A Sierra Monitor Company

Driver Manual
(Supplement to the FieldServer Instruction Manual)

FS-8700-82 Carrier DataLink

APPLICABILITY & EFFECTIVITY

Effective for all systems manufactured after May 1, 2001

Driver Version:	**1.03**
Document Revision:	**5**

FieldServer Technologies 1991 Tarob Court Milpitas, California 95035 USA **Web**:www.fieldserver.com
Tel: (408) 262-2299 **Fax**: (408) 262-9042 **Toll_Free**: 888-509-1970 **email**: support@fieldserver.com

TABLE OF CONTENTS

1. Carrier DataLink Description

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

The Carrier DataLink Serial Driver allows variables to be read and written in system elements connected to a Carrier Comfort Network communication bus. Up to 15 system elements may be connected to a DataLink Device. Display, Occupancy, Set Point Tables and Carrier Comfort Network (CCN) variables may be read or written.

The DataLink device provides a gateway to CCN devices. This driver polls the DataLink device which in turn reads/writes data to the CCN devices.

Carrier limits the number of CCN devices that can polled from a DataLink Device as well as the data that can be transferred between some CCN devices and the DataLink device. For information on these limitations please consult the Carrier Corporation.

The driver is an active client driver. This means that it initiates read/write polls with the DataLink device which is expected to provide responses. Server functionality is not intended to provide full emulation of DataLink Devices but can be extended on request.

The driver is configured to allow data tables to be read from the CCN devices via the DataLink device. As the tables typically contain more than one data element, the retrieved data is stored in a number of consecutive FieldServer data array locations in the FieldServer. The individual values cannot be scaled before they are stored. The driver can provide descriptions for each of the table values retrieved. Descriptions are stored in ASCII in a separate data array.

The driver can be configured to read a specific variable from a CNN device and store its value using optional scaling in a configurable location in a FieldServer data array.

The driver can be configured to write a value (using optional scaling) from a FieldServer data array to a specific variable in a CNN device, provided that the variable allows its value to be written.

The driver has no advanced knowledge of the CCN devices and their data tables. Therefore it cannot validate table names or variable names specified in the configuration file. This means that the driver handles each table in a generic way, without regard for the particular variables that constitute the tables.

It is important that you understand the limitations and exclusions of this driver. Refer to Section 5.4 for more information.

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 RS-232 use
FS-8917-17	RJ45 to DB25M connection adapter
FS-8700-82	Driver Manual.

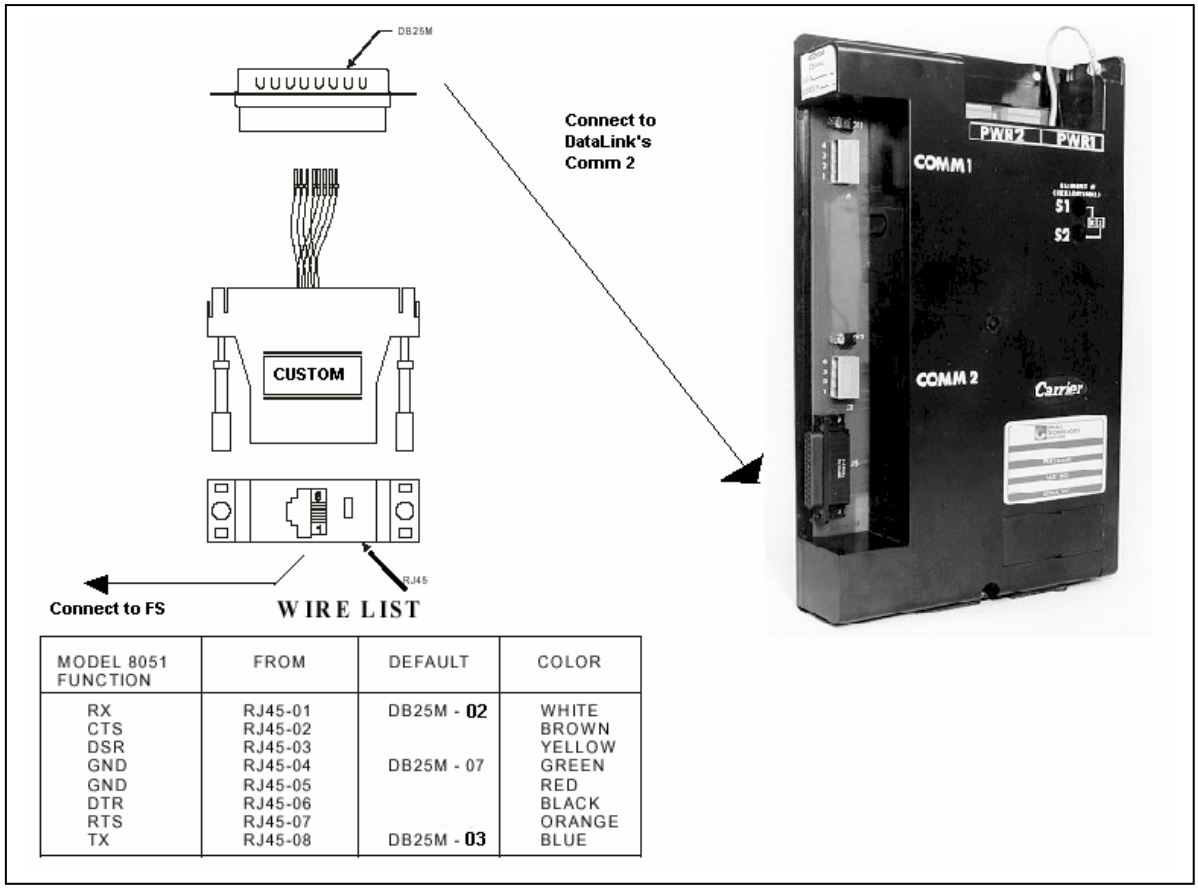
2.2. Provided by Suppler of 3rd Party Equipment

PART #	DESCRIPTION
	Carrier DataLink Card

3. Hardware Connections

The FieldServer is connected to the Carrier DataLink device as shown below. Ensure that the device to be connected to is in fact the DataLink device and not the DataPort device which is similar looking.

Configure the DataLink Device according to manufacturer's instructions. This driver requires that the DataLink device's DTPCONFIG table has been configured prior to connection with a FieldServer. Consult the manufacturer's information on connecting Carrier Device's to CCN network.



WIRE LIST

MODEL 8051 FUNCTION	FROM	DEFAULT	COLOR
RX	RJ45-01	DB25M - **02**	WHITE
CTS	RJ45-02		BROWN
DSR	RJ45-03		YELLOW
GND	RJ45-04	DB25M - 07	GREEN
GND	RJ45-05		RED
DTR	RJ45-06		BLACK
RTS	RJ45-07		ORANGE
TX	RJ45-08	DB25M - **03**	BLUE

4. Configuring the FieldServer as a Carrier DataLink Client

It is not possible to configure communication with a DataLink device until familiar with the data available from the devices connected to the DataLink. The DataLink device does not provide a method for discovering the data tables and variables that are available in all the Carrier devices.

In order to configure the Client it is necessary to know the names of the data tables available in the devices connected via the CCN network to the data link. A partial list of available tables is provided in Appendix A.1. If you know the table names and the variable names that you wish to poll then you have enough information to complete the configuration. If you have table names but do not know variable names then we provide a method of discovering the variable names. This method is discussed in chapter 6. If you do not have the table names then you will need to consult with the Carrier Corporation before proceeding.

For a detailed discussion on FieldServer configuration, please 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 Carrier DataLink Server.

The configuration file tells the FieldServer about its interfaces, and the routing of data required. In order to enable the FieldServer for Carrier DataLink 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/Descriptors

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_AI_01,               UInt16,         200
DA_AO_01,               UInt16,         200
DA_DI_01,               Bit,            200
DA_DO_01,               Bit,            200
```

4.2. Client Side Connection Descriptions

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	300, 600 , 1200, 2400, **9600** (Vendor limitation)
Parity*	Specify parity.	**None** (Vendor Limitation)
Data_Bits*	Specify data bits	**8** (Vendor Limitation)
Stop_Bits*	Specify stop bits	**1** (Vendor limitation)
Protocol	Specify protocol used	CarrrierDL
Handshaking*	Specify hardware handshaking	**None**
Poll _Delay*	Time between internal polls	0-32000 seconds, **1 second**
Timeout*	When reading entire tables it is recommended that the timeout be set to between 15 and 30 seconds. Additional information is provided in Appendix A.5	0.1 to 60.0 seconds, **2.0seconds**
IC_Timeout*	When reading entire tables it is recommended that the ic_timeout be set to approx 15.0 seconds. Additional information is provided in Appendix A.5	

Example

```
//  Client Side Connections

Connections
Port,   Baud,   Parity,   Data_bits,   Stop_Bits,   Protocol,    Handshaking,   Poll_Delay
P8,     9600,   None,     8      ,     1      ,     CarrierDL,   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. These correspond to the 'devices' configured in the DTPConfig. Thus the Node_ID is not the address of the final CCN device. The DataLink DTPConfig table maps a device number (1..15) to a bus number (0-239). Use the Node_ID to tell the driver which device to use.	1-15
Protocol	Specify protocol used	CarrierDL
Port	Specify which port the device is connected to the FieldServer	P1-P8, R1-R2[2]

Example

```
//    Client Side Nodes

Nodes
Node_Name,          Node_ID,          Protocol ,          Port
FAN1     ,          1      ,          CarrierDL,          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	RDBC, WRBC, WRBX

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

4.4.2. 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 "Client Node Descriptor" above
Data_Type	Data type. This parameter is not required except when defining Map Descriptors which write DOW (Day of Week) and Time value to the Carrier devices. When a write Map Descriptor checks the data type it uses the information to convert the value extracted from the FieldServer's Data Arrays into a suitable format for a write. Refer to Section 4.4.7.	- Occupancy_Time Occupancy_DOW
Length	Length of Map Descriptor. When reading a compete table, set the length to the maximum number of data values you want stored. Additional information on the length parameter is provided in Appendix A.5.	1 – 1000
Address	This commonly used parameter is not required for this driver.	
Table_Name	The name of the table to be polled, e.g. DISPLAY. Some system elements have multiple instances of the same table name. For example, the Terminal System Manager has 64 Temperature Zone configuration tables named TZONE1 though TZONE64. These tables are accessed by using both the primary & secondary table names, e.g. 'TZCONF TZONE1'	Only ASCII characters are permitted. When using the table name parameter to specify a primary and secondary table, leave a single space between the two names.
Field_Name*	If you wish to read the whole table, leave the field name blank or specify 'EVERYTHING'.	Only ASCII characters are permitted. Field names never contain spaces. They may not be longer than 10 characters long.
Store_As*	Use the Ascii or AsciiLog format when you are discovering the variables contained in a table by reading a table. Refer to 1.1	Ascii, AsciiLog, **Values**

4.4.3. Timing Parameters

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

4.4.4. Map Descriptor Example 1 – Read a Table

This example illustrates how to read the entire contents of a table called 'DISPLAY'. The Map Descriptor is connected to a node using the node name and a node definition provides a connection to a port. Thus this Map Descriptor is connected to a port via its node. The FieldServer will use that port to send this poll which will be generated every 5 seconds.

The value extracted from the response will be stored in the array called DA_DISPLAY. Ensure that it is suitable format for storing a number – FLOAT is suggested. The driver stores the value of the 1st element at offset 1; the 2nd element at offset 2… The number of table elements whose values have been successfully stored at is stored at offset zero.

```
//  Client Side Map Descriptors

Map Descriptors
Map_Descriptor_Name,  Data_Array_Name,  Data_Array_Offset,  Function,  Node_Name,  Length,  Scan_Interval,  Table_Name,  Field_Name
Read_Table_Md      ,  DA_DISPLAY     ,  0                 ,  rdbc      ,  FAN01    ,  100   ,  5.0s          ,  DISPLAY   ,  EVERYTHING
```

Responses are stored in this data array. Table element x gets stored at offset x, where x is a non-zero integer.

Offset zero is used by the driver to report how many values were stored.

The driver generates a read poll continuously.

The node name connects the Map Descriptor to a node definition.

The length is set to a number big enough to contain all the table's variables.

The name of the table to be read.

By using the keyword 'EVERYTHING' you are telling the driver to read the whole table.

4.4.5. Map Descriptor Example 2 – Read a Variable

This example illustrates how to read single variables from a Carrier Device. A Map Descriptor needs to be defined for each variable. Reading variables is effective for reading a limited set of variables from a table(s). Scaling can be applied when the driver stores the value by using the additional parameters: Data_Array_Low_Scale, Data_Array_High_Scale, Device_Low_Scale, Device_High_Scale

```
Map_Descriptor_Name,  Data_Array_Name,  Data_Array_Offset,  Function,  Node_Name,  Length,  Scan_Interval,  Table_Name,  Field_Name
Read_Temps_Md1    ,   DA_TEMPS      ,   0                 ,   rdbc    ,   FAN01    ,   1       ,   5.0s          ,   DISPLAY  ,   RAT
Read_Temps_Md2    ,   DA_TEMPS      ,   1                 ,   rdbc    ,   FAN01    ,   1       ,   5.0s          ,   DISPLAY  ,   SAT
```

As we are using the same data array for both variables, we change the offset.

The 'RAT' temperature will be stored at index 0 (first element), the SAT temperature will be stored at index 1 (2nd element of array).

The pneumonic variable name.

Space Temperature	-40.0 dF	RAT
Supply Air Temperature	-40.0 dF	SAT

The variable names are supplied by the Carrier Corporation. Refer to Appendix A.2 for assistance in determining version names.

4.4.6. Map Descriptor Example 3 – Write A Variable

This example illustrates how to force a variable. Always set the length to 1 for a write. If the variable cannot be written then an error will be reported. In this example the variable being written is called 'FAN'. It can be set on/off because is a discrete point.

```
Map_Descriptor_Name,  Data_Array_Name,  Data_Array_Offset,  Function,  Node_Name,  Length,  Scan_Interval,  Table_Name,  Field_Name
Write_Md1         ,   DA_DISPLAY    ,   0                 ,   wrbc    ,   FAN01    ,   1       ,   5.0s          ,   DISPLAY  ,   FAN
```

If the first element of this array is set to 1, the fan will be set ON; if set to zero the fan will be set OFF.

This function ensures that the driver writes the Setpoint to the device continuously (every 5 seconds in this example.) Use WRBX to generate a write message each time the value in the array is updated.

Take care to specify variables that can be forced.

4.4.7. Map Descriptor Example 4 – Write Day of Week (DOW) or Time

This example illustrates how to write values to an occupancy table. Occupancy tables contain amongst other variables, Day of Week (DOW) settings and time settings. The driver needs to be configured so that it formats these values correctly.

When the DOW write occurs, the driver extracts the value from the Data Array and converts it to a binary string whose bit value is the binary representation of the Data Array value, e.g. DA value = 131 DOW string = 10000011 e.g. DA value = 1 DOW string = 00000001

Occupancy's are set using a BCD value. Write the value decimal equivalent of 01111100 (62 decimal) to signify an occupancy of Monday to Friday. The bit string starts on Sunday, runs through Monday to Saturday and the final bit is for holidays.

When a time write occurs the driver extracts the value from the Data Array and coverts the value to a time string. The value extracted from the DA is considered to be the number of minutes from midnight, e.g. DA value = 1 - Time = 00:01; DA value = 959 - Time=15:59

```
Map_Descriptors
Map_Descriptor_Name,    Scan_interval,  Data_Array_Name,    Data_Array_Offset,  Function,  Node_Name,  Table_Name,              Field_Name ,  Length,  Data_Type
MapDesc13       ,       1.0s    ,       TABLE_OCCPC64S,     100         ,       wrbx    ,       Node_A  ,       OCCDEFCS OCCPC64S,      DOW3    ,       1       ,       Occupancy_DOW
MapDesc14       ,       1.0s    ,       TABLE_OCCPC64S,     101         ,       wrbx    ,       Node_A  ,       OCCDEFCS OCCPC64S,      UNOCC3  ,       1       ,       Occupancy_Time
```

The Data type tells the driver how to format the value for the write.

5. Configuring the FieldServer as a Carrier DataLink Server

For a detailed discussion on FieldServer configuration, please refer to the FieldServer Instruction 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 Carrier DataLink Client.

The FieldServer can be configured to emulate a Carrier DataLink Device. The use is able to define a variable number of data tables, allocating a table name a variable number of variables. The FieldServer may be polled and will respond like a DataLink device. Remote clients can force variable states by ending write variable commands.

The FieldServer does not emulate any of the alarm buffer/history features of the DataLink Device.

All variables may be read or written without restriction.

The configuration file tells the FieldServer about its interfaces, and the routing of data required. In order to enable the FieldServer for Carrier DataLink 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. Server 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[3]
Baud*	Specify baud rate	300, 600 , 1200, 2400, **9600** (Vendor limitation)
Parity*	Specify parity	**None** (Vendor limitation)
Data_Bits*	Specify data bits	**8** (Vendor limitation)
Stop_Bits*	Specify stop bits	**1** (Vendor limitation)
Protocol	Specify protocol used	CarrrierDL
Handshaking*	Specify hardware handshaking	**None**

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

Example

```
//  Server Side Connections

Connections
Port,  Baud  Parity,  Data_bits,  Stop_Bits,  Protocol ,  Handshaking,  Poll_Delay
P8,    9600,  None ,  8       ,  1        ,  CarrierDL,  None      ,  0.100s
```

5.2. 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. These correspond to the 'devices' configured in the DTPConfig. Thus the Node_ID is not the address of the final CCN device. The DataLink DTPConfig table maps a device number (1...15) to a bus number (0-239). Use the Node_ID to tell the driver which device to use.	1-15
Protocol	Specify protocol used	CarrierDL

Example

```
//  Server Side Nodes

Nodes
Node_Name,  Node_ID,  Protocol ,  Port
FAN1     ,  1      ,  CarrierDL,
```

It is common to leave Server nodes unconnected to a port. This means that the FieldServer can respond with the node's data irrespective of which port the request is received on.

5.3. Server Side Map Descriptors

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 "Data Array" section above
Data_Array_Location	Starting location in Data Array	0 to maximum specified in "Data Array" section above
Function	Map Descriptor function	Passive

5.3.2. 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 "Server Node Descriptor" above
Data_Type	This commonly used parameter is not required for this driver.	
Length	Length of Map Descriptor. When reading a compete table, set the length to the maximum number of data values to be stored. Refer to 1.1.	1 – 1000
Address	This commonly used parameter is not required for this driver.	
Table_Name	The name of the table to be polled, e.g. DISPLAY. Some system elements have multiple instances of the same table name, e.g. the Terminal System Manager has 64 Temperature Zone configuration tables named TZONE1 though TZONE64. These tables are accessed by using both the primary & secondary table names, e.g. 'TZCONF TZONE1'	ASCII characters only. When using this parameter to specify a primary and secondary table, leave a single space between the two names.
Field_Name	This is the field/table variable pneumonic.	ASCII characters only. No spaces, maximum length 10 characters.
Field_Description Field_Units	Returned when a client does a table read. If units have no meaning for the variable then leave this field blank or use a single dash.	ASCII string which may contain spaces.
On_String	If the variable has a discrete state, use this parameter to define the word that describes the state when the variable's value is 1. OR Use one of the keywords to tell the driver to send the value of the associated array elements as a number (keyword=NUMERIC),	NUMERIC TIME STRING DOW Or any other ASCII string which may not contain spaces.

Column Title	Function	Legal Values
	as a time value formatted as hh:mm (keyword=TIME) or as a string of bytes (keyword=STRING) or as an occupancy string (keyword=DOW). When using the string keyword the driver reads x consecutive array elements and treats them like ASCII character values in forming the response. X is defined by the length parameter.	
Off_String	If the variable has a discrete state, use this parameter to define the word that describes the state when the variable's value is zero. If you have used one of the above keywords, simply put a dash in this field.	An ASCII string which may not contain spaces.
Length	Always set the length to 1 unless you use the key word 'STRING' as the *On_String* parameter.	1

5.3.3. 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

5.3.4. Map Descriptor Example.

This example shows a number of server Map Descriptors used to define a table called 'DISPLAY'. If a client sends a read table request the driver responds by sending all the appropriate data for every Map Descriptor with the same table name (provided that the node's match).

Map_Descriptor_Name,	Function,	Node_Name,	Table_Name,	Field_Name,	Field_Description,	Field_Units,	On_String,	Off_String,	Data_Array_Name,	Data_Array_Offset,	Length,
Display_MD01,	passive,	Node_1,	DISPLAY,	MODE,	Desired Mode,	,	ON ,	OFF ,	TABLE DISPLAY,	01,	1
Display_MD02,	passive,	Node_1,	DISPLAY,	ALARM ,	Equipment Status,	,	ALARM ,	NORMAL,	TABLE DISPLAY,	02,	1
Display_MD03,	passive,	Node_1,	DISPLAY,	CSPT ,	Controlling Setp,	dF,	NUMERIC,	-,	TABLE DISPLAY,	03,	1
Display_MD04,	passive,	Node_1,	DISPLAY,	SPT ,	Controlling Temp,	dF,	NUMERIC,	-,	TABLE DISPLAY,	04,	1
Display_MD05,	passive,	Node_1,	DISPLAY,	RAT ,	Space Temp,	dF,	NUMERIC,	-,	TABLE DISPLAY,	05,	1
Display_MD06,	passive,	Node_1,	DISPLAY,	SAT ,	Supply Air Temp,	dF,	NUMERIC,	-,	TABLE DISPLAY,	06,	1
Display_MD07,	passive,	Node_1,	DISPLAY,	FANSTAT,	Fan Mode,	,	ON ,	OFF,	TABLE DISPLAY,	07,	1
Display_MD08,	passive,	Node_1,	DISPLAY,	CCAP ,	Cooling Capacity,	% ,	NUMERIC,	-,	TABLE DISPLAY,	08,	1
Display_MD09,	passive,	Node_1,	DISPLAY,	HCAP ,	Heating Capacity,	% ,	NUMERIC,	- ,	TABLE DISPLAY,	09,	1
Display_MD10,	passive,	Node_1,	DISPLAY,	FLTSTAT,	Filter Status ,	,	DIRTY ,	CLEAN,	TABLE DISPLAY,	10,	1

All the server Map Descriptors are passive.

If the value of the array named TABLE_DISPLAY, offset 10 is 1 then the FieldServer will report he FLTSTAT variable as DIRTY. If it is zero then the variable's state will be reported as clean.

By using the NUMERIC keyword, the driver is told to report the value of the variable HCAP as a number. Thus the driver sends the value of the array named TABLE_DISPLAY offset 9 to the client.

5.4. Driver Limitations and Exclusions

As a client:

- The Carrier DataLink driver is not capable of polling for alarm data. (The RA and AV functions are not implemented).
- The Carrier DataLink driver dos not validate table or variable names.
- The Carrier DataLink driver does not support the Read Table command with the /C or /N options.
- The Carrier DataLink driver records a timeout if the DataLink device does not provide the '?' prompt within the user-configurable timeout period.
- Write-thru's are not supported. The reason for this is that the driver read a composite data set and the client side Map Descriptors do not contain enough information to format a write.

As a server:

- The Carrier DataLink driver is not capable of serving alarm data. (RA and AV polls produce an error response)
- The driver cannot set a discrete variable's state unless it is set a value of one or zero.
- The driver server does not understand write table or write variable requests where the requested state is a word like ON or OFF.
- The Carrier DataLink driver does not validate values or states, it simply stores the values.
- All variables are may be read and written without limitation. The driver has not implemented the Courier notion of *Force Levels*.
- There is no command buffer. One command can be processed at a time.
- The driver responds to all Read Table commands as if the /Y option was used. The driver does not support the /C or /N options.
- The Carrier DataLink driver does not support the control character commands CTRL-C/S/Q

General

- The Carrier DataLink driver is not capable of configuring the DataLink device. Software provided by the Carrier Corporation is required to do this. The DataLink device requires configuration, so that connects to the appropriate CCN devices on the CCN communications network.
- The driver cannot be configured to act as what the Carrier Corporation identify as an 'Alarm Acknowledger'.
- If the total length of the response from a read table query is more than 3000 bytes long, the driver will produce an error.

Appendix A. Advanced Topics

Appendix A.1. Table Names

The following list of table names is provided as a reference. Carrier may add tables and new devices may become available.

Equipment Type	Table Name
17/19EX CHILLER:	STATUS01
	STATUS02
	STATUS03
	STATUS04
	SETPOINT
	OCCDEFCS OCCPC01S
	OCCDEFCS OCCPC02S
	OCCDEFCS OCCPC03S
23XL CHILLER:	COMPRESS
	CVC_PSWD
	HEAT_EX
	ISM_STAT
	MAINSTAT
	POWER
	STARTUP
	SETPOINT
	OCCDEFCS OCCPC01S
	OCCDEFCS OCCPC02S
	OCCDEFCS OCCPC03S
30GTNHW CHILLER:	A_UNIT
	CIRCADIO
	CIRCA_AN
	OPTIONS
	SETPOINT
	OccDefcS OCCPC01S
30RA AQUASNAP CHILLER:	A_UNIT
	CIRCADIO
	CIRCA_AN
	CIRCBDIO (some units)
	CIRCB_AN (some units)
	OPTIONS
	SETPOINT
39N AIR HANDLER:	BASEUNIT
	DXCOOL
	ELECHEAT

Equipment Type	Table Name
	OPTIONS
	SETPOINT
	OCCDEFCS OCCPC01S
	OCCDEFCS OCCPC02S
48/50HG CENTURION ROOFTOP:	ECONOMZR
	GENERAL
	TSTAT
	SETPOINT
	OccDefcS OCCPC01S
AIR MANAGER AHU CONTROLLER:	DISPLAY
	SETPOINT
CHILLERVISOR:	BASESYS
	BYPASS
	POINTS1
	POINTS2
	SETPOINT
	OCCDEFCS OCCPC01S
COMFORTID VAV TERMINAL CONTROL:	POINTS
	SETPOINT
	OCCDEFCS OCCPC64S
COMFORTID FAN COIL CONTROLLER:	DISPLAY
	SETPOINT
	OCCDEFCS OCCPC64S
FLOTRONIC SYSTEM MANAGER:	FSMSTAT1
	FSMSTAT2
	FSMSTPT
	OCCDEFCS OCCPC01S
	OCCDEFCS OCCPC02S
50 VPAC:	HWP01-32
	HWP33-64
	SWP65-96
	SETPOINT SETPT01
	SETPOINT SETPT02
	SETPOINT

Equipment Type	Table Name
	SETPT03
	SETPOINT SETPT04
	SETPOINT SETPT05
	SETPOINT SETPT06
	SETPOINT SETPT07
	SETPOINT SETPT08
	OCCPCxxC OCCPC01
19XR PIC II CHILLER:	COMPRESS
	CVC_PSWD
	HEAT_EX
	ISM_STAT
	MAINSTAT
	POWER
	STARTUP
	SETPOINT
	OCCDEFCS OCCPC01S
	OCCDEFCS OCCPC02S
	OCCDEFCS OCCPC03S
30GTN CHILLER:	A_UNIT
	CIRCADIO
	CIRCA_AN
	CIRCBDIO
	CIRCB_AN
	OPTIONS
	SETPOINT
	OccDefcS OCCPC01S
30GXNHXA CHILLER:	A_UNIT
	CIRCADIO
	CIRCA_AN
	CIRCBDIO
	CIRCB_AN
	OPTIONS
	SETPOINT
	OccDefcS OCCPC01S
33CS VVT MONITOR:	ZONESTAT
	SETPOINT
	OccDefcS OCCPC01S
40UV UNIT VENTILATOR	DISPLAY
	SETPOINT
48/50F SERIES ROOFTOP:	STATUS01
	STATUS02
	STATUS03
	STATUS04
	STATUS05
	STATUS06
	SETPOINT
	OCCDEFCS OCCPC01S
	OCCDEFCS OCCPC02S
48/50M SERIES ROOFTOP:	STATUS01
	STATUS02
	STATUS03
	STATUS04
	STATUS05
	STATUS06
	SETPOINT
	OCCDEFCS OCCPC01S
	OCCDEFCS OCCPC02S
COMFORTID FAN TERMINAL CONTROL:	POINTS
	SETPOINT
	OCCDEFCS OCCPC64S
CONQUEST ROOFTOP:	CV_TSTAT
	DXCOOL
	STATUS01
	STATUS02
	SETPOINT
	OccDefcS OCCPC01S
PREMIERLINK ROOFTOP CONTROLLER:	CV_TSTAT
	STATUS01
	SETPOINT
	OCCDEFCS OCCPC64S
VVT BYPASS CONTROLLER:	BYPSSTAT
	SETPOINT
VVT SLAVE STAT:	ZONESTAT
	SETPOINT
	OccDefcS OCCPC01S
COMFORT CONTROLLER 6400/1600:	HWP01-32
	HWP33-64
	SWP65-96
	SETPOINT SETPT01
	OCCPCxxC OCCPC01

Appendix A.2. Using the Carrier Datalink Driver to Obtain Field Names

The Carrier Datalink driver can be used to obtain a list of variable names for a given table using the following Map Descriptors. It is recommended that these Map Descriptors are removed from the configuration after the variable names have been obtained as they will consume resource and processing time.

The following example illustrates a Map Descriptor which reads a table and dumps the response in ASCII format to a Data Array which can be browsed using RUINET (Refer to RuiNet Manual)

```
Data_Arrays
Data_Array_Name,  Data_Format,  Data_Array_Length
DA_DUMP       ,   BYTE       ,  2000
```

```
Map Descriptors
Map_Descriptor_Name,  Data_Array_Name,  Data_Array_Offset,  Function,  Node_Name,  Length,  Scan_Interval,  Table_Name  Field_Name,  Storet_As
Md1                ,  DA_DUMP       ,  0                ,  rdbc      ,  FAN01    ,  1000  ,  5.0s,          DISPLAY ,  EVERYTHING,  Ascii
```

The s=data array DA_DUMP has been defined with format BYTE. When viewing this table with RUINET display the array in 'STRING' format so that you can read it.

This is what tells the driver to dump the table, its variables names and variable descriptions in ASCII format.

This example illustrates a Map Descriptor which reads a table and dumps the response in ASCII format to the error log. The error log can be dumped to a file on your computer using RUIDEBUG (Refer to FieldServer Utilities Manual)

```
Data_Arrays
Data_Array_Name,    Data_Format,    Data_Array_Length
DA_DUMP        ,    BYTE        ,    2000
```

```
Map_Descriptor_Name,  Data_Array_Name,  Data_Array_Offset,  Function,  Node_Name,  Length,  Scan_Interval,  Table_Name,  Field_Name ,  Store_As
Md1                ,  DA_DUMP      ,  0                 ,  rdbc    ,  FAN01    ,  1000  ,  5.0s          ,  DISPLAY  ,  EVERYTHING,  AsciiLog
```

The following fragment from the error log is typical of the response to a Map Descriptor like the one above.

```
Ascii Dump Requested for MD=<Md1> Table=<DISPLAY>
DEV 02 DISPLAY
Desired Mode                  Off              MODE
Equipment Status           NORMAL              ALARM
Controlling Setpoint         75.0 dF           CSPT
Controlling Temperature     -40.0 dF           SPT
Space Temperature           -40.0 dF           RAT
Supply Air Temperature      -40.0 dF           SAT
Fan Mode                      Off              FANSTAT
Cooling Capacity              0.0 %            CCAP
Heating Capacity              0.0 %            HCAP
```

These are variable names. Use these names in your Map Descriptors.

Variable descriptions.

Appendix A.3. Map Descriptor Length Explained

The Length parameter is specified as part of the Map Descriptor.

Appendix A.3.1. Client Reads a Table:

The length determines the number of table variables whose values are to be stored when the response is received. If you do not know the length of the table in advance, set the length to a large number (e.g. 100). If the table contains more elements than the number defined in the length parameter, the additional data will be discarded.

Appendix A.3.2. Client reads a variable:

Response to a variable read takes many forms. The 'value' returned can be a number, a state (like ON), a time or some other string. To store a number or the time set the length to 1. When a state or string is returned, the driver looks it up in a table of state words. If it makes a match then it returns the value that has been associated with the state word, (e.g. ON = 1, OFF = 0). If it cannot make a match, it stores the string byte for byte in the data array. The number of bytes stored is determined by the length parameter. Thus, if the length is 1 and the string is 'INTERRUPTED' then the driver will store the 'I' in the data array. If the length was two, then the driver would use two consecutive locations and store 'I' and 'N'.

Appendix A.3.3. Client writes a variable:

Always set the length to 1

Appendix A.3.4. Server:

The length parameter is used when the server returns a value that is a string. The length is used to tell the driver how many characters to extract from consecutive array location to form the response string. When the server returns a value that is a number, state or time the length should always be 1.

Appendix A.4. How the Client stores the states/values of the Table Variables.

Appendix A.4.1. Discrete States

When a Carrier DataLink device reports the state of a table variable which has a discrete state, it reports the state as a keyword like on/off. This driver has to convert the keywords to numbers as most other remote devices are interested in the state as a numeric value of 1 or 0.

The driver is programmed to recognize certain keywords. A list is provided below. The driver is also capable of allowing the user to add to the table of predefined keywords by specifying additional information in the configuration file.

State Word	Value	State Word	Value
CLEAN	1	ALERT	2
DIRTY	0	EMSTOP	0
HEAT	1	ENABLE	1
COOL	0	Service	0
ON	1	OFF-local	1

State Word	Value	State Word	Value
OFF	0	OFF-CCN	2
OPEN	1	OFF-time	3
CLOSE	0	Emergency	4
CLOSED	0	ON-local	5
ABNORMAL	1	ON-CCN	6
NORMAL	0	ON-time	7
ALARM	1	Redline	1
STOP	0	Shed	2
START	1	DSABLE	0
YES	1	LOCAL	0
NO	0	REMOTE	1

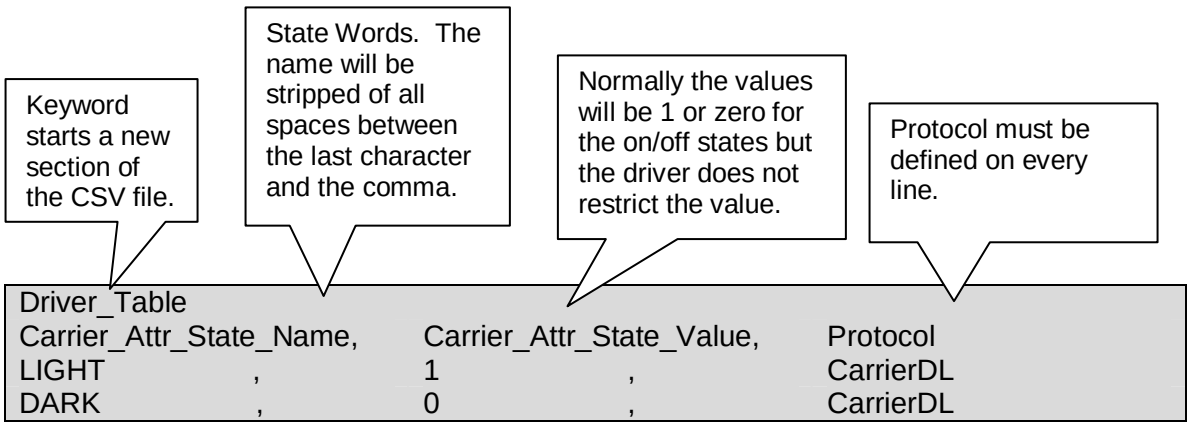
If the driver doesn't recognize the state word it stores the characters of the state word as decimal values based on their ASCII value. The number of characters stored is dependent on the length parameter.

For example, say the driver responds, reporting a variable to be a state 'INCREDIBLE'. If the length parameter of the polling map descriptor is 1 then the driver stores a value of 73 (An uppercase 'I' is the seventy third character in the ASCII alphabet.). If the length was 2 then the driver would also store the value 78 in the next array. The driver will store a maximum of 100 characters. The driver recognizes discrete state words by checking the 1st character of the value field. If it is a non-digit then it is regarded as a state word.

Appendix A.4.2. Extending the List of Discrete State Words

You can extend the list of state words the driver recognizes by modifying the configuration file.

The following example adds three state words. If a device reports the state of the variable as LIGHT then the driver will store the value of 1, if on the other hand the state is reported as DARK then the driver will store the value zero.



You can use this method to change the values of any of the driver's default state words by duplicating the word and specifying a new value in the configuration file.

Appendix A.4.3. Time Values

If the driver receives a variable value reported in the format hh:mm then the driver stores a number obtained by multiplying the hours by 60 and adding the minutes. The driver recognizes a time value by checking the 3rd character for a colon and checking that the 1st character is a digit.

Value_Stored = hh * 60 + mm

Eg. 5:30pm is reported as 17:20 and is stored as 17*60 + 20 = 1040.

Appendix A.4.4. Numeric Values

The driver recognizes numeric values by checking the first character of the value field. If it is a digit then the field is treated as a number.

Appendix A.4.5. Occupancy Strings / Values

If the value returned for a variable is 8 characters long and each of the characters is a one or a zero then the driver regards this as an occupancy string and converts it to a binary coded decimal value and stores this value.

Eg: 00101010 = 42 decimal

Appendix A.5. Timing Considerations

Reading a whole table can take several seconds depending on the table length. In addition the response from the Carrier device can have small pauses after each line of response. Thus, if you use the default timeout and IC_Timeout settings the communications may fail when either one of the pauses exceeds the IC_Timeout setting or when the whole response hasn't been received by the time the timeout setting has elapsed.

For this reason it is recommended that the timeout be set to a value between 15 and 30 seconds and the IC_Timeout be set to a value between 7 and 15 seconds. Before setting either of these to the maximum consider that the in the event of a real communications failure, this set amount of time must elapse before the driver recognizes the timeout and reports the node offline. Thus, it is best to start with a smaller value and increase it until reliable communications have been achieved with the timeout settings.

The timeout can be set for the Map Descriptor, the node or the connection. This is more fully explained in the FieldServer Configuration Manual. In summary, if you wish to apply a single timeout to all messages to a device then set the timeout for the connection. If you wish to override the connection timeout settings for a particular node the set the timeout for that node too and if you wish to override the settings for a particular Map Descriptor then set the timeout for that Map Descriptor.

The example below illustrates how to set the timeout parameters at the connection level.

```
// Client Side Connections

Connections
Port,   Baud,   Parity,  Data_bits,  Stop_Bits,  Protocol ,  Handshaking,  Poll_Delay,  Timeou,t,  IC_Timeout
P8,     9600,   None,    8       ,   1       ,   CarrierDL,  None      ,   0.100s   ,   30.0s  ,   15.0s
```

Appendix B. Carrier DataLink Device Error Response

The following notes are provided from the protocol specification.

Message	Explanation	Action
#1 Invalid Command	The last command sent to the DataLink device is not recognized.	Check the spelling of the command and verify that it contains the required parameters.
#2 Invalid or Non-Existent Table	The specified table does not exist in the specified element.	Check the spelling of the table name(s). If using a WV or RV command; verify that the table contains the specified variables.
#3 Level II Communications Error	A Communication error has occurred due to other activity on the CCN communication bus, a system element failure, or an incorrect address. The command will not be executed.	Verify the COMM2 connector is not disconnected and retype the command.
#6 Device Not Configured	The specified system element's bus and the element number were not found in the DTP-CONFG table.	Verify that the correct bus and element number have been entered in the DTP-CONFG table corresponding to the device number.
#7 Variable Does Not Exist	The variable does not exist in the specified table.	Check the spelling of the variable name and verify that the CCN variable exists in the specified system element.
#8 Invalid Data	The data was in an incorrect format.	Verify the format of the data.
#9 Access Restricted	An attempt was made to read or write to a table or variable type that is not supported by the DataLink device.	Check variable.
#10 Limits Exceeded	The value that was typed is outside the specified variable's upper or lower limits	Verify the limits and adjust the force value.
#11 Alarms Not Available	There are no alarms currently in the alarm buffer.	Reissue the command at later time or adjust the alarm priority level.
#12 Cannot Force Variable	The CCN system element does not allow write access to the specified variable.	Choose different variable.
#13 Parameter Not Found	The 8-character point (variable) name used in the RT or WT command does not exist.	Check the spelling of the point (variable) name and verify that the point name exists in the specified system element.
#14 Level II Communications NACK	The command was accepted but not understood by the specified system element, an attempt was made to write to or read a variable with a higher force level, or an attempt was made to write or read from a UT203 controller.	Retype the command after checking the force priority and after verifying that the correct bus and element number is entered in the DTP-CONFG table.

Appendix C. Driver Notes

Appendix C.1. Driver Stats

The driver reports statistics according to the FieldServer standards. The following notes describe some aspects of standard statistic reporting which are peculiar to this driver.

All errors responses from the Carrier DataLink device are recorded as a PROCOL ERROR,

In addition to the standard FieldServer communication statistics described above and in the FieldServer User's Manual, this 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 "Carrier-stats".

The following example shows how this special Map Descriptor can be configured. You can copy this section of text directly into your CSV file.

```
Nodes
Node_Name  ,                    Station ,                    Protocol
Carr_stats ,                    1       ,                    CarrierDL

Data_Arrays
Data_Array_Name  ,              Data_Format,                 Data_Array_Length
DA_CARRIER_STATS ,              UINT32     ,                 2000

Map_Descriptors
Map_Descriptor_Name,  Data_Array_Name ,  Data_Array_Offset  Function  Node_Name ,  Length
Carrier-Stats,        DA_CARRIER_STATS   0,                 passive,   Carr_stats,  500
```

When the driver sees this Map Descriptor it uses the data array DA_CARRIER_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. The location in the data array is obtained by multiplying the port number by 50 and then using the location offset indicated in the table below.

OFFSET	DESCRIPTION
0	Number of Read Table messages sent
1	Number of Read Variable messages sent
2	Number of Write Table messages sent
3	Number of Write Var Messages sent
4	Number of bytes sent by client driver
5	Number of messages sent by client
6	Number of response messages received by client
7	Number of times client tries write with alternate format because original method failed
9	Most recent response error
8	Number of times client receives an error response
10	Carrier Error: #1 Invalid Command"
11	Carrier Error: #2 Invalid or Non-Existent Table"
12	Carrier Error: #3 Level II communications Error"
13	Carrier Error: #4 Error not defined"
14	Carrier Error: #5 Error not defined"
15	Carrier Error: #6 Device Not Configured"
16	Carrier Error: #7 Variable Does not Exist"
17	Carrier Error: #8 Invalid Data"
18	Carrier Error: #9 Access Restricted"
19	Carrier Error: #10 Limits Exceeded"
20	Carrier Error: #11 Alarms not Available"
21	Carrier Error: #12 Cannot Force Variable"
22	Carrier Error: #13 Parameter Not Found"
23	Carrier Error: #14 Level II Communications NACK"
24	Some other error
25	Number of response bytes received by client
26	Number of times client has timeout out waiting for (response) prompt
27	When set then the server sends response whose var names have leading spaces. This is used for QAA testing and diagnostic purposes only. Customers should never set this value.
28	Number of times client cannot parse / recognize the value sting in a response
29	Number of times the server had to store a time greater than 24:00
30	Number of times the client sent a message with a time greater than 24:00
31	Number of times the client sent a message to a probable occupancy table without the correct data types

Appendix D. Driver Error Messages

The driver reports information and errors in the form of messages printed to the error log. Those messages marked with a * are only printed once even if they occur repeatedly.

Error Message	Explanation
Carrier:#1 FYI. The MapDesc called <%s> is too short	The length of the Map Descriptor used to expose driver statistics is too short. Set to at least 1000. This message may be ignored[§].
Carrier:#2 FYI. You could have used a MapDesc called <%s> to expose diagnostic info.	This message is a prompt and may be ignored. Read Appendix C.1 for more information.
Carrier:#3 Err. Illegal Node_ID=%d Valid=1..15	Valid node numbers are in the range 1 to 15 inclusive. Read sections 4.3 or 5.3 for more information. [§]
Carrier:#4 FYI. Address has no meaning. Best set to 1 MapDesc=<%s>	You can ignore this message. The address parameter has no meaning in the configuration of this driver. Remove the parameter from the configuration or set its value to 1[§]
Carrier:#5 Err. For write, set length to 1. MapDesc=<%s>	
Carrier:#6 FYI. Field Name Blank. Assumed 'EVERYTHING'. MapDesc=<%s>	If the Field_Name parameter is left unspecified then the driver assumes you intend reading the whole table by filling in the Field_Name with the keyword EVERYTHING. Suppress this message by specifying the Field_Name .[§]
Carrier:#7 Err. Table name required. MapDesc=<%s>	Every Map Descriptor for this driver requires that you specify a table name. Section 4.4.2 and Appendix A.1 provide additional information. [§]
Carrier:#8 Err. Length required. MapDesc=<%s>	The length parameter must be set in the configuration file to a value greater than zero. Refer to Appendix A.3.[§]
Carr:#9 FYI. Duplicate state=<%s>. Value has been updated from=%d to=%d	You have specified a discrete state word in the configuration file which is a duplicate of one already in the list. The driver uses the new value, thus changing the values for the driver's default discrete state words. You can ignore this message.
Carr:#10 Err. No space. Driver rejects value state=<%s> value=%d	The driver has limited space to store discrete state keywords added in the configuration file. The maximum is 150 words including the driver's defaults. Remove some of the keywords you have added to the configuration file. [§]
Carr:#11 FYI. User added value state=<%s> value=%d	You can ignore this message; it is for information only. Each time a new discrete state word is added to the driver from the configuration file, the driver reports the new word and its value.
Carr:#12 Err. Length too short to store all. MD=<%s>	The read table command resulted in more variables being returned than you have reserved space for (with the length parameter). Increase the length parameter[§].
Carrier:#13 FYI. Diagnostic send error #1 response.	These messages are for FieldServer engineers. If printed in the error log please call FieldServer support and report the message.
Carrier:#14 FYI. Diagnostic cancelled slave response	

[§] Correct the error by editing the configuration CSV file, downloading the corrected file to the FieldServer and then resetting the FieldServer.

Error Message	Explanation
Carrier:#15 Err. Alarm commands not supported.	The server does not support alarm functions. Polls beginning AV or RA produce these errors. Re-configure your client not to request this information. [§]
Carrier:#16 Err. Alarm commands not supported.	
Carr:#17 Err. MD=<%s> discrete state word not recognized.	On the line immediately following this error the driver reports the response that generated the error. The driver will store a value that is the ASCII code for the first character of the unrecognized word. Add a new discrete state word to the .CSV file as described in Appendix A.4. [§]
CarrDL:#18 FYI. Timeout probably too short. Read Manual.	This message may be ignored if your communications are operating reliably. Read Appendix A.5 for additional information on timing considerations. The message is printed when the driver detects that the configuration requires that a whole table be read and the timeout value is set below the recommended minimum of 15s for this operation.
CarrDL:#19 Err. Diagnostic. Call Support.	This error message should only be produced by FST's QA testing procedure. If you see this error, call Tech Support after taking a log. Instructions for taking a log may be found in the Trouble Shooting Guide.
CarrDL:#20* FYI. Invalid Time(%02d:%02d) being written. MD=%s	The message is printed once and suppressed for subsequent occurrences. The message is printed when writing a time to a Carrier Device which is invalid as it is greater than 23:59. Check the value in the Data Array being used for the write. It's possible an upstream device sent an invalid value. The driver sends the message with the invalid time and it's up to the Carrier device to reject the setting.
CarrDL:#21* FYI. Invalid Time(%02d:%02d) being stored. MD=%s	The message is printed once and suppressed for subsequent occurrences. The message is printed when the Server is required to store an invalid time (>23.59). Check the value in the Data Array being used for the write. The data is stored despite the warning.
CarrDL:#23* Err. CarrDL device reported errors. Check exposed stats 8-23.	These errors occur when communications are operating correctly but the CarrDL device cannot respond to the poll. The reason error is reported in the driver stats (See Appendix C.1) Stat #9, reports the error number of the most recent error reported. The message is printed once and then suppressed. These errors are most commonly produced when a table name/ variable name does not exist or is mis-spelled
CarrDL:#22* FYI. Read notes for #22 in Manual. MD=%s	DOW and times require that the Data_Type be specified when writing to enable the Driver to convert the value extracted from the Data Array for formatting in the write message. This error is printed if the table name or the field name contain the sub-string "OCC", the function is a write, and a Data_Type has not been specified. In most cases the write will be rejected by the Carrier device or the value may not be what you expect. Refer to Section 4.4.7 for more information.

[§] Correct the error by editing the configuration CSV file, downloading the corrected file to the FieldServer and then resetting the FieldServer.

THIS PAGE INTENTIONALLY LEFT BLANK

FieldServer Technologies 1991 Tarob Court Milpitas, California 95035 USA **Web**:www.fieldserver.com
Tel: (408) 262-2299 **Fax**: (408) 262-9042 **Toll_Free**: 888-509-1970 **email**: support@fieldserver.com