
- **Block\_Number** – Specifies lowest panel number from which events will be received. Refer to **Section 7.2** for more information.
- **Length** – Specify length in multiples of 19 events according to Block Number used. Refer to **Section 7.2** for more information.
- **Msg\_Type** – Specify poller message type for this Map Descriptor.

## Configuring the FieldServer

Passive Map Descriptors must be added for each event per panel to be monitored:

**NOTE: CD – Client Descriptor.**

```
// Client Side Map Descriptors

Map_Descriptors
Map_Descriptor_Name , Data_Array_Name , Data_Array_Offset , Function , Node_Name , Block_Number , Length , Msg_Type
CD_Fire1           , Fire           , 0           , Passive , Panel1   , 1           , 1024 , Fire
CD_Fire_Ack1      , Fire_Ack       , 0           , Passive , Panel1   , 1           , 1024 , Fire_Ack
CD_Trouble1       , Trouble        , 0           , Passive , Panel1   , 1           , 1024 , Trouble
CD_Trouble_Ack1   , Trouble_Ack    , 0           , Passive , Panel1   , 1           , 1024 , Trouble_Ack
CD_Super1         , Super          , 0           , Passive , Panel1   , 1           , 1024 , Super
CD_Super_Ack1     , Super_Ack      , 0           , Passive , Panel1   , 1           , 1024 , Super_Ack
CD_Secur1         , Secur          , 0           , Passive , Panel1   , 1           , 1024 , Secur
CD_Secur_Ack1     , Secur_Ack      , 0           , Passive , Panel1   , 1           , 1024 , Secur_Ack
CD_Status1        , Status         , 0           , Passive , Panel1   , 1           , 1024 , Status
CD_Test1          , Test           , 0           , Passive , Panel1   , 1           , 1024 , Test
CD_Audible1       , Audible        , 0           , Passive , Panel1   , -           , 1     , Audible
CD_Reset1         , Reset          , 0           , Passive , Panel1   , -           , 1     , Reset
```

In the above example:

- **Node\_Name** – Specifies the panel number for which events will be stored.
- **Block\_Number** – Specifies the starting module number under which points are mapped.
- **Length** – Specifies the number of points. There are 256 points per module. A length of 1024 holds points for 4 modules.
- **Msg\_Type** – The stored value for only the Test Msg\_Type, represents the following events:

Event_Type	Value
Alarm (Fire)	1
Trouble	2
Supervisory	3
Security	4
Status	5

**NOTES:**

1. All passive Map Descriptors are optional. Specific Map Descriptors may be defined for particular events to be monitored as per the Msg\_Type. Should events happen for which no Map Descriptors are defined, error messages will occur on the FieldServer. It may therefore be useful to add dummy Map Descriptors especially for ack responses to prevent nuisance error messages.
2. Passive Map Descriptors with Node\_Name the same as the poller Map Descriptor will only be able to store events from a panel with the same Node\_ID e.g. Node\_ID of 1 will only store events from panel 1. To store events from panel 2, it is necessary to add passive Map Descriptors with Node ID of 2 etc. Do not add poller Map Descriptors for other panels.
3. **Important! – First alarm point and Data Array offsets:**  
Most SBT-FSI panels have device point numbers starting at one instead of zero. The FieldServer MXL driver reserves Data Array offset zero for points with device numbers of zero although they are currently not used. Triggering the first alarm point on an MXL panel will thus result in a point status change at Data\_Array\_Offset of one instead of zero.

## 5.4.2 Monitoring Analog Loop Devices

Analog devices on a loop can be monitored for detector voltage, sensitivity voltage and threshold voltage using three analog Map Descriptors:

```
// Client Side Map Descriptors

Map_Descriptors
Map_Descriptor_Name, Data_Array_Name, Data_Array_Offset, Function, Node_Name, Block_Number, Length, Msg_Type, Analog_Device, Scan_Interval
CD_Analog_Volts1, Analog_Volts, 0, Rdbc, Panel1, 1, 60, Analog_Volts, 0, 300s
CD_Analog_Sens1, Analog_Sense, 0, Rdbc, Panel1, 1, 60, Analog_Sense, 0, 300s
CD_Analog_Thres1, Analog_Thres, 0, Rdbc, Panel1, 1, 60, Analog_Thres, 0, 300s
```

In the above example:

- **Msg\_Type** – Use Float Data Arrays for the analog Map Descriptors to get the full analog value.
- **Analog\_Device** – 0: devices 1-60; 1: devices 61-121; 2: devices 122-182; 3: devices 183-243; 4: devices 244-252.
- **Scan\_Interval** – This Map Descriptor on analog pollers may not be shorter than 300s to prevent flooding the Foreign Systems Interface.

**NOTE:** If necessary, use more analog Map Descriptors with other Analog\_Device numbers to poll other devices on the loop.



## 5.4.3 Enumerated Message Type

The Enumerated mapdesc will match incoming events from the Panel number as set by the Node\_ID under Nodes, and from the module as by the Block\_Number and from the submodule as set by the Sub\_Module mapdesc parameter. Component\_Address is an unused field.

The following point events with their storage values will be stored in the Data array specified under Data\_Array\_Name at the offset set by Data\_Array\_Offset:

```
// Client Side Map Descriptors
Map_Descriptors
Map_Descriptor_Name , Data_Array_Name , Data_Array_Offset , Function , Node_Name , Block_Number , Length , Msg_Type
CD_Enumerated1 , Enumerated , 0 , Passive , Panel1 , 1 , 1024 , Enumerated
```

**NOTE: Make sure the data array format is not bit otherwise it will only show a 1 or a 0.**

Event_Type	Value
Normal	0
Alarm (Fire)	1
Trouble	2
Supervisory	3
Security	4
Status	5

## 6 Server Side Configuration

For detailed information on FieldServer configuration, refer to the FieldServer Configuration Manual. The information that follows describes how to expand upon the factory defaults provided in the configuration files included with the FieldServer (see “.csv” files provided with the FieldServer).

This section documents and describes the parameters necessary for configuring the FieldServer to communicate with an SBT-FSI Client.

The driver can be used to emulate a MXL/XLS Server device. Other protocol drivers could then poll other remote devices and access the local MXL/XLS Server data to set or clear events. In this configuration an existing SBT-FSI system could be replaced with an MXL/XLS emulation. Existing Clients could poll the emulation driver on the FieldServer to get the same data as from a conventional MXL/XLS Server.

Up to 99 panels can be emulated with the driver. A Map Descriptor has to be defined for each type of remote device that would normally exist in a MXL/XLS Server. The following types can be used:

Device Type
Fire Alarm In
Fire Alarm Out
Fire Alarm Acknowledge
Trouble In
Trouble Out
Trouble Acknowledge
Supervisory In
Supervisory Out
Supervisory Acknowledge
Security Alarm In
Security Alarm Out
Security Alarm Acknowledge
Status Event
Test Event
Analog Voltage
Analog Sensitivity Voltage
Analog Threshold Voltage

The configuration file tells the FieldServer about its interfaces, and the routing of data required. In order to enable the FieldServer for SBT-FSI communications, the driver independent FieldServer buffers need to be declared in the “Data Arrays” section, the FieldServer virtual node(s) needs to be declared in the “Server Side Nodes” section, and the data to be provided to the Clients needs to be mapped in the “Server Side Map Descriptors” section. Details on how to do this can be found below.

**NOTE: In the tables below, \* indicates an optional parameter, with the bold legal value as default.**

## 6.1 Server Side Connection Parameters

Section Title		
Connections		
Column Title	Function	Legal Values
Port	Specify which port the device is connected to the FieldServer.	P1-P2, R1-R2 <sup>12</sup>
Protocol	Specify protocol used.	Cerberus, SBT-FSI
Enhanced_Protocol	Enables legacy or enhanced protocol. MXL panels do not currently support Enhanced_Protocol, so this parameter must be set to “No” for MXL. The parameter is normally set to “Yes” for XLS but may be set to “No” if problems are being experienced.	Yes (XLS) <b>No</b> (MXL)
System_Type	Specifies Siemens Panel type.	MXL, XLS
Echo	Specifies if the server should echo the client’s polls.	Yes, No
Baud*	Specify baud rate.	110 – 115200, standard baud rates only, <b>19200</b>
Parity*	Specify parity.	<b>Even</b> , Odd, None, Mark, Space
Data_Bits*	Specify data bits.	<b>7</b> , 8
Stop_Bits*	Specify stop bits.	<b>1</b> (Vendor limitation)
Server_Hold_Timeout	Specifies time FieldServer will reserve Server Side connection while waiting for the Client side to update data in Data_Array (if necessary).	>1.0s

### Example

```
// Server Side Connections
Connections
Port , Protocol , Baud , Parity , Data_Bits
P1 , SBT-FSI , 19200 , Even , 7
```

## 6.2 Server Side Node Parameters

Section Title		
Nodes		
Column Title	Function	Legal Values
Node_Name	Provide name for node.	Up to 32 alphanumeric characters
Node_ID	Panel number.	1-99
Protocol	Specify protocol used.	Cerberus, SBT-FSI
Server_Hold_Timeout	Specifies time FieldServer will reserve Server Side connection while waiting for the Client side to update data in Data_Array (if necessary).	>1.0s

### Example

```
// Server Side Nodes
Nodes
Node_Name , Node_ID , Protocol 13
Panel1 , 1 , SBT-FSI
```

<sup>12</sup>Not all ports shown are necessarily supported by the hardware. Consult the appropriate Instruction manual for details of the ports available on specific hardware.

<sup>13</sup>No port information is necessary on the Server side.

## 6.3 Server Side Map Descriptor Parameters

### 6.3.1 FieldServer Specific Map Descriptor Parameters

Column Title	Function	Legal Values
Map_Descriptor_Name	Name of this Map Descriptor.	Up to 32 alphanumeric characters <sup>14</sup>
Data_Array_Name	Name of Data Array where data is to be stored in the FieldServer.	One of the Data Array names from <b>Section 4</b>
Data_Array_Offset	Starting location in Data Array.	0 to maximum as specified in <b>Section 4</b>
Function	Function of Server Map Descriptor.	Server
Server_Hold_Timeout*	Specifies time FieldServer will reserve Server Side connection while waiting for the Client side to update data in Data_Array (if necessary).	>1.0s

### 6.3.2 Driver Related Map Descriptor Parameters

Column Title	Function	Legal Values
Node_Name	Name of Node or panel for which this Map Descriptor holds points.	One of the Node names specified in <b>Section 6.2</b>
Block_Number	Module number <b>NOTE: This only specifies one module number as opposed to a starting module number as in the Client Map Descriptor).</b>	1 – 256
Sub_Module	Sub module in XLS system.	0 – 99
Component_Address	Component address in XLS system.	0 – 99
Length	Length of Map Descriptor Specifies number of points for the indicated module. Note that MXL systems typically have only 256 points per module.	1 – 256
Msg_Type*	If configured, the driver will store a value that indicates the point type of the event if the Data Array used on the Map Descriptor is not of a bit type. <b>NOTE: A poller message type may not be used in a Server Map Descriptor.</b>	Fire_Alarm; Fire_Ack; Trouble; Trouble_Ack; Secur; Secur_Ack; Super; Super_Ack; Status; Test; Audible; Reset; Analog_Volts; Analog_Sense; Analog_Thres, -

<sup>14</sup> Object\_Name values of any length may be written via BACnet (subject only to memory and message length constraints).

## 6.4 Map Descriptor Examples

### 6.4.1 Emulating a MXL/XLS Device with Events

The following list of Map Descriptors emulates a MXL Server:

**NOTE: SD – Server Descriptor**

```
// Server Side Map Descriptors

Map Descriptors
Map_Descriptor_Name , Data_Array_Name , Data_Array_Offset , Function , Node_Name , Block_Number , Length , Msg_Type
SD_Fire1 , Fire , 0 , Server , Panel1 , 1 , 256 , Fire
SD_Fire_Ack1 , Fire_Ack , 0 , Server , Panel1 , 1 , 256 , Fire_Ack
SD_Trouble1 , Trouble , 0 , Server , Panel1 , 1 , 256 , Trouble
SD_Trouble2 , Trouble , 256 , Server , Panel1 , 2 , 256 , Trouble
SD_Trouble_Ack1 , Trouble_Ack , 0 , Server , Panel1 , 1 , 256 , Trouble_Ack
SD_Super1 , Super , 0 , Server , Panel1 , 1 , 256 , Super
SD_Super_Ack1 , Super_Ack , 0 , Server , Panel1 , 1 , 256 , Super_Ack
SD_Secur1 , Secur , 0 , Server , Panel1 , 1 , 256 , Secur
SD_Secur_Ack1 , Secur_Ack , 0 , Server , Panel1 , 1 , 256 , Secur_Ack
SD_Status1 , Status , 0 , Server , Panel1 , 1 , 256 , Status
SD_Audible1 , Audible , 0 , Server , Panel1 , - , 1 , Audible
SD_Reset1 , Reset , 0 , Server , Panel1 , - , 1 , Reset
```

**NOTE: Trouble modules 1 and 2 on Panel 1 are emulated with two separate Map Descriptors.**

In the above example:

- **Node\_Name** – Specifies the panel number being emulated by this Server Map Descriptor.
- **Block\_Number** – Specifies the module number. A typical Cerberus Server has 256 points per module.
- **Length** – Specifies the number of points represented within the indicated module.

**NOTES:**

1. MXL/XLS Server Map Descriptor fields differ in many respects from MXL/XLS Client Map Descriptors. Please follow the examples clearly and read all notes to ensure a working configuration. No poller Map Descriptors may be defined in a configuration file that is used to configure a MXL/XLS Server.
2. Only one module specified by Block\_Number can be emulated with a Server Map Descriptor whereas a Client Map Descriptor specifies a starting module number with one module for every 256 counts of the Map Descriptor length. You have to use separate Map Descriptors for each module to emulate when doing a Server configuration.
3. More than one Msg\_Type Map Descriptor may be used to represent other modules.

## 6.4.2 Emulating an MXL/XLS Device with Analogs

The following list of Map Descriptors emulates an MXL Server:

```
// Server Side Map Descriptors

Map Descriptor
Map_Descriptor_Name , Data_Array_Name , Data_Array_Offset , Function , Node_Name , Block_Number , Length , Msg_Type
SD_Analog_Volts1    , Volts            , 0                , Passive , Panel1    , 1        , 252    , Analog_Volts
SD_Analog_Sense1   , Sense            , 0                , Passive , Panel1    , 1        , 252    , Analog_Sense
SD_Analog_Thres1   , Thres            , 0                , Passive , Panel1    , 1        , 252    , Analog_Thres
```

**In the above example:**

- **Data\_Array\_Name** – Use Float Data Arrays for the analog Map Descriptors to get the full analog value.

## 7 Useful Features

### 7.1 Specifying a Poller Map Descriptor Length

The poller Map Descriptor is responsible for checking a remote MXL/XLS device for events. When events are received the poller Map Descriptor increments counters at various offsets in the Data Array specified by the poller Map Descriptor. The offsets and their meanings are explained under the Client configuration section of this Driver Manual.

The poller Map Descriptor's Data Array contains sets of 19 elements or counters. Each set of counters represents the events from a panel. The Block\_Number is used to specify the lowest panel number from which events will be received. The highest Node\_ID of any passive Map Descriptor on the same port as the poller Map Descriptor specifies the highest panel number from which events will be received. The poller Map Descriptor's length has to be at least 19 and has to be incremented by 19 for every extra panel above the Block\_Number from which events will be received.

If a poller Map Descriptor length is set below 19 and for example, event "System Reset" is received which needs to be mapped at offset 18 in the Data Array, the following kernel message will be printed:

```
"Cannot store alarm event type counts!", 11010" 'offset=%u, da=%s"
```

#### Example 1 – Only receive events from panel 7:

- Passive event Map Descriptor's Node Id = 7 (X)
- Poller Map Descriptor's Block\_Number = 7 (Y)
- Poller Map Descriptor's Length =  $(X - Y) * 19 + 19 = 19$
- Event counters for panel 7 will reside in the first 19 elements.

#### Example 2 – Receive events from panel 7 and 6:

- Passive event Map Descriptor's Node Id = 7 (X)
- Poller Map Descriptor's Block\_Number = 6 (Y)
- Poller Map Descriptor's Length =  $(X - Y) * 19 + 19 = 38$
- Event counters for panel 6 will reside in the first 19 elements and those for panel 7 in the next 19 elements.

**Warning:** An error will be generated when an event arrives from a panel with a number lower than the Block\_Number of the poller. The poller Map Descriptor's Block\_Number must equal the lowest panel number for which a passive Map Descriptor exists on the same port.

## 7.2 Interpreting Panel Error Messages

A NAK is sent by the panel when it is unable to respond to a message. The format will be as follows:

```
T02> DRV-> CER : Panel replied with NAK on last message!
T02> DRV-> CER : Reason: 1
```

The following table lists the reason for the failure and the suggested corrective action.

**NOTE: The messages are generated by the panel, not the FieldServer.**

Reason #	Error	Corrective Action
Reason 1	Checksum or Parity fail	Ensure connection baud rate, start, stop bits and parity match the panel's settings.
Reason 2	Receive buffer full	Increase scan interval on the poller Map Descriptor.
Reason 3	Syntax Error	Check integrity of physical connection to panel, route cables away from electromagnetic sources that can generate noise.
Reason 4	Unrecognized message failure	Replace hardware?

## 7.3 Obtaining Point Addresses from the Panel Report

In order to configure the FieldServer to capture data from the panel, the point addresses being used in the panel need to be identified. The panel report that can be printed out from the panel can be used for this purpose, but matching the report parameters with the FieldServer configuration parameters is not a trivial exercise.

The following notes should assist in identifying the addresses needed for FieldServer configuration:

- The preamble to the report is not relevant to the FieldServer driver.
- The first Module can be identified by the title NET Address =1. This corresponds to FieldServer's "Block 1".
- The Module Subaddress corresponds to individual points in the Block Array.

This information is depicted in the following file extract:

```
NET ADDR: 1 ==> ALD-2  2 Analog Loop Module
=====
Module Subaddress ----->  1
DEVICE TYPE -----> TRI-60/TRX-2/3  1 Switch Input
Usage -----> TROUBLE
Message -----> RCP-01 Trouble Input MM-1
Switch Contact Type -----> N.O.
```

Corresponds to Block Number 1

Corresponds to Block Array point 1



## 7.4 Activating and De-activating Entities

These examples demonstrate the methodology to activate or de-activate equipment on the MXL/XLS panel as per the SBT-FSI protocol specification. This relates to the opcode Activate/Deactivate Entity 0x7b/0x7c.

The configuration example in **Section 4** can be modified as per the example below for Activating and De-activating Entities on the MXL Panel.

```
// Client Side Connections
//
Connections
Port      , Baud , Parity , Data_Bits , Enhanced_Protocol , Protocol , System_Type
P1        , 19200 , Even , 7 , No , Cerberus , MXL
```

```
Map_Descriptors
Map_Descriptor_Name , Scan_Interval , Data_Array_Name , Data_Array_Offset , Function , Node_Name
POLLER_1 , 2.0s , Counts_1 , 0 , Rdbc , Panel1
// this is for activation and deactivation of pseudo points
PNL_1_Act_Entity , 0s , Act_Ent , 0 , Awt , Panel1
```

```
 , Block_Number , Block_Offset , Length , Msg_Type
 , 1 , 0 , 19 , Poller
 , 1 , 1 , 10 , Entity_Activate_Deact
```

The configuration example in **Section 4** can be modified as per the example below for Activating and De-activating Entities on the XLS panel.

```
// Client Side Connections
//
Connections
Port      , Baud , Parity , Data_Bits , Enhanced_Protocol , Protocol , System_Type
P1        , 19200 , Even , 7 , Yes , Cerberus , XLS
```

```
Map_Descriptors
Map_Descriptor_Name , Scan_Interval , Data_Array_Name , Data_Array_Offset , Function , Node_Name , Block_Number
POLLER_1 , 2.0s , Counts_1 , 0 , Rdbc , Panel1 , 1
// this is for activation and deactivation of pseudo points
PNL_1_Act_Entity , 0s , Act_Ent , 0 , Awt , Panel1 , 1
```

```
 , Sub_Module , Block_Offset , Length , Msg_Type
 , - , 0 , 19 , Poller
// this is for activation and deactivation of pseudo points
 , 1 , 1 , 10 , Entity_Activate_Deact
```

**NOTE: This msg\_type requires Command Access Control to be unlocked. Contact FieldServer technical support to obtain Command Access Control.**

## 7.5 Arming and De-Arming Entities

This example demonstrates the methodology to arm or disarm equipment on the MXL/XLS panel as per the SBT-FSI protocol specification. This relates to the opcodes Disarm/manually control entity – opcode 0x77; Arm entity – opcode 0x78.

The configuration example in **Section 4** can be modified as per the example below for Arming and De-arming Entities on the MXL Panel.

```
// Data Arrays
//
Data_Arrays
Data_Array_Name , Data_Format , Data_Array_Length
Counts_1          , Uint16          , 19
Arm_Disarm_Ent   , Bit             , 10
```

```
// Client Side Connections
//
Connections
Port          , Baud , Parity , Data_Bits , Enhanced_Protocol , Protocol , System_Type
P1            , 19200 , Even , 7          , No                , Cerberus , MXL
```

```
// Client Side Map Descriptors
//
Map_Descriptors
Map_Descriptor_Name , Scan_Interval , Data_Array_Name , Data_Array_Offset , Function , Node_Name , Block_Number , Length , Msg_Type
POLLER_1           , 2.0s          , Counts_1        , 0                  , Rdbc     , Panel1   , 1            , 19          , Poller
```

```
Map_Descriptors
Map_Descriptor_Name Scan_Interval , Data_Array_Name , Data_Array_Offset , Function , Node_Name , Block_Number , Block_Offset , Length , Msg_Type , Entity_Types
PNL_1_Arm_Disarm   , 0s            , Arm_Disarm_Ent , 0          , Awt      , Panel1   , 1            , 1          , 10      , Entity_Arm_Disarm , Inputs
```

```
Map_Descriptors
Map_Descriptor_Name , Scan_Interval , Data_Array_Name , Data_Array_Offset , Function , Node_Name , Block_Number , Block_Offset
PNL_1_Arm_Disarm   , 0s            , Arm_Disarm_Ent , 0                  , Awt     , Panel1   , 1            , 1
, Length , Msg_Type , Entity_Type , Entity_Energization
, 10     , Entity_Arm_Disarm , Outputs , De-energize
```

```
Map_Descriptors
Map_Descriptor_Name , Scan_Interval , Data_Array_Name , Data_Array_Offset , Function , Node_Name , Block_Number , Block_Offset , Length
PNL_1_Arm_Disarm   , 0s            , Arm_Disarm_Ent , 0                  , Awt     , Panel1   , 1            , 1            , 10
, Msg_Type , Entity_Types , Entity_Energization
, Entity_Arm_Disarm , Outputs , Energize
```

**NOTE: This msg\_type requires Command Access Control to be unlocked. Contact FieldServer technical support to obtain Command Access Control.**

The configuration example in **Section 4** can be modified as per the example below for Arming and De-arming Entities on the XLS panel.

```
// Data Arrays
//
Data_Arrays
Data_Array_Name , Data_Format , Data_Array_Length
Counts_1      , Uint16      , 19
Arm_Disarm_Ent , Bit        , 10
```

```
// Client Side Connections
//
Connections
Port      , Baud , Parity , Data_Bits , Enhanced_Protocol , Protocol , System_Type
P1       , 19200 , Even , 7 , Yes , Cerberus , XLS
```

```
// Client Side Map Descriptors
//
Map_Descriptors
Map_Descriptor_Name , Scan_Interval , Data_Array_Name , Data_Array_Offset , Function , Node_Name , Block_Number , Length , Msg_Type
POLLER_1           , 2.0s , Counts_1 , 0 , Rdbc , Panel1 , 1 , 19 , Poller
```

```
Map_Descriptors
Map_Descriptor_Name , Scan_Interval , Data_Array_Name , Data_Array_Offset , Function , Node_Name , Block_Number , Sub_Module , Component_Address
PNL_1_Arm_Disarm , 0s , Arm_Disarm_Ent , 0 , Awt , Panel1 , 1 , 1 , 1
```

```
 , Block_Offset , Length , Msg_Type , Entity_Type
 , 1 , 10 , Entity_Arm_Disarm , Inputs
```

```
Map_Descriptors
Map_Descriptor_Name , Scan_Interval , Data_Array_Name , Data_Array_Offset , Function , Node_Name , Block_Number , Sub_Module , Component_Address
PNL_1_Arm_Disarm , 0s , Arm_Disarm_Ent , 0 , Awt , Panel1 , 1 , 1 , 1
```

```
 , Block_Offset , Length , Msg_Type , Entity_Types , Entity_Energizations
 , 1 , 10 , Entity_Arm_Disarm , Outputs , De-energize
```

```
Map_Descriptors
Map_Descriptor_Name , Scan_Interval , Data_Array_Name , Data_Array_Offset , Function , Node_Name , Block_Number , Sub_Module , Component_Address
PNL_1_Arm_Disarm , 0s , Arm_Disarm_Ent , 0 , Awt , Panel1 , 1 , 1 , 1
```

```
 , Block_Offset , Length , Msg_Type , Entity_Types , Entity_Energizations
 , 1 , 10 , Entity_Arm_Disarm , Outputs , Energize
```

```
Map_Descriptors
Map_Descriptor_Name , Scan_Interval , Data_Array_Name , Data_Array_Offset , Function , Node_Name , Block_Number , Sub_Module , Component_Address
PNL_1_Arm_Disarm , 0s , Arm_Disarm_Ent , 0 , Awt , Panel1 , 1 , 1 , 1
```

```
 , Block_Offset , Length , Msg_Type
 , 1 , 10 , Entity_Arm_Disarm
```

## 7.6 Request ASD or Sensitivity

Data_Arrays			
Data_Array_Name	Data_Format	Data_Array_Length	
Current_ASD	Uint16	60	
Current_Sens	Uint16	60	

Map_Descriptors									
Map_Descriptor_Name	Scan_Interval	Data_Array_Name	Data_Array_Offset	Function	Node_Name	Block_Number	Length	Msg_Type	
PNL_1_ASD	300s	Current_ASD	0	Rdbc	Panel1	1	60	Current_ASD	
PNL_1_Sens	300s	Current_Sens	0	Rdbc	Panel1	2	60	Current_Sensitivity	

In the above example:

- **Block\_Number** – Sets the module number for which this analog device data pertains.
- **Length** – Data values will be stored for device addresses from 0 to Length which can be from 1 to 253.

The values stored are as follows:

	Current_ASD	Current_Sensitivity
0	does not apply	does not apply
1	disabled	2.45%/foot
2	office	2.59%/foot
3	warehouse	2.72%/foot
4	lobby	2.86%/foot
5	computer room	3.00%/foot
6	dormitory	3.13%/foot
7	health care	3.27%/foot
8	garage	
9	utility room	
10	precious storage	
11	hostile environment	

## 7.7 Request % Compensation Used / % Dirty

Data_Arrays			
Data_Array_Name	Data_Format	Data_Array_Length	
Comp_Used	Uint16	253	
Dirty	Uint16	253	

Map_Descriptors									
Map_Descriptor_Name	Scan_Interval	Data_Array_Name	Data_Array_Offset	Function	Node_Name	Block_Number	Length	Msg_Type	
PNL_1_Comp_Used	300s	Comp_Used	0	Rdbc	Panel1	1	253	Compensation_Used	
PNL_1_Dirty	300s	Dirty	0	Rdbc	Panel1	2	253	Dirty	

In the above example:

- **Block\_Number** – Sets the module number for which this analog device data pertains (must be DLC or ALD).
- **Msg\_Type** – The data values stored are % values per device address (e.g. 1, 10 etc.).

## 7.8 Opcode 45

### Request List Map Descriptors that trigger at startup or node offline and stop at node online:

Request List Map Descriptors can be added to ensure the current alarm, trouble, supervision, security or status events that have not been cleared or reset in the remote panel are recorded again after a power-cycle or a re-connection to the panel. Request List Map Descriptors are optional and may be defined per type as needed.

A Passive Map Descriptor of the indicated type needs to be defined and linked to the Request List Map Descriptors to allow storage of these events.

Map_Descriptors								
Map_Descriptor_Name	Scan_Interval	Data_Array_Name	Data_Array_Offset	Function	Node_Name	Block_Number	Length	Msg_Type
PNL_1_Req_List_ALR	,2.0s	,-	,0	,ARS	,Panel1	,-	,1	,req_list_alarm
PNL_1_Req_List_TRB	,2.0s	,-	,0	,ARS	,Panel1	,-	,1	,req_list_trouble
PNL_1_Req_List_SUP	,2.0s	,-	,0	,ARS	,Panel1	,-	,1	,req_list_superv
PNL_1_Req_List_SEC	,2.0s	,-	,0	,ARS	,Panel1	,-	,1	,req_list_secur
PNL_1_Req_List_STA	,2.0s	,-	,0	,ARS	,Panel1	,-	,1	,req_list_status
// for networked panels								
PNL_2_Req_List_ALR	,2.0s	,-	,0	,ARS	,Panel2	,-	,1	,req_list_alarm
PNL_2_Req_List_TRB	,2.0s	,-	,0	,ARS	,Panel2	,-	,1	,req_list_trouble
PNL_2_Req_List_SUP	,2.0s	,-	,0	,ARS	,Panel2	,-	,1	,req_list_superv
PNL_2_Req_List_SEC	,2.0s	,-	,0	,ARS	,Panel2	,-	,1	,req_list_secur
PNL_2_Req_List_STA	,2.0s	,-	,0	,ARS	,Panel2	,-	,1	,req_list_status

## 7.9 Node Status Function

Refer to the FieldServer Configuration manual for information on how to configure Node status. The only status that can be monitored is the direct connection from the FieldServer to the main panel since there is no direct communication with the other networked panels.

## 7.10 XLS Addressing

The XLS panel uses component addresses to identify the Hardware source of event messages. The following table shows the component addresses to be used in the Map Descriptors per XLS device.

Component	Component Address
<b>XLS Device: HFP-11</b>	
Smoke	1
Thermal	2
Neural	3
Relay Output	4
<b>XLS Device: HFPT-11</b>	
Thermal	1
Neural	2
Relay Output	3
<b>XLS Device: HMS</b>	
Switch 1	1
<b>XLS Device: HSM-2S</b>	
Switch 1	1
Switch 2 (keyswitch)	2
<b>XLS Device: HTRI-R</b>	
Switch Input	1
Relay Output	2

Component	Component Address
<b>XLS Device: HTRI-D</b>	
Switch 1	1
Switch 2	2
<b>XLS Device: HTRI-S</b>	
Switch Input	1
<b>XLS Device: HZM</b>	
Zone Input	1
<b>XLS Device: SIM-16 Devices</b>	
Switch Input	1
Relay Output	2
<b>XLS Device: ILED</b>	
Output	1
<b>XLS Device: PSC-12 Relay</b>	
Output/Relay	1

The XLS panel has the ability to have an addressable field device configured as either Fire Alarm, Status, Supervisory or Trouble. A device programmed as something other than TROUBLE, have a component address of "0". A device programmed as a TROUBLE however seem to have a "COMPONENT ADDRESS of "1".

When specifying the length of the Map Descriptor, the component address remains fixed. Specifying block 1, sub module 0, component address 0, length 100, will only obtain data for loop 1, sub module 0, component address 0, address (devices) 0 – 99.

Map_Descriptors				
Map_Descriptor_Name	Data_Array_Name	Data_Array_Offset	Function	Node_Name
PNL_1_Fir_001_0	DA_Fir_001_0	000	Passive_Client	Panel_1
	Block_Number	Sub_Module	Component_Address	Length
	001	000	0	100
				Msg_Type
				Fire

In order to get data for component address 1, specify another map descriptor, block 1, sub module 0, component address 1, length 100.

Map_Descriptors				
Map_Descriptor_Name	Data_Array_Name	Data_Array_Offset	Function	Node_Name
PNL_1_Fir_001_0	DA_Fir_001_0	000	Passive_Client	Panel_1
	Block_Number	Sub_Module	Component_Address	Length
	001	000	1	100
				Msg_Type
				Fire

## 8 Troubleshooting

### 8.1 Checksum Errors

Messages using special characters with ASCII values higher than 127 (e.g. Portuguese ç, á, ó, ã, etc.), provide a bad checksum at 7 data bits. This is resolved if 8 data bits are used.

### 8.2 Recording Active Events – XLS Panel

When there is an active event in the XLS panel, the FieldServer will report this event in the driver log of the FS-GUI regardless of whether there is a Map Descriptor to store the event or not. The report may state the location of the event as follows:

```
1^10^0^26^1
```

This translates to the following: Panel Number ^ Block (Loop) ^ sub-Module ^ Address ^ Component Address

Alternatively, the location of the event could be recorded as:

```
P:L-D
```

This translates to the following: Panel Number: Block (Loop) - Device

### 8.3 Using Trouble Codes

In order to see the trouble codes for incoming troubles it is necessary to use a Uint16 or Int16 data array for the trouble instead of the normal Bit. When the driver stores a trouble code of for example 15 into a Bit data array, it will be displayed as 1 since Bit only holds either a zero or a one value. Using a Uint16 or Int16 data array will ensure the value of 15 is displayed.

The following information is taken from Siemens Fire Safety documentation.

#### MXL/XLS Trouble Codes:

These codes are sent in Field 5 of the Trouble in, Trouble out and Trouble Acknowledge messages 24H, 25H and 26H. Each code is sent as up to 3 ASCII bytes. Leading zeros are not sent. The Request Trouble Type message (43H) can be used to obtain the text strings shown below for each of these codes. The trouble text string is returned in the Trouble Type message (53H).

Trouble Type	Trouble Type Text
0	
1	Multiple Response
2	Communication Error
3	Input Dev. Response too low
4	Device Receive Error
5	Dev Detected Waveform Error
6	Not Responding
7	Not Responding
8	Unspecified Dev. Responding
9	Incorrect ID
10	INPUT DE-ISOLATED
11	INPUT ISOLATED
12	Input Disarmed
13	Addressable Relay disarmed
14	Relay Disarmed
15	Signal Circuit Disarmed

Trouble Type	Trouble Type Text
16	Solid State Output Disarmed
17	Relay Supervision Failure
18	Signal Circuit Loop Open
19	Loop Disarmed
20	Class A Trouble
21	Relay Driver Inoperative
22	Signal Circuit Shorted
23	ALD Reception Error
24	Coil Open
25	Zone Power Off
26	Power Cycling
27	Short Circuit
28	Open Circuit
29	Voltage Low
30	Zone Power Loss
31	Loop Node

Trouble Type	Trouble Type Text
32	Not defined
33	Multiple Response
34	MOI-1 Local Trouble
35	MOI-1 Zone Trouble
36	No database – default in use
37	Too Many ALD-2's
38	PSR-1 Driver Disarmed
39	Output Disarmed
40	Not Responding
41	Unspecified Module Responding
42	Communication Failure
43	RAM Failure
44	ROM Failure
45	A/D Convert Fail
46	Single Module Zone Code
47	Command Not Valid
48	Network A Pair Fail
49	
50	Loss of 24V Input Power
51	Act Power Voltage Low
52	Default Trouble Active
53	Network Disabled
54	Alarm from Isolated Device
55	Printer off-line
56	Warm Reset
57	Printer out of Power
58	Tamper Switch
59	Battery Backup Invalid
60	Positive Ground Fault
61	Negative Ground Fault
62	AC Fail/brownout
63	System Signal Power Failure
64	Battery Fuse/Wiring Open
65	
66	24 Volt Supply Over/Under
67	High Battery Voltage
68	Battery Charger Off
69	24 Volts on Battery Backup
70	Low Battery voltage
71	Watchdog Triggered
72	Device LED Manually Activated
73	Alarm queue full
74	Supervisory queue full
75	Trouble queue full
76	Security queue full
77	Memory error in Ram text
78	Memory error in ROM text
79	Aux Pwr Class A Fail Pos
80	Aux Pwr Class A Fail Neg
81	Supervisory Point
82	Security Point
83	BELL 2 ISOLATED
84	BELL 2 DE-ISOLATED

Trouble Type	Trouble Type Text
85	Configuration Error
86	Addressable Relay Isolated
87	Relay Isolated
88	Single Circuit Isolated
89	Solid State Output Isolated
90	Class A Fault
91	Noise Fault
92	MOI-1 Local Fault
93	MOI-1 Zone Isolated
94	PSR-1 Driver Isolated
95	Fault queue full
96	Isolated Alarm queue full
97	Isolated Alarm point
98	Requires Service
99	MVP General Trouble
100	Valve Tamper queue full
101	Valve Tamper Point
102	WALKTEST Activated
103	Function Recursion Limit Exceeded
104	Trouble Causing Input
105	Carrier Loss
106	
107	Missing Device ID Response
108	Annunciator Trouble
109	CMI-300 general trouble
110	ROM SW version error
111	Lamp Supervision Trouble
112	Keypad Activated
113	Ancillary Cntrl Funcs Isolated
114	Ancillary Cntrl Funcs De-Isolated
115	Pre-dirty Analog volt reached
116	IEC limit reached. Service Req
117	IEC memory error
118	NET-7 General Trouble
119	Network B Pair Failure
120	MOI-1 Voice System Trouble
121	Style 4 Network mode
122	Aux. RS232 Transmit Fail
123	Module Address Buffer Fail
124	Incorrect Enclosure
125	OCC-1 General Trouble
126	OCC-1 Degrade Mode
127	Backup Tone Card (BTC)
128	Daily detector test abnormal
129	Event log 80% full
130	Event log 90% full
131	Log FULL, events will be lost
132	Audio Riser 1 Trouble
133	Audio Riser 2 Trouble
134	Audio Riser 3 Trouble
135	Mic Key Trouble
136	Mic Pre-amp Trouble
137	Tone Generator Trouble



Trouble Type	Trouble Type Text
138	Mic Input Trouble
139	Telephone Riser Trouble
140	Mic Key Trouble
141	Amplifier 1 Trouble
142	Amplifier 2 Trouble
143	Amplifier 3 Trouble
144	Amplifier 1 Backed Up
145	Amplifier 2 Backed Up
146	Amplifier 3 Backed Up
147	Multiple Response
148	Communication Error
149	Not Responding
150	Unspecified Sub-module Responding
151	Incorrect ID
152	Audio Riser 1 Class A Active
153	Audio Riser 2 Class A Active
154	Audio Riser 3 Class A Active
155	Backup Amplifier Trouble
156	Daily Detector Test Abnormal
157	Low IEC voltage. Service Req.
158	Audio Output Shorted
159	Audio Output Class A Active
160	Amplifier 1 Signal Loss
161	Amplifier 2 Signal Loss
162	Amplifier 3 Signal Loss
163	Generic Input Terminal
164	DMU 1 Failure
165	DMU 2 Failure
166	Out of RAM. Call factory
167	Incorrect Node Address
168	Not Responding
169	Unspecified Node Responding
170	Node Disconnected
171	Network CSGM Upload Needed
172	Style 7 Network Mode
173	Network input disarmed
174	Out of task – Call factory
175	Stack Fault – Call factory
176	Software Key Violation
177	Airflow too low
178	Airflow too high
179	Detector Trouble
180	AnaLASER General Trouble
181	Pre1 level
182	Pre2 level
183	Incorrect library version
184	Out of calibration

Trouble Type	Trouble Type Text
185	AnaLASER Power Trouble
186	Pre-alarm
187	+15volt fault
188	Strobe power fault
189	Audio fault iso-St 7Tx
190	Audio fault iso-St 7Rx
191	Audio fault iso-St 4Tx
192	Audio fault iso-St 4Rx
193	Multiple Response
194	Global audio riser-St 7
195	Global audio riser-St 4
196	Local audio riser
197	Database Out of Date
298	ASD disabled for test
199	Sys halt – too many ALD/ACM
200	DMC-1 in record mode
201	DMC-1 tone trouble
202	DMC-1 phrase trouble
203	Troubles summarized
204	
205	Incorrect custom phrase control
206	Remote node disconnected
207	History manually turned off
208	History manually turned on
209	History manually erased
210	Memory upgrade required
211	Event queue corrupt
212	Current sensor failure
213	EEProm error
214	No application firmware
215	Component disarmed
216	Zone 1 open
217	Zone 2 open
218	Zone 1 short
219	Zone 2 short
220	Zone 1 Class A return fault
221	Zone 2 Class A return fault
222	Input source malfunction
223	Over current fault
224	Log failure, events lost
225	24V 4amp supply shutdown
226	24V 12amp supply shutdown
227	Backplane power shutdown
228	No battery detected
229	Thermistor Trouble
230	Configuration process failure
231	Device(s) with no address

## 8.4 Setting the Panel Language to French

If the panel is set to the French language, the Communication parameters 19200, N81 need to be applied to the RPM Module before it will communicate to the FieldServer.

Property	Value
Type	Serial Port
Device Address	1
Base Language Cust. Message	Serial Port @ address 2-1
Alternate Language Cust. Message	
Serial Device Type	FSI
Printer Type	
Supervised	<input checked="" type="checkbox"/>
Serial Port Baud Rate	19200
Asynchronous Comm. Setting	N81
Print Level	Device/Primitive
Global Printer	<input type="checkbox"/>
Print Alarm Events	<input type="checkbox"/>
Print Supervisory Events	<input type="checkbox"/>
Print Security Events	<input type="checkbox"/>
Print Trouble Events	<input type="checkbox"/>
Print Status Events	<input type="checkbox"/>
Reset Before Printing Complete	<input type="checkbox"/>
Language Setting	
FSI Scope	FSI Global
FSI Non-Network System ID	1
FSI Enhanced Protocol	<input checked="" type="checkbox"/>
FSI List	<input checked="" type="checkbox"/>
FSI Acknowledge Alarm	<input checked="" type="checkbox"/>
FSI Acknowledge Supervisory	<input checked="" type="checkbox"/>
FSI Acknowledge Trouble	<input checked="" type="checkbox"/>

```
// Client Side Connections
//
Connections
Port , Protocol , Enhanced_Protocol , System_Type , Baud , Parity , Data_Bits , Stop_Bits
P1 , SBT_FSI , Yes , XLS , 19200 , None , 8 , 1

, Poll_Delay , Language_Pack
, 0.001s , Yes
```