



A Sierra Monitor Company

Driver Manual
(Supplement to the FieldServer Instruction Manual)

FS-8700-67 Russelectric Model 2000

APPLICABILITY & EFFECTIVITY

Effective for all systems manufactured after May 1, 2001

Instruction Manual Part Number FS-8700-67

5/6/2002

TABLE OF CONTENTS

1.	Russelectric Model 2000 Description	1
2.	Driver Scope of Supply	2
2.1	Supplied by FieldServer Technologies for this driver	2
2.2	Provided by user.....	2
3.	Hardware Connections.....	3
4.	Configuring the FieldServer as a Russelectric Model 2000 Client	4
4.1	Data Arrays	4
4.2	Client Side Connections.....	5
4.3	Client Side Nodes.....	6
4.3.1	FieldServer Specific Map Descriptor Parameters	7
4.3.2	Driver Specific Map Descriptor Parameters	7
4.3.3	Timing Parameters	8
4.3.4	Map Descriptor Example.	9
5.	Configuring the FieldServer as a Russelectric Model 2000 Server	10
5.1	Data Arrays	10
5.2	Server Side Connections.....	11
5.3	Server Side Nodes.....	12
5.4	Server Side Map Descriptors.....	13
5.4.1	FieldServer Specific Map Descriptor Parameters	13
5.4.2	Driver Specific Map Descriptor Parameters	13
5.4.3	Map Descriptor Example.	14
6.	Driver Notes.....	15

1. Russelectric Model 2000 Description

The Serial Russelectric Model 2000 driver allows the FieldServer to transfer data to and from devices over either RS232 or RS485 using RTU protocol. There are eight RS232 and two RS485 ports standard on the FieldServer. The FieldServer can emulate either a Server or Client.

The Russelectric Model 2000 drivers implement a Model 2000 client and a Model 2000 server. The client driver can both read data from a remote server as well as send write data commands. The server driver emulates a Model 2000 device and responds to data read and write poll commands.

Messages

The client driver implements the following RTU message or command types:

1. Read_Output_Table – Used to read the bit values of the discrete outputs on the Model 2000.
2. Read_Registers – Used to read the integer values of the Model 2000's internal registers.
3. Force_Single_Output – Used to set or clear the bit value of a single discrete output on the Model 2000.
4. Preset_Single_Register – Used to load an integer value into a specific single register of the Model 2000.

Addressing

Every Model 2000 has a unique address assigned to it. The client driver can send messages to a specific Model 2000 by using the unique address in the message or by using a special broadcast address of value 0. Only the following messages can be sent in broadcast by the client (other messages are read data types and would cause a communications collision when all Model 2000 devices respond):

1. Force_Single_Output
2. Preset_Single_Register

All Model 2000 devices have to receive and process a broadcast message. No device may send a response message in reply. Subsequent data read commands by the client driver will update the data that was changed or loaded by a broadcast message.

2. Driver Scope of Supply

2.1 Supplied by FieldServer Technologies for this driver

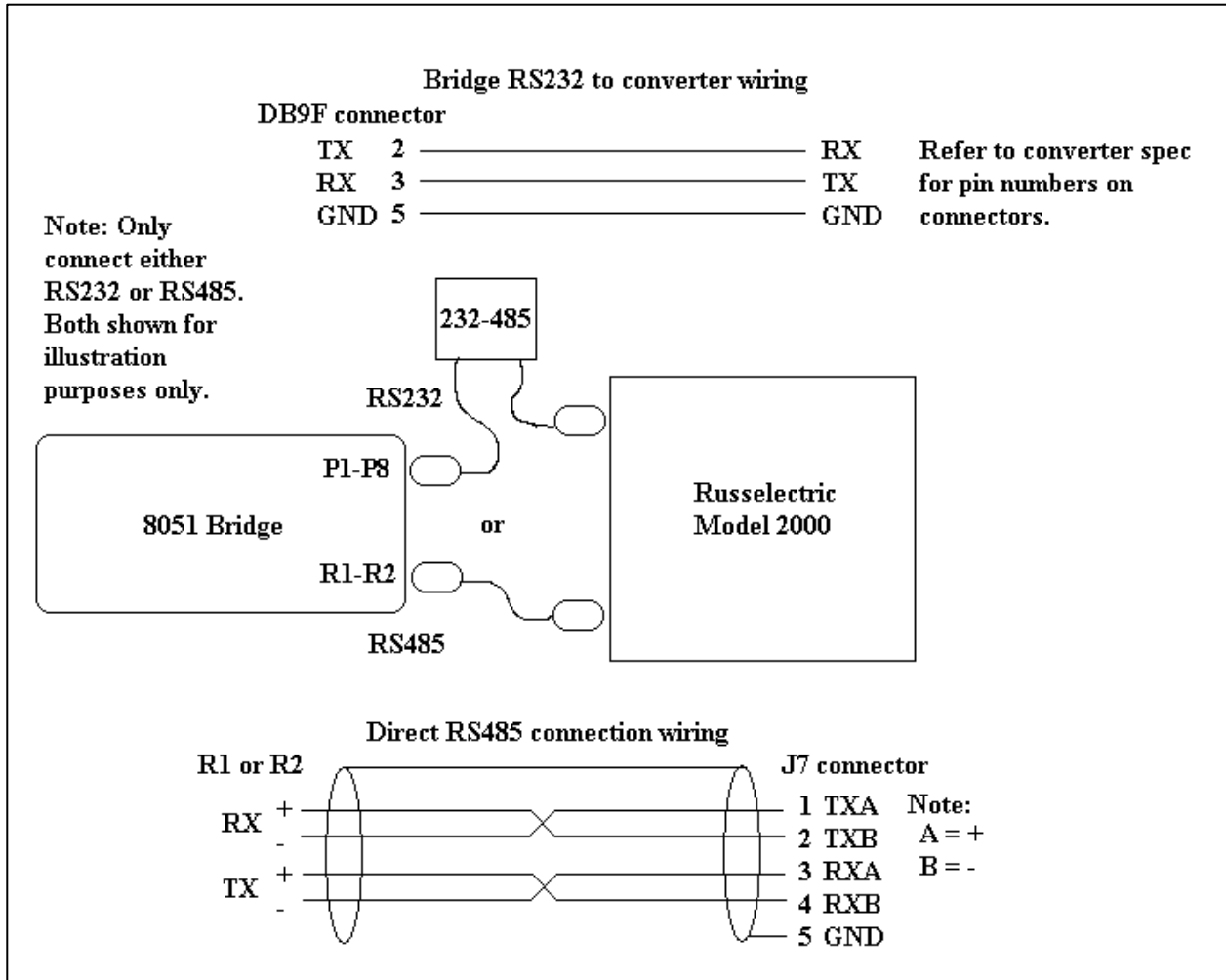
RJ45 to DB9F connection adapter (for connecting a 8051 RS232 port to a converter)
Driver Manual.

2.2 Provided by user

The user has to provide a suitable cable with appropriate connectors that will connect the Model 2000 either directly to the FieldServer or to the RS232 to RS485 converter. When connecting to a converter, consult your converter's documentation for pinouts.

3. Hardware Connections

The FieldServer is connected to the Model 2000 as shown below.
 Configure the Model 2000 according to manufacturer's instructions.



4. Configuring the FieldServer as a Russelectric Model 2000 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” files on the driver diskette).

This section documents and describes the parameters necessary for configuring the FieldServer to communicate with a Russelectric Model 2000 Server.

The configuration file tells the FieldServer about its interfaces, and the routing of data required. In order to enable the FieldServer for Russelectric Model 2000 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.

4.1 Data Arrays

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.	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_Format,      Data_Array_Length
RTU_Registers,       UInt16,          79
```

4.2 Client Side Connections

Section Title		
Connections		
Column Title	Function	Legal Values
Port	Specify which port the device is connected to the FieldServer	P1-P8, R1-R2 (P1-P8 can be used when using the RS232 to RS485 converter)
Baud*	Specify baud rate	110 – 115200, standard baud rates only (set to same value as used on Model 2000)
Parity*	Specify parity	Even, Odd, None , Mark, Space (refer to Model 2000 setup)
Data_Bits*	Specify data bits	8
Stop_Bits*	Specify stop bits	1
Protocol	Specify protocol used	Rus
Handshaking*	Specify hardware handshaking	None
Poll Delay*	Time between internal polls	0.5s

Example

```
// Client Side Connections

Connections
Port, Baud, Parity, Protocol, Handshaking, Poll_Delay
P1, 9600, None, Rus, None, 1.0s
```

4.3 Client Side Nodes

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	0-247 (Node 0 used exclusively to broadcast)
Protocol	Specify protocol used	Rus
Port	Specify which port the device is connected to the FieldServer	P1-P8, R1-R2 (P1-P8 can be used when using the RS232 to RS485 converter).

Example

```
// Client Side Nodes

Nodes
Node_Name, Node_ID, Protocol, Port
Node_0, 0, Rus, P1
Node_1, 1, Rus, P1
```

Note that a node with Node_ID equal to zero (0) is reserved for a client's ability to transmit a message in broadcast. Other nodes specified refer to remote Model 2000 devices that will be communicated with.

Client Side Map Descriptors

4.3.1 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 "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	RDBC, WRBC, WRBX

4.3.2 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 "Client Node Descriptor" above. Node_0 can be used to broadcast a command. Only Preset_xx and Force_xx (eg. Write type commands) can be used with Node_0.
Rus_command	Defines a type of RTU message that will be sent	<i>Register commands:</i> Read_Registers, Preset_Single_Register, <i>Point commands:</i> Read_Discrete_Outputs, Force_Single_Output
Start_point	Denotes the start register number or start point number that will be used in a command	1-79 for register commands 1-34 for point commands
Num_points	Denotes the number of registers or the number of points that will be used in a command. The value is inclusive of the start point number.	1-79 for register commands 1-34 for point commands For xx_Single_xx commands num_points defaults to 1. The number of points from start_point to (start_point+num_points) must be less than or equal to the maximum number of points or registers available, which is 34 and 79 respectively.
Output_value	Denotes the value that will be output in a xx_Force_xx or xx_Preset_xx command	0-65,535 for register commands 0,1 for point commands Output_value is ignored for other commands.

4.3.3 Timing Parameters

Column Title	Function	Legal Values
Scan_Interval	Rate at which data is polled	>0.5s

4.3.4 Map Descriptor Example.

Map_Descriptor_Name, Scan_Interval, Data_Array_Name, Data_Array_Offset, Function, node_name, rus_command, start_point, num_points, output_value
 RUS_MBA1, 3.0s RTU_Discrete_Outputs, 0, rdbc, Node_1, Read_Discrete_Outputs, 1, 34, 0

This can be any name but each name must be unique. Name will appear in FieldServer map descriptor status information screens.

Scan interval must be adapted for multiple map descriptor scans to prevent a situation where not all map descriptors can be executed in a certain time. Take remote device response time into consideration here.

The data array name must be one found under Data_Arrays. Data from the scan will be stored into the array at Data_Array_Offset.

This value specifies the offset into the data array (RTU_Discrete_Outputs) where the data fetched must be stored.

The function may be read for read commands and write for write type commands. Continuous operations are allowed.

Node name must be one found under Nodes, Node_name. Data will be fetched from this node and port during a scan. Use Node_0 with id of 0 to execute a broadcast command.

rus commands may be one of several commands specified in the table eg. Preset_Single_Register

Start point of 1 to 34 allowed. If start point was 34, then num_points can only be 1, since the maximum number of points on the Model 2000 is 34. For registers the maximum number is 79.

Num_points is the number of points from start point. Thus endpoint will be start_point plus num_points minus 1. One is subtracted since num_points includes start_point

The output value for write type commands. Point commands accept 0=off or 1=on. Register commands accept a positive value from 0 to 65,535

5. Configuring the FieldServer as a Russelectric Model 2000 Server

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” files on the driver diskette).

This section documents and describes the parameters necessary for configuring the FieldServer to communicate with a Russelectric Model 2000 Client.

The configuration file tells the FieldServer about its interfaces, and the routing of data required. In order to enable the FieldServer for Russelectric Model 2000 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 that in the tables, * indicates an optional parameter, with the bold legal value being the default.

5.1 Data Arrays

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.	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_Format,      Data_Array_Length
DA_AI1,              UInt16,          79
DA_AI2,              byte,           5
    
```

5.2 Server Side Connections

Section Title		
Connections		
Column Title	Function	Legal Values
Port	Specify which port the device is connected to the FieldServer	P1-P8, R1-R2 (P1-P8 can be used when using the RS232 to RS485 converter)
Baud*	Specify baud rate	110 – 115200, standard baud rates only (set to same value as used on Model 2000)
Parity*	Specify parity	Even, Odd, None , Mark, Space (refer to Model 2000 setup)
Data_Bits*	Specify data bits	8
Stop_Bits*	Specify stop bits	1
Protocol	Specify protocol used	Rus
Handshaking*	Specify hardware handshaking	None

Example

```
//      Server Side Connections

Connections
Port, Baud, Parity, Protocol,      Handshaking
P1,   9600, None,   Rus,          None
```

5.3 Server Side Nodes

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	0-247 (Node 0 used exclusively to receive broadcast messages)
Protocol	Specify protocol used	Rus

Example

```
//      Server Side Nodes

Nodes
Node_Name, Node_ID, Protocol
Node_0,    0,      Rus
Node_1,    1,      Rus
```

5.4 Server Side Map Descriptors

5.4.1 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 "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

5.4.2 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
Rus_command	When processing a command from a client, use the associated map descriptor properties. It uniquely identifies from what data array data should be read or stored in response to a specific command from a client.	Read_Discrete_Outputs, Force_Single_Output, Read_Registers, Preset_Single_Register

5.4.3 Map Descriptor Example.

Map_Descriptor_Name, Data_Array_Name, Data_Array_Offset, Function, node_name, rus_command
SMB_AI3, DA_AI2, 0, Passive, Node_1, Read_Discrete_Outputs

This can be any name but each name must be unique. Name will appear in FieldServer map descriptor status information screens.

The data array name must be one found under Data_Arrays. Data from the forth script file will be stored into the array at Data_Array_Offset. This data will be sent to a requesting client.

This value specifies the offset into the data array (DA_AI2) where the data from the forth script will be stored. Note that the script can offset the data in addition to this offset value.

Function may not be read or write since it implements a server. Function may only be passive.

Node name must be one found under Nodes, Node_name. This defines the data array for node name and polls from a client to this node will be answered with data from this data array.

The command message name here associates a specific command from a client with this map descriptor.

6. Driver Notes

None.