Driver Manual

FS-8700-65 Silent Knight

APPLICABILITY & EFFECTIVITY

Effective for all systems manufactured after March 2021.

Contents

1   Description

The Silent Knight Gateway Command (SKgwcmd) Driver allows the FieldServer to transfer data to and from devices over either RS-232 or RS-485 using Silent Knight Gateway Command (SKgwcmd) Driver protocol. The Silent Knight Gateway Command (SKgwcmd) Driver is based on the protocol specification "Gateway Command Set Specification 5820 Intelliknight System rev D.2 dated 11 July 2000"

The Silent Knight Gateway Command driver can serve as an active Client as well as emulate a Silent Knight 5820 device in Server mode.

The Silent Knight unit's serial port settings are configurable. The FieldServer device can be configured to match all Silent Knight Port settings.

2   Driver Scope of Supply

2.1   Supplied by MSA Safety

Part #	Description
FS-8917-15	RJ45 to DB9F connector adapter
FS-8917-16	UTP cable (7 foot)
	Driver Manual

3   Hardware Connections

The FieldServer is connected to the Silent Knight Gateway Command as shown below. Configure the Silent Knight Gateway Command Driver according to manufacturer's instructions.
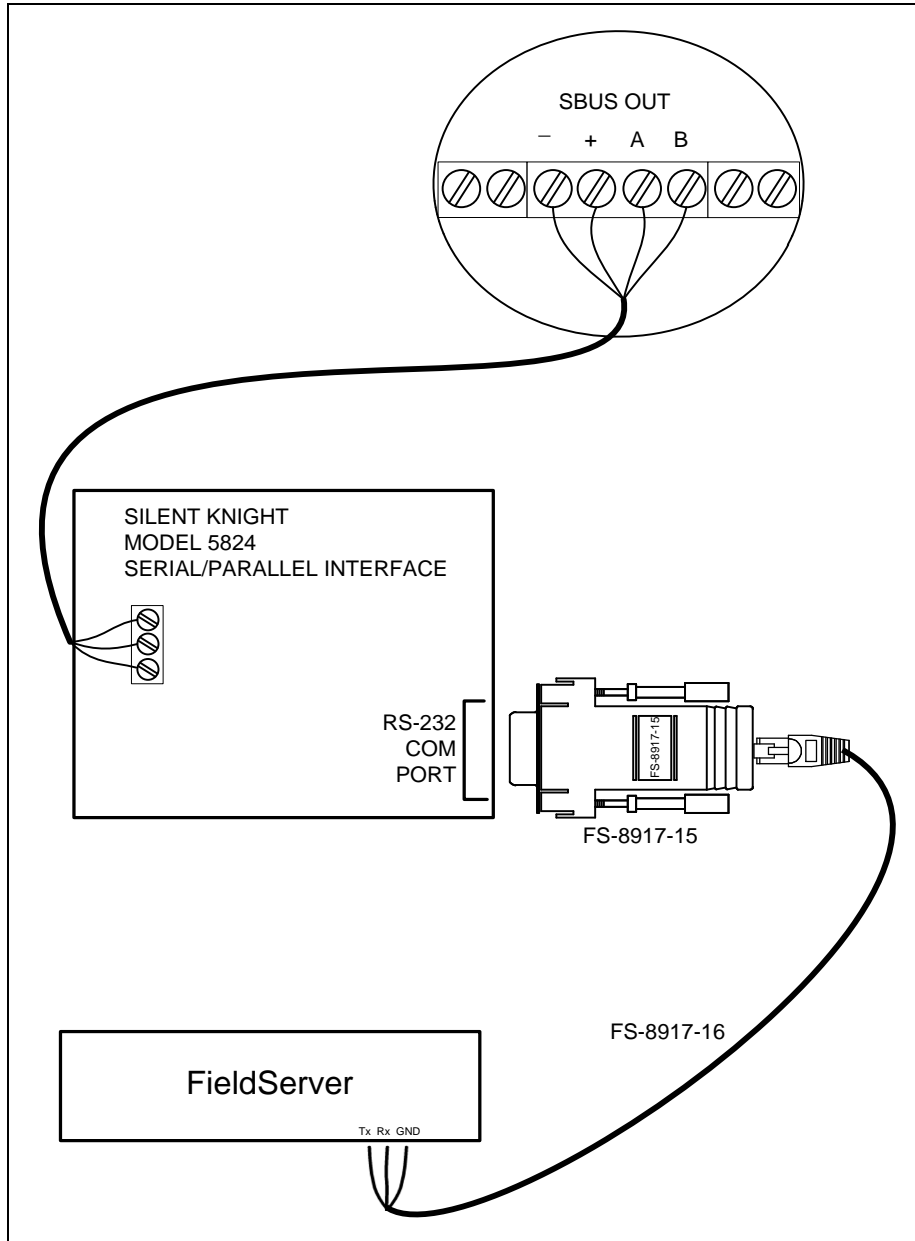
Pinouts		
Function	**Color**	**RJ45 #**
Rx/-	Brown	1
Tx/+	Orange/White	8
GND	Blue/White	4

3.1   Connection Notes

To enable communications, the 5824 module needs to be set as a parallel connection in the Silent Knight panel, despite being connected to the serial port on the FieldServer.

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

Example

//   Data Arrays
Data_Arrays
Data_Array_Name   , Data_Array_Format   , Data_Array_Length
DA_AI_01   , UInt16,   , 200
DA_AO_01   , UInt16   , 200
DA_DI_01   , Bit   , 200
DA_DO_01   , Bit   , 200

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 a Silent Knight Gateway Command (SKgwcmd) Server.

The configuration file tells the FieldServer about its interfaces, and the routing of data required. In order to enable the FieldServer for Silent Knight Gateway Command (SKgwcmd) Driver 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 following tables, * indicates an optional parameter and bold legal values are 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
Baud*	Specify baud rate.	Standard Baud Rates up to 19200 (SK Device Limitation), **19200**
Parity*	Specify parity.	Even, Odd, **None**, (Device Limitation)
Data_Bits*	Specify data bits.	**7**, 8
Stop_Bits*	Specify stop bits.	**1**,2
Protocol	Specify protocol used.	SKgwcmd
Poll Delay*	Time between internal polls.	0-32000 seconds, **1s**

Example

```
//   Client Side Connections
Connections
Port                        , Baud      , Parity     , Protocol
R1                          , 19200     , None       , SKgwcmd
```

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
Protocol	Specify protocol used.	SKgwcmd
Port	Specify FieldServer port the device is connected to.	P1-P2, R1-R2

Example

```
//   Client Side Nodes
Nodes
Node_Name      , Protocol     , Port
SK-Device1     , SKgwcmd      , R1
```

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 **Section 4**
Data_Array_Offset	Starting location in Data Array.	0 to (Data_Array_Length -1) as specified in **Section 4**
Function	Function of Client Map Descriptor.	Rdbc, Wrbc, Wrbx

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 **Section 5.2**
Length*	Length of Map Descriptor.	**1** – 1000. The length must be sufficient to store all data polled to prevent data being discarded by the driver. Certain commands, e.g. description queries use 20 array elements to store the response. Thus, the length must be at least 20. Refer to **Section 8.2**.
Sk_Function	This mandatory field tells the driver what query or command is required.	Refer to **Section 8.1**
Da_Byte_Name*	Used when Sk_Function is one of the following: *Output Group Status* *Point Status* *Point Description* Refer to **Sections 5.4.3** and **5.4.5.**	One of the Data Array names from **Section 4**
DA_Bit_Name*	Only used when Sk_Function is *Point Description.* See **Section 5.4.3**	One of the Data Array names from **Section 4**
Sk_Zone*	This parameter is required for *General Zone Status* *Expanded Zone Status* *Zone Description* Specify the number of the zone being queried.	Whole numbers limited to the range of addressable zones configured in the Silent Knight device
Sk_Group*	This parameter is required for the following commands. *Output Group Status* *Output Group Description*	Whole numbers limited to the range of addressable zones configured in the Silent Knight device

Sk_Module*	This parameter is always used in conjunction with the Sk_Point parameter to uniquely define the address of a point to be queried, enabled or disabled. Refer to **Sections 5.4.3** and **5.4.5**.	0, 1 ,2 … Whole numbers limited to the addressable expander Module ID's
Sk_Point*	When querying a specific point both the sk_module and sk_point parameters must be defined.	0 -199 are sensors (e.g. S010 is SK_Point 10). 200-399 are modules (e.g. M010 is Sk_Point 210). Whole numbers limited to the range of addressable points in the Silent Knight Unit.

5.3.3   Timing Parameters

Column Title	Function	Legal Values
Scan_Interval	Rate at which data is polled	>0.1s

5.4 Map Descriptor Examples

5.4.1 General System Status Query

The Map Descriptor causes the driver to poll the Silent Knight Device every 1.0 seconds, read the general system status and store the result in an array called UNIT_STATUS.

Map_Descriptor_Name	Data_Array_Name	Data_Array_Offset	Function	Node_Name	Length	Scan_Interval	Sk_Function
GEN_STATUS	UNIT_STATUS	3	Rdbc	SK-Dev3	1	1.0s	General System Status

In the above example:

- Map_Descriptor_Name – Map Descriptor names need not be unique but since some driver error messages use the name it is helpful to have meaningful names.

- Data_Array_Name – This is the name of the array which the driver will use to store the status information returned by the Silent Knight Unit. Ensure that the array is long enough and that the data type is meaningful with respect to the type of data being stored. This function obtains one 16 bit element of status data; therefore the array should be a UINT16 or similar data type.

- Data_Array_Offset – In this example, the offset into the Data Array is set to 3 to correspond to the device (Node_Name) being polled using this Map Descriptor.

- Function – Read Continuously.

- Node_Name – The Node name must connect this Map Descriptor to a previously defined Node which in turn must connect to a previously defined port. That is how the driver knows which port to use in processing this Map Descriptor.

- Length – Only one element of data is being polled and only one element of data will be stored. Thus, the length is set to 1.

The bits provided in the response will exactly mimic the state of the five indicator LED's on the IntelliKnight front panel.

Bit	Meaning	Bit	Meaning
0	Alarm condition present	8	Not Used
1	Supervisory condition present	9	Not Used
2	Trouble condition present	10	Not Used
3	System silenced	11	Not Used
4	AC Power present	12	Not Used
5	Not Used	13	Not Used
6	Not Used	14	Not Used
7	Not Used	15	Not Used

5.4.2   Alarm Silence/Reset Commands

This example illustrates the use of the silence command. The alarm reset command is used in an identical fashion. The Driver watches the value of the specified Data_Array (Wrbx function). When the value changes, the command is triggered. Typically, users configure the FieldServer to connect a MMI or field push button on a remote device to set the array element to 1 to trigger the silence command and then set it back to zero.

```
//   Client Side Map Descriptors

Map Descriptors
Map_Descriptor_Name   , Data_Array_Name  , Data_Array_Offset  , Function  , Node_Name  , Length  , Sk_Function
SILENCE-CMD           , CMD_TRIGGERS     , 0                  , Wrbx      , SK-DEV1     , 2       , Silence
```

In the above example:

- Map_Descriptor_Name – Have some remote device set the value of this Data Array to trigger this command.

- Data_Array_Name – Write on change. When the value of the 1st element of the array called CMD_TRIGGERS changes then the command will be triggered.

- Length – Use a length of 2 as a later release of the driver may write the return code from the SK device to the second element of the Data Array.

- Sk_Function – The Alarm Silence command will be sent to the SK device.

5.4.3   Point Enable/Disable

In this example point 10 of module 1 is disabled. The command is executed each time the value of the array DA01 element index 10 (11th element) changes. To cause this command to be executed continuously, the function should be changed to Wrbc.

```
Map_Descriptor_Name  , Data_Array_Name  , Data_Array_Offset  , Function  , Node_Name  , Length  , Scan_Interval  , Sk_Function    , Sk_Module  , Sk_Point
ENABLE1              , DA01             , 10                 , Wrbx      , SK-DEV1     , 1       , 1.0s           , Point Disable  , 1          , 10
```

In the above example:

- Data_Array_Offset– This is a triggered command. The Data Array offset = 10. Thus, the element of the Data Array indexed by 10 (11th element) will trigger the command when it changes.

- Sk_Function – Change this to *Point Enable* to enable a point. The remainder of the Map Descriptor remains the same.

- Sk_Module – Sk_Module and Sk_Point must be non-zero. Only one point can be enabled or disabled per Map Descriptor.

5.4.4 Point Status

NOTE: The Driver does not support polls for multiple points. A separate Map Descriptor needs to be defined to return point status for each point.

```
Map Descriptors
Map_Descriptor_Name  , Data_Array_Name  , Data_Array_Offset  , Function  , Node_Name  , Length  , Scan_Interval  , Sk_Function  , Sk_Module  , Sk_Point  , Da_Byte_Name
POINT_STAT1          , POINT_STATUS     , 0                  , Rdbc      , Node_A     , 1       , 10.0s          , Point Status , 33        , 1         , -
POINT_STAT2          , POINT_STATUS     , 1                  , Rdbc      , Node_A     , 1       , 10.0s          , Point Status , 33        , 2         , EXTRA_DATA1
```

In the above example:

- Data_Array_Name – The Status data is stored in these arrays. The offset is based on the point number and the offset defined in the Map Descriptor.

- Data_Array_Offset– A separate Map Descriptor is required to return point status for each point.

- Sk_Module – Set sk_module & sk_point to non-zero values and length to 1.

- Da_Byte_Name – The accessory data associated with each point is stored in the corresponding location of these Data Arrays. If this parameter is not defined, the accessory data is discarded.

<u>Point Status Values</u>

Value	Meaning
0x00	Normal
0x01	Alarm
0x02	Pre-Alarm
0x03	Supervisory
0x04	Trouble
0x05	Output Active
0x06	Output Inactive
0x07	Output Disconnected
0x8000	Unused
0x8001	Undefined

<u>Accessory Status Values</u>

Value	Meaning
0x00	Accessory Normal
0x01	Accessory Trouble
0x8000	Accessory Unused

5.4.5 Point Descriptions

Requests the text description & type designator of any individual point in the system – the panel returns the point description record for the point specified. One Map Descriptor is required for each point description requested.

When a point description is polled the returned data contains the point type, the accessory type and the description. The driver loads the description into 40 consecutive elements of the Map Descriptor's Data Array. The point type data is only stored if the parameter *Da_Byte_Name* is specified by providing the name of another Data Array in which the point type data should be stored. The accessory type data is only stored if the parameter *DA_Bit_Name* is specified by providing the name of another Data Array in which the accessory type data should be stored.

The Map Descriptor below illustrates a query for a point description. Each query returns three data components – a point type, accessory type and a point description. Only one point can be queried per Map Descriptor.

```
//   Client Side Map Descriptors

Map Descriptors
Map_Descriptor_Name  , Data_Array_Name  , Data_Array_Offset  , Function  , Node_Name  , Length  , Scan_Interval  , Sk_Function
PNT_DESC1            , DESC_ARRAY       , 0                  , Rdbc      , SK-DEV1    , 20      , 1s             , Point Description
```

```
, Sk_Module  , Sk_Point  , Da_Byte_Name  , DA_Bit_Name
, 1          , 5         , TYPE_ARR      , ACC_ARR
```

In the above example:

- Function – The point's description will be placed in 20 consecutive elements of this Data Array starting at this offset position.

- Length – 20 elements of the Data Array are controlled by this Map Descriptor.

- Sk_Function – Function name- spelling and spacing are important.

- Da_Byte_Name – When using the *Point Description* function a single non-zero value must be specified and the Sk_Module parameter must be set to a non-zero value. This is because this function only queries one specific point per Map Descriptor.

- Da_Bit_Name – The additional data components are stored in these Data Arrays. The point type is stored as a number at the offset specified by Data_Array_Offset in the array called TYPE_ARR. The accessory type is stored at the same offset in the array called ACC_ARR.

6   Server Side Configuration

The Silent Knight Gateway Command (SKgwcmd) Driver can act as a Server. As the protocol does not support device or node identification it is not possible for this driver to handle more than one Client per FieldServer unless the driver is specifically tied to a port by using the port keyword when defining the Server side node.

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 a Silent Knight Gateway Command (SKgwcmd) Driver Client.

The configuration file tells the FieldServer about its interfaces, and the routing of data required. In order to enable the FieldServer for Silent Knight Gateway Command (SKgwcmd) Driver 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 must 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[1]
Baud*	Specify baud rate.	110 – 115200, standard baud rates only
Parity*	Specify parity.	Even, Odd, **None,** Mark, Space
Data_Bits*	Specify data bits.	7, **8**
Stop_Bits*	Specify stop bits.	**1**
Protocol	Specify protocol used.	SKgwcmd

Example

```
//   Server Side Connections
Connections
Port             , Baud    , Protocol
R1               , 19200   , SKgwcmd
```

---

[1] Not all ports shown are necessarily supported by the hardware. Consult the appropriate Instruction manual for details of the ports available on specific hardware.

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
Protocol	Specify protocol used.	SKgwcmd

Example 1

```
//   Server Side Nodes


Nodes
Node_Name                , Protocol
FieldServer              , SKgwcmd
```

Example 2 – Use this example when the FieldServer is being used to emulate more than one Silent Knight Device

```
//   Server Side Nodes


Nodes
Node_Name                , Protocol      , Port
Sk-emulation1            , SKgwcmd     , R1
Sk-emulation2            , SKgwcmd     , R2
```

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[2]
Data_Array_Name	Name of Data Array where data is to be stored in the FieldServer.	One of the Data Array names from **Section 4**
Data_Array_Offset	Starting location in Data Array.	0 to (Data_Array_Length-1) as specified in **Section 4**
Function	Function of Server Map Descriptor.	Passive

---

[2] Object_Name values of any length may be written via BACnet (subject only to memory and message length constraints).

6.3.2 Driver Specific Map Descriptor Parameters

Column Title	Function	Legal Values
Node_Name	Name of Node to fetch data from.	One of the Node names specified in **Section 6.2**
Length	Length of Map Descriptor – The length parameter's meaning is dependent on the value of the Sk_Function parameter. Refer to **Section 8.2**.	1 - 1000
Da_Byte_Name*	Only used when Sk_Function is one of the following: *Output Group Status* *Point Status* *Point Description*	One of the Data Array names from **Section 4**
DA_Bit_Name*	Only used when Sk_Function is Point Description.	One of the Data Array names from **Section 4**
Sk_Function	This mandatory field tells the driver what query or command is required.	Refer to **Section 8.1**
Sk_Zone*	Use one Map Descriptor for each zone to be emulated.	1, 2, 3 ….
Sk_Group*		Positive Integers
Sk_Module*	These parameters are linked as points belong to modules. Thus, the point and module define each point uniquely.	1, 2, 3 ….
Sk_Point*		Positive Integers

6.3.3 Timing Parameters

Column Title	Function	Legal Values
Scada_Hold_Timeout	Specifies time Server side waits before responding to Client that Node is offline on FieldServer Client side.	>1.0s

6.4　Map Descriptor Example – FieldServer Responds to Point Status Requests

This example illustrates a number of Map Descriptors used to allow the FieldServer to emulate 3 modules of 10 points each. These Map Descriptors can only respond to **Point Status** requests. Other Map Descriptors are required to allow responses to other queries like **point description** queries.

Map_Descriptor_Name	, Data_Array_Name	, Data_Array_Offset	, Function	, Node_Name	, Length	, Sk_Function	, Sk_Module	, Sk_Point	, Da_Byte_Name
SERVER1	, PNT_STAT	, 0	, Passive	, Node_A	, 10	, Point Status	, 1	, 1	, PNT_ACC_STATUS
SERVER2	, PNT_STAT	, 10	, Passive	, Node_A	, 20	, Point Status	, 2	, 1	, PNT_ACC_STATUS
SERVER3	, PNT_STAT	, 30	, Passive	, Node_A	, 10	, Point Status	, 3	, 1	, PNT_ACC_STATUS

In the above example:

- Data_Array_Name – Status data is served from the Data Array called PNT_STAT.

- Data_Array_Offset – The offset increases by the length of each Map Descriptor. Thus Module 2's data will be served from elements 10-19 of the array.

- Length – The length is very important. If a request arrives for all point data then the driver will report status data for:

  module 1 points 1-10

  module 2 points 1-20

  module 3 points 1-30

  If a request arrived for the status of module 2 point 25 the driver would not respond as a Map Descriptor defining this point has not been defined.

- Sk_Function – The Point Status request returns two data components per point. The first is the status, a 16 bit value. The second is the accessory status also a 16 bit value. Unless you define the da_byte_name parameter the driver will return zero for this second data component. By defining an array where the driver can get this data the driver looks in the corresponding element of this array (PNT_ACC_STATUS in this example) and the PNT_STATUS array to return the status of the point requested.

7 Troubleshooting

7.1 Driver Error Messages

Message	Explanation
SKgwcmd: #1 Error. Don't know this function=%d %x(h)	The value of the Sk_Function has been set incorrectly. The driver does not recognize it.[3]
SKgwcmd: #3. Error. Too much data to send=%d max=%d.	The Silent Knight Gateway Command protocol can send a maximum of 7*255 bytes of data per message. Try breaking one Map Descriptor up so that less data is requested at one time. It may not be possible to set Sk_Module, Sk_Point, Sk_Zone or Sk_Group equal to zero as this requires too much data to be sent.
SKgwcmd: #4. Error. Cannot do Exp. Zone Stat. Mapdesc=<%s>	The command code is defined but the command and response format is not defined in the protocol specification. Change the Sk_Function parameter to another suitable command. [3]
SKgwcmd: #5 FYI. The mapDesc called <skgwcmd-stats> is too short.	Increase the value of the length parameter to at least 100.[3] Read **Section 7.2** for additional information.
SKgwcmd: #6 Error. Dont know this function=%d %x(h)	See notes for error 1.
SKgwcmd: #7. Error. Cannot do Exp. Zone Stat.	The command code is defined but the command & response format is not defined in the protocol specification.
SKgwcmd: #8 Error Incoming point status data from module=%d is being abandoned. MapDescriptors are required to define storage for. No further messages about data being abandoned will be provided.	If a Map Descriptor was defined which generated a poll for all point data then a Map Descriptor is required to specify where the point status data for each module must be stored. If this has not been done then this error message will be produced. [3]
SKgwcmd:#9 Error. Don't know this function=%d %x(h)	See notes for error 1.
SKgwcmd:#11* FYI. MapDesc Length too short rqd/actual=%d/%d MapDesc=<%s> Some incoming data will be abandoned	If a Map Descriptor generates a poll for all module points, zone or group data and the SK device returns more data than provided for by the length parameter of the Map Descriptor then this error will be generated. [3]
SKgwcmd:#12* FYI. Point Accessory Status Data abandoned. Define field=da_byte_name in mapDesc=<%s>	To eliminate this error, link the Map Descriptor to two Data Arrays, one for status data and the other for accessory data. See **Section 5.4.5**.[3]
SKgwcmd:#13* FYI. Data Array too short rqd/actual=%d/%d MapDesc=<%s> Some incoming data will be abandoned.	This error is similar to #11 but in this case the Data Array itself is too short. [3]

---

[3] Edit the CSV file, download it to the FieldServer and reset the FieldServer for the changes to take effect.
* Subsequent similar messages will be suppressed.

Message	Explanation
SKgwcmd:#14* FYI. Point Accessory Status Data abandoned. Define field=da_byte_name in mapDesc=<%s>	See error #12.
SKgwcmd:#15* FYI. Port Checking Disabled ! If there is more than one SK device connected to this FieldServer this may produce errors.	If the FieldServer is to be connected to more than one Silent Knight device, the node definitions in the CSV file need to link to a port definition. Although normally not required for a Server side connection, this is not invalid.
SKgwcmd:#16 FYI. Resetting/Silencing Array too short. <%s>	The value of the length parameter and the length of the Data Array conflict. [4]
Skgwcmd:#17 Error. Incoming data is being abandoned. No further messages about data being abandoned will be provided.	Same error as #8 except that the driver produces this error in a different context.
Skgwcmd:#18 Error. Requested too much data from mapDesc=<%s>rqstd/avail: start=%d/%d. len=%d/%d.	More Zone Status data is contained in the response than the Map Descriptors length or starting zone number allows the driver to process. This error is produced by the Client driver.[4]
SKgwcmd:#19 Error. Incoming zone status data being abandoned. MapDesc's are required to define storage. No further messages about data being abandoned will be provided.	More Zone Status data is contained in the response than the Map Descriptors length or starting zone number allows the driver to process. This error is produced by the Server driver.[4]
SKgwcmd:#20 FYI*. Data Array too short rqd/actual=%d/%d MapDesc=<%s> Some incoming data will be abandoned.	This message is produced by the Server driver. It arises while processing zone status data. The message indicates that some data is being stored correctly but that 'surplus' data is being abandoned. Correct the length or starting zone number in the CSV file. [4]
SKgwcmd:#21* FYI. Data Array too short rqd/actual=%d/%d MapDesc=<%s> Some incoming data will be abandoned.	This message is produced by the Server driver. It arises while processing group status data. The message indicates that some data is being stored correctly but that 'surplus' data is being abandoned. Correct the length or starting zone number in the CSV file. [4]
Skgwcmd:#22 Error. Incoming group status data being abandoned. MapDesc's are required to define storage. No further messages about data being abandoned will be provided.	This message is produced by the Server driver when the Sk_Group is set to zero and the Map Descriptor length is too small to store the incoming group data. Add a passive Map Descriptor to define the storage of the incoming data. [4]

---

[4] Edit the CSV file, download it to the FieldServer and reset the FieldServer for the changes to take effect.

Message	Explanation
SKgwcmd:#23* FYI. Additional data was discarded. You could have defined storage using the <da_byte_name> parameter in MapDesc=<%s>.	Group Status data consists of the group status and the number of troubles. The number of troubles will be discarded unless the Map Descriptor is connected to a second Data Array using the Da_Byte_Name parameter. This message is a warning only - no action is necessarily required.
Skgwcmd:#24* Error. Requested too much data from mapDesc=<%s> rqstd/avail: start=%d/%d. len=%d/%d.	This message is produced by the Client driver. It arises while processing group status data. The message indicates that some data is being stored correctly but that 'surplus' data is being abandoned. Correct the length or starting zone number.[4]
Skgwcmd:#25 FYI. You could have used a MapDesc called <skgwcmd-stats> to expose diagnostic info.	This is a warning message only. No action is required. Read **Section 7.2** for additional information.
DRV->SKgwcmd : #29 Err. No write for Func=%d %#x. MD=%s	Write is not possible for specified function. Also see Err #30
DRV->SKgwcmd : #30 Err. MD could not be created.	This is further information for #29. Avoid Wrbc or Wrb Map Descriptors in the configuration for the function specified in #29. If this error occurred during operation, avoid write thru's on the specified Map Descriptor.

7.2   Driver Stats

In addition to the standard FieldServer communication statistics described in the FieldServer Configuration Manual the DNP 3 Driver can also expose some driver statistics by writing data to a Data Array. A special Map Descriptor is required. The driver recognizes the Map Descriptor by its name which must be "**Skgwcmd-stats**". The following example shows how this special Map Descriptor can be configured.

Example

```
Nodes
Node_Name               , Protocol
Null_Node               , SKgwcmd

Data Arrays
Data_Array_Name         , Data_Format        , Data_Array_Length
SK_STATS                , UINT32             , 300

Map Descriptors
Map_Descriptor_Name     , Data_Array_Name    , Node_Name           , Length
Skgwcmd-stats           , SK_STATS           , Null_Node           , 300
```

The driver uses the Data Array SK_STATS (in this example) to store driver specific statistics. Only one of these Map Descriptors may be specified per FieldServer.

The driver stores the following data.

Array Element	Contents
0	Not Used
1	SK_STAT_COMPL_HDR1
2	SK_STAT_COMPL_HDR2
3	SK_STAT_COMPL_CMD
4	SK_STAT_COMPL_BLK
5	SK_STAT_COMPL_CRC
6	SK_STAT_COMPL_CRC1
7	SK_STAT_COMPL_CRC2
8	SK_STAT_COMPL_BLK2
9	SK_STAT_MSTR_BAD
10	SK_STAT_MSTR_GOOD
11	SK_STAT_COMPL_SLV
12	SK_STAT_COMPL_GWAY
12	SK_STAT_SEND_MST_POLL
14	SK_STAT_SEND_SLV_IGNORE
15	SK_STAT_MST_IGNORE_GROUP
16	SK_STAT_MST_IGNORE_ZONE
17	SK_STAT_MST_IGNORE_POINT
18	SK_STAT_SEND_SLV_MSG
19	SK_STAT_SEND_SLV_MSG_BYTES
20	SK_STAT_SEND_SLV_CONCT_MSG
21	SK_STAT_SEND_SLV_RCVD_MSG
22	SK_STAT_SEND_SLV_RCVD_BYTES

8   Reference

8.1   Protocol Commands

The following list of commands has been implemented in the Silent Knight Gateway Command (SKgwcmd) Driver. The table contains the keywords that may be used in the .CSV files to specify the Sk_Function parameter.

	Commands Implemented		Commands Implemented
1	General System Status	9	Point Description
2	System Trouble Status	10	Time Status
3	General Zone Status	11	gateway Version
4	Expanded Zone Status	12	Disable Point
5	Zone Description	13	Enable Point
6	Output Group Status	14	Reset Alarms
7	Output Group Description	15	Silence
8	Point Status	16	Time Synch

Most of the commands consist of more than one word. Ensure that there is only a single space between the words and that the space is not a tab character. Leading and trailing spaces are not important but the highest level of user confidence can be achieved by delimiting the Sk_Function with a comma on either side. Spelling is important but the driver is case-insensitive.

8.2   Length Parameter Settings Based on Sk_Function Parameter

Sk_function command	Length Parameter Meaning
Silence *Reset Alarms*	The length parameter tells the driver how much of the Data Array to reset to zero when one of these commands is received
Output Group Status *General Zone Status*	The length parameter tells the driver how many zones or groups to return data for (if all zones or groups are requested) when one of these commands is received.
Point Status	The length tells the driver how many points to report if the poll requests data for all points.

For all other values of Sk_Function the length should be set to 1 as the driver does not use the parameter.

8.3   Legal Values for Sk_Function

Command	Description
General System Status	Requests the status of the entire system - the bits provided in the response will exactly mimic the state of the five indicator LED's on the IntelliKnight front panel. Refer to **Section 5.4**.
System Trouble Status	Monitors the system for trouble or service requirements - the panel responds with the system trouble status record. Refer to **Section 8.4**.
General Zone Status	Requests the status of one or more zones in the system - the panel responds with a zone status byte for each of the zone(s) specified. Refer to **Section 8.5**. Set zone number to zero and length to 1 to request all possible zones
Expanded Zone Status	Set zone number to a specific zone and length to 1 to read status data from one zone. Set zone number to a specific zone and length greater than one to read consecutive zones starting at the zone number specified.
Zone Description	Requests the text description & type designator for any individual zone in the system - the panel returns the zone description record for the zone specified. Only one specific zone may be queried. 40 consecutive array elements are loaded with the description returned by the device.
Output Group Status	Requests information about the state of one or more output groups present in the system. Returns the active/inactive state and the number of troubles for each group. Set group number to zero and length to 1 to request all possible groups. Set group number to a specific group and length to 1 to read status data from one group. Set group number to a specific group and the length greater than one to read consecutive groups starting at the group number specified. <table><tr><th>Value</th><th>Meaning</th></tr><tr><td>1</td><td>Group is Active</td></tr><tr><td>0</td><td>Group is Inactive</td></tr><tr><td>8000h</td><td>Group is not defined.</td></tr></table> When an output group status is requested the SK device returns the status and the number of troubles in the group. The 'number of troubles' is discarded by this driver unless a second Data Array is provided to store this quantity. Specify the second Data Array by putting the array name as the value of a field called Da_Byte_Name in the Map Descriptor. (See the usage for point status illustrated in **Section 5.4.4** and **6.4**).
Output Group Description	Requests the text description & type designator for any individual output group in the system - the panel returns the output group description record for the output group specified. See **Section 5.4.5** for additional information.
Point Status	Requests the state of one or all points in the system - the panel returns the point status for the point(s) specified. This enables the interfacing computer to select for appropriate data from the large number of points possible in the IntelliKnight system
Point Description	Requests the text description & type designator of any individual point in the system - the panel returns the point description record for the point specified.

Time Status	The time and date are stored in 6 consecutive element of the Data Array specified.		
	Element	**Contents**	
	1	Second	
	2	Minute	
	3	Hour	
	4	Day	
	5	Month	
	6	Year - 1900	
Gateway Version	Requests the gateway version number string - the panel responds with gateway version in a null terminated string. This command can be used to determine the command set supported by the connected panel. The driver writes the version character by character to consecutive elements of the Data Array specified. The version is variable length so ensure that the Map Descriptor length is sufficient to cover all contingencies.		
Disable Point	A specific point must be specified by setting Sk_Module and Sk_Point to non-zero values.		
Enable Point	A specific point must be specified by setting Sk_Module and Sk_Point to non-zero values.		
Reset Alarms Silence	Used to reset or silence any alarms occurring in the system - the panel returns a non-zero error code if the command could not be completed.		
Time Synch	Used to set the time and date at the panel - the driver uses 6 consecutive elements from the Data Array specified.		
	Element	**Contents**	
	1	Second	
	2	Minute	
	3	Hour	
	4	Day	
	5	Month	
	6	Year - 1900	

8.4 System Trouble Status Bitmap

Bit	Meaning	Bit	Meaning
0	AC Trouble	8	Printer Trouble
1	Battery Trouble	9	Auxiliary Power Trouble
2	Earth Ground Fault Trouble	10	System Aux 1 Switch Trouble
3	unused	11	System Aux 2 Switch Trouble
4	Phone Line 1 Trouble	12	System Switch Trouble
5	Phone Line 2 Trouble	13	Undefined SLC Point Trouble
6	IO Device Trouble	14	SBUS Class A Trouble
7	Account Trouble	15	unused

8.5 Zone Status Bitmap

Bit	Meaning	Bit	Meaning
0	Alarm condition present in the zone	8	Not Used
1	Supervisory condition present in the zone	9	Not Used
2	Trouble condition present in the zone	10	Not Used
3	Not Used	11	Not Used
4	Not Used	12	Not Used
5	Not Used	13	Not Used
6	Not Used	14	Not Used
7	Not Used	15	Zone Not Defined