A Sierra Monitor Company

Driver Manual
(Supplement to the FieldServer Instruction Manual)

FS-8700-58 ILCO 9000

APPLICABILITY & EFFECTIVITY

Effective for all systems manufactured after May 1, 2001

Driver Version:	**1.02**
Document Revision:	**1**

TABLE OF CONTENTS

1. Ilco-9000 Description

The Ilco-9000 driver allows the FieldServer to transfer data to and from devices over either RS-232 or RS-485 using Ilco-9000 protocol. The FieldServer can emulate either a Server or Client.

The Ilco-9000 driver is a passive Client driver. The driver waits passively for incoming data messages from an Ilco-9000 type device and processes these messages, updating data in the FieldServer's data arrays. Thus, the data read by this driver is visible to other driver's loaded on the FieldServer.

The Ilco-9000 driver recognizes room *entries* and *exits* from the incoming log messages and stores the state of each room in a data array.

The Ilco-9000 driver is not capable of acting as a Server.

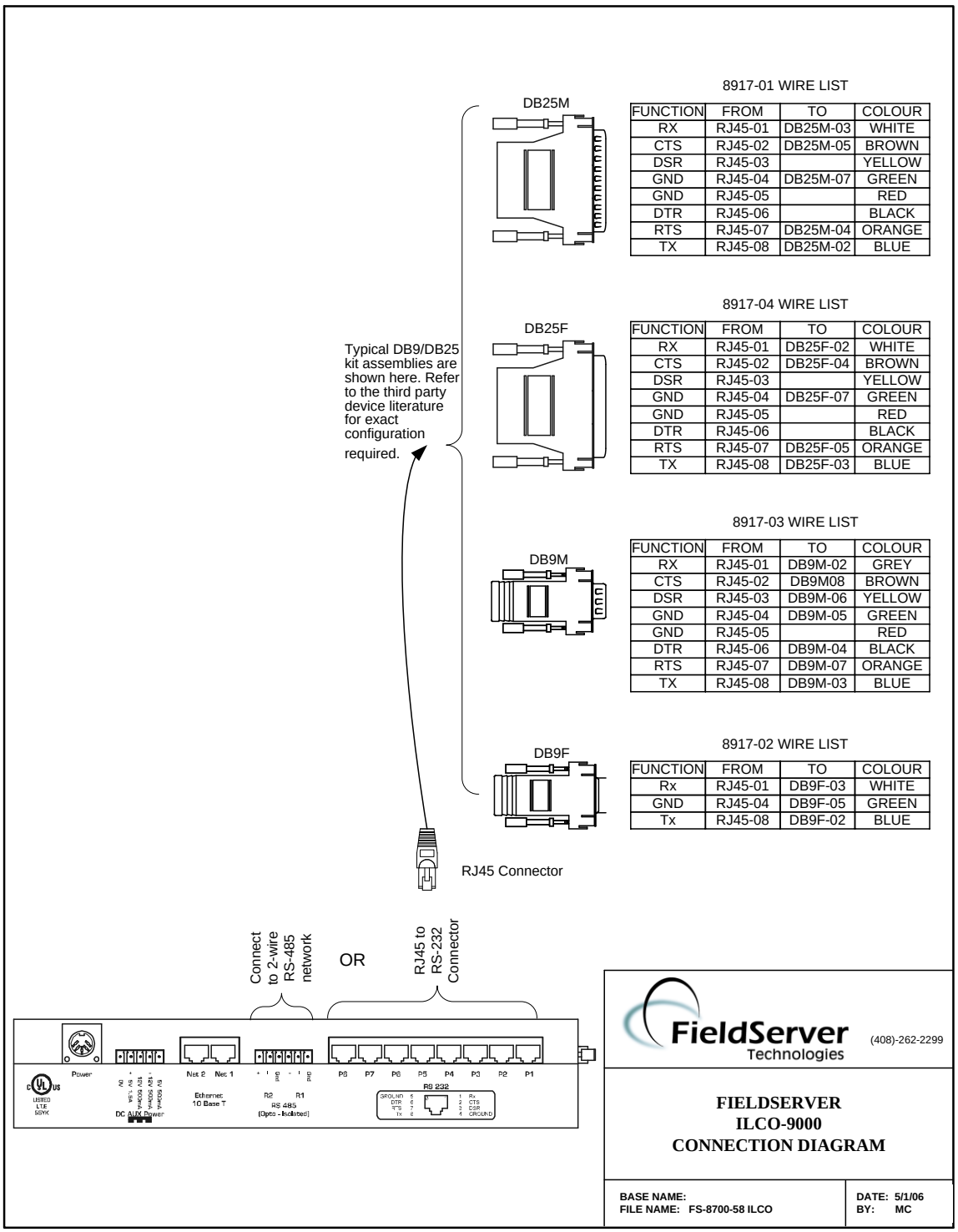
2. Driver Scope of Supply

2.1. Supplied by FieldServer Technologies for this driver

FieldServer Technologies PART #	Description
FS-8915-10	UTP cable (7 foot) for Ethernet connection
FS-8915-10	UTP cable (7 foot) for RS-232 use
FS-8917-02	RJ45 to DB9F connector adapter
FS-8917-01	RJ45 to DB25M connection adapter
	RS-485 connection adapter.
FS-8700-58	Driver Manual.

3. Hardware Connections

The FieldServer is connected to the Ilco-6000 as shown in connection drawing. Configure the Ilco-6000 according to manufacturer's instructions



8917-01 WIRE LIST

FUNCTION	FROM	TO	COLOUR
RX	RJ45-01	DB25M-03	WHITE
CTS	RJ45-02	DB25M-05	BROWN
DSR	RJ45-03		YELLOW
GND	RJ45-04	DB25M-07	GREEN
GND	RJ45-05		RED
DTR	RJ45-06		BLACK
RTS	RJ45-07	DB25M-04	ORANGE
TX	RJ45-08	DB25M-02	BLUE

8917-04 WIRE LIST

FUNCTION	FROM	TO	COLOUR
RX	RJ45-01	DB25F-02	WHITE
CTS	RJ45-02	DB25F-04	BROWN
DSR	RJ45-03		YELLOW
GND	RJ45-04	DB25F-07	GREEN
GND	RJ45-05		RED
DTR	RJ45-06		BLACK
RTS	RJ45-07	DB25F-05	ORANGE
TX	RJ45-08	DB25F-03	BLUE

8917-03 WIRE LIST

FUNCTION	FROM	TO	COLOUR
RX	RJ45-01	DB9M-02	GREY
CTS	RJ45-02	DB9M08	BROWN
DSR	RJ45-03	DB9M-06	YELLOW
GND	RJ45-04	DB9M-05	GREEN
GND	RJ45-05		RED
DTR	RJ45-06	DB9M-04	BLACK
RTS	RJ45-07	DB9M-07	ORANGE
TX	RJ45-08	DB9M-03	BLUE

8917-02 WIRE LIST

FUNCTION	FROM	TO	COLOUR
Rx	RJ45-01	DB9F-03	WHITE
GND	RJ45-04	DB9F-05	GREEN
Tx	RJ45-08	DB9F-02	BLUE

4. Configuring the FieldServer as a Ilco-9000 Client

For a detailed discussion on FieldServer configuration, please refer to the instruction manual for the FieldServer. 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 Ilco-9000 Server.

4.1. Data Arrays/Descriptors

The configuration file tells the FieldServer about its interfaces, and the routing of data required. In order to enable the FieldServer for Ilco-9000 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 that in the tables, * indicates an optional parameter, with the bold legal value being the default.

Section Title		
Data_Arrays		
Column Title	**Function**	**Legal Values**
Data_Array_Name	Provide name for Data Array	Up to 15 alphanumeric characters
Data_Format	Provide data format. Each data array can only take on one format. Rooms are regarded as being in state-0;empty or state-1;occupied. Thus bit data arrays are capable of storing a complete state record for each room.	FLOAT, BIT, UInt16, SInt16, Packed_Bit, Byte, Packed_Byte, Swapped_Byte
Data_Array_Length	Number of Data Objects. Must be larger than the data storage area required for the data being placed in this array.	1-10,000

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
```

4.2. Client Side Connection descriptors

Section Title
Connections

Column Title	**Function**	**Legal Values**
Port	Specify which port the device is connected to the FieldServer	P1-P8, 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	Ilco9000
Handshaking*	Specify hardware handshaking	RTS, RTS/CTS, **None**
Poll Delay*	Time between internal polls	0-32000 seconds **1 second**

Example

```
//      Client Side Connections

Connections
Port,       Protocol,       Baud,       Parity,     Handshaking,    Poll_Delay
P8,         Ilco9000,       9600,       None,       None,           0.100s
```

[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.

4.3. Client Side Node Descriptors

Section Title
Nodes

Column Title	Function	Legal Values
Node_Name	Provide name for node	Up to 32 alphanumeric characters
Node_ID	Modbus station address of physical Server node	1-255
Protocol	Specify protocol used	Ilco9000
Port	Specify which port the device is connected to the FieldServer	P1-P8, R1-R2

Example

```
//  Client Side Nodes

Nodes
Node_Name,      Node_ID,    Protocol,    Connection
Logger1,        1,          Ilco9000,    P8
```

4.4. Client Side Map Descriptors

4.4.1. FieldServer Related Map Descriptor Parameters

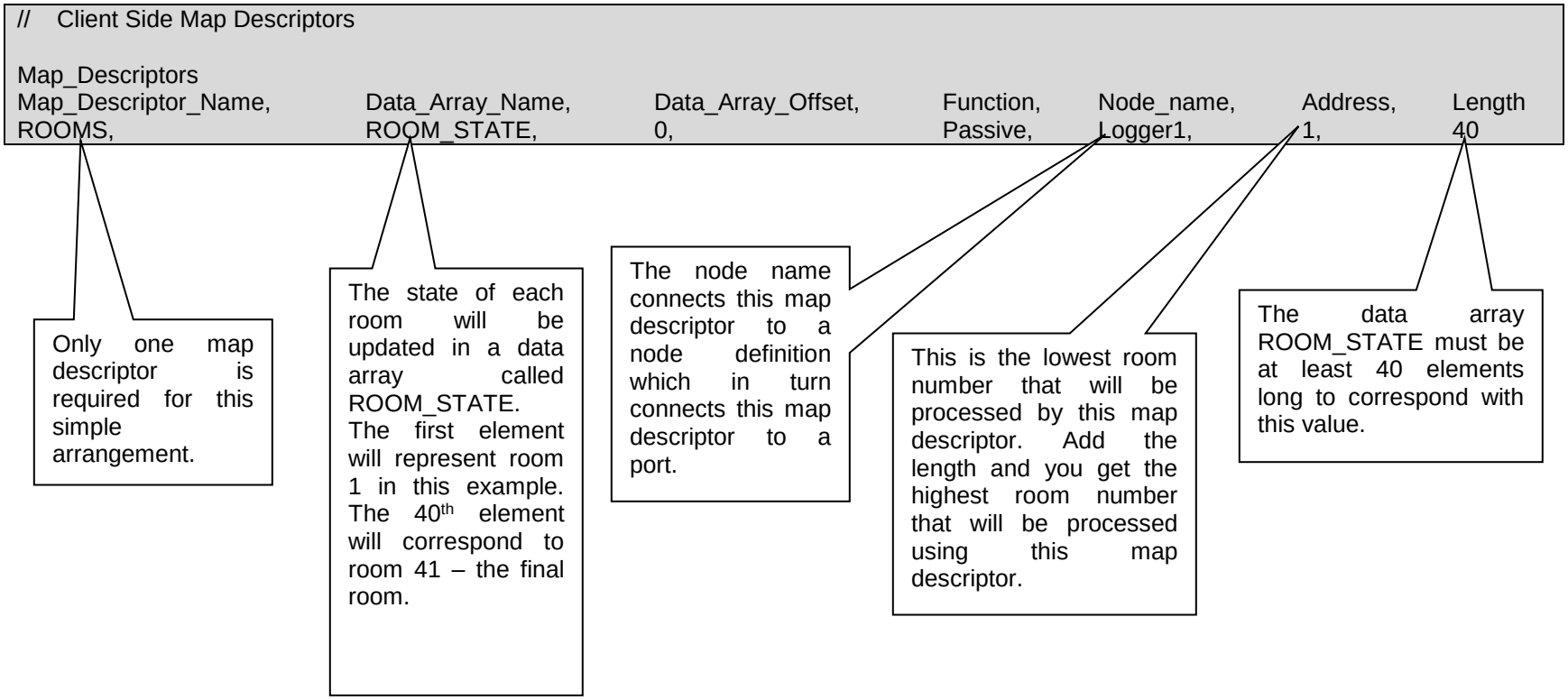
Column Title	Function	Legal Values
Map_Descriptor_Name	Name of this Map Descriptor	Up to 32 alphanumeric characters
Data_Array_Name	Name of Data Array where data is to be stored in the FieldServer	One of the Data Array names from “Data Array” section above
Data_Array_Location	Starting location in Data Array	0 to maximum specified in “Data Array” section above
Function	Function of Client Map Descriptor	Passive.

4.4.2. 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 “Client Node Descriptor” above
Data_Type	Data type	Register, Coil, AI, DI
Length	Number of rooms to be processed by this map descriptor. Refer to examples below.	1 - 1000
Address	Starting room number.	101, 201, 29001 …. etc

4.4.3. Map Descriptor Example 1

In this example, Assume the hotel has 40 rooms starting at room 1, which are to be monitored. The room numbers are consecutive indicating a range of room numbers from 1 to 41. The banquet hall and conference facilities are identified as rooms 1000, 1001 … and are not monitored.



4.4.4. Map Descriptor Example 2.

In this example, Assume the hotel has 40 rooms per floor starting on floor number 3. There are 5 consecutively numbered floors. The rooms are numbered 301,302 on the 3rd floor, 401,402… on the 4th floor. One approach is to use the previous example and set up the length as being 900 long. Set the start address to 1. This will mean that room 301's state will be found in the 301st element of the data array. Room 741's state will be found in the 741st element of the data array. This will result in a big empty data array. This doesn't present a problem to the FieldServer but it may present a problem to the data user. nThis example illustrates a different approach that allows the data to be packed more efficiently.

```
//  Client side Map Descriptors

Map_Descriptors
Map_Descriptor_Name,    Data_Array_Name,    Data_Array_Offset,    Function,    Node_name,    Address,    Length
FLR3,                   ROOM_STATE,         0,                    Passive,     Logger1,      301,        40
FLR4,                   ROOM_STATE,         40,                   Passive,     Logger1,      401,        40
FLR5,                   ROOM_STATE,         80,                   Passive,     Logger1,      501,        40
FLR6,                   ROOM_STATE,         120,                  Passive,     Logger1,      601,        40
FLR7,                   ROOM_STATE,         160,                  Passive,     Logger1,      701,        40
```

There is one map descriptor per range of room numbers.

Each map descriptor uses the same data array - all that changes is the offset.

The array offset controls the location of the data array that the room states are stored in.

Room 301's state will be stored at index location zero in the data array (the first array element).

Room 401's state will be stored in element index 40 which is the 41st element.

This is the lowest room number that will be processed by this map descriptor. Add the length and you get the highest room number that will be processed using this map descriptor.

Make sure the array is large enough to store the maximum room number.

Appendix A. Driver Notes

Appendix A.1. How the driver determines the room state.

The driver is programmed to processes the log records produces by the Ilco-9000/Millennium device and look for certain keywords in order to process the room state.

The functionality described below cannot be configured.

The room number is determined by inspecting a log record for the keyword *ROOM*. The inspection is not case sensitive. If the keyword is found then the driver interprets the next string of characters as the room number. The first occurrence of the word *ROOM* on a line is used for this determination.

Examples:

Log record	Driver determined room number
01/24/01 13:42:47 Room 29029 exit for maid GRA-29N2	29029
01/24/01 15:44:56 Room 29029 unlocked for guest VIP128	29029
01/24/01 13:42:47 Room 29029 exit for maid GRA-29N2	29029
01/24/01 15:39:17 Con 33 Op VIP LOG KEY Room 29030 VIP127 Assign Guest	29030
01/24/01 15:44:56 Room A29030 unlocked for guest VIP128	0
01/24/01 08:57:03 Room invalid entry attempt key 85021B	0
01/24/01 08:57:03 Room Entry Room 29030 unlocked for maid GRA-29N2	0
01/24/01 08:57:03 Room 29030 invalid entry attempt to room key 85021B	29030
01/24/01 13:42:47 Room Entry 29030	0

The room state is determined by looking for an action key word within a log record. Obviously the same record must also have a room number in a format the driver can process. The inspection is not case sensitive.

Examples:

Action Key Word	Room State Set
CLEAR or DELETED	Empty (State 0)
REPLACE or ASSIGN	Occupied (State 1
Change	One Room Empty and 1 room occupied according to following example: Wed 09/14/00 14:14:26 Con 2 Op Sally Change from 15089 to 17114. Room 15089 will be set to empty (state 0) Room 17114 will be set to occupied (state 1)

Log records are processed in sequential order. The time stamp on the line is not considered in determining the state of a room.

THIS PAGE INTENTIONALLY LEFT BLANK