

# **LOGGERNET USER'S MANUAL**

**Version 1.2**

**REVISION: 4/02**

**COPYRIGHT (c) 1999-2002 CAMPBELL SCIENTIFIC, INC.**



# ***License for Use***

---

This software is protected by both United States copyright law and international copyright treaty provisions. The installation and use of this software constitutes an agreement to abide by the provisions of this license agreement.

You may copy this software onto a computer to be used and you may make archival copies of the software for the sole purpose of backing-up CAMPBELL SCIENTIFIC, INC. software and protecting your investment from loss. This software may not be sold, included or redistributed in any other software, or altered in any way without prior written permission from Campbell Scientific. All copyright notices and labeling must be left intact.

This software may be used by any number of people, and may be freely moved from one computer location to another, so long as there is not a possibility of it being used at one location while it's being used at another. The software, under the terms of this license, cannot be used by two different people in two different places at the same time.



# ***Limited Warranty***

---

CAMPBELL SCIENTIFIC, INC. warrants that the installation media on which the accompanying computer software is recorded and the documentation provided with it are free from physical defects in materials and workmanship under normal use. CAMPBELL SCIENTIFIC, INC. warrants that the computer software itself will perform substantially in accordance with the specifications set forth in the instruction manual published by CAMPBELL SCIENTIFIC, INC. CAMPBELL SCIENTIFIC, INC. warrants that the software is compatible with computers running Microsoft Windows NT and 2000.

CAMPBELL SCIENTIFIC, INC. will either replace or correct any software that does not perform substantially according to the specifications set forth in the instruction manual with a corrected copy of the software or corrective code. In the case of significant error in the installation media or documentation, CAMPBELL SCIENTIFIC, INC. will correct errors without charge by providing new media, addenda or substitute pages.

If CAMPBELL SCIENTIFIC, INC. is unable to replace defective media or documentation, or if CAMPBELL SCIENTIFIC, INC. is unable to provide corrected software or corrected documentation within a reasonable time, CAMPBELL SCIENTIFIC, INC. will either replace the software with a functionally similar program or refund the purchase price paid for the software.

The above warranties are made for ninety (90) days from the date of original shipment.

CAMPBELL SCIENTIFIC, INC. does not warrant that the software will meet licensee's requirements or that the software or documentation are error free or that the operation of the software will be uninterrupted. The warranty does not cover any diskette or documentation that has been damaged or abused. The software warranty does not cover any software that has been altered or changed in any way by anyone other than CAMPBELL SCIENTIFIC, INC. CAMPBELL SCIENTIFIC, INC. is not responsible for problems caused by computer hardware, computer operating systems or the use of CAMPBELL SCIENTIFIC, INC.'s software with non-CAMPBELL SCIENTIFIC, INC. software.

ALL WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED AND EXCLUDED. CAMPBELL SCIENTIFIC, INC. SHALL NOT IN ANY CASE BE LIABLE FOR SPECIAL, INCIDENTAL, CONSEQUENTIAL, INDIRECT, OR OTHER SIMILAR DAMAGES EVEN IF CAMPBELL SCIENTIFIC HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

CAMPBELL SCIENTIFIC, INC. IS NOT RESPONSIBLE FOR ANY COSTS INCURRED AS A RESULT OF LOST PROFITS OR REVENUE, LOSS OF USE OF THE SOFTWARE, LOSS OF DATA, COST OF RE-CREATING LOST DATA, THE COST OF ANY SUBSTITUTE PROGRAM, CLAIMS BY ANY PARTY OTHER THAN LICENSEE, OR FOR OTHER SIMILAR COSTS.

LICENSEE'S SOLE AND EXCLUSIVE REMEDY IS SET FORTH IN THIS LIMITED WARRANTY. CAMPBELL SCIENTIFIC, INC.'S AGGREGATE LIABILITY ARISING FROM OR RELATING TO THIS AGREEMENT OR THE SOFTWARE OR DOCUMENTATION (REGARDLESS OF THE FORM OF ACTION; E.G., CONTRACT, TORT, COMPUTER MALPRACTICE, FRAUD AND/OR OTHERWISE) IS LIMITED TO THE PURCHASE PRICE PAID BY THE LICENSEE.



## **CAMPBELL SCIENTIFIC, INC.**

---

815 W. 1800 N.  
Logan, UT 84321-1784  
USA  
Phone (435) 753-2342  
FAX (435) 750-9540  
[www.campbellsci.com](http://www.campbellsci.com)

Campbell Scientific Canada Corp.  
11564 -149th Street  
Edmonton, Alberta T5M 1W7  
CANADA  
Phone (780) 454-2505  
FAX (780) 454-2655

Campbell Scientific Ltd.  
Campbell Park  
80 Hathern Road  
Shepshed, Loughborough  
LE12 9GX, U.K.  
Phone +44 (0) 1509 601141  
FAX +44 (0) 1509 601091

# ***LoggerNet Table of Contents***

---

<b>1. Introduction.....</b>	<b>1-1</b>
<b>2. System Requirements .....</b>	<b>2-1</b>
2.1 Hardware and Software .....	2-1
2.2 Configuration of TCP/IP Services .....	2-1
<b>3. Installation.....</b>	<b>3-1</b>
3.1 CD-ROM Installation .....	3-1
<b>4. The LoggerNet Server and the LoggerNet     Clients .....</b>	<b>4-1</b>
4.1 What is Meant by Client and Server .....	4-1
4.2 LoggerNet Communication Server .....	4-1
4.3 Clients .....	4-2
4.3.1 NetAdmin.....	4-2
4.3.2 Communication Status Monitor .....	4-3
4.3.3 Control Panel .....	4-3
4.3.4 Baler.....	4-3
4.3.5 Socket Data Export .....	4-4
4.3.6 Security .....	4-4
4.3.7 Hole Monitor.....	4-4
4.3.8 Edlog.....	4-4
4.3.9 CRBasic Editor .....	4-4
4.3.10 Tools .....	4-5
4.4 Getting Help for LoggerNet Applications .....	4-5
4.5 LoggerNet Operations and Backup Procedures .....	4-5
4.5.1 Backing up Data.....	4-5
4.5.2 Loss of Computer Power.....	4-7
4.5.3 Program Crashes .....	4-8
4.5.4 Restoring from Backup .....	4-8
4.5.5 Computer System Security.....	4-9
4.5.6 Directory and File Descriptions .....	4-9
<b>5. Network Administration.....</b>	<b>5-1</b>
5.1 Setting Up a Datalogger Network.....	5-1
5.1.1 Adding Devices to the Network.....	5-3
5.1.2 Renaming Network Devices .....	5-4
5.2 Device Configuration Settings.....	5-5
5.2.1 Serial Port.....	5-5
5.2.2 Internet Serial Port .....	5-5
5.2.3 Datalogger.....	5-6
5.2.4 RF Base .....	5-15
5.2.5 RF Modem .....	5-17
5.2.6 MD9 Base .....	5-18
5.2.7 MD9 Modem.....	5-19
5.2.8 Phone Modem .....	5-20
5.2.9 Remote Phone Modem.....	5-21

5.3	Setting the Clock .....	5-22
5.4	Setting Up Scheduled Data Collection .....	5-23
5.4.1	Data Collection Scheduling Considerations .....	5-23
5.4.2	Setting Up Scheduled Data Collection .....	5-25
5.5	Data Management .....	5-26
5.5.1	CSILoggerNet Data Cache .....	5-26
5.5.2	Collection of Data Holes .....	5-27
5.6	Communications Testing .....	5-27
5.6.1	Network Communications Test .....	5-27
5.6.2	RF Communications Test .....	5-28
5.7	Server Settings .....	5-30

## **6. Communication Status Monitor ..... 6-1**

6.1	Graphic Displays .....	6-1
6.1.1	Status Icon .....	6-1
6.1.2	Graphical Status History .....	6-2
6.2	Custom Status Monitoring .....	6-2
6.3	Obtaining Datalogger Status Information .....	6-4
6.4	Monitoring Operational Logs .....	6-4
6.4.1	Transaction Log .....	6-5
6.4.2	Communication Log .....	6-5
6.4.3	Object State Log .....	6-5
6.5	Monitoring Low Level I/O .....	6-6
6.6	Reset Statistics .....	6-6

## **7. Control Panel..... 7-1**

7.1	Selecting the Datalogger .....	7-1
7.2	Program Management .....	7-2
7.3	Manually Checking and Setting the Clock .....	7-2
7.4	Control Panel Options .....	7-3
7.4.1	Connect to Station – Enable Connection Management .....	7-3
7.4.2	Settings Override .....	7-4
7.4.3	Table Definitions .....	7-5
7.4.4	Retrieve Cached Program .....	7-5
7.4.5	Terminal Emulation .....	7-5
7.5	Monitoring Data Collected from the Datalogger .....	7-6
7.6	Graphing Data Collected from the Datalogger .....	7-8
7.6.1	Graph Setup .....	7-9
7.6.2	Scale Factor .....	7-11
7.7	Saving Collected Data to a Data File .....	7-11
7.7.1	Update Data Cache .....	7-11
7.7.2	Retrieval Options .....	7-12
7.7.3	File Formats .....	7-13
7.7.4	File Names and File Save Modes .....	7-14
7.8	File Management for CR5000 and CR9000 (File Control Tab) .....	7-14

## **8. Baler – Timed Save to Data File ..... 8-1**

8.1	Baler Settings .....	8-1
8.2	File Naming and Directory Structure .....	8-3
8.3	Number of Files to Keep .....	8-3
8.4	Baler Operation .....	8-3
8.4.1	Baler Controls .....	8-3
8.4.2	Table Status and Messages .....	8-4
8.4.3	Automatic Start .....	8-4



8.4.4 Command Line Options .....	8-4
8.5 Advanced Settings .....	8-5
8.5.1 Colletion Order .....	8-5
8.5.2 Select the Baling Starting Point .....	8-5
8.5.3 Historical Data Storage Selection .....	8-5
8.5.4 Launching a Program .....	8-6
8.6 Importing Files into Other Applications .....	8-6
<b>9. Socket Data Export .....</b>	<b>9-1</b>
9.1 Functional Overview.....	9-1
9.2 Theory of Operation.....	9-2
9.3 Custom Data Retrieval Client .....	9-3
9.4 Custom Client/Socket Export Interface Description .....	9-3
9.5 RTMS Format Description .....	9-7
9.6 Standard Format Description .....	9-8
<b>10. Security.....</b>	<b>10-1</b>
10.1 Security Setup and Operation .....	10-1
10.1.1 Group Permissions .....	10-2
10.1.2 Group Members .....	10-4
10.1.3 Enabling Security .....	10-5
10.2 Resetting Security .....	10-5
<b>11. Hole Monitor .....</b>	<b>11-1</b>
11.1 Tracking Data Hole Collection .....	11-1
11.2 Refresh the List of Holes .....	11-2
<b>12. Datalogger Program Creation with Edlog .....</b>	<b>12-1</b>
12.1 Overview.....	12-1
12.1.1 Creating a New Edlog Program .....	12-2
12.1.2 Editing an Existing Program .....	12-12
12.1.3 Library Files.....	12-14
12.1.4 Documenting a DLD File.....	12-14
12.1.5 Display Options.....	12-14
12.2 Input Locations .....	12-16
12.2.1 Entering Input Locations.....	12-16
12.2.2 Repetitions .....	12-17
12.2.3 Input Location Editor .....	12-17
12.2.4 Input Location Anomalies.....	12-19
12.3 Final Storage Labels .....	12-20
<b>13. Datalogger Program Creation with CRBasic Editor .....</b>	<b>13-1</b>
13.1 Overview.....	13-1
13.1.1 Inserting Instructions.....	13-2
13.1.2 Parameter Dialog Box .....	13-3
13.1.3 Right Click Functionality .....	13-4
13.1.4 Toolbar.....	13-5
13.1.5 Compile.....	13-6
13.1.6 Templates .....	13-7
13.1.7 CRBasic Editor Options .....	13-7
13.1.8 Available Help Information .....	13-11

13.2 CRBasic Programming.....	13-12
13.2.1 Programming Sequence.....	13-12
13.2.2 Program Declarations .....	13-13
13.2.3 Mathematical Expressions.....	13-13
13.2.4 Measurement and Output Processing Instructions .....	13-14
13.2.5 Inserting Comments Into Program .....	13-15
13.3 Example Program.....	13-16
13.3.1 Data Tables.....	13-16
13.3.2 The Scan — Measurement Timing and Processing.....	13-18
13.4 Numerical Entries .....	13-19
13.5 Logical Expression Evaluation.....	13-19
13.5.1 What is True? .....	13-19
13.5.2 Expression Evaluation .....	13-20
13.5.3 Numeric Results of Expression Evaluation .....	13-20
13.6 Flags.....	13-21
13.7 Parameter Types.....	13-21
13.7.1 Expressions in Parameters.....	13-21
13.7.2 Arrays of Multipliers and Offsets for Sensor Calibration.....	13-22
13.8 Program Access to Data Tables .....	13-22
<b>14. Real-Time Monitor and Control.....</b>	<b>14-1</b>
14.1 Overview.....	14-1
14.2 Development Mode.....	14-1
14.2.1 The RTMC Workspace.....	14-2
14.2.2 Display Components .....	14-3
14.2.3 RTMC Operations .....	14-5
14.2.4 Expressions.....	14-8
14.2.5 Remote Connections.....	14-11
14.3 Run-Time .....	14-11
<b>15. Troubleshooting Guide.....</b>	<b>15-1</b>
15.1 LoggerNet Server Problems .....	15-1
15.1.1 Running and Connecting to the Server.....	15-1
15.1.2 Data Collection Issues .....	15-2
15.2 Client Application Problems .....	15-2
15.3 General Communication Link Problems.....	15-3
15.4 RF Communication Link Issues.....	15-3
15.4.1 Checking RF Components and Connections .....	15-3
15.4.2 RF Signal Strength Testing .....	15-4
15.4.3 Troubleshooting with Attenuation Pads .....	15-5
15.5 Ping .....	15-8
15.6 Telnet .....	15-9
15.7 Using Data Table Monitor .....	15-10
<b>16. Implementing Advanced Communications Links.....</b>	<b>16-1</b>
16.1 Table-based Dataloggers via RF .....	16-1
16.1.1 Setup.....	16-1
16.1.2 Data Collection Options .....	16-3
16.1.3 Operational Considerations .....	16-6
16.1.4 Special Considerations .....	16-9
16.1.5 Error Messages.....	16-9
16.1.6 RF Communications Test.....	16-10

16.2 Phone to RF .....	16-11
16.2.1 Setup .....	16-11
16.2.2 Operational Considerations .....	16-12
16.3 Phone to MD9 .....	16-12
16.3.1 Setup .....	16-12
16.3.2 Operational Considerations .....	16-13
16.4 TCP/IP to RF .....	16-13
16.4.1 Setup .....	16-14
16.4.2 Operational Considerations .....	16-14
16.4.3 Special Considerations .....	16-15

## Appendices

### A. Glossary of Terms ..... A-1

### B. Table-Based Dataloggers ..... B-1

B.1 Memory Allocation for Final Storage .....	B-1
B.1.1 CR10X-TD Family Table-Based Dataloggers .....	B-1
B.1.2 CR5000/CR9000 Memory Programs and Data Storage .....	B-2
B.2 Converting an Array-Based Program to a CR10X-TD Table-Based Program using Edlog .....	B-3
B.2.1 Steps for Program Conversion .....	B-3
B.2.2 Program Instruction Changes .....	B-4
B.3 Table Data Overview .....	B-5
B.4 Default Tables .....	B-6

### C. Software Organization ..... C-1

C.1 LoggerNet/Client Architecture .....	C-1
C.2 LoggerNet Server Data Cache .....	C-1
C.2.1 Organization .....	C-1
C.2.2 Operation .....	C-3
C.2.3 Retrieving Data from the Cache .....	C-3
C.2.4 Updating Table Definitions .....	C-4
C.3 Directory Organization .....	C-4
C.3.1 C:\CampbellSci\LoggerNet Directory .....	C-5
C.3.2 C:\Program Files\CampbellSci\LoggerNet Directory .....	C-6

### D. Log Files ..... D-1

D.1 Event Logging .....	D-1
D.1.1 Log Categories .....	D-1
D.1.2 Enabling Log Files .....	D-1
D.1.3 Log File Message Formats .....	D-2

### E. Using RTDM with LoggerNet ..... E-1

E.1 RTDM .....	E-1
E.2 Setting up RTDM .....	E-2

### F. CoraScript ..... F-1

F.1 CoraScript Fundamentals .....	F-1
F.2 Useful CoraScript Operations .....	F-2

F.2.1 Connecting to the LoggerNet Server.....	F-2
F.2.2 Checking and Setting Device Settings .....	F-2
F.2.3 Creating and using a Network Backup Script .....	F-3
F.2.4 Hole Management .....	F-3
F.2.5 Scripting CoraScript Commands.....	F-3

## **G. Importing Files into Excel.....G-1**

### **Figures**

9-1 Client State Diagram.....	9-4
9-2 Socket Data Export Server State Diagram.....	9-5

### **Tables**

5-1 Example of Status Table Entries (CR10T) .....	5-10
5-2 CR5000 Status Table Entries .....	5-10
12.1-1 Operators and Functions .....	12-8
12.1-2 Editor Keystrokes .....	12-13
13.3-1 Formats for Output Data .....	13-18
13.4-1 Formats for Entering Numbers in CRBasic .....	13-19
13.5-1 Synonyms for True and False .....	13-20
13.7-1 Rules for Names .....	13-21
B-1 Example of Status Table Entries (CR10T).....	B-7
B-2 CR5000 Status Table Entries .....	B-8

# Section 1. Introduction

---

LoggerNet is a software application which enables users to set up, configure, and retrieve data from a network of Campbell Scientific table-based dataloggers and share this data over an Ethernet communications network. This software application is designed to run under Microsoft Windows 95, 98, ME, Windows NT version 4.0, and Windows 2000.

The heart of the LoggerNet software is an application known as the communication server. All of the communication with the dataloggers, programming, variable read and set, and data retrieval, is managed through the communication server.

The LoggerNet communication server also maintains a data cache of the data collected from the dataloggers in the network. The data in the cache is available to multiple clients simultaneously for viewing, analysis, and archival of the collected data.

The client applications provide a set of tools for the user to work with the datalogger network and retrieve the data. These applications include network configuration, network communication monitoring, datalogger configuration, data retrieval to file and data export to other computer applications. Each of these client applications is explained in subsequent sections of this manual.

One significant benefit of the software design is that the clients do not have to be run on the same computer as the communication server. The clients can be run on any computer that is connected to the main computer by an Ethernet computer network connection. Some of examples of these networks are Local Area Network (LAN), Wide Area Network (WAN), or the Internet.

---

**NOTE**

The LoggerNet communication server must be running for scheduled data collection to occur and for use of the client applications.

---

Security capabilities have been built into the communication server such that the datalogger network administrator can determine the level of access for specific groups or users. In this way the administrator can determine who has the ability to change the setup of the network, configure the dataloggers, or even access the data.

LoggerNet is an ideal solution for users desiring a reliable data collection system which is also flexible enough to meet the needs of a variety of users.



## Section 2. System Requirements

---

### 2.1 Hardware and Software

The LoggerNet server and clients are 32-bit programs designed to run on Intel based computers running Microsoft Windows Operating systems. The recommended computer configuration for running the LoggerNet communication server is Windows NT or Windows 2000. The client applications can run on Windows 95, Windows 98, Windows NT, or Windows 2000. All installations require a Pentium or equivalent processor, with 45 MB free space on the hard disk, and TCP/IP support installed.

Access to the LoggerNet communication server is accomplished through a TCP/IP connection. Therefore, TCP/IP services must be installed on the communication server computer, even if the applications will be run locally.

### 2.2 Configuration of TCP/IP Services

TCP/IP services must be running on the computer for the software to work, even if LoggerNet and the clients will be running on a local computer. Following are the procedures for enabling TCP/IP communication on a Windows 95, 98, or NT system. For Windows 2000 the same things need to be set up, but they are accessed in different ways. See the documentation and help for Windows 2000 to add a dial-up connection and associate it with TCP/IP.

---

**NOTE**

**Before beginning this procedure make sure that you have your Windows installation CDROM or floppy disks as appropriate for your computer.**

---

As you install these options you will be prompted to insert various disks or the CDROM to complete the installation.

1. Click on the Start button and select Settings | Control Panel
2. When the Control Panel window comes up double click on the Add/Remove Programs icon.
3. Select the Windows Setup tab.
4. Select “Communications” and click on the “Details” button.
5. On the Communications options screen click the box by “Dial-Up Networking” (Win 98/95) or “Phone Dialer” (NT)
6. Click OK on the Communications options screen and on the Windows setup screen.

7. Provide the Windows installation software as prompted and then follow the directions.
8. When you are prompted to reboot the computer choose Yes.
9. After the computer boots, go to the Windows control panel and double click on the “Network” icon.
10. In the list box on the “Configuration” tab (Win95/98) or “Protocols” tab (NT) of the “Network” window which comes up, see if there is an entry “TCP/IP -> Dial-Up Adapter” or “TCP/IP protocol”. If this entry exists, skip the next steps.
11. Click on the “Add” button. In the “Select Network Component Type” window which comes up select “Protocol” or “TCP/IP protocol” and click on the “Add” or “OK” button.
12. When the “Select Network Protocol” window comes up select “Microsoft” under “Manufacturers:”, and “TCP/IP” under “Network Protocols:”. Click OK.



## ***Section 3. Installation***

---

### **3.1 CD-ROM Installation**

The following instructions assume that drive D: is a CD-ROM drive on the computer from which the software is being installed. If the drive letter is different, substitute the appropriate drive letter.

1. From the Windows system menu, select Start | Run.
2. Type D:\Disk1\Setup.exe in the Open field or use the Browse button to access the CD-ROM drive and select the setup executable in the Disk1 folder.
3. This activates the LoggerNet Installation Utility. Follow the prompts on the screen to complete the installation.

Items are added to your computer's start menu under Programs | LoggerNet that start the communication server and each of the individual clients. If the default directories are used, LoggerNet executable files and help files are placed in the C:\Program Files\CampbellSci\LoggerNet directory. The directory C:\CampbellSci\LoggerNet is a working directory and contains initialization files, log files and other files used by LoggerNet for operation (including data collection files from the dataloggers).



# Section 4. The LoggerNet Server and the LoggerNet Clients

---

*This section provides an overview on the LoggerNet server and the LoggerNet clients. Each of the clients is explained in detail in subsequent sections.*

## 4.1 What is Meant by Client and Server

Throughout this manual and the on-line documentation, reference is often made to the LoggerNet "clients" and the LoggerNet "communication server" (or LoggerNet server).

The LoggerNet communication server is the application that provides the software interface between the client applications and the datalogger network. It may be run on a computer that is solely dedicated to the task of retrieving and storing data from a network of dataloggers, or it may be a computer that is used for multiple tasks. **The LoggerNet server software must be running on the server computer for scheduled data collection to occur, and to allow any of the client applications to be used to access information from the dataloggers.**

The terms client and server are derived from computer software terminology. The server is taken to mean a software program which provides a connection for other programs to get data or other information from the server program. The client program attaches to the server program to receive the data 'served' by the server program.

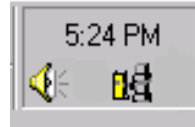
The clients in this system are the individual applications in the LoggerNet software package. The clients include NetAdmin, CommStatus, Control Panel, Baler, Socket Data Export, Hole Monitor, and Security. Each client is used to perform a specific set of tasks. The clients can be run from the same computer that is running the communication server, or they can be run from a computer that is networked to the communication server computer. The Edlog and CRBasic Editor applications are not clients of the server but stand-alone utilities to create and edit programs for a datalogger.

In the discussion on Socket Data Export (Section 9) reference is made to the "remote client". In this instance, the remote client is an application not provided by CSI, and to which the Socket Data Export client is transferring data from the data cache.

## 4.2 LoggerNet Communication Server



The LoggerNet communication server, CSILgrNet, must be running before the other clients are enabled. The communication server is started by choosing Start | Programs | LoggerNet | CSILgrNet from the Windows start menu. When the program is started, an icon will appear in the system tray on the task bar similar to that shown on the right below:



The current version information along with a display showing the number of client connections is available by clicking the right mouse button over the icon in the system tray and choosing the 'About' list item.

Server operation is stopped by clicking on the icon with the right mouse button and choosing 'Close'.

There is no user interface for the LoggerNet server software. It is simply started and left running on the server computer. If the LoggerNet server software is not running, no data from the dataloggers will be stored in the data cache and users will be unable to use clients to communicate with the dataloggers.

The LoggerNet communication server performs two functions:

- It is the software that communicates directly with the dataloggers.
- It is an interface between the applications running on end-user computers and the dataloggers.

For data to be collected from a datalogger and stored in the data cache, the communication server must be running. However, it is not necessary for any of the clients to be running unless a user needs to interact with the datalogger. The server collects data without any of the clients running.

The communication server is responsible for all communication with the dataloggers and other devices in the network. Clients do not communicate with the datalogger network directly, but query the server for data collection, device status, etc. Therefore, the LoggerNet server must be active for the clients to interact with the datalogger network.

The LoggerNet communication server software can be loaded on a networked computer and the clients can access the server from different computers through the TCP/IP network, or it can be loaded on a stand-alone machine with the clients also running on that machine. It can access the datalogger network via a serial port (directly or via modems) or TCP/IP connection.

## 4.3 Clients

### 4.3.1 NetAdmin (see section 5)



NetAdmin is used to define the dataloggers in the network and the communications path between the server computer and the dataloggers, as well as set up what data should be collected and a scheduled interval on which the data should be collected. Data collection and schedule information are set up separately for each datalogger. The collected data is stored on the

communication server computer's hard drive in a data cache (refer to Section 5.5.1 for an explanation of the data cache).

Another function that can be set up by NetAdmin is having the server automatically checking the datalogger's clock, comparing it with the communication server's clock, and setting it if it exceeds a specified variation. NetAdmin can also be used to test communication with each datalogger in the communications network to confirm the integrity of the server-datalogger network communications link.

### 4.3.2 Communication Status Monitor (see section 6)



CommStatus is the application used to monitor the health of datalogger network communications. The integrity of the communications link can be verified quickly from the color depicted by a status icon for each device. Columns can be set up to display information on communication error rate and data collection. The network can also be queried for more specific information on each datalogger such as battery voltage, availability of data storage space, table overruns, processor errors, program signature, and operating system or version number.

For trouble-shooting purposes, menu items are available to view I/O communication between the datalogger and the server.

### 4.3.3 Control Panel (see section 7)



The Control Panel is used primarily for troubleshooting or checking operation of a datalogger. This client provides near real-time communication with a datalogger. Utilities are available for sending programs to or retrieving programs from a datalogger, checking or setting a datalogger clock, reviewing data either graphically or in numeric format, manually retrieving data in various formats, and communicating with a datalogger in terminal emulation mode. As with the other clients, the end-use computer running the Control Panel does not communicate with the datalogger network directly, but initiates a transaction with the server to communicate with the datalogger network.

### 4.3.4 Baler (see section 8)



Baler is an application used to perform scheduled data retrieval from the data cache and save it to a file (table ASCII) on the end-use computer. The data can then be used for analyses or archive purposes. The Baler was designed especially for RTMS-compatible files. (Baler gets its name from an analogous operation on a farm where a machine packs hay into bales of uniform size and shape.)

---

**NOTE**

RTMS is an older version of Campbell Scientific's table-based software application that ran in the OS/2 environment.

---

Baler's main window offers an easy way to select the data to be saved to file. The names of the dataloggers in the network are listed in the **Available Dataloggers** field. When a datalogger is highlighted in this field, its associated

tables appear in the **Tables From** field. The user then selects the desired tables from which to collect data and sets up the file save schedule. The data file is saved based on this schedule.

Advanced Settings options allows the user to specify whether or not data collection should proceed if there are holes in the data (refer to Section 5.5.2 for a discussion on data holes), whether Baler should attempt to collect these holes at a later time, and how historical data should be stored on the initial data collection attempt. An option is also available for launching another program after a baling event.

#### 4.3.5 Socket Data Export (see section 9)



The Socket Data Export application is used to output data collected from the data cache to a network port. The intent is to make the data available to another “receiver” application that is built to accept the exported data. Data is output in a standard ASCII or RTMS-compatible format. The interface for selecting a datalogger’s output tables is similar to the interface used in Baler. Once the data to be exported is specified, the application begins “listening” for another application to attach to it. As data is received by the server from the datalogger network, it will be output to the specified network port.

#### 4.3.6 Security (see section 10)



The Security application allows a network administrator to restrict access to certain functions in the LoggerNet clients. Groups are set up in the security application, and individual users are assigned to these groups. Separate security rights are assigned to each of the groups. Using this function, the administrator can determine who has the rights to change the setup of the network, configure the dataloggers, or even access the data.

#### 4.3.7 Hole Monitor (see section 11)



The Hole Monitor provides a way to see historical data that is contained in the datalogger but has not been collected by the server. This data missing from the data cache is called a hole and can be collected by the server through a set of special commands. The Hole monitor displays all the holes that have been identified for each table on each datalogger for the entire network.

#### 4.3.8 Edlog (see section 12)



Edlog is the tool that is used to create and edit datalogger programs for CR10X-TD, CR510-TD, and CR23X-TD dataloggers. Instructions are included for sensor measurement, intermediate processing, program and peripheral control, and data output. The compiler provides error checking and warns of potential problems in the program.

#### 4.3.9 CRBasic Editor



The CRBasic Editor is a tool used to create and edit programs for the CR5000 and CR9000 dataloggers. Instructions are included for sensor measurement,

program and peripheral control, data output, and file storage. On-line help is available from the editing screen and should provide the information needed to get started. There is no documentation for CRBasic programming included in this user manual.

#### **4.3.10 Tools**

There are other client applications that provide other capabilities in working with the LoggerNet server. Among these are Data Table Monitor (section 13.7) to review data in the data cache and Cora Script (Appendix F) to work with the server using text commands or scripts. These applications are most often used for troubleshooting or special applications. These applications are accessed from the Start menu in the Programs | LoggerNet | Tools folder.

### **4.4 Getting Help for LoggerNet Applications**

Detailed information on each application is included in the following sections. Additionally, each application has an on-line help system. On-line help can be accessed by pressing the F1 key or by selecting Help from the application's menu.

A troubleshooting guide is provided in Section 13 of this manual. If you are unable to resolve your problem after reviewing the above noted resources, contact your Campbell Scientific Representative or Campbell Scientific directly.

### **4.5 LoggerNet Operations and Backup Procedures**

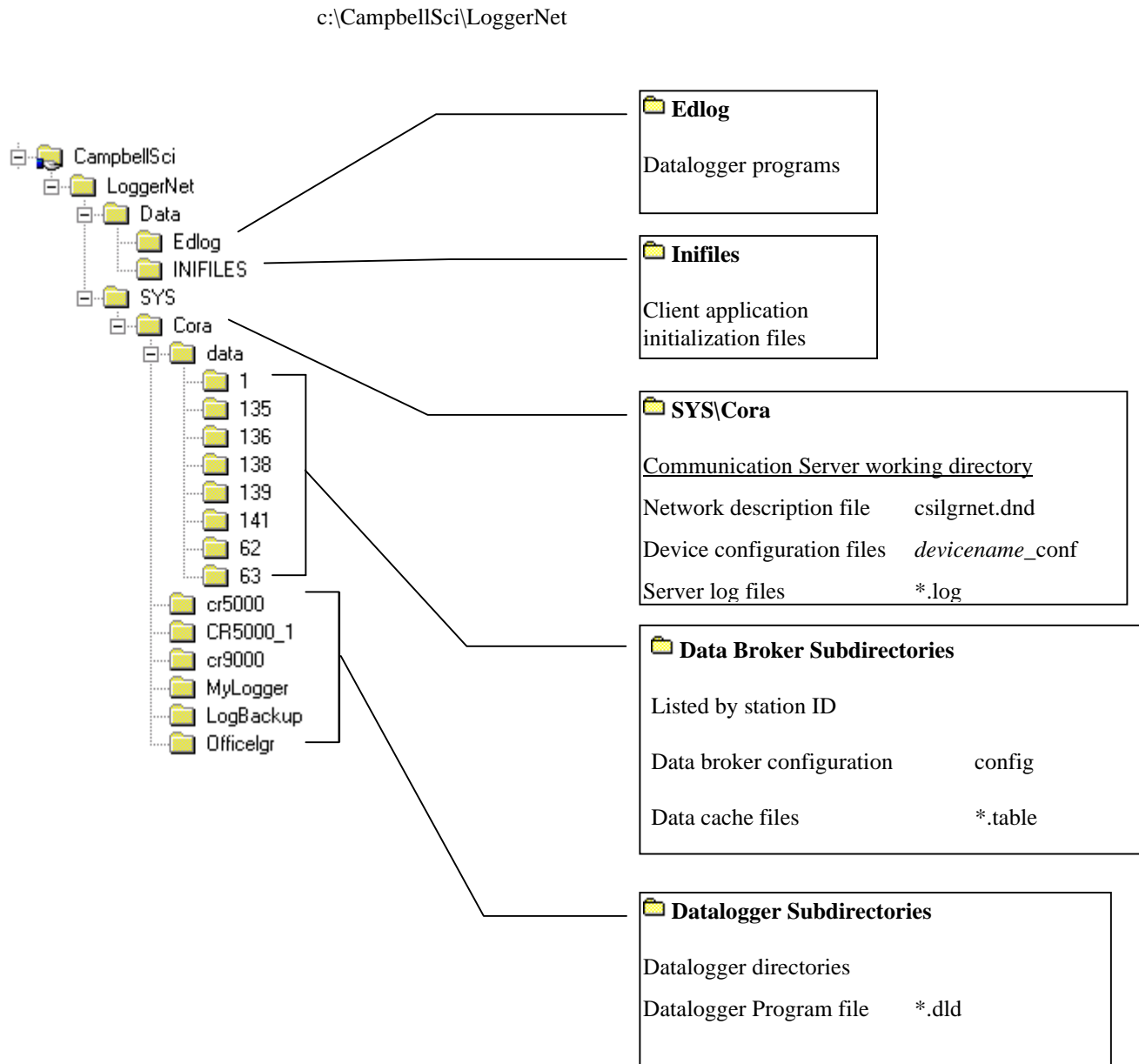
This section describes some of the concepts and procedures recommended for routine operation and security of the LoggerNet software. If software and computer systems were perfect this section would not be necessary. However, since this software is required to run with predictable results in the real world on real computers, the following guidelines and procedures will be helpful in minimizing possible problems that may occur.

#### **4.5.1 Backing up Data**

As with any computer system that contains important information the data stored in the LoggerNet data files should be backed up to a secure archive or transferred to another system. This is a prudent measure in case the hard disk crashes or the computer suffers some other hardware failure that prevents access to the stored data on the disk.

To back up the files, the client applications should be shut down and the LoggerNet server closed. This is necessary to prevent the server from trying to access the files while they are being copied for the backup. Once the files have been backed up, the server and client applications may be restarted.

The most direct approach is to back up the entire working directory as shown below. The default directory name is:



This will back up all of the working files for the server and the client applications along with any datalogger program files stored in the 'Edlog' subdirectory. For a description of the directories and the files see section 4.5.6.

If storage space is limited and the entire directory cannot be backed up, the most important files that should be backed up from the server working directory are:

- The network description file:  
C:\CampbellSci\LoggerNet\SYS\Cora\CsiLgrNet.dnd
- The security configuration file (if security has been set up):  
C:\CampbellSci\LoggerNet\SYS\Cora\CsiLgrNet.sec



- The custom modem settings file (if a custom modem has been configured):  
C:\CampbellSci\LoggerNet\SYS\Cora\wmodem.cust
- All of the device configuration files of the form *devicename\_conf*:  
C:\CampbellSci\LoggerNet\SYS\Cora\\*\_conf
- The subdirectories for each of the dataloggers containing the datalogger program file.
- The SYS\Cora\data directory containing the subdirectories for each station and the data cache files. The numbers used for the station subdirectories correspond to the station IDs assigned by the server. Cora\_Script (see Appendix F) can be used to get the station ID.

Note that the above directories assume that the installation used the default directory structure suggested by the install utility. If a different working directory was used, then the files will be contained in the same set of subdirectories, under the main working directory.

To maintain the information about which records were last retrieved for various applications you should also back up the client initialization (\*.INI) files. The setups on the graph and numeric display of the Control Panel are also contained in these initialization files.

The client application initialization files are in a subdirectory called c:\CampbellSci\LoggerNet\Data\Inifiles (if the default working directory was chosen during installation).

The maximum interval for backups depends primarily on the amount of data maintained in the datalogger memory. The datalogger's final storage is configured as ring memory that will overwrite itself once it is full. If the data is backed up more often than the oldest records in the datalogger are overwritten, a complete data record can still be maintained by restoring the data from the backup and then re-collecting the newest records from the datalogger.

There is a function available in Cora\_Script to save a script file containing all the commands to rebuild the network map and restore the configuration settings for all of the devices. This can be used in case some computer glitch causes the network map or configuration files to be corrupted. It will not, however, store any of the data cache or restore the system state. (see Appendix F)

## 4.5.2 Loss of Computer Power

The LoggerNet communications server writes to several files during normal operations. The most critical files are the data cache table files and the device configuration files. The data cache files contain all of the data that has been collected from the dataloggers by the LoggerNet Server. These files are kept open (or active) as long as data is being stored to the file.

The configuration files contain information about each device in the datalogger network, including collection schedules, device settings, and other parameters.

These files are written to frequently to make sure that they reflect the current state and configuration of the each device. The configuration files are only opened as needed.

If computer system power is lost while the LoggerNet server is writing data to the active files, the files can become corrupted, making the files inaccessible to the server. This is particularly a problem for Windows 95 and 98 machines using the FAT32 disk file formatting. Windows NT offers the choice of NTFS that provides a greater protection for this type of event. Thus Windows NT offers more robust operation.

While loss of power won't always cause a file problem, having files backed up as described above will allow you to recover if a problem occurs. If a file does get corrupted, all of the working files need to be restored from backup to maintain the synchronization in the server state.

### 4.5.3 Program Crashes

If the communication server crashes, there is a possibility that files can be corrupted. This is much less likely than problems due to power loss since the computer operating system remains in control and can close the files left open by the failed program. Again this is handled better with Windows NT than on Windows 95 and 98. If, after a program crash, the server does not run properly, you may need to restore the data from backup.

If you have problems restarting the LoggerNet server after a program crash or it crashes as soon as it starts, on Windows NT systems make sure that the LoggerNet server has not left a process running. You can check this by going to the NT Task Manager and selecting the Process tab. In the list of processes look for one called CsiLgrNet. If this process exists but the communication server is not running, select this process and click on "End Process"; you will be asked to confirm the end process.

### 4.5.4 Restoring from Backup

To restore server operation from a backup copy of the data and configuration you must close any client applications and the communication server. You can then copy and replace the files in the server working directory with the files from the backup. Any data collected or changes made to the network since the last backup will be lost.

Once all the files have been copied, you can restart the LoggerNet server and let the server start data collection for all of the stations. If all of the stations are using scheduled data collection, the server will automatically call at the next scheduled collection interval, and collect as much of the missing data as the dataloggers have available. For any dataloggers not scheduled for data collection you should use the LoggerNet Control Panel application to update the data cache (Retrieve to File tab).

### 4.5.5 Computer System Security

Since all of the working files, system configuration and security information for the communication server operation are contained in the server working directory (c:\CampbellSci\LoggerNet\SYS\Cora), administrators should consider the issue of system security. If there are a large number of users needing access to the data, or there are users with only restricted access to the data, the system administrator should enable the LoggerNet Security client and take advantage of the security offered by Windows NT.

On a Windows NT machine any file or directory can have restricted access by owner or group. For details on this feature, see the documentation with the Windows NT operating system.

### 4.5.6 Directory and File Descriptions

The following descriptions for the file names and directories assume the default working directory was selected during the installation. If another working directory was selected substitute the directory name and path for c:\CampbellSci\LoggerNet wherever it appears in the description below.

c:\CampbellSci\LoggerNet	Working directory. All working files used by LoggerNet are in this directory.
\Data	Files used by the client application
\BaledFiles	This directory is created if the Baler is used
\Logger1	Data files from the Baler for Logger1
\Logger2	Data files from the baler for Logger2
\IniFiles	Initialization files for client applications
\Edlog	Datalogger program files created using Edlog
\SYS\Cora	Application server working directory
\Logger1	Last logger program file
\Logger2	Last logger program file
\Data	Directory containing station data cache files
\2	Directory containing data cache for station 2
\3	Directory containing data cache for station 3



# Section 5. Network Administration

---



*NetAdmin provides a way to create and maintain a network of dataloggers. The datalogger network map is used to define all of the devices and communications links to get from the LoggerNet server to the datalogger stations. The settings for all of the devices are displayed and can be modified on the configuration pages. Many of the settings for the LoggerNet server are also available from NetAdmin.*

*NetAdmin provides utilities to test the communications with dataloggers or RF modems.*

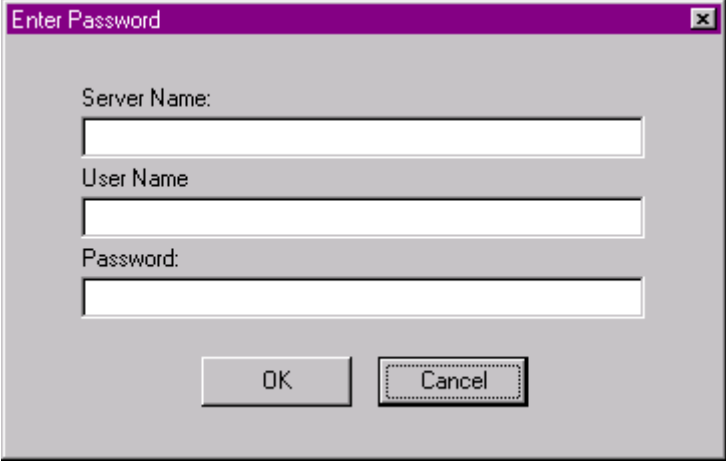
## 5.1 Setting Up a Datalogger Network

The Network Administrator is used to configure your datalogger network, define the communications link that exists between the host computer and the datalogger, and to set up the data collection schedule.

The following are viable datalogger communications methods for the LoggerNet software:

- Direct Connect – Simple serial communications typically on demand and close to the computer.
- Phone Modem – Connection from a phone modem in the computer to a datalogger attached to a remote phone modem. Cell phone communication is also supported.
- Radio Frequency (RF) – Connection over RF using antennas for line of sight communications. (Not available for CR5000/9000 dataloggers.)
- Networked Direct Connect – Direct connection over dedicated cables. (Not available for CR5000/9000 dataloggers.)
- TCP/IP Ethernet or Internet – Connection over a computer local area network or over the Internet using TCP/IP modems at the datalogger.

When you first start the NetAdmin software, the Server Selection dialog box may appear. This dialog box is used to identify the computer on which the LoggerNet communication server is running.

A screenshot of a Windows-style dialog box titled "Enter Password". The dialog box has a purple title bar with a close button (X) in the top right corner. The main area is light gray and contains three text input fields. The first field is labeled "Server Name:" and is empty. The second field is labeled "User Name" and is empty. The third field is labeled "Password:" and is empty. At the bottom of the dialog box, there are two buttons: "OK" and "Cancel". The "Cancel" button is outlined with a dashed border.

In the Server Name field, type the name of the host computer. This must be the valid network name of the host computer or its TCP/IP address (in the form ###.###.###.###) consisting of the IP network number and the host number. If you are on the same computer where the LoggerNet server is running you can use "localhost" as the server name.

The User Name and Password fields are only used if the LoggerNet administrator has enabled security. For more information on the Security application, refer to Section 10.

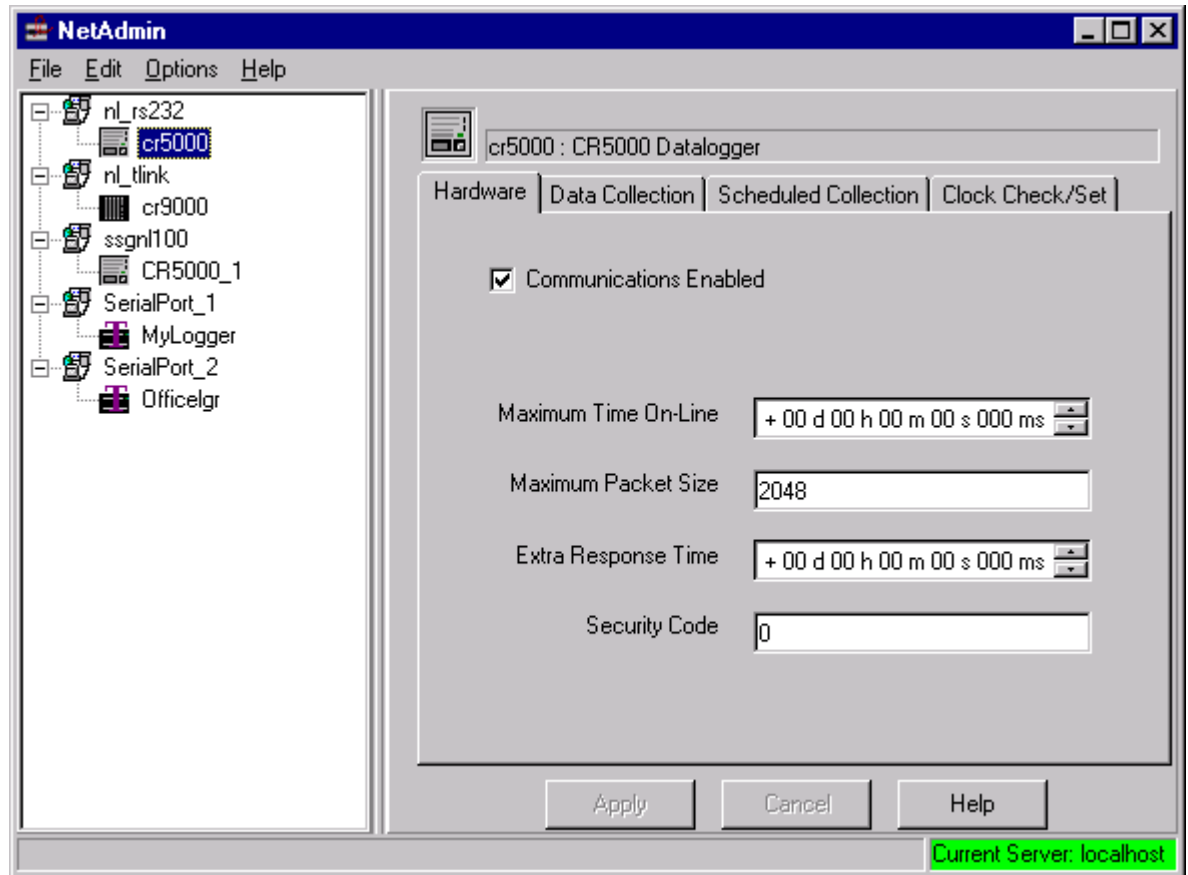
---

**NOTE**

If the Server Selection Dialog box keeps reappearing after the server name is typed in, ensure the LoggerNet communication server is running and TCP/IP communications are enabled.

---

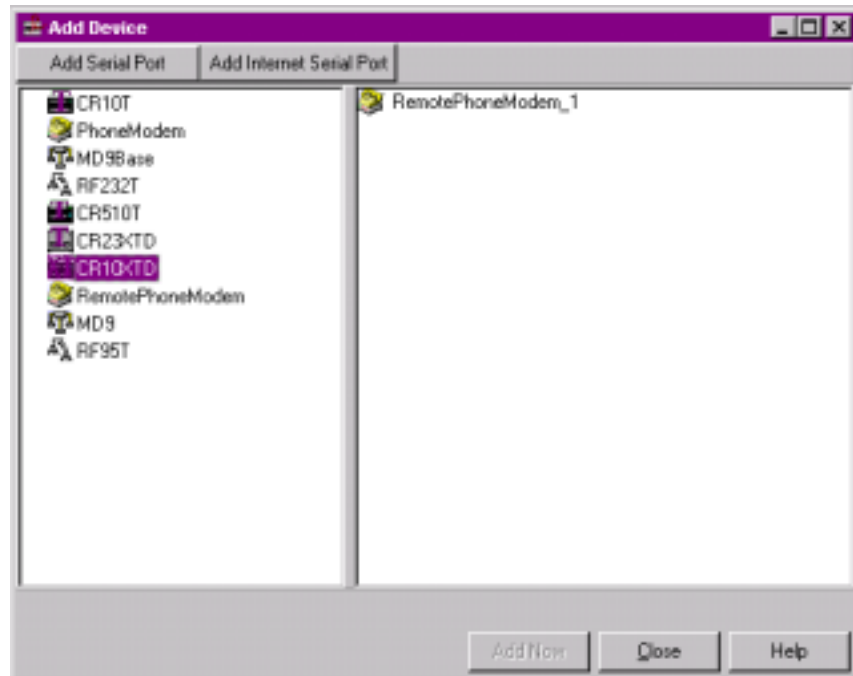
After connecting to the server, NetAdmin's main screen will appear. The screen is divided into two parts: the device map (left side of the screen) and the set up tabs (right side of the screen).



### 5.1.1 Adding Devices to the Network

Begin adding devices to the device map in the order that they appear in your communications link. Let's assume that your server computer is connected to the datalogger via a telephone modem. You would first add a Serial Port, then the telephone modem, the remote phone modem, and the datalogger.

A new device can be added to the datalogger network device map by choosing Edit | Add from the NetAdmin menu. The Add Device window will appear.



When you select an item from the left side of the Add Device window, valid connections will be displayed in the right-hand column. Highlight the item to which you want to attach the device and select the Add Now button. Continue to add devices in this manner until your network map is complete.

An alternate way to quickly add items is to press the right mouse button while your cursor is on a device within the device map window. A shortcut menu will appear that will provide a list of valid devices for connection to the device you have right clicked. For instance, if you right-click within the white space of the device map, the list will present serial port options. If you right-click on a serial port, only valid connections to serial ports will be presented. This right-click menu also provides a quick way to delete items from the device map.

Once all devices are added to the device map, complete the forms associated with each device. Refer to the sections that immediately follow for information on setting up devices.

### 5.1.2 Renaming Network Devices

Device names are often set to reflect a location or some other label that relates to the layout or physical location of network devices.

The names of all of the devices can be changed by either clicking once with the left mouse button on a selected device, or going to the menu item Edit | Rename. The name of the selected device will change to a text edit box and the new device name can be entered. Valid names consist of letters, numbers and the underscore (\_). The device name must be unique in the network and the first character must be a letter.

Some careful thought is appropriate in naming the devices since these names are used throughout the LoggerNet server to refer to the devices.



## 5.2 Device Configuration Settings

When you highlight any device on the network shown on the left side of the NetAdmin screen, forms appear on the right side with the relevant settings. These settings are different for different devices and are described in detail below.

### 5.2.1 Serial Port

The serial port has only a Hardware tab to configure.

**Communications Enabled** - Before communications can take place, all devices in the communications chain must be enabled. The default setting for this check box is Enabled.

**Com Port Connection** - This field designates the communications port through which you will be connecting to the datalogger. Select the arrow to the right of the field with a mouse to display a list containing Com 1 through Com 12. (If you need a comport above 12, simply type the desired comport into the text box.)

**Baud Rate** - Select the arrow to the right of this field to choose a maximum baud rate for communication over the serial port. Note that the actual rate of communication may be limited by the capability of other devices in the communications chain.

**Extra Response Time** - LoggerNet is preconfigured to allow time for responses based on type of device and baud rates. In this field, specify only the additional time that the server should delay before breaking the communications link if there is no response from the serial port. Additional time may be needed in instances where the communications link is noisy or network traffic is heavy. If extra response time is needed, it is typically set to 100 milliseconds.

---

**NOTE**

LoggerNet waits a certain amount of time for a response from each device in a communications path. The extra response times defined for the communications link are cumulative. Therefore, the amount of time spent waiting for a device to respond is the sum of all Extra Response Times defined, plus the default response time for each device in the link. Add only the minimum time necessary since very long response times can delay other scheduled events while waiting for a device that is not responding.

---

### 5.2.2 Internet Serial Port

Like the standard serial port, configuration for the Internet Serial Port also has only one tab, the Hardware tab. Following is an explanation of each of the fields on this form.

**Communications Enabled** - Before communication can take place, all devices in the chain must be enabled. When this box is selected, the Internet serial port is enabled for communication.

**Internet IP Address** - In this field, enter the TCP/IP address and port through which the server will communicate with the datalogger network. The address is entered in the form ###.###.###.###. (Alternately, a valid machine name can be entered.) The port is in the form of :####. A typical entry might be 123.456.789.001:1024.

**Baud Rate** - Select a valid baud rate at which the communication will occur over the Internet serial port. Note that the actual rate of communication may be limited by the capability of other devices in the communications link.

**Extra Response Time** - In this field, specify the additional time that the LoggerNet server should delay before breaking the communications link if there is no response from the Internet Serial Port. Additional time may be needed in instances where the communications link is noisy or network traffic is heavy. With Internet communications, this value should be set to at least 300 milliseconds.

---

**NOTE**

LoggerNet waits a certain amount of time for a response from each device in a communications path. The extra response times defined for the communications link are cumulative. Therefore, the amount of time spent waiting for a device to respond is the sum of all Extra Response Times defined, plus the default response time for each device in the link. Add the minimum time necessary since very long response times can delay other scheduled events while waiting for a device that is not responding.

---

## 5.2.3 Datalogger

Dataloggers have four different tabs. Similar to a serial port, a hardware tab is completed to specify communications settings. There are also tabs to define the data to be collected, how often data should be collected, and whether to automatically update the datalogger's clock.

### 5.2.3.1 Hardware Tab

**Communications Enabled** - Before communication can take place, all devices in the chain must be enabled. When this box is selected, the datalogger is enabled for communication.

**Maximum Time On-line** - A time limit can be set for the length of time the server will stay connected to the datalogger on any one call. Once this time limit has been exceeded, communications with the device will be terminated. This time limit can be disabled by setting the time to zero. This setting should be set to zero for communications links other than phone modems.

**NOTE**

---

Where more than one device in a communications chain has a Maximum Time On-line, the shortest time set in the chain will prevail.

---

**Maximum Packet Size** - Data is transferred in "chunks" called packets. For most devices the default value is 2048 bytes. The value entered in this field can be changed in 32 byte increments. If a communications link is marginal, reducing the packet size may improve reliability.

**Extra Response Time** - In this field, specify the additional time that the LoggerNet server should delay before breaking the communications link if there is no response from the datalogger. Additional time may be needed in instances where the communications link is noisy or network traffic is heavy.

**NOTE**

---

LoggerNet waits a certain amount of time for a response from each device in a communications path. The extra response times defined for the communications link are cumulative. Therefore, the amount of time spent waiting for a device to respond is the sum of all Extra Response Times defined, plus the default response time for each device in the link. Add the minimum time necessary since very long response times can delay other scheduled events while waiting for a device that is not responding.

---

**Security Code** - A datalogger can have a security code to restrict access to the datalogger. This helps prevent inadvertent changes to the datalogger's program or memory. A valid security code is any four digit, non-zero number. This security code is set by the datalogger program, or through a keyboard display or the terminal emulation utility. If a datalogger program that sets or changes security is loaded into the datalogger, the Security Code for that datalogger in LoggerNet must be changed to match so that the clients can access the datalogger. (Security is not available in the CR5000/CR9000 dataloggers.)

**NOTE**

---

Refer to your datalogger operator's manual for complete information on its security functions.

---

### 5.2.3.2 Data Collection Tab

Each time the datalogger executes an instruction to output data, the information is stored in a table-based format. The data collection tab is used to define what data tables will be copied from the datalogger.

**Table Definitions** - This field will read either Current or Changed. When Current is displayed, the tables that are stored by the datalogger match what the server expects. When Changed is displayed, some change has been made to the datalogger program so its tables no longer match those expected by the server. Refer to Section 7.4.3 for information on updating table definitions.

**Database Table Size Factor** - This field is used to specify the size of the data cache in the server, and is based on the number of records in a table. If a factor of 5 is entered into the field, data equal to 5 times the number of records in the datalogger table will be saved in the cache. The amount of data that can be cached is limited only by the server computer's storage capacity.

If the default of 2 is left for the Database Table Size Factor, all that will be stored for each input location in the Inlocs table, or variable in the Public table, is the current value and one value immediately before it. If additional historical input location or variable data is required, the values should be directed to an output table with each scan interval.

---

**NOTE**

The Database Table Size Factor is only used when table storage areas are newly created in the data cache by the server. This will not affect data cache tables currently collecting data. For this setting to take effect it must be set before a new program is sent or table definitions are updated.

---

**Collection Via Data Advise** - When this setting is enabled, the first time data collection is attempted the server tells the datalogger which tables to transmit. After this first collection, the server prompts the datalogger to send the specified data. This method of data collection is recommended for large RF networks using the table based RF modems. If Collect Via Data Advise is disabled, the server collects data using a standard data collection. With standard data collection, the server requests data by table name based on the collection schedule specified on the Scheduled Collection tab in NetAdmin. The server will attempt to collect data starting with the oldest record not yet collected. Standard data collection is the normal means of data collection for most types of communications links. See Appendix B.4 for a more complete description of Data Advise and data collection.

---

**NOTE**

Collection via Data Advise and Hole Collection are not available for the CR5000 and CR9000 dataloggers.

---

**Hole Collection Enabled** - When a data collection attempt is made by the server, the data may not necessarily be retrieved in the order it was recorded by the datalogger. During data collection, the server requests a packet of data from the datalogger. The datalogger sends the packet to the server, along with a checksum which is used to verify the integrity of the data. The server then echoes back the checksum to the datalogger. If the checksums do not match, the packet is considered to be corrupt. If a data packet is corrupted while being transmitted, the server will discard this packet and request that it be sent again. In the meantime, other packets of data may be received successfully by the server.

The Data Advise collection method may create areas in the data cache where data is missing, but can still be collected from the datalogger. The discontinuity in the collected data is referred to as a hole. When this check box is cleared, the server will not attempt to collect these holes in the data. If hole collection is enabled, data collection may be significantly slower as the server attempts to query the datalogger to fill in the missing data. The user must decide if it is

more important to view near real-time data as quickly as possible or to have a complete, archived data set that can be used for further analyses.

Normal operation with a properly configured datalogger network will create few, if any, data holes. In these cases, hole collection will not likely have a significant impact on the speed of data collection.

**Tables to be Collected** - When data collection occurs, the information in the tables listed in the **Included Tables** field will be part of the data copied to the server's cache. Conversely, information in the tables listed in the **Excluded Tables** field will not be part of the collected data. Select a table with a left mouse click, and use the arrow buttons to move the selected table(s) from one field to the other. Alternately, double-clicking a table name will transfer it to the opposite field.

Each datalogger has a set of default tables plus the tables created by the datalogger program. The four default tables in the CR10X-TD family of dataloggers, are Timeset, Errorlog, Inlocs, and Status. The default tables in CR5000/9000 dataloggers are Status and Public.

- **Timeset Table** - The Timeset table contains a history of clock sets for the datalogger. It includes three fields: TimeStamp, RecordNumber, and OldTime. TimeStamp is the time and date the clock was set. RecordNumber is incremented each time the clock is set. When the datalogger is reset or a new program is loaded, RecordNumber is reset to 1. OldTime is the datalogger's clock values before the time was set. (CR10X-TD family dataloggers only)
- **Errorlog Table** - The Errorlog table contains any errors that occur in the datalogger. It includes three fields: TimeStamp, RecordNumber, and ErrorCode. TimeStamp is the time and date the error occurred. RecordNumber is incremented each time an error occurs. When the datalogger is reset or a new program is loaded, RecordNumber is reset to 1. ErrorCode is the code returned by the datalogger when an error occurs. Refer to the datalogger user's manual for a list of all error codes. (CR10X-TD family dataloggers only)
- **Inlocs Table** (CR10X-TD family dataloggers) or **Public Table** (CR-X000 dataloggers) - When a datalogger measures a sensor, the sensor reading is stored in a temporary register called an input location or variable. With each new measurement, the old value is overwritten by the new value. The Inlocs or Public table contains a time stamp, record number, flag status, port status, and the reading from each sensor scanned or user created input locations.
- **Status Table** - The Status table contains information on the datalogger. Data is written to the table with each datalogger program execution. Note that the actual fields contained in the table are datalogger-specific. Table 5-1 below describes each of the fields in the table for a CR10T. CR10X-TD, CR510TD and CR23X-TD status tables are similar. See the datalogger operator's manual for specifics. Table 5-2 describes the fields for the CR5000 datalogger. CR9000 is similar.

**Table 5-1. Example of Status Table Entries (CR10T)**

<b>TMStamp</b>	Date and time the status information was recorded.
<b>RecNBR</b>	The record number in the table.
<b>Battery</b>	Datalogger battery voltage.
<b>Watchdog</b>	The watchdog checks the processor state, software timers, and program related counters. If an error occurs, the watchdog counter is incremented.
<b>Overruns</b>	A table overrun occurs when the datalogger has insufficient time between execution intervals to complete one pass through the program. This counter is incremented with each table overrun.
<b>InLocs</b>	Number of input locations allocated for the program.
<b>PrgmFree</b>	Amount of remaining program memory, in bytes.
<b>Storage</b>	Number of final storage locations available.
<b>Tables</b>	Number of user-created data tables.
<b>DaysFull</b>	Estimated number of days of data the tables using automatic record allocation can hold. (NOTE: this number is only based on tables stored at intervals. Automatically allocating an event based table will often result in very small interval tables.)
<b>Holes</b>	Number of missed records in all data storage tables.
<b>PrgmSig</b>	Signature of the datalogger program. The signature is a unique number derived from the size and format of the datalogger program. If this signature changes, the program has been altered.
<b>PromSig</b>	Signature of the datalogger PROM. As with the PrgmSig, if this signature changes, the datalogger instruction set has somehow been changed.
<b>PromID</b>	Version number of the datalogger PROM.
<b>ObjSrlNo</b>	Revision number of the datalogger PROM.

**Table 5-2. CR5000 Status Table Entries**

<b>ROMVersion</b>	Version of the ROM code. This value is stored in the ROM and read by the OS at compile time.
<b>OSVersion</b>	Current version of the operating system.
<b>OSItem</b>	The CSI item number for the operating system.
<b>OSDate</b>	Date that the Operating System was compiled.
<b>StationName</b>	String stored as the Station Name of the CR5000.
<b>ProgName</b>	The Name of the currently running program.
<b>StartTime</b>	Time that the program began running.
<b>Battery</b>	Current value of the battery voltage. This measurement is made in the background calibration.

Table 5-2. CR5000 Status Table Entries

<b>PanelTemp</b>	Current Panel temperature measurement.
<b>LithiumBattery</b>	A Boolean variable signaling "True" (-1) if the lithium battery is OK and "False" (0) if not. The lithium battery is loaded and a comparator checked every 4 seconds to verify that the battery is charged.
<b>CPUSignature</b>	The Operating System signature. The value should match the value obtained by running the CSI sig program on the <i>name.obj</i> operating system file.
<b>DLDSignature</b>	Signature of the current running program file.
<b>ProgSignature</b>	Signature of the compiled binary data structure for the current program. This value is independent of comments added or non functional changes to the program file.
<b>PC-CardBytesFree</b>	Gives the number of bytes free on the PC-Card.
<b>MemoryFree</b>	Amount (in bytes) of unallocated memory on the CPU (SRAM). The user may not be able to allocate all of free memory for data tables as final storage must be contiguous. As memory is allocated and freed there may be holes that are unusable for final storage, but that will show up as free bytes.
<b>DLDBytesFree</b>	Amount of free space in the CPU RAM disk that is used to store program files.
<b>ProcessTime</b>	Time in microseconds that it took to run through processing on the last scan. Time is measured from the end of the EndScan instruction (after the measurement event is set) to the beginning of the EndScan (before the wait for the measurement event begins) for the subsequent scan.
<b>MaxProcTime</b>	The maximum time required to run through processing for the current scan. This value is reset when the scan exits.
<b>MeasureTime</b>	The time required by the hardware to make the measurements in this scan. The sum of all integration times and settling times. Processing will occur concurrent with this time so the sum of measure time and process time is not the time required in the scan instruction.
<b>SkippedScan</b>	Number of skipped scans that have occurred while running the current program.
<b>SlowProcTime</b>	Time required to process the current slow scan. If the user has slow scans then this variable becomes an array with a value for the system slow scan and each of the user's scans.
<b>MaxSlowProcTime</b>	The maximum Time required to process the current slow scan. If the user has slow scans then this variable becomes an array with a value for the system slow scan and each of the user's scans.
<b>LastSlowScan</b>	The last time that this slow scan executed. If the user has slow scans then this variable becomes an array with a value for the system slow scan and each of the user's scans.
<b>SkippedSlowScan</b>	The number of scans that have been skipped in this slow sequence. If the user has slow scans then this variable becomes an array with a value for

**Table 5-2. CR5000 Status Table Entries**

	the system slow scan and each of the user's scans.
<b>MeasureOps</b>	This is the number of task sequencer opcodes required to do all measurements in the system. This value includes the Calibration opcodes (compile time) and the system slow sequence opcodes.
<b>WatchdogErrors</b>	The number of Watchdog errors that have occurred while running this program. This value can be reset from the keyboard by going to status and scrolling down to the variable and pressing the DEL key. It is also reset upon compiling a new program.
<b>Low12VCount</b>	Keeps a running count of the number of occurrences of the 12VLow signal being asserted. When this condition is detected the logger ceases making measurements and goes into a low power mode until the system voltage is up to a safe level.
<b>StartUpCode</b>	A code variable that allows the user to know how the system woke up from poweroff.
<b>CommActive</b>	A variable signaling whether or not communications is currently active (increments each time the autobaud detect code is executed).
<b>ProgErrors</b>	The number of compile (or runtime) errors for the current program.
<b>ErrorCalib</b>	A counter that is incremented each time a bad calibration value is measured. The value is discarded (not included in the filter update) and this variable is incremented.
<b>VarOutOfBound</b>	Flags whether an array was accessed out of bounds.
<b>SkippedRecord</b>	Variable that tells how many records have been shipped for a given table. Each table has its own entry in this array.
<b>SecsPerRecord</b>	Output interval for a given table. Each table has its own entry in this array.
<b>SrlNbr</b>	Machine specific serial number. Stored in FLASH memory.
<b>Rev</b>	Hardware revision number. Stored in FLASH memory.
<b>CalVolts</b>	Factory calibration numbers. This array contains twenty values corresponding to the 20 integration / range combinations. These numbers are loaded by the Factory Calibration and are stored in FLASH.
<b>CalGain</b>	Calibration table Gain values. Each integration / range combination has a gain associated with it. These numbers are updated by the background slow sequence if the running program uses the integration / range.
<b>CalSeOffset</b>	Calibration table single ended offset values. Each integration / range combination has a single ended offset associated with it. These numbers are updated by the background slow sequence if the running program uses the integration / range.
<b>CalDiffOffset</b>	Calibration table differential offset values. Each integration / range combination has a differential offset associated with it. These numbers are updated by the background slow sequence if the running program uses the integration / range.
<b>CardStatus</b>	Contains a string with the most recent card status information.



Table 5-2. CR5000 Status Table Entries

<b>CompileResults</b>	Contains any error messages that were generated by compilation or during run time.
-----------------------	--

### 5.2.3.3 Scheduled Collection Tab

The Scheduled Collection tab defines when the server will automatically query the datalogger for new data.

**Scheduled Collection Enabled** - This check box activates the data collection schedule defined on this tab. No data will be saved to the data cache if the schedule is disabled.

**Initial Date** - The initial date field is used to define the first date for scheduled data retrieval. If the date entered in this field has already passed, a data collection attempt will be made at the end of the next scheduled collection interval.

**Initial Time** - This field is used to define the first time for scheduled data retrieval. As with the Initial Date field, if the time has already passed, a data collection attempt will be made when communication is established. This setting is also used with the Collection Interval to determine the time data collection will be done.

---

#### NOTE

Entering a zero for any of the intervals below will cause the server to try and collect as fast as it can.

---

**Collection Interval** - This is the interval at which the datalogger will be queried for new data. If this interval is set at 1 hour, new data will be collected from the datalogger every hour.

Example: If the Initial Date and Time are 1/1/99, 12:15 p.m., with an interval of one hour, data collection attempts will be made at 15 minutes past the hour, each hour.

---

#### NOTE

For RF systems see the notes on network setup and configuration in Section 14 before entering these settings.

---

**Primary Retry Interval** - If a data collection attempt is made but fails, you can specify an interval on which another attempt will be made. This primary retry interval starts at the time of failure, not on the original calling time and interval. "Failures" may be caused by busy phone lines, noisy RF environments, low batteries, damaged hardware, etc.

**Number of Primary Retries** - The number entered into this field is the number of times the server will attempt to contact the datalogger on the Primary Retry Interval. If this number of collection attempts fail, then the server will resume calling on the Secondary Retry Interval.

**Secondary Retry Interval** - The secondary retry interval is a calling interval that will be attempted if all Primary Retries fail. Data collection attempts will continue on the Secondary Interval until a data collection attempt is successful, at which time, all retry statistics are reset. The Secondary Retry Interval is based on the initial date and time settings, not the time of the last failure.

Typical use is to set the Primary Retries fairly close together, and the Secondary Retry at a longer interval. For instance, if you are calling on an hourly basis, the Primary Retries might be set to three tries, spaced five minutes apart. The Secondary Interval then might be set at 2 hours.

#### 5.2.3.4 Clock Check/Set Tab

**Automated Clock Check** - Enable this to compare the datalogger's clock to the server PC's clock based on the schedule defined by the other parameters on this tab. If the datalogger's time differs from the server's time by more than a specified amount, the datalogger's clock will be set to the server's time.

A separate call to the datalogger will not be made exclusively to process a clock check. A clock check will be made when the server contacts the datalogger for some other function.

Setting up a clock check may not be desirable. It is possible to end up with missing data or duplicate data if the datalogger's clock is set forward or backward enough to skip or duplicate a data storage time.

Refer to Section 5.3 for additional information on setting and checking the clock.

**Time Zone Offset** - A value can be entered into this field to set an offset for the datalogger's clock from the server's clock. This may be useful if the server and the datalogger are in different time zones.

**Initial Date** - The initial date field is used to define the date on which the first clock check will occur. If the date entered in this field has already passed, the datalogger's clock will be checked at the next scheduled time.

**Initial Time** - This field is used to define the time at which the first clock check will occur. As with the Initial Date field, if the time has already passed, the clock will be checked at the next scheduled time.

**Interval** - The interval at which a clock check should be performed is specified in the Interval field. If this interval is set at 24 hours, the datalogger's clock will be checked daily, based on the initial date and time.

**Allowed Clock Deviation** - The Allowed Clock Deviation field is used to specify the number of seconds the datalogger's clock can differ from the server's before the server resets the datalogger's clock.

---

#### NOTE

The Allowed Clock Deviation setting will prevent a manual clock set from being carried out if the difference between the datalogger's and server's clocks is less than the specified deviation.

---

## 5.2.4 RF Base

The RF base modem acts as a master device that communicates with the remote RF modems connected to dataloggers at the field site. It has three tabs that must be configured.

### 5.2.4.1 Hardware Tab

**Communications Enabled** - Before communication can take place, all devices in the chain must have the Communications Enabled box checked. When this box is selected, communication to the RF base is enabled.

When the user disables communications to an operating RF base, the server will attempt to send a new network map to the RF Base that will suspend the polling operation. This stops RF polling until communications on the RF base are enabled.

An alternate method to suspend polling by the RF base is to disable scheduled collection for each of the dataloggers in the RF network. Communications must be disabled for the remote RF modems prior to disabling communications with the RF base. Otherwise, the RF base will continue polling and attempting to collect data until it times out (up to two hours).

**Maximum Time On-line** - A time limit can be set for the length of time the server will stay connected to the RF base on any one call. Once the time limit entered in this field has been exceeded, communications with the device will be terminated. In RF network configurations where the RF base is connected to the server by a direct or TCP/IP serial connection this should be set to zero. This setting is most often used to limit phone time in a phone to RF communications link. Setting the Maximum Time On-line to zero disables the time limit.

---

**NOTE**

Where more than one device in a communications chain has a Maximum Time On-line, the shortest time set in the chain will prevail.

---

**Maximum Packet Size** - Data is transferred in "chunks" called packets. For most devices the default value is 2048 bytes. The value entered in this field can be changed in 32 byte increments. If a communications link is marginal, reducing the packet size may improve the quality.

**Extra Response Time** - In this field, specify the additional time that the server should delay before breaking the communications link if there is no response from the RF base. Additional time may be needed in instances where the communications link is noisy or network traffic is heavy.

**NOTE**

---

LoggerNet waits a certain amount of time for a response from each device in a communications path. The extra response times defined for the communications link are cumulative. Therefore, the amount of time spent waiting for a device to respond is the sum of all Extra Response Times defined, plus the default response time for each device in the link. Add the minimum time necessary since very long response times can delay other scheduled events while waiting for a device that is not responding.

---

#### 5.2.4.2 Scheduled Collection

When collecting data using Data Advise via an RF link, the RF base queries the dataloggers on a specified interval. It buffers this data until it is queried by the server computer. The Interval is used in conjunction with the Router Offset and the Computer Offset to time data collection for optimum performance.

**Interval** - The Interval field is used to set up the schedule on which the RF base will query the devices in the datalogger network for new data. This is also the interval at which the computer will contact the RF base for new data.

**NOTE**

---

This interval sets the fastest rate for scheduled data collection using Data Advise from the dataloggers in this RF network.

---

**Router Offset** - An offset from the Interval can be specified at which the RF base will query the devices in the datalogger network for new data. This offset can be used to allow the dataloggers time to make the sensor readings before the data is transferred.

**Computer Offset** - An offset from the Interval can be specified at which the server will retrieve any data that has been collected, and is being buffered, by the RF base. This offset can be used to allow the RF Base time to complete data collection so the server gets the most recent data.

#### 5.2.4.3 Clock Check/Set

The RF base has its own internal real-time clock that is used to schedule the data collection polling and for timing. This clock can only be set, it cannot be checked. A clock set to the RF base will cause it to resynchronize the data collection polling.

**Automated Clock Check** - Enable this to set the RF base clock to the server PC's clock based on the schedule defined by the other parameters on this tab.

A separate connection to the RF Base will not be made exclusively to process an automated clock set. A clock set will be made when the server contacts the RF Base for some other function.

Frequent setting of the RF base clock is not desirable because it causes a resynchronization of the data collection polling schedules and loss of any data waiting to be transferred to the computer.

**Time Zone Offset** - A value can be entered into this field to set an offset for the RF Base clock from the server's clock. This may be useful if the server and the RF Base are in different time zones.

**Initial Date** - The initial date field is used to define the date on which the first clock set will occur. If the date entered in this field has already passed, the RF Base clock will be set at the next scheduled interval.

**Initial Time** - This field is used to define the time at which the first clock set will occur. As with the Initial Date field, if the time has already passed, the clock will be checked at the next scheduled interval.

**Interval** - The interval at which a clock set should be performed is specified in the Interval field. If this interval is set at 24 hours, the RF Base clock will be checked daily, based on the initial date and time.

**Allowed Clock Deviation** – This field is not used for the RF Base. Any value entered here is ignored.

## 5.2.5 RF Modem

The RF modem has only a Hardware tab. Many of its settings are similar to the Hardware tabs for the other devices.

**Communications Enabled** - Before communications can take place, all devices in the chain must have the Communications Enabled box checked. When this box is selected, communication to the RF modem is enabled.

**Switch ID** - The hardware for each RF modem is configured with internal switches for a certain address. This address acts as an identification for the device in an RF network. Each RF modem in the network must have a unique address; this number is entered in the Switch ID field. RF modems used as repeater only sites still need their own unique switch ID.

**Maximum Time On-line** – The communication with an RF modem cannot be controlled by the server so this setting doesn't apply to RF modems. This setting should be disabled by entering zero for the time. If this setting is set to something other than zero, it can force the server to terminate the connection with the RF base.

---

**NOTE**

Where more than one device in a communications chain has a Maximum Time On-line, the shortest time set in the chain will prevail.

---

**Maximum Packet Size** - Data is transferred in "chunks" called packets. For most devices the default value is 2048 bytes. The value entered in this field can be changed in 32 byte increments. If a communications link is marginal, reducing the packet size may improve the quality.

**Extra Response Time** - In this field, specify the additional time that the LoggerNet server should delay before breaking the communications link if there is no response from the RF modem. Additional time may be needed in instances where the communications link is noisy or network traffic is heavy. Extra response time is particularly important when the RF modem is used as a

repeater to other RF nodes. Extra response times of 30 to 45 seconds may be needed at the remote RF nodes.

**NOTE**

LoggerNet waits a certain amount of time for a response from each device in a communications path. The extra response times defined for the communications link are cumulative. Therefore, the amount of time spent waiting for a device to respond is the sum of all Extra Response Times defined, plus the default response time for each device in the link. Add the minimum time necessary since very long response times can delay other scheduled events while waiting for a device that is not responding.

---

## 5.2.6 MD9 Base

The MD9 base modem has only a Hardware tab. Many of its settings are similar to the Hardware tabs for the other devices.

**NOTE**

LoggerNet assumes an MD9 base modem address of 255. Therefore, the MD9 base modem must have the hardware switch ID set to 255 for communication to work.

---

**Communications Enabled** - Before communications can take place, all devices in the chain must have the Communications Enabled box checked. When this box is selected, communication to the MD9 base is enabled.

**Maximum Time On-line** - A time limit can be set for the length of time the server will stay connected to the MD9 base on any one call. Once the time limit entered in this field has been exceeded, communication with the device will be terminated. When setting this parameter, consideration should be given to the number of devices in the MD9 network and the length of time it will take to collect data from each of the attached dataloggers. Maximum Time On-line can be disabled for this device by entering zero for the time.

**NOTE**

Where more than one device in a communications chain has a Maximum Time On-line, the shortest time set in the chain will prevail.

---

**Maximum Packet Size** - Data is transferred in "chunks" called packets. For most devices the default value is 2048 bytes. The value entered in this field can be changed in 32 byte increments. If a communications link is marginal, reducing the packet size may improve the quality.

**Extra Response Time** - In this field, specify the additional time that the LoggerNet server should delay before breaking the communications link if there is no response from the MD9 base. Additional time may be needed in instances where the communications link is noisy or network traffic is heavy.

**NOTE**

---

LoggerNet waits a certain amount of time for a response from each device in a communications path. The extra response times defined for the communications link are cumulative. Therefore, the amount of time spent waiting for a device to respond is the sum of all Extra Response Times defined, plus the default response time for each device in the link. Add the minimum time necessary since very long response times can delay other scheduled events while waiting for a device that is not responding.

---

## 5.2.7 MD9 Modem

The MD9 modem is the remote MD9 device that is connected to the datalogger at the field site. It has a Hardware tab only.

**Communications Enabled** - Before communications can take place, all devices in the chain must have the Communications Enabled box checked. When this box is selected, communication to the MD9 modem is enabled.

**Switch ID** - The hardware for each MD9 modem is configured for a certain address using internal hardware switches. This address acts as an identification for the device in an MD9 network. Each MD9 modem in the network must have a unique address; this number is entered in the Switch ID field.

**Maximum Time On-line** - A time limit can be set for the length of time the server will stay connected to the MD9 modem on any one call. Once the time limit entered in this field has been exceeded, communications with the device will be terminated. When setting this parameter, consideration should be given to the length of time it will take for the MD9 modem to collect data from the attached datalogger. Maximum Time On-line can be disabled for this device by entering zero for the time.

**NOTE**

---

Where more than one device in a communications chain has a Maximum Time On-line, the shortest time set in the chain will prevail.

---

**Maximum Packet Size** - Data is transferred in "chunks" called packets. For most devices the default value is 2048 bytes. The value entered in this field can be changed in 32 byte increments. If a communications link is marginal, reducing the packet size may improve the quality.

**Extra Response Time** - In this field, specify the additional time that the LoggerNet server should delay before breaking the communications link if there is no response from the MD9 modem. Additional time may be needed in instances where the communications link is noisy or network traffic is heavy.

**NOTE**

---

LoggerNet waits a certain amount of time for a response from each device in a communications path. The extra response times defined for the communications link are cumulative. Therefore, the amount of time spent waiting for a device to respond is the sum of all Extra Response Times defined, plus the default response time for each device in the link. Add the minimum time necessary since very long response times can delay other scheduled events while waiting for a device that is not responding.

---

## 5.2.8 Phone Modem

A telephone modem can be connected to one of the server's serial ports to provide access to other devices in the datalogger network. The telephone modem has only the Hardware tab. This device must be properly installed and configured in the operating system to use one of the computer's COM ports before it can be used.

**Communications Enabled** - Before communication can take place, all devices in the chain must have the Communications Enabled box checked. When this box is selected, communication to the phone modem is enabled.

**Modem Type** - Use the drop down list box to select the type of modem that is attached to the server computer's communications port.

**Edit Server's Modem Database** - The modem connected to the server computer may not be listed in the database, or the user may desire to change the modem configurations. When the Edit Server's Modem Database button is selected, the reset and initialization strings for the selected modem are displayed. You can change these settings or add a custom modem to the list. The only modems that can be deleted from the list are modems that have been added by the user.

**Extra Response Time** - In this field, specify the additional time that the LoggerNet server should delay before breaking the communications link if there is no response from the phone modem. Additional time may be needed in instances where the communications link is noisy or network traffic is heavy.

**NOTE**

---

LoggerNet waits a certain amount of time for a response from each device in a communications path. The extra response times defined for the communications link are cumulative. Therefore, the amount of time spent waiting for a device to respond is the sum of all Extra Response Times defined, plus the default response time for each device in the link. Add the minimum time necessary since very long response times can delay other scheduled events while waiting for a device that is not responding.

---



## 5.2.9 Remote Phone Modem

The Hardware tab of the remote phone modem is used to set up the dialing string for the remote device attached to it. The hardware tab has the following controls:

**Communications Enabled** - Before communication can take place, all devices in the chain must have the Communications Enabled box checked. When this box is selected, communication to the remote phone modem is enabled.

**Phone Number/Delay Field** - This field lists all of the telephone numbers or dialing strings that have been entered for the remote modem. Multiple entries are allowed to accommodate entry of access codes, credit card numbers, or other numbers that may be needed for communication to the remote modem and its attached device.

**Add Phone Number** - When the Add Phone Number button is selected, a dialog box appears. This dialog box is used to enter the dialing string for the remote modem. It also includes a field in which to add additional time to the end of the dialing string. This is useful if a delay is needed before the program sends the next part of the dialing string. The number entered in the delay field is the amount of time, in seconds, for the delay.

**Edit Phone Number Button** - If a dialing string is highlighted, selecting the Edit Phone Number button will invoke the dialing string dialog box. Any required changes can then be made to the existing dialing string.

**Delete Phone Number Button** - If a dialing string is highlighted, press the Delete Phone Number button to remove it from the dialing string list.

**Maximum Time On-line** - A time limit can be set for the length of time the server will stay connected to the remote phone modem on any one call. Once the time limit entered in this field has been exceeded, communication with the device will be terminated. When setting this parameter, consideration should be given to the length of time it will take for the remote phone modem to collect data from the attached datalogger. Maximum Time On-line can be disabled for this device by entering zero for the time.

---

**NOTE**

Where more than one device in a communications chain has a Maximum Time On-line, the shortest time set in the chain will prevail.

---

**Maximum Packet Size** - Data is transferred in "chunks" called packets. For most devices the default value is 2048 bytes. The value entered in this field can be changed in 32 byte increments. If a communications link is marginal, reducing the packet size may improve the quality.

---

**NOTE**

With older modems communicating at slower speeds or with small data buffers, it is possible to overflow the internal buffer when sending programs. Changing to a smaller packet size will allow the logger programs to be sent.

---

**Extra Response Time** - In this field, specify the additional time that the LoggerNet server should delay before breaking the communications link if there is no response from the remote phone modem. Additional time may be needed in instances where the communications link is noisy or network traffic is heavy.

**NOTE**

---

LoggerNet waits a certain amount of time for a response from each device in a communications path. The extra response times defined for the communications link are cumulative. Therefore, the amount of time spent waiting for a device to respond is the sum of all Extra Response Times defined, plus the default response time for each device in the link. Add the minimum time necessary since very long response times can delay other scheduled events while waiting for a device that is not responding.

---

## 5.3 Setting the Clock

A datalogger's Clock Check/Set tab can be used to define a schedule at which a clock check will be performed (refer to Section 5.2.3.4). Based on the information entered for this tab, the datalogger's clock will be updated if it varies from the LoggerNet server's clock more than the amount of time specified in the Allowed Clock Deviation field.

Because it is important to maintain accurate time-stamping of your data, there are a few things to take into consideration when setting up a clock check schedule. First of all, let's look at how the time stamp is derived for each record of data stored in the interval based data tables.

**NOTE**

---

Event based data storage does not rely on a calculated time stamp. Data stored to a table based on an event includes a time stamp in the table.

---

For the CR10X-TD family of dataloggers, when an instruction P84 Output Record is executed in the datalogger program to create an interval based table and a record is stored, the time stamp is not stored along with the record. Instead, when data is retrieved from the datalogger, the datalogger uses the time stamp of the last record stored and the table interval to calculate the time stamp for any previous records. This calculated time stamp is then stored by the server as part of the data record along with the other data values when data is collected. Because of this time stamping method, if the datalogger clock is changed such that it passes an output interval a discontinuity could occur in the records that could cause the time stamps to be incorrect.

Your datalogger clock should deviate no more than  $\pm 1$  minute per month. Typically, this drift is less than what will be experienced with a personal computer. Therefore, unless you have a scheduled task on your computer which synchronizes your computer's clock with an atomic clock or other accurate time keeping device, the datalogger's clock may be more accurate than the server's clock.

**NOTE**

---

Changing the computer system clock while clients are connected to the server will terminate the connection for most client applications. This can also affect server operations or even crash the server.

---

Another point to consider is how the clock checks may affect the time stamp for your data. Let's say, for instance, that you have a data collection schedule of one minute with a clock set if the two clocks deviate more than two minutes. Over time, the clocks may drift sufficiently that the datalogger's clock is set. If the datalogger's clock is 12:02:00, and the LoggerNet server's clock is 12:04:15 the datalogger's clock will be set to 12:04:15. Therefore, there will be no data for the time stamps 12:03 and 12:04. Conversely, if the datalogger's clock is a few minutes faster than the LoggerNet server's clock, the result would be duplicate time stamps that contained different data.

**NOTE**

---

The record number can be used along with the time stamp to assure that records are in order, and no data has been missed.

---

## 5.4 Setting Up Scheduled Data Collection

### 5.4.1 Data Collection Scheduling Considerations

For any data to be available for viewing using any of the clients, the data has to be collected from the dataloggers into the LoggerNet server's data cache. This includes data from the default tables of the datalogger such as Inlocs and Status. Even the Control Panel information depends on the data cache for its data since the idea of "connected" simply means that the server will try to keep the datalogger on line. The data must still be collected and subsequently displayed from the data cache.

For data to be collected it has to be selected for collection on the Data Collection tab in NetAdmin for each datalogger (Section 5.2.3.2). This setting is in turn dependent on the server having the information from the datalogger about what tables are available.

In summary, all data is displayed only from the data cache. The data cache gets data from the datalogger based on the information the server has about the datalogger tables, and which of those tables are selected for collection.

#### 5.4.1.1 Intervals

One of the most significant considerations for setting up data collection is all of the intervals associated with reading, storing, collecting and retrieving data. The intervals and their significance for data handling are described below.

##### 5.4.1.1.1 Datalogger Program Intervals

There are two types of intervals written into the datalogger program which affect the availability and collection of data:

- **Program Execution Interval** – The execution, or scan, interval determines how often the datalogger carries out the instructions in the datalogger program. It is specified in seconds and determines the fundamental rate at which data is available. In typical programs the sensor readings are taken at this rate and the values are stored in corresponding Input Locations or variables. This execution interval is the fastest that data measurements can be updated and data stored. (Depending on how the program is written, sensor readings may occur at specified intervals and not on every program execution.)
- **Table Storage Interval** – Most data tables are set up to store data records at regular intervals. The data record consists of a record number and “implied time stamp”, followed by the output processing (i.e., sample, average, min, max, etc.) of the input location or variable values. This interval is specified in seconds and must be a multiple of the program execution interval or storage intervals will be skipped. For example, if the program execution interval is 5 seconds and the table interval is set to 3 seconds, there will only be an entry in the table every 15 seconds. The interval specified determines the fastest rate the server can collect new data from a data table.

#### **5.4.1.1.2 Data Collection Setting Intervals**

The collection interval at which the LoggerNet server gets new data from the datalogger is set up on the Scheduled Data tab for that datalogger in NetAdmin (Section 5.2.3.3). If the collection interval set up is faster than the rate at which data is being stored to a data table by the datalogger program, data will not be collected every call, but only at intervals when new data is available.

If data collection is enabled for input location data (Inlocs) or Public tables, the current values will be collected every time a scheduled collection occurs, whether or not the values have been updated.

#### **5.4.1.1.3 Communications Path Considerations**

When setting up data collection intervals in NetAdmin you should consider the communications path to the datalogger and how it will affect how often you can retrieve data.

Phone modems require anywhere from 20 – 30 seconds or more to establish communication, negotiate baud rate, and start transferring data. Therefore, it is not a good idea to schedule collection by phone modem more often than once every two minutes. The time to make a call and start communications should be considered when you are collecting from multiple stations by phone. Sufficient time should be allowed for the server to dial the station, retrieve the data, and then contact the next station. There should be enough time between calls to the first station that the calls to the other stations using the same modem can be completed. Otherwise the collection schedule will be delayed waiting for the previous stations to finish.

RF communications introduces another scheduling interval into the setup when Data Advise data collection is enabled. The RF base has a programmed polling interval which determines how often it polls the stations in its RF network for

Data Advise data. Even though data collection can be set up independently for the datalogger stations, the rate at which data is actually collected is determined by the polling interval of the RF base.

The presence of repeaters in the RF network needs to be considered since switching times from station to station are longer when the station has repeaters in the RF path. On average you should allow about 5 seconds/repeater switching times between stations which use repeaters. Typical station to station time using Data Advise without repeaters is about 1 second.

Consideration should also be given to any manual operations such as clock sets, program downloads, or directed queries to the datalogger. These require significantly more time and can affect RF network responsiveness.

Any network setup using a phone to RF should be reviewed for collection scheduling issues due to the combinations of time delays and the extra polling interval associated with RF data collection.

### 5.4.2 Setting Up Scheduled Data Collection

The data to be collected and the method used to collect it are specified on the datalogger's Data Collection tab (Section 5.2.3.2). The Scheduled Collection tab (refer to Section 5.2.3.3) is used to define the interval on which the LoggerNet server will query the datalogger for new data. If new data exists, it will be stored in the LoggerNet server's data cache, which is accessible to the other clients.

To set up a data collection schedule for a datalogger, first ensure that your device map has been configured with all of the devices listed as they actually exist. Next, determine which tables should be collected from the datalogger each time a data collection attempt is made. If no tables are selected for the Included Tables field of the Data Collection tab, no data will be collected from the datalogger.

---

**NOTE**

If no table names appear in the Tables to Be Collected fields on the Data Collection Tab, open the Control Panel application, connect to the datalogger, and update the table definitions (Section 7.4.3).

---

The data collection schedule should be set up next. Set the initial date and time to when you would like the first data collection attempt to occur and set the interval at which subsequent data collection attempts should occur. Make sure that communications are enabled for all devices in the communications path, and that scheduled collection is enabled. If the initial date and time is set to a time that has already passed, data collection will begin at the next scheduled interval,

The Communications Status Monitor (Section 6) can be used to ensure that data collection is occurring on the defined schedule. If data is not being collected, check the following:

- The Scheduled Collection Enabled box on the Scheduled Collection tab for the datalogger must be selected. This turns the schedule "on". You can temporarily disable data collection by clearing this check box and applying the change.
- The tables from which you desire data should be listed in the Included Tables field of the Data Collection Tab.
- All devices in the communications path to the datalogger must have the Communications Enabled check box on the Hardware tab enabled.
- Unsuccessful attempts to communicate with the datalogger may exhaust the number of Primary Retries specified so that the Secondary Retry interval is in effect. Check the date and time listed for the next data collection in the CommStatus Communications Monitor.
- Look at the collection state data for the datalogger in the CommStatus Communications Monitor. This is displayed as one of four states.
  - Normal collection
  - Primary retry
  - Secondary retry
  - Collection disabled
- Check on the Options | Server Settings form in NetAdmin to make sure that the Allow Automated Operations and Network Communications Enabled check boxes are both checked. See section 5.7.

## 5.5 Data Management

### 5.5.1 CSILoggerNet Data Cache

When data is collected from the devices in the datalogger network, it is stored on the LoggerNet server's computer in a data cache. This is simply an area of hard disk space that is used to store all of the collected tables from all of the devices. The size of the data cache is determined by the value entered into the Database Table Size Factor on the dataloggers' data collection tabs (Section 5.2.3.2), and the table size defined for each datalogger in its program. The available hard disk space on the LoggerNet server should be sufficient to accommodate the table size factors specified.

The use of a data cache has several advantages over accessing the data directly from the datalogger:

- Large quantities of historical data can be stored for each datalogger. Access to this data is not limited by the datalogger's memory capacity.
- Multiple client applications may access cached data simultaneously without necessarily initiating communication to the datalogger.
- A range of data can be retrieved by entering time stamps.

- Client applications can be advised automatically when new data has been collected.
- Data collection is more efficient since the client applications do not access the datalogger network directly, and therefore are not limited by another application's communications with the datalogger.

### 5.5.2 Collection of Data Holes

Something that should be considered when collecting data using the LoggerNet clients is the method of data collection and the concept of "holes". A hole is any discontinuity in the collected data.

Data collection using Data Advise results in the most recent records being retrieved first. Therefore, occasionally there may be holes of missing data, which the server will attempt to collect along with any scheduled collections.

There are other instances that could produce holes in the data cache:

- Corrupted data packets that result in the data being collected "out of order" by the LoggerNet server.
- Interruption of data collection.
- Computer power failure.

Data collection options allow you to ignore or to attempt to collect these holes of data. Depending upon your application, loss of data because of holes may be insignificant.

Standard data collection tries to collect all the records that the server has not yet received, starting with the next expected record number. This causes the oldest data in the datalogger to be collected first. Using this method of data collection, any holes in the data cache will not be collected, because the data has already been overwritten in the datalogger.

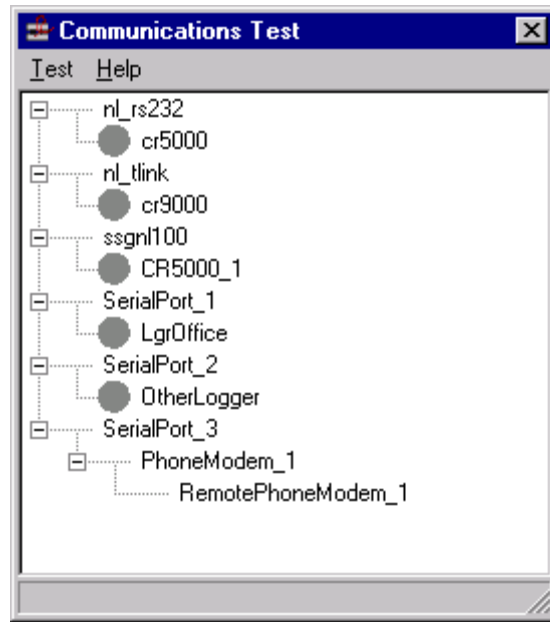
## 5.6 Communications Testing

The NetAdmin client provides two utilities for testing the datalogger network communications link. One is for testing a complete network connection to a datalogger and the other is solely for testing RF communications links. Both of these tests are accessed from NetAdmin's Options menu.

### 5.6.1 Network Communications Test

The Network Communications test is used to check the communications path to one or more dataloggers. Communication is verified by attempting to check the datalogger clock.

Selecting Options | Communications Test from the main menu will invoke a screen similar to following dialog box:



All devices in the network are displayed in the window. Each datalogger has a gray circle beside it. Once the communications test is run, the color of the circle will reflect the state of the communications link to the datalogger.

From the Communications Test dialog box, select a datalogger to be tested. When a datalogger is selected the circle beside it will turn black. If you want to select more than one datalogger, select the first and press and hold the CTRL key to select additional dataloggers. To select all the dataloggers in a device map at once, choose the menu item Test | Select All. To deselect all the dataloggers choose Test | Clear Selections.

To begin testing, select the menu item Test | Start Test. As a device is tested, the status icon to its left will change colors. Green signifies a good communications link, blue is marginal, and red is critical. When a device is being tested the icon will be yellow. If the status is unknown or the datalogger has not been chosen for testing the icon will remain gray.

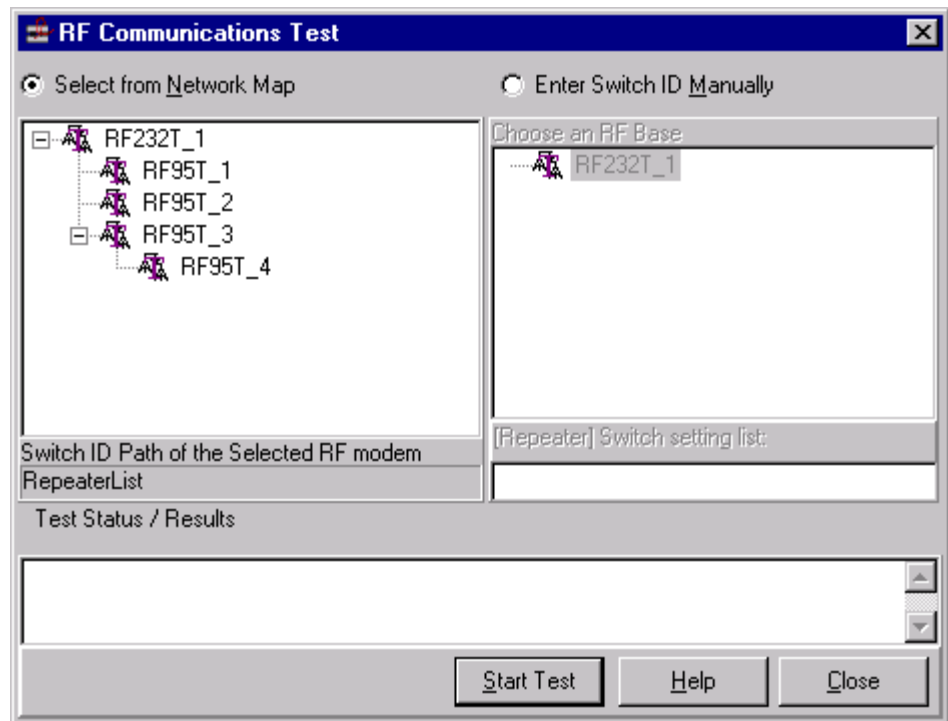
The results of the communications tests can be saved in the server's log files. From the Test menu, select Logging. When Logging is enabled the results of the communications test are placed in the server's transaction log.

## 5.6.2 RF Communications Test

To invoke the RF Communications Test dialog box, select Options | RF Communications Test from the main menu.

The RF communications test can be performed to test links established in the network map, or RF Modem switch IDs can be entered manually to test alternate communications paths.





At the top of the screen choose Select from Network Map or Enter Switch ID Manually.

To test a communications path in the existing RF map, select one of the RF devices in the communications network and press the Start Test button. The results of the test will be displayed in the Test Status/Results field. (Be patient, it may take some time for the test to complete.)

To test communications with RF modems by entering the switch IDs, first select the base to use, then enter the list of switch settings, separated by commas, for the RF modems to test. The switch ID settings for RF modems in the RF network map can be viewed by selecting the RF modem from the network map at the left.

A typical test result for an RF station with one repeater might be:

```
Prom Signature: 41125
243 35 145 59 177
243 30 144 62 178
243 28 138 58 167
```

The first entry, Prom Signature, is the PROM signature of the last RF modem in the communications link. The next strings are RF link quality reports. If there is one remote RF station, there will be two strings displayed in the quality report. The first string is an indication of the remote RF's reception and the second string is an indication of the base RF's reception. If there is an RF station with one repeater, there will be three strings displayed in the quality report. The first string is an indication of the furthestmost RF remote's reception, the second is the repeater's RF reception, and the third is the base reception.

The quality record is made up of five values:

test packet size, front of 2T envelope, back of 2T envelope, front of 1T envelope, back of 1T envelope.

The test packet size is normally around 238, but it will vary depending on network layout. If this value is half or less than 238, packets are being lost in transmission and are being resent at a smaller size.

In RF communications, data is transferred as a stream of bits encoded into short and long periods of time between transitions. The short period of time is considered "1T" and the longer period of time "2T". Each transition is expected to fall within a certain window of tolerance to be valid; if the transition is outside of this window, the packet will be resent. As a packet is received the RF modem keeps track of the transition that occurred closest to the front of the allowable window and closest to the back. These values are kept for both the 2T and 1T windows. Both windows are 204 ticks long, therefore, if a transmission were perfect, the front and back numbers of each envelope would be close to 102. The closer the front value is to 0 and the closer the back value is to 204, the worse the quality of the RF communications link.

## 5.7 Server Settings

Operational settings can be configured for the server to determine how the system responds to regional clock changes, whether the system is allowed to collect data automatically, and the generation of troubleshooting logs. These settings are accessed from the Options | Server Settings menu. Accessing this menu invokes the Server Settings dialog box, which has the following fields.

**Server Settings**

☒ Allow Automated Operations

☒ Network Communications Enabled

System Clock Options: Always Use Computer Clock

**Server Log Settings**

**Transaction Log Settings**

☒ To Disk

File Count: 5

File Size: 1200000

**Communication Log Settings**

☒ To Disk

File Count: 5

File Size: 1200000

**State Log Settings**

☒ To Disk

File Count: 5

File Size: 1200000

**Low Level I/O Log Settings**

☒ To Disk

File Count: 5

File Size: 12000000

Ok Apply Cancel Help

**Allow Automated Operations** - When this check box is selected, scheduled data collection and automated clock settings are allowed for all devices in the datalogger network. If this check box is not selected, scheduled data collection will be disabled, regardless of the individual devices' setups. The only method of data collection will be manually via the Control Panel Update Data Cache (see Section 7.7).

**Network Communications Enabled** - When this field is selected, communications between the server and the datalogger network is enabled. This option is often turned off during system maintenance to temporarily disable communications to the entire network.

#### NOTE

Before performing network maintenance on RF systems, verify that RF polling has been shut down. To manually suspend polling by an RF base, scheduled collection must be disabled for each of the dataloggers in the RF network, and communications must be disabled for the remote RF modems prior to disabling communication with the RF base. Otherwise, the RF base will continue polling and attempting to collect data until it times out (up to two hours).

**System Clock Options** - The server can be set to one of three options to determine how the time will be reflected for the LoggerNet server during regional time changes (such as daylight savings time, DST). The selected option will affect all time/date functions in the clients. The data time stamps in the datalogger will also be affected when clock sets are done to change the datalogger clock to match the server time.

- **No Adjust for DST** - The time reflected for the LoggerNet server will remain on standard time when the computer automatically adjusts for DST (default setting). This adds an offset to the computer time during DST. This setting should be used only if your computer is set to adjust for DST, but the datalogger network should not adjust for DST.
- **Always Use Computer Time** - The time reflected for the LoggerNet server adjusts for DST along with the computer. You should also use this setting if your computer is set to Greenwich Mean Time (GMT) or if your computer does not automatically adjust for DST.
- **GMT** - The time reflected for the LoggerNet server is GMT. The LoggerNet server will use the time zone setting in your computer to calculate this time value. There is no adjustment for DST. Use this setting only if your computer is not set to GMT. Unintended offsets may result if both the computer and this setting indicate GMT.

**Server Log Settings** - There are four logs that can be saved to disk. Each of these logs contains different information relative to the interaction between the LoggerNet server and the datalogger network. The log files are stored in the server's working directory. These logs are explained below.

- **Transaction Log** (tran\$.log) - This log includes information on the transactions that occur between the LoggerNet communication server and devices in the datalogger network. Examples of these types of events are clock checks/sets, program downloads, and data collection. The

transaction log can also contain messages from client applications attached to the server.

- **Communications Log** (comms\$.log) - This log records information on the quality of communications in the datalogger network. Three types of messages are recorded: status messages, warning messages, and fault messages.
- **Object State Log** (state\$.log) - This log is used for troubleshooting an object in the datalogger network. The information in this log conveys the state of an object at a given time. Information from end use applications may also be written to this log.
- **Low Level I/O Log** (io\$SerialPort\_1 log) - Low level incoming and outgoing communications for a root device (i.e., serial port) are recorded in this log. A separate log file is kept for each root device.

Appendix D provides examples of the log files and an explanation of their messages.

If the log files are being saved to disk they will have a filename as noted above.

The following settings are used enable saving the logs to disk as well as to control the number and size of the log files.

**To Disk** – Turning on this check box enables saving the associated logs to files on the server computer hard disk.

**File Count** – This setting determines the number of log files to be saved to disk for this type of log. The server will store up to the number specified before overwriting the oldest log. The \$ sign identifies the active file. Once a file reaches the specified File Size, it is saved to disk with a sequential number beginning with 0 (i.e., tran0.log, tran1.log, etc.).

**File Size** – This setting determines how big the log file is allowed to grow before being saved to an archived file.

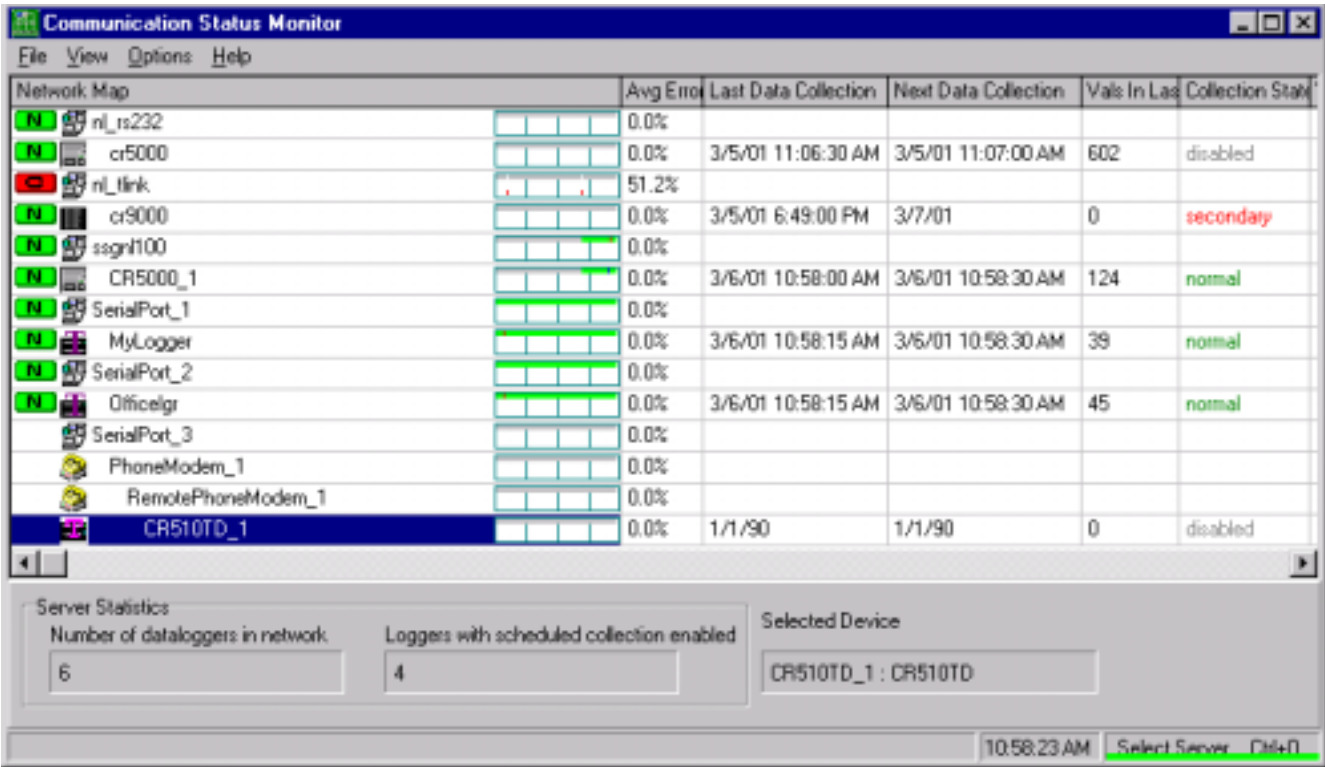
# Section 6. Communication Status Monitor



The Communication Status Monitor client provides a way to monitor communications statistics for the datalogger network. Information can be viewed about data collection attempts and communications failures by the LoggerNet server. Status information from the datalogger can also be obtained.

## 6.1 Graphic Displays

When Communication Status Monitor is first opened, the network map is displayed in the leftmost column. The name of each device is displayed, along with a status icon and a graphical histogram. Both of these provide some indication of the health of the communications link for the associated datalogger.



### 6.1.1 Status Icon

The status icon, **N**, is displayed to the left of the device name. The color and letter displayed are a way to quickly verify the LoggerNet communication server's connection with the device. A device has four states: Normal (green N), Marginal (blue M), Critical (red C), or Unknown (gray U). The status icon will change accordingly to reflect the device's current status.

## 6.1.2 Graphical Status History

The Status History is displayed to the right of the device name in the network map column. It uses color to provide a history of the communications status between the LoggerNet server and the device for the last 12 hours. As with the status icon, the four states (or colors) are: Normal (green), Marginal (blue), Critical (red), or Unknown (gray). There are 72 points in the display, each representing 10 minutes of communications. A single retry or failure during 10 minutes will change the color to blue or red.

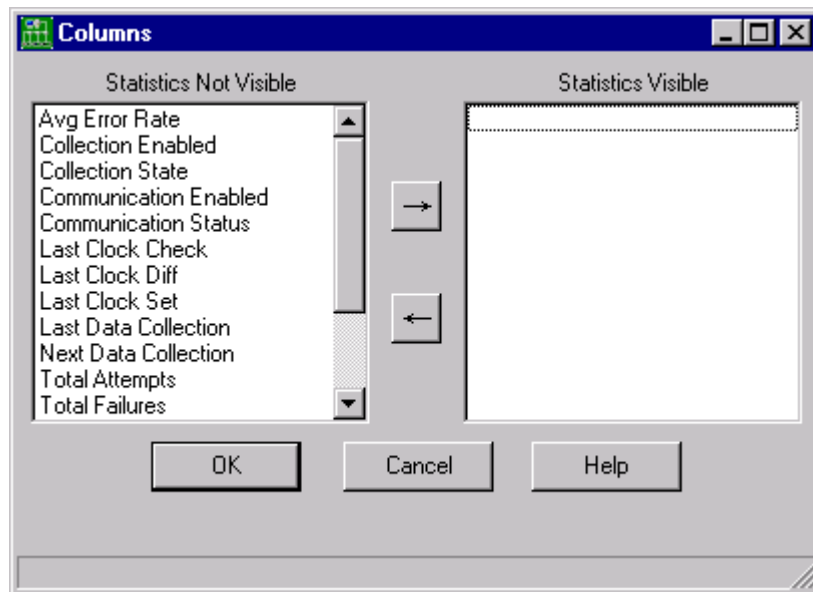
In addition, the level of the line indicates the status in terms of successful data communications. A display that shows Blue (marginal communications link) with high data communications success is probably not a problem. Even a station with significant amounts of red in the history may be getting all of the data.

### NOTE

Error messages from the datalogger are interpreted as a successful communication and will show green even with the failure.

## 6.2 Custom Status Monitoring

Communication Status Monitor can be customized by the user to display only those columns containing communications data of interest. To add columns to the Status Monitor window, select View | Columns (or right-click in the middle of the window). The Column Modification window appears.



Entries in the Statistics Not Visible field will not be displayed on the main screen. Entries in the Statistics Visible field will be displayed on the screen. The arrow buttons are used to move entries between the two columns. Alternately, an entry can be moved from one column to the other by double-clicking it.

A variety of status information and statistics are available:

- **Avg Error Rate** - A running average of the number of communication failures and retries. Each failure increments the average 5%. The error rate decreases slowly with more successful communications. This is done to allow marginal link errors to be displayed long enough for an administrator to observe.
- **Collection Enabled** - Indicates whether or not scheduled data collection is enabled for the device. (Data collection is set up and enabled in NetAdmin.)
- **Collection State** - Indicates the current state of scheduled data collection. Normal, Primary Retry, Secondary Retry, Disabled.
- **Communication Enabled** - Indicates whether or not communication is enabled for the device. (Communication is set up and enabled in NetAdmin.)
- **Communication Status** - The current status of the device: Normal, Marginal, Critical, or Unknown.
- **Last Clock Check** - The date and time of the last clock check for the device.
- **Last Clock Diff** - The amount of time the datalogger clock deviated from the LoggerNet server's clock when the last clock check was performed.
- **Last Clock Set** - The date and time that the datalogger's clock was last set to match the LoggerNet server's clock.
- **Last Data Collection** - The date and time that data was last collected from the device by the LoggerNet server.
- **Next Data Collection** - The date and time of the next scheduled data collection for the device.
- **Total Attempts** - The total number of times the LoggerNet server has attempted to communicate with the device since the server was started or the last time statistics were reset.
- **Total Failures** - The total number of times the LoggerNet server has attempted to communicate with the device but has failed device since the server was started or the last time statistics were reset.

- **Total Retries** - The total number of times the LoggerNet server has attempted to communicate with a device after the original attempt failed device since the server was started or the last time statistics were reset.
- **Values (Vals) In Last Collect** - The number of values that were collected during the last data collection attempt.
- **Values in Holes** - The number of uncollected values in holes. For further information on holes refer to Section 5.5.2.
- **Values in Uncollectable Holes** - The number of values in holes that cannot be retrieved from the datalogger. If the datalogger's memory becomes completely full it will begin overwriting its oldest data; thus, any holes of data that were overwritten will be uncollectable.

## 6.3 Obtaining Datalogger Status Information

Communication Status Monitor can be prompted to return all of the information provided in the datalogger's status table, which is stored with each program scan. With the datalogger highlighted in the device map, select View | Status Table from the menu. This will display a dialog box with a view of the Status parameters. Select the Get Datalogger Status button and wait a few moments while the LoggerNet server calls the datalogger to get its most current status. The information returned on this screen is retrieved from the datalogger's default status table (see Table 5-1 and Table 5-2, Section 5.2.3.2).

## 6.4 Monitoring Operational Logs

Three operational logs are available that provide information on the communications between the LoggerNet server and the datalogger network: Transaction Log, Communication Log, and Object State Log. The information in these logs is used for trouble-shooting communications links.

Each log is initially configured in NetAdmin (refer to Section 5.7). NetAdmin Server Settings has settings for enabling the server to save the logged information to disk, and setting the file size and number of log files.

Once in Communication Status Monitor, a display of the server logs can be brought up by selecting View | Server Logs from the menu. Individual log displays can be turned on or off by selecting the On/Off check box, or temporarily paused by selecting the Pause check box. Saving the log file to disk can also be turned on or off using the check boxes. All log files are stored to the hard disk of the computer where the server is running.

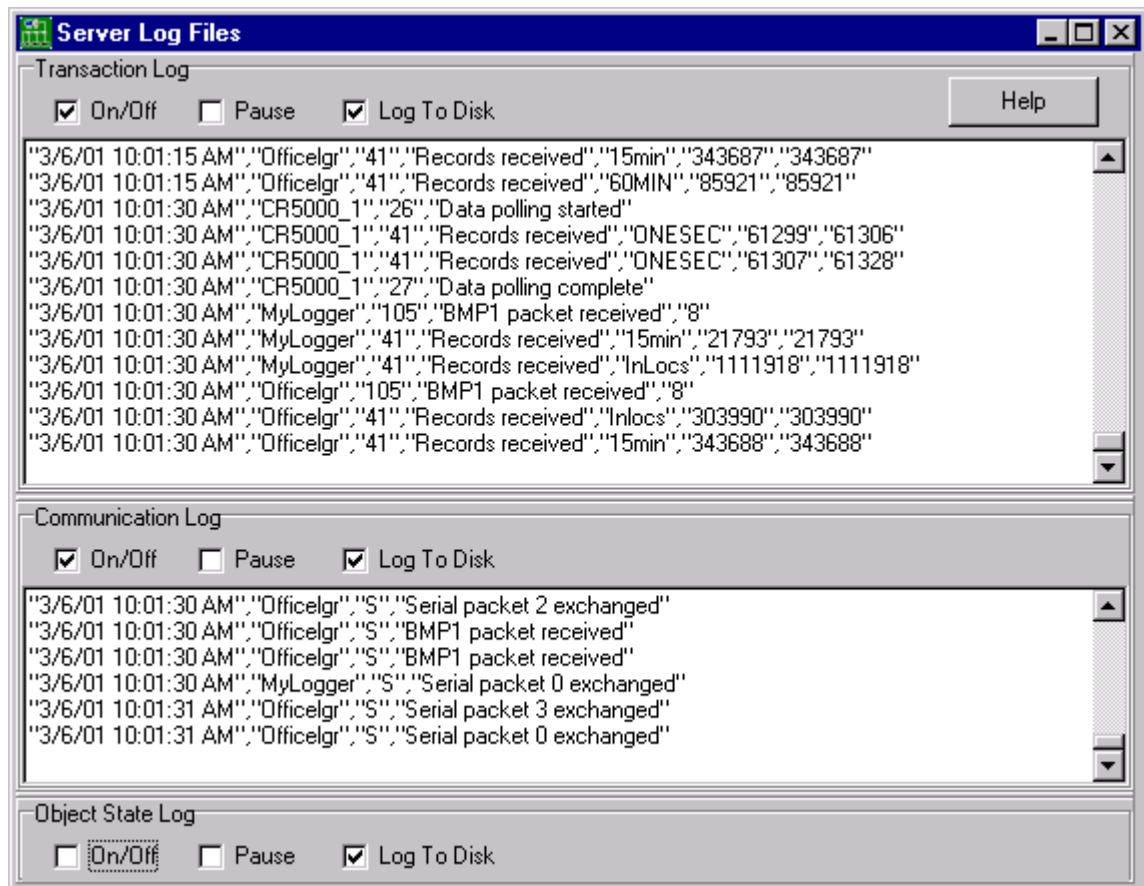
---

### NOTE

Examples of the log files, and a brief explanation of messages, can be found in Appendix D.

---





### 6.4.1 Transaction Log (TRAN\$.LOG)

This log includes information on the transactions that occur between the LoggerNet server and devices in the datalogger network. Examples of these types of events are clock checks/sets, program downloads, and data collection. When saved to disk, the filename for the log will be tran\$.log, where \$ denotes the active log, and tran1.log, tran2.log, etc. denote historical logs.

### 6.4.2 Communication Log (COMMS\$.LOG)

This log records information on the quality of communications in the datalogger network. Three types of messages are recorded: status messages, warning messages, and fault messages. When saved to disk, the filename for the log will be comms\$.log, where \$ denotes the active log, and comms1.log, comms2.log, etc. denote historical logs.

### 6.4.3 Object State Log (STATE\$.LOG)

This log is used for troubleshooting an object in the datalogger network. The information in this log conveys the state of an object at a given time. When

saved to disk, the filename for the log will be state\$.log, where \$ denotes the active log, and state1.log, state2.log, etc. denote historical logs.

## 6.5 Monitoring Low Level I/O

A low-level log is available for each serial communications device used for LoggerNet communications on the server computer. The low-level log shows all of the communication code being passed between the communication server and the devices in the datalogger network over the communications port. As with the operational logs, this log's initial settings are specified using the Server Settings in the NetAdmin Client under the Options menu (Section 5.7).

The low-level log is displayed by highlighting the serial port in the device map, and selecting View | Low Level Log from the menu. Each line in the log includes a time stamp. An R indicates information that has been received from a device in the network; a T indicates information that is being transmitted to a device.

Multiple low level log windows can be displayed at the same time so you can view low level messages for more than one serial communications device.

## 6.6 Reset Statistics

Some of the statistics displayed on the Communication Status Monitor can be reset to allow tracking of errors from a specific time. The statistics are reset by selecting Options | Reset Statistics from the menu.

Resetting Statistics will reset the Average Error Rate, Communication Status, Total Attempts, Total Failures, Total Retries and Values in Uncollectable Holes.

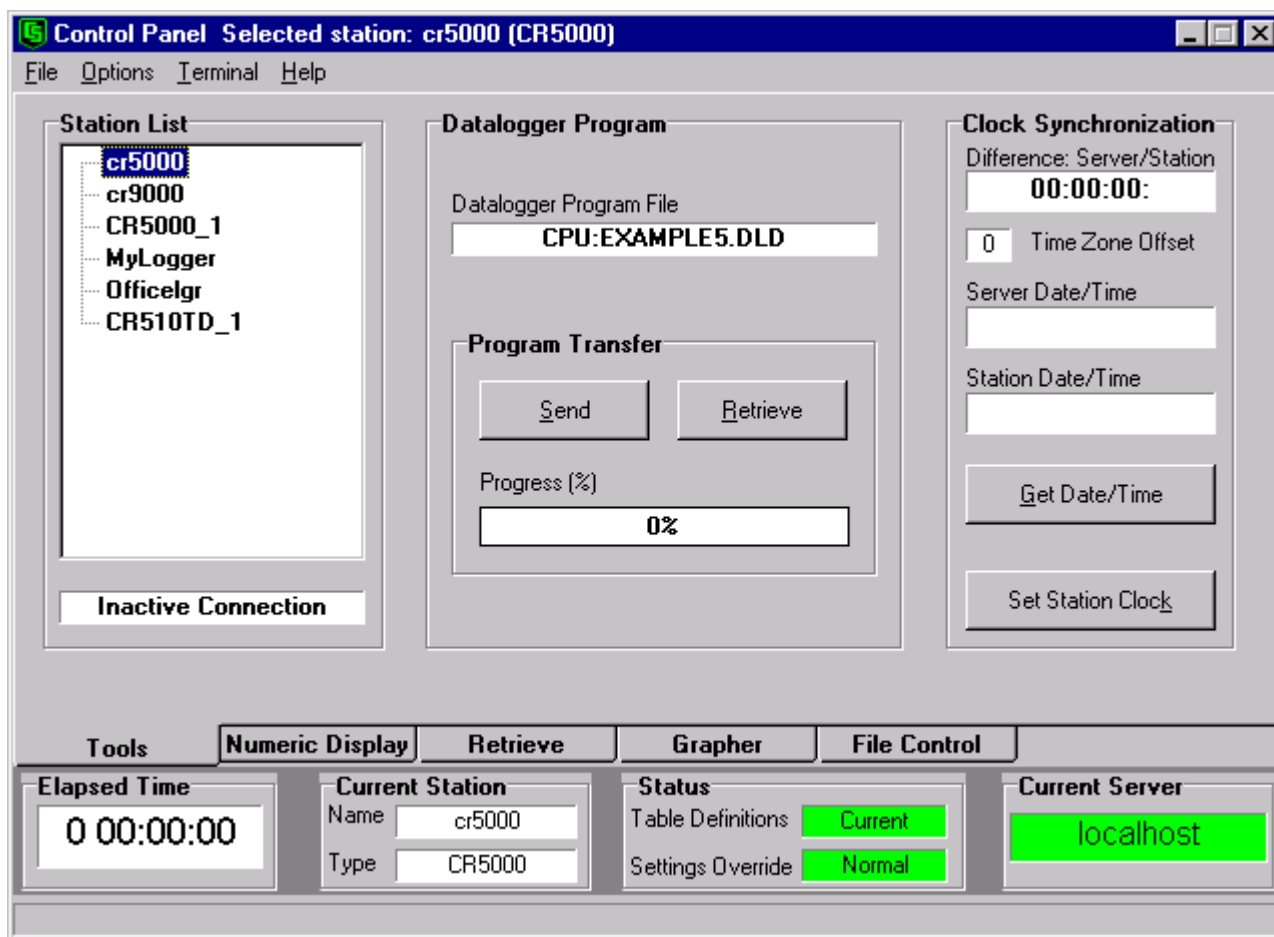
Resetting the statistics does not affect the datalogger network operation or change the operating state of any of the stations.



## Section 7. Control Panel

The Control Panel provides a near real-time connection to the datalogger network by communicating through the LoggerNet communication server to the dataloggers. Tools are provided for transferring programs to the datalogger, manually setting and checking the datalogger's clock, viewing and collecting data, and communicating with the datalogger in terminal mode.

The Control Panel is typically used by an organization's LoggerNet administrator for datalogger network maintenance and troubleshooting. Under normal circumstances, a typical end-user would not run the Control Panel to interact with the datalogger network, but would rely on one of the other clients for data collection, device status, etc. Security features are available (using the Security application, see Section 10) to restrict access to certain Control Panel functions.



### 7.1 Selecting the Datalogger (Tools Tab)

The Control Panel works with and displays data from only one datalogger at a time. A datalogger must be selected before any of the Control Panel's other tabs are active. The name of the selected datalogger appears in the text window at the bottom of the screen along with status information about

Settings Override and Table Definitions. Selecting the datalogger enables all of the Control Panel functions.

## 7.2 Program Management (Tools Tab)

After a program is created and compiled using Edlog (Section 12.2) or the CRBasic Editor (for CR5000 and CR9000 dataloggers), the Program Management utility in the Control Panel is used to transfer it to the datalogger. To transfer a program, first choose the datalogger, then select the Send button from the Tools tab. A warning will appear that the server data cache will be erased when the program is sent. Any uncollected data from the data cache should be retrieved at this time. Selecting Cancel will stop the program send process and allow you to do so.

If OK is selected at the warning, a standard file dialog box will appear from which to select the program to send. A confirmation message will appear when the program has been sent and compiled successfully. For CR5000/9000 dataloggers, the running program is cleared and the run now flag is set after the program is compiled by the datalogger. (See Section 7.8 for more detail on file control for CR5000/9000 dataloggers.)

---

### NOTE

If a program downloaded to the datalogger sets or changes the active security code, make sure to change the security setting for the datalogger in the NetAdmin application (Section 5.2.3.1). Otherwise, access to the datalogger may be limited or completely denied.

---

A program running in the datalogger can be retrieved and saved to a file if needed. This is done by selecting the datalogger and clicking on the Retrieve button from the Tools tab. A retrieved file from a CR10X-TD family datalogger can be imported into Edlog for editing by using Edlog's Document DLD feature (Section 12.2.4). Program files for the CR5000 or CR9000 can be opened directly in the CRBasic editor. This is useful if the original CSI file has been corrupted, lost, or erased.

## 7.3 Manually Checking and Setting the Clock (Tools Tab)

The schedule for an automatic clock check and set can be set up in the NetAdmin client (Section 5.2.3.4). Manual clock checks or sets can be accomplished using the Control Panel.

To check the current time, select the Get Date/Time button from the Control Panel's Tools tab. The current server date and time and datalogger date and time are displayed on the right side of the window, along with the difference in the two clocks and any time zone offset that might be in effect.

To set the datalogger's clock based on the communication server's clock, select the Set Station Clock button. A slight difference in the clocks might exist after the clock is set because of the communications time delay.

**NOTE**

If a manual clock set does not change the datalogger clock, check the allowed deviation specified in NetAdmin for the datalogger's automatic clock check (Section 5.2.3.4). If the differences in clocks is less than the specified deviation, a manual clock set will not have any effect.

Setting the clock may affect your data. Refer to Section 5.3 for further discussion.

## 7.4 Control Panel Options

The following options are provided on the Control Panel to allow a LoggerNet administrator greater access to the dataloggers and tools to manage the datalogger network. These settings each have specific warnings and cautions since they allow changes to the operation of the network and can cause the loss of data. Care should be exercised in using these options.

### 7.4.1 Connect to Station – Enable Connection Management

Connection management is enabled for the selected datalogger, by highlighting the datalogger in the network map displayed on the left side of the Control Panel window, and clicking in the “Inactive Connection” bar underneath the station list. The bar will turn yellow and indicate “Attempting Connection” while the server tries to check the clock of the selected datalogger. If the clock check is successful, Connection Management is enabled and the bar turns green with the indicator “Active Connection”. Connection Management can also be enabled/disabled using the menu item Options | Connect to Station.

Connection Management is a function built into the LoggerNet server for situations when a connection to the datalogger must be maintained.

Connection Management is used to keep the communications link active. If no communication is detected over the communications link for 15 seconds, the server will send a message to keep the datalogger on line. Otherwise, the datalogger would time out and terminate communications.

Connection Management is not needed to use the features of the Control Panel and would normally not be enabled. Two situations that benefit from enabling Connection management:

- The communications path to the datalogger includes a phone modem. Without Connection Management, each action such as a clock check, a clock set, or override settings, would initiate a new call to the datalogger. Connection Management will establish the connection and keep it active until it is terminated.
- Fast data collection is being done with a direct connection to the datalogger (especially using the SC32A). Without Connection Management the data can only be retrieved from the datalogger every 2-3 seconds.

**NOTE**

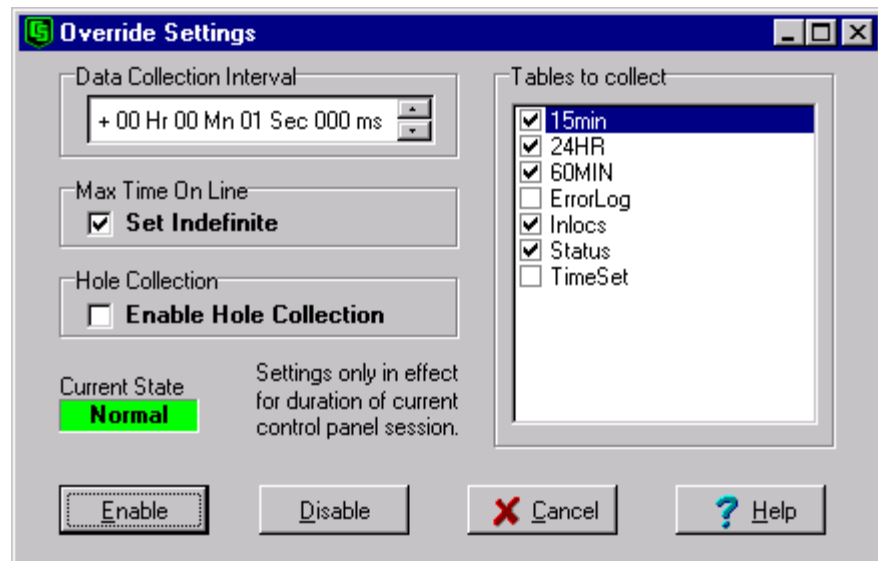
Once a connection is made to a datalogger in the Control Panel, this connection takes precedence and may prevent communications with other devices sharing the same serial port or base modem. This should be taken into consideration when using Connection Management or when enabling the Override Settings (Section 7.4.2) to change the collection interval or the tables of data collected from the datalogger.

**Connection Management cannot maintain a link to the dataloggers in an RF95T based radio network, because all communications through the RF modems are handled by the RF base. The server can only connect to the RF base.**

## 7.4.2 Settings Override

You can direct the communication server to retrieve data from the datalogger network on a schedule other than what has been defined in NetAdmin by enabling Override Settings from the Control Panel menu. This alternate data collection schedule remains in effect only for the Control Panel session in which it is enabled. Once you disconnect from the datalogger, either by selecting another station, or by closing the Control Panel, the data collection schedule reverts to the schedule set up in the NetAdmin client. If you had Connection Management (Section 7.4.1) active for the selected datalogger, canceling Connection Management will cancel the Override Settings.

To configure Override Settings, select Options | Override Settings from the menu. A dialog box similar to the one below will appear:



The new interval at which data should be collected is specified in the Data Collection Interval field. The Tables to Collect field lists all the tables in the datalogger. If a table is selected (as noted by the check mark), data will be collected from it.

Setting the Maximum Time On Line property to Set Indefinite will override any of the settings for the devices in the datalogger network, and will keep the communications link from timing out over a period of time. Hole collection (see Section 5.5.2) can also be toggled on or off.

Once configured, press the Enable button to put the settings into effect. The Override Settings field on the Tools tab will read Overridden. Settings can be returned to normal by revisiting the Override Settings window and pressing the Disable button. The Override Settings field on the Tools tab will then read Normal.

### 7.4.3 Table Definitions

The LoggerNet server retains information about the tables in the datalogger based on the Table Definitions retrieved from the datalogger. If the tables in the datalogger have changed (for example, by using a keyboard display to alter the program at the datalogger site), LoggerNet will warn the user of the change and data collection will be disabled. On the Control Panel, when the datalogger with changed table definitions is selected, a warning box will appear indicating that table definitions are no longer current and data collection has been suspended. The Table Definitions indicator will also change from green with the word Current, to red with the word Invalid.

To update the table definitions and reinitialize data collection, first save all unsaved data in the data cache to files. The data cache is erased when the table definitions are updated, because the data cache must be reconfigured to accommodate the tables found. Select Options | Advanced | Get Table Definitions from the menu to begin the table definition update.

---

**NOTE**

Even if the Table Definitions have not changed, the data cache will be erased when the Get Table Definitions function is executed.

---

### 7.4.4 Retrieve Cached Program

When the server sends a program to a datalogger, a copy of the program is kept by the server, as part of the datalogger information. The Retrieve Cached Program menu option will get the version of the program stored by the server for the selected datalogger.

### 7.4.5 Terminal Emulation

The datalogger can be put into a remote terminal mode so that the user can send low level communication instructions to the datalogger. Often this is used for troubleshooting applications, such as viewing memory settings in the datalogger or changing the program execution interval temporarily. When terminal emulation is active, all other communication with the datalogger is suspended, including scheduled data collection. Information on communicating with the datalogger in terminal mode is provided in the datalogger user's manual.

To activate the terminal emulation screen select Terminal | Terminal Emulation from the menu.

**CAUTION**

The Terminal Emulation function should be used with care. It is possible to affect the settings of the datalogger, such as changing the datalogger program and tables. Changes of this type will cause data collection to be suspended and possibly result in lost data.

## 7.5 Monitoring Data Collected from the Datalogger (Numeric Display Tab)

Values from the datalogger tables can be displayed in numerical format on the Numeric Display tab of the Control Panel. Once the Numeric Display is set up for viewing the desired values, this setup information is retained between Control Panel sessions.

**Control Panel Selected station: Officelgr (CR10XTD)**

File Options Terminal Help

	Column 1	Column 2	Column 3
Inlocs.IntTempC	24.660		
Inlocs.TCTempC	23.563		
Inlocs.BattVolt	13.922		
Inlocs.TCTempF	74.413		
Inlocs.TestSet1	345.000		
Inlocs.TestSet2	0.000		
Inlocs.TestSet3	0.000		
Inlocs.Count	330.000		
Inlocs.SineOut	-0.500		

Add Delete Cell Properties... Add Column Delete Column

**Tools** **Numeric Display** **Retrieve** **Grapher** **File Control**

**Elapsed Time**  
0 00:00:00

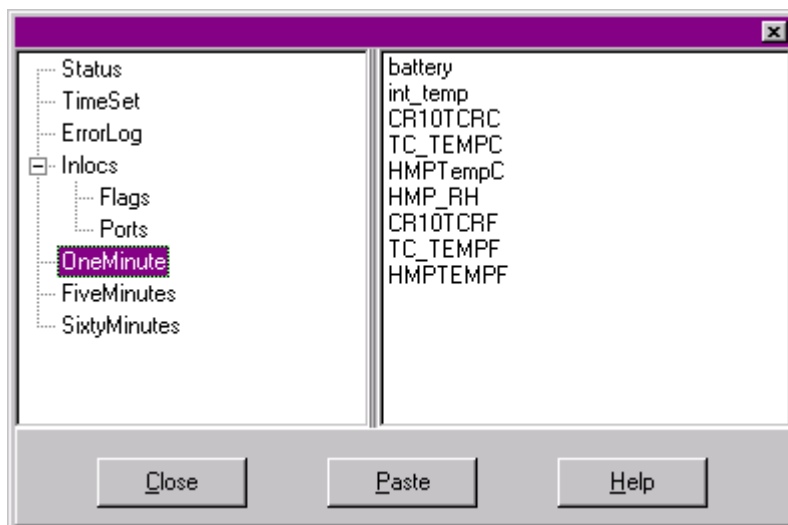
**Current Station**  
Name Officelgr  
Type CR10XTD

**Status**  
Table Definitions Current  
Settings Override Normal

**Current Server**  
localhost



When you first open the Numeric Display, there are three columns of ten cells; all of the cells are blank. The values to be displayed are chosen by first selecting a cell on the display then pressing the Add button, which brings up a tree-type view of all the tables in the datalogger.



When a table is highlighted on the left, all the data fields in this table are listed on the right. To insert a data value into the Numeric Display, highlight it and press the Paste button. Once all of the desired values have been added to the Numeric Display, press the Close button. Fields can also be highlighted and dragged to the Numeric Display.

More columns can be added to the Numeric Display by pressing the Add Column button. Up to 10 columns are available. Columns can be deleted by pressing the Delete Column button and selecting the column to be deleted from the resulting list. The Numeric Display defaults to three columns. If you delete enough columns so that less than three are displayed, they will be added back automatically to the Numeric Display.

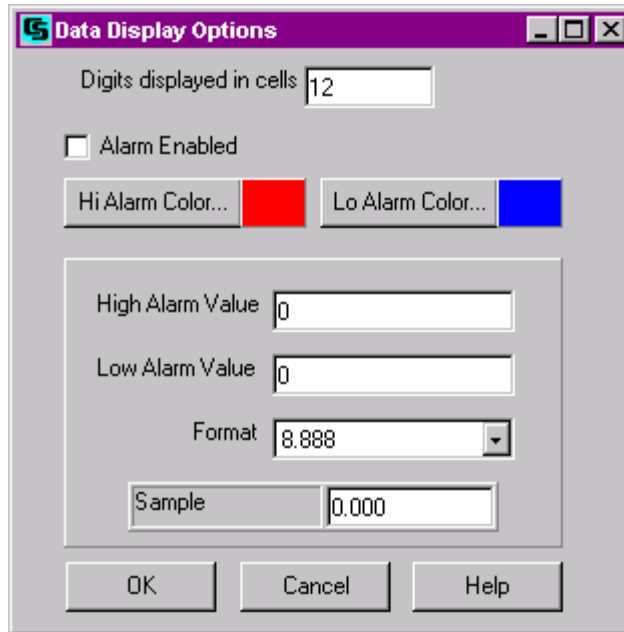
---

**NOTE**

Data must be collected by the LoggerNet server before it can be displayed.

---

The cells of the Numeric Display can be formatted by selecting a cell, or range of cells, and pressing the Options button. The Numeric Display Options window appears.



The Digits Displayed in Cells field allows you to specify the width of the display area in terms of the number of digits that can be displayed. The Format field is used to set up the number of decimal places to be displayed.

A visual alarm can be set for a cell by entering a value in the High Alarm Value field and the Low Alarm Value field. When the data value displayed in the cell exceeds the limits set, the cell color will change based on the colors selected for each of the alarm values. To turn on the alarm, select the Alarm Enabled check box.

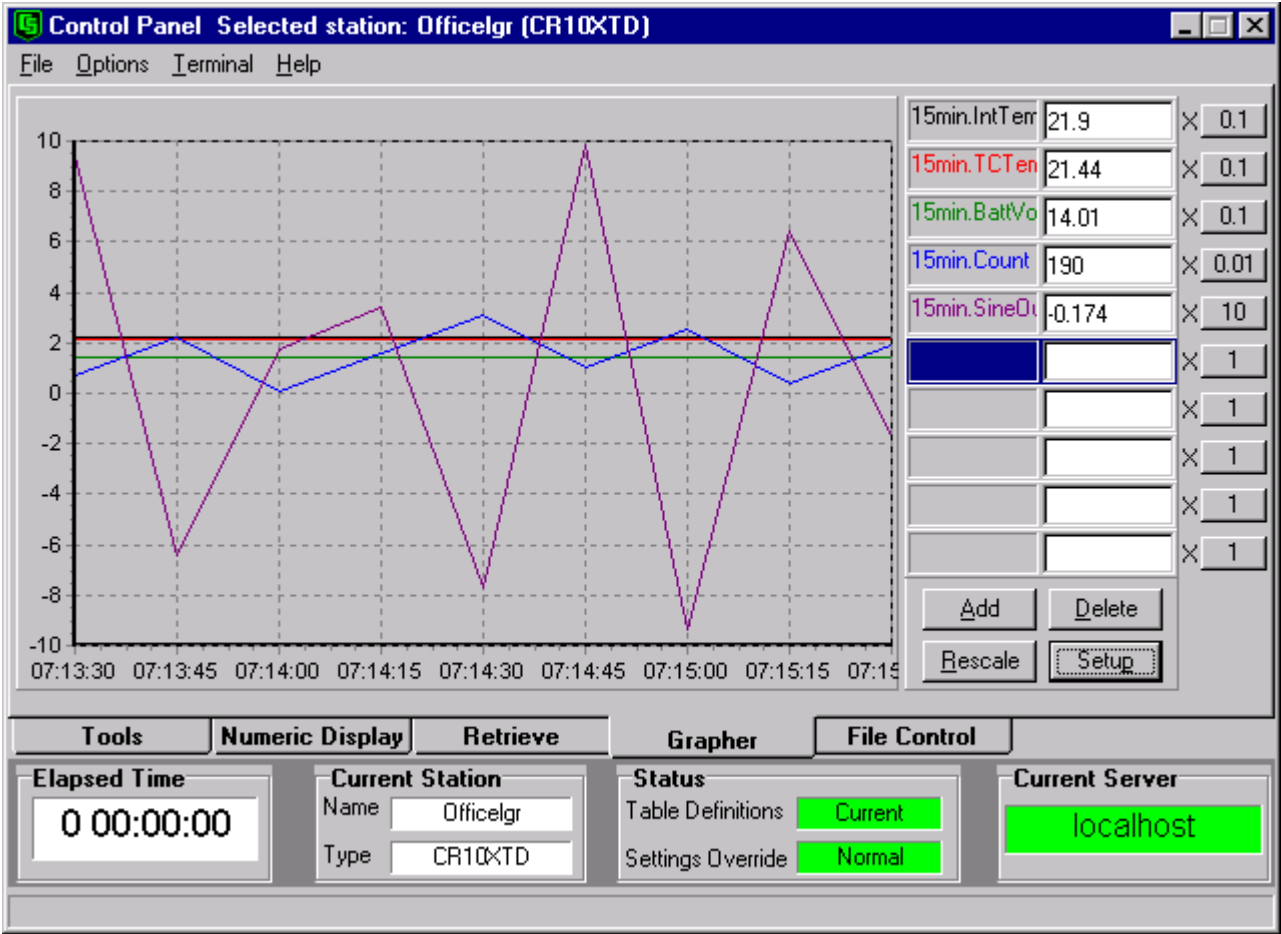
## 7.6 Graphing Data Collected from the Datalogger (Graph Tab)

Data values collected from the datalogger can also be plotted on a line graph and displayed in the Control Panel. Like the Numeric Display, the Control Panel retains the graph setups between sessions.

### NOTE

Data must be collected by the communication server before it can be graphed.

The Graphical Display is initially blank; the values to be plotted must be selected. Press the Add button to bring up the dialog box that lists the available datalogger tables and their data values, and add the values in the same manner they were added for the Numeric Display (refer to Section 7.5 above). Up to ten data values can be graphed simultaneously.



Once data values have been added, the graphical display will start automatically. If there is historical data in the server data cache, it will be displayed on the graph. Note that Input Location or Public tables typically only have two records of history in the data cache; therefore, no historical data will be displayed.

To delete data values from the graphical display, select the data values on the graphical display and press the Delete button. Adding new data values to the graphical display will overwrite existing data values.

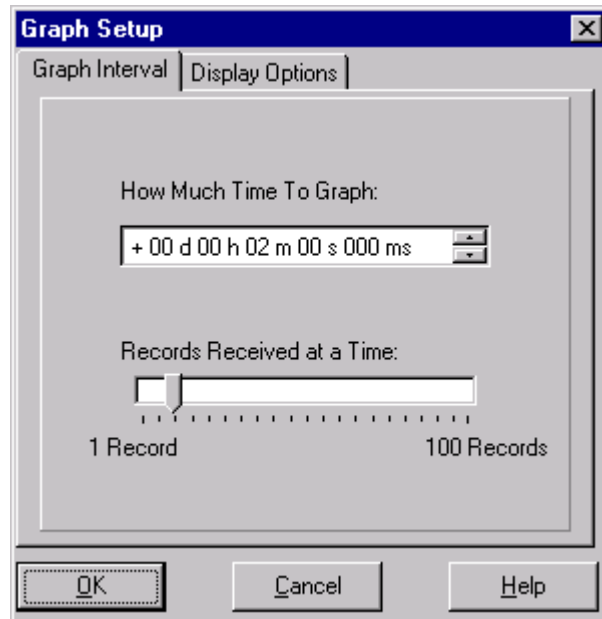
7.6.1 Graph Setup

To customize the Control Panel's graphical display, press the Setup button. The Graph Setup dialog box has two tabs.

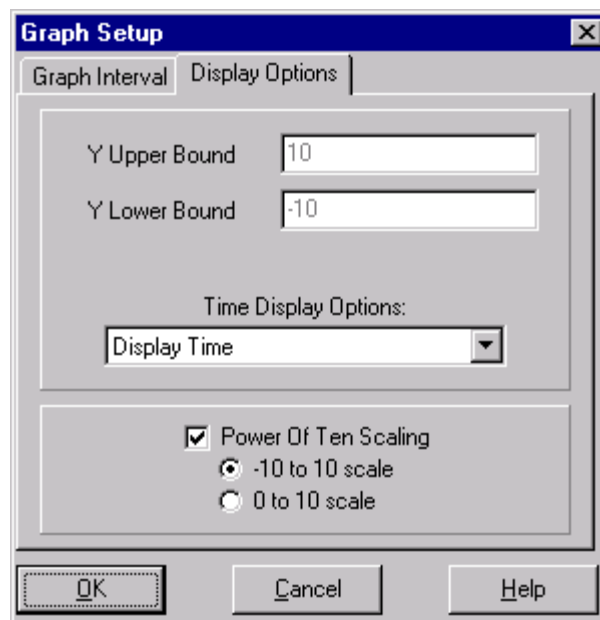
**Graph Interval** - The Graph Interval tab configures the span of time to be displayed and how often the graph will be updated.

**How Much Time to Graph** determines the amount of time displayed across the width of the graph. The default setting of 30 minutes will display 30 minutes of data if it is available in the data cache.

**Records Received at a Time** defines how many data points the graph will obtain and display for each update. If 10 records are selected, the graph will only be updated every 10 data points. This can speed up graph processing for rapidly collected data. This also reduces the load on the processor.



**Display Options** - When **Power of Ten Scaling** is selected, the graph will display numbers as a single place value times some power of ten on a scale of 0 to 10 or -10 to 10, depending upon the option chosen. When this option is in effect, pressing the Rescale Button on the Graph tab resets the power of ten so all values selected for display appear on the graph.

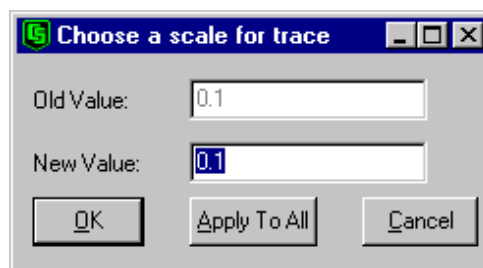


If Power of 10 Scaling is not selected, the Y Upper and Lower Bound fields are used to specify the range of the Y axis. When this option is in effect, the Rescale button on the Graph tab will be disabled. Any values that do not fall within the specified range will not be displayed, unless a scale factor is applied.

There are three Time Display Options for the X axis. You can choose to display date and time, or only date or only time.

## 7.6.2 Scale Factor

On the main graph form next to each data value being graphed is a Scale Factor button. This button indicates the scale factor that is being applied to the data being graphed. These scale factors are set automatically with Power of 10 scaling. They can also be adjusted manually. Clicking on the Scale Factor button next to the data value brings up a scale change screen. Enter the desired scale factor in the New Value text box and click on OK. You can also choose to apply the selected scale factor to all of the data values currently selected for graphing.



## 7.7 Saving Collected Data to a Data File (Retrieve Tab)

The Retrieve to File tab is used to manually copy data from the LoggerNet server's data cache to a file on the computer where the Control Panel application is running.

The Available Tables field contains a list of all tables that are being collected from the datalogger to which you are connected. Tables are selected for data retrieval by highlighting them and using the arrow buttons to move the tables into the Selected Tables field. Alternately, tables can be selected by double-clicking them.

### 7.7.1 Update Data Cache

Before initiating a data collection attempt, you can update the communication server's data cache by pressing the Update Data Cache button. This retrieves any uncollected data from the datalogger and stores it in the data cache, which ensures you have all of the available data from the datalogger. Update Data Cache will also collect the data from the datalogger for any holes in the server's data cache.

**NOTE**

Update Data Cache runs at a high priority and will postpone scheduled data collection until it has completed. This function should not be used if there are many large holes in the data pending collection.

## 7.7.2 Retrieval Options

There are three options for selecting which data to copy to a file.

- **Get New Data** retrieves only the data from the data cache that has not yet been saved to a file. The Control Panel retains information on the last data record retrieved, and begins with the very next record. If no data has been previously retrieved, all data in the data cache will be retrieved.
- **Get All Data** retrieves all of the data from the data cache regardless of whether the data has been previously retrieved.
- **Retrieve by Date/Time** is the last option. When this option is selected the Starting Date/Time and Ending Date/Time fields are enabled. These two fields are used to specify the range of data to be retrieved from the datalogger. If the specified range contains no records, a file will be saved

with a header only and no data. If the Ending Date/Time is in the future, the file will be saved with data up through the most current record in the cache. It will not wait for more data to be stored and collected.

### 7.7.3 File Formats

There are four formats for retrieved data: Table ASCII, Comma Delimited, Table ASCII Tab Delimited, and Tab Delimited. All formats store the time stamp and the record number followed by the data values. Table ASCII data is formatted with Table and Header information in the first few lines, and then the time stamp, record number, and data, are separated by commas. Comma Delimited data is similar, but does not include the header information. Table ASCII Tab Delimited data is formatted with Table and Header information in the first two lines, and then the time stamp, record number, and data, are separated by tabs. Tab Delimited data includes no header and the time stamp, record number, and data values are separated by tab characters.

The file formats provided with the formatted timestamp allow these data files to be easily imported into most spreadsheet or database applications. The interpretation of the timestamp will allow data manipulation and graphing in many of these applications. An example of importing a comma separated file into Excel is provided in Appendix G. The procedure for importing to other applications is similar.

Following are samples of each type of data.

#### **Table ASCII**

```
"TOACI1","CR10XTD_1","SixtyMinutes"
"TMSTAMP","RECNR","battery","int_temp","CR10TCRC","TC_TEMPC","HMPTempC"
"1999-06-16 00:00:00",34,13.99,25.89,24.55,24.04,23.87
"1999-06-16 01:00:00",35,13.99,25.75,24.38,23.82,23.72,
"1999-06-16 02:00:00",36,13.99,25.66,24.3,23.8,23.64
"1999-06-16 03:00:00",37,13.99,25.55,24.21,23.73,23.55
"1999-06-16 04:00:00",38,14,25.46,24.03,23.58,23.35
"1999-06-16 05:00:00",39,14.01,25.21,23.76,23.27,23.08,
"1999-06-16 06:00:00",40,14.01,25.01,23.65,23.34,23.08
```

#### **Comma Delimited**

```
"1999-06-16 00:00:00",34,13.99,25.89,24.55,24.04,23.87
"1999-06-16 01:00:00",35,13.99,25.75,24.38,23.82,23.72
"1999-06-16 02:00:00",36,13.99,25.66,24.3,23.8,23.64
"1999-06-16 03:00:00",37,13.99,25.55,24.21,23.73,23.55
"1999-06-16 04:00:00",38,14,25.46,24.03,23.58,23.35
"1999-06-16 05:00:00",39,14.01,25.21,23.76,23.27,23.08
"1999-06-16 06:00:00",40,14.01,25.01,23.65,23.34,23.08
```

#### **Tab Delimited**

```
"1999-06-16 00:00:00"    34    13.99    25.89    24.55    24.04    23.87
"1999-06-16 01:00:00"    35    13.99    25.75    24.38    23.82    23.72
"1999-06-16 02:00:00"    36    13.99    25.66    24.3     23.8     23.64
"1999-06-16 03:00:00"    37    13.99    25.55    24.21    23.73    23.55
"1999-06-16 04:00:00"    38    14      25.46    24.03    23.58    23.35
"1999-06-16 05:00:00"    39    14.01    25.21    23.76    23.27    23.08
"1999-06-16 06:00:00"    40    14.01    25.01    23.65    23.34    23.08
```

**Table ASCII Tab Delimited**

```
"TOACI1","CR10XTD_1","SixtyMinutes"
"TMSTAMP","RECNR","battery","int_temp","CR10TCRC","TC_TEMPC","HMPTempC"
"1999-06-16 00:00:00"    34    13.99  25.89  24.55  24.04  23.87
"1999-06-16 01:00:00"    35    13.99  25.75  24.38  23.82  23.72
"1999-06-16 02:00:00"    36    13.99  25.66  24.3   23.8   23.64
"1999-06-16 03:00:00"    37    13.99  25.55  24.21  23.73  23.55
"1999-06-16 04:00:00"    38    14     25.46  24.03  23.58  23.35
"1999-06-16 05:00:00"    39    14.01  25.21  23.76  23.27  23.08
"1999-06-16 06:00:00"    40    14.01  25.01  23.65  23.34  23.08
```

### 7.7.4 File Names and File Save Modes

When data is retrieved from the data cache, by default the data files are saved in the LoggerNet\Data working directory. The directory where the files are saved can be changed, by selecting the Browse button to the right of the Output Directory field, and making a new directory selection.

A separate file is created for each selected table. The name of the file is based on the datalogger name and the table name. For instance, the file name LgrOffice\_15min\_01.dat indicates that the data in the file came from the 15min table of the LgrOffice datalogger, and it is the first file with this name saved in this directory. Though the directory in which the file is saved can be changed, the name of the file cannot.

There are three options available for how the file is saved when data is retrieved: Append to Files, Create New Files, Overwrite Existing Files.

When **Append to Files** is selected, the first time data is retrieved for that datalogger in the selected directory, a new file is created. Thereafter, all new data retrieved is appended to the existing file.

When **Create New Files** is selected, all data retrieved is saved to a file with the same name, but with a numerical suffix of \_01, \_02, \_03, etc. In the above example, the second file saved would be named LgrOffice\_15min\_02.dat.

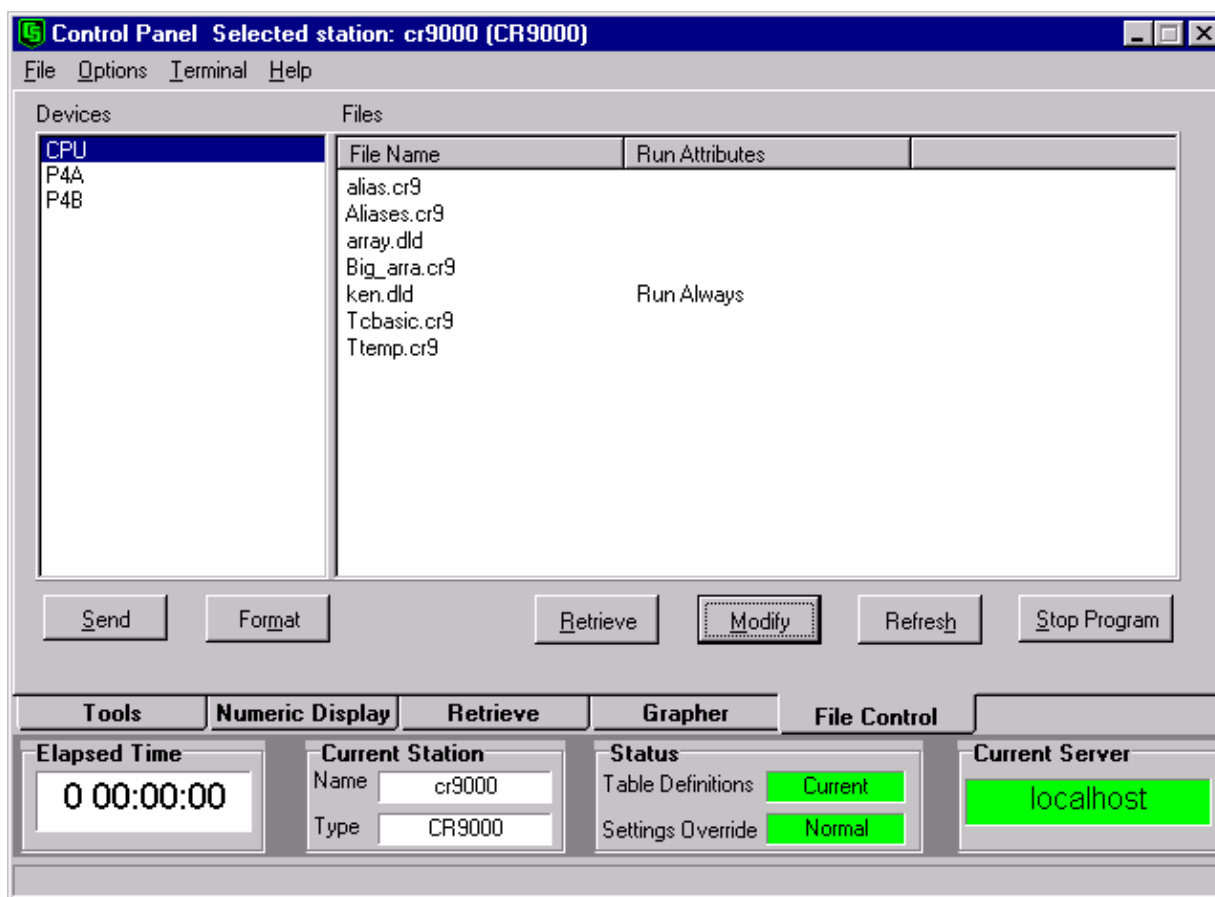
When **Overwrite Existing Files** is selected the old file will be overwritten with a file containing the new data.

## 7.8 File Management for CR5000 and CR9000 (File Control Tab)

The File Control tab is used to display a list of files stored on a CR5000 or a CR9000's CPU or PC card. The window on the left lists all of the data storage areas available for the selected datalogger. Selecting one shows a list of the files stored in that area.

If you attempt to access this tab while a datalogger other than a CR5000 or CR9000 is selected, a message will be displayed that the functions on this tab are not available for the selected datalogger type.





The run attributes for a file indicate whether it is set to Run Always, Run Now, or Run On Power-up. The currently executing program is indicated by the Run Now attribute.

There are several options to work with the files and directories on the CR5000 and CR9000 dataloggers. Highlighting a storage area and clicking on Send brings up a standard file selection dialog box. A new file can be chosen to send to the highlighted area.

On the example screen above, the CPU card is the selected device. There are a variety of files that can be sent to the datalogger. These include datalogger operating system, datalogger program, and data files.

#### NOTE

Because of the number of special operations that run in the background that don't happen in a normal file transfer, a datalogger operating system should be sent to the datalogger using the Send button on the Tools tab of the Control Panel.

**Format** is used to format the selected device. Just like the format on a computer, all of the files in the file storage of the device are deleted and the device storage is initialized.

**Retrieve** will get the selected file from the datalogger and store it on the computer where the Control Panel is running. A Save As dialog box comes up allowing you to specify the directory and file name for the saved file.

**Modify** brings up a dialog box to change the attributes of the selected file. The device name and filename appear at the top of the dialog screen. Run Now will make the selected file the active datalogger program. Run on Power Up will run the selected program the next time power is cycled to the datalogger.



**Refresh** will update the list of files for the selected device. This is used when the files on the datalogger have changed because of files being created by the datalogger or file attributes being changed by the datalogger under program control.

**Stop Program** halts execution of the currently running datalogger program. This is accomplished by loading a blank program into the datalogger.

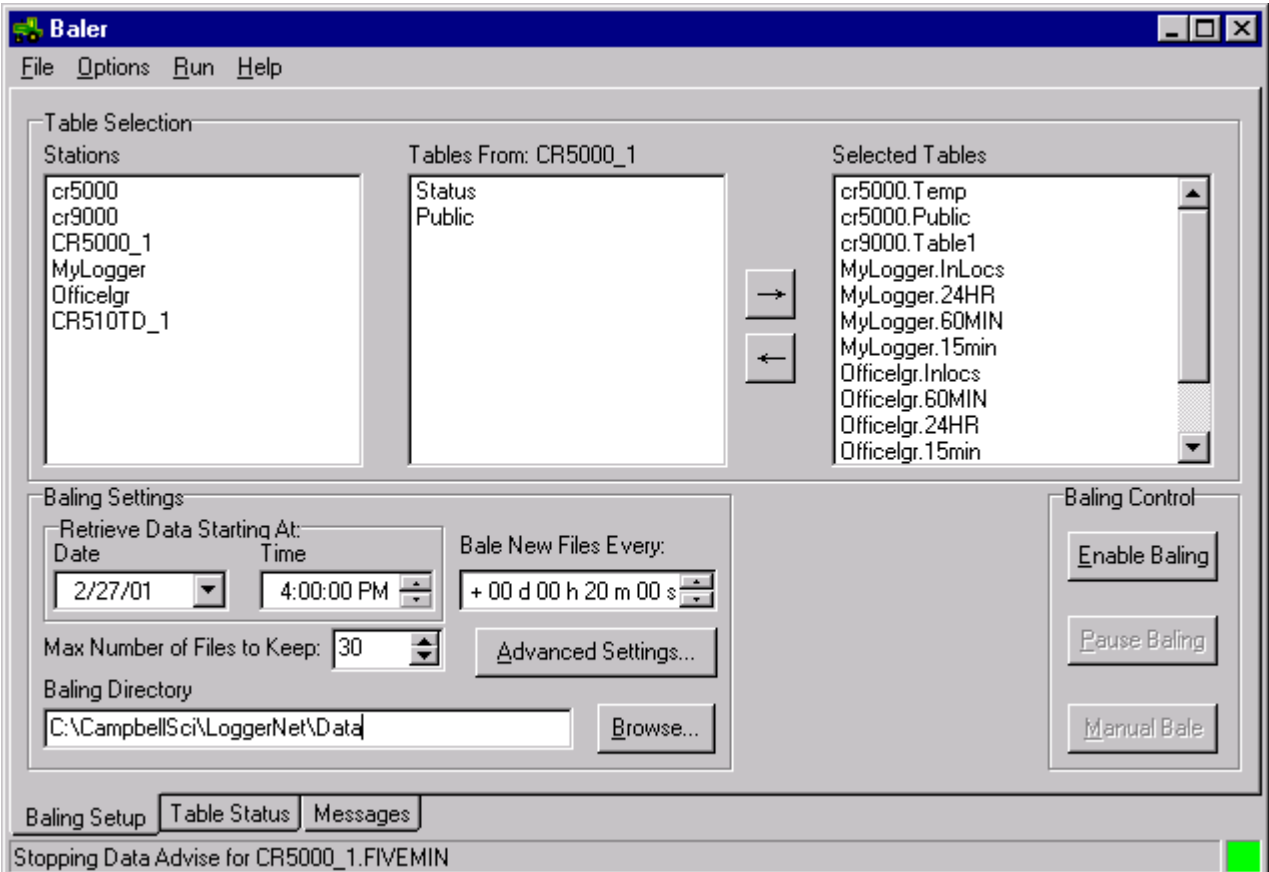
## Section 8. Baler - Timed Save to Data File

*Baler is a client application that allows you to set up a schedule on which data will be retrieved from the LoggerNet server's data cache and saved to a file in a Table ASCII format (refer to Section 7.7.3). This format is most often used with RTMS, but saved files can also be imported quite easily into most spreadsheets for further analysis. The Baler will create a separate file for each selected table.*

*The term "baling" is used in this application because of how files are saved to disk. The user specifies an interval on which data will be retrieved from the data cache, and once "baling" is started, data is saved in interval-sized packets or "bales".*

### 8.1 Baler Settings

The first step in setting up the Baler is to select the tables that will be saved to file. Each datalogger in the network will be listed in the Available Dataloggers field. When a datalogger is selected with the mouse pointer, all of its tables will be listed in the Tables From field. To select the tables that will be saved to file, highlight them with a click of the mouse pointer and use the arrow buttons to move them into the Selected Tables field.



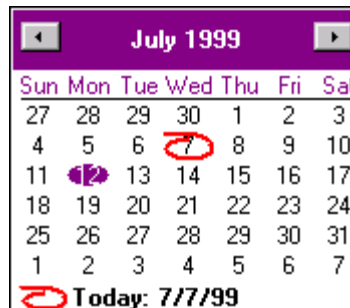
The screenshot shows the Baler application window with the following components:

- Table Selection:**
  - Stations:** cr5000, cr9000, CR5000\_1, MyLogger, Officelgr, CR510TD\_1
  - Tables From:** CR5000\_1 (selected), Status, Public
  - Selected Tables:** cr5000.Temp, cr5000.Public, cr9000.Table1, MyLogger.InLocs, MyLogger.24HR, MyLogger.60MIN, MyLogger.15min, Officelgr.InLocs, Officelgr.60MIN, Officelgr.24HR, Officelgr.15min
- Baling Settings:**
  - Retrieve Data Starting At:** Date: 2/27/01, Time: 4:00:00 PM
  - Bale New Files Every:** + 00 d 00 h 20 m 00 s
  - Max Number of Files to Keep:** 30
  - Baling Directory:** C:\CampbellSci\LoggerNet\Data
  - Buttons:** Advanced Settings..., Browse...
- Baling Control:**
  - Enable Baling
  - Pause Baling
  - Manual Bale
- Bottom Bar:** Baling Setup | Table Status | Messages
- Status Bar:** Stopping Data Advise for CR5000\_1.FIVEMIN

Only tables that are enabled for scheduled data collection can be selected for baling. All of the tables for the datalogger are listed, whether enabled for collection or not. If you try to select a table that is not enabled for collection a message will be displayed stating that the chosen table has not been enabled for scheduled collection.

All tables available for baling from all of the dataloggers in the network can be selected from the menu by selecting Option | Select All Tables.

Next, set the initial time to start baling and the interval on which to bale. If the arrow button to the right of the Date field is selected, a calendar will appear.



The current date is enclosed in a red circle and is listed at the bottom of the calendar. The date on which to start baling is indicated by a solid circle. To select a date, double-click it with the mouse pointer. The arrow buttons to the left and right of the Month/Year can be used to move forward or backward in the calendar, month-by-month.

If you do not wish to use the calendar, you can type the date directly into the Date field.

The time to start baling is specified by the entry in the time field. A time element is selected by clicking on it. The tab key will move the cursor to the next element. The arrow buttons to the right of the time field can be used to increase or decrease the selected time element, or you can enter a time into the field directly. Note that all subsequent baling attempts will be based on the time set up in this field.

The schedule for how often baling will occur is entered into the Bale New Files Every field. This field is in the format 00 d(ays) 00 h(ours) 00 m(inutes) 00 s(econds). As with the time field, the arrow buttons to the right of the field can be used to scroll through each time element, or the interval can be entered into the field directly.

As an example, the Time and Interval shown above would set the first baling event to begin at 4:00 p.m., February 27, 2001, and subsequent baling events would occur every 20 minutes. Each file saved to disk would contain 20 minutes of data. If there is no new data available in the data cache for a particular table, no bale file will be created for that table.

**NOTE**

With the above set up, if a record were written to a table every 24 hours, the file would be saved only when a new record was found; i.e., only once a day.

## 8.2 File Naming and Directory Structure

A subdirectory is created in the chosen base directory for each datalogger from which data will be retrieved. The subdirectory name reflects the name of the datalogger. The file naming convention used by the Baler is based on the table name and the current date. For instance, the file name OneMinute\_1999\_06\_16\_0001.dat indicates that the table name in the datalogger was OneMinute, the file was created on June 16, 1999, and it is the first file saved that day for this datalogger's table. Each time a Baling event occurs on a particular day, the file number is incremented (i.e., the next file to be saved would be OneMinute\_1999\_06\_16\_0002). The user does not have the option to change the file name, but can specify the directory.

The default base directory for saving files is in a directory called Baled\_Files in the c:\Campbellsci\LoggerNet\Data working directory. The user can change the base directory by pressing the Browse button to the right of the Base Directory field and selecting a new directory from the resulting dialog box.

## 8.3 Number of Files to Keep

The Baler is used to retrieve data from the LoggerNet server's data cache, and save this information to disk. However, it is expected that this data will be used for archive and analysis purposes, and that the data files will be consolidated or manipulated and saved to a different filename or a different device.

To prevent filling up your hard disk with data files, the Baler has a Number of Files to Keep per Table field. As the field name suggests, enter the maximum number of files that should be saved to the hard disk for each table. When this maximum number of files is reached, the oldest files will be deleted as newer files are saved.

## 8.4 Baler Operation

When Baling is enabled, temporary files are created as soon as data is available in the data cache. As new data is collected by the server, these records are placed in this temporary file and are saved to a permanent file when the baling event occurs. If the Baler is paused or a Manual Bale is initiated, the temporary files will be saved to a permanent file, just as if the baling interval had been reached. When Baling is resumed, a new file will be created.

### 8.4.1 Baler Controls

The Baler Controls can be accessed from buttons or the Run menu and include:

- **Enable Baling** - Begin baling on the defined schedule. This control will be disabled if the baling schedule is active. If any changes are made to the

baling interval or selected table settings, the baling schedule will be turned off and the control will become active again.

- **Pause Baling** - Pauses the baling schedule. A set of baled files will be created. The control will be disabled if the baling schedule is not currently enabled.
- **Manual Bale** - Immediately initiate a baling attempt, regardless of the schedule. This control is only enabled when baling is enabled. If the baling schedule has been paused, the control will be disabled.

## 8.4.2 Table Status and Messages

Once baling has been initiated, you can review the status of the tables set up for baling by switching to the Table Status tab. This tab provides a list of the tables selected for baling, the time and number of records saved the last time the table was baled, and whether or not the table was included in the last baling event.

The Messages tab displays the status messages that indicate baler operation. Messages showing baling events as well as communications with the server are displayed. The most recent message is also displayed on the status bar at the bottom of the application. The messages can be optionally included in the server's transaction log by selecting Options | Log Baling Status.

## 8.4.3 Automatic Start

If Options | Automatic Start is enabled, the baler will attempt to connect to the server and start baling as soon as it comes up. This will allow an automatic restart to be set up in case of a power failure or machine crash.

### NOTE

---

Automatic Start cannot be used if security is enabled for the LoggerNet server. Security requires a username and password login.

---

## 8.4.4 Command Line Options

Multiple instances of the baler can be run, and separate directories can be specified for the data files. This is done by setting up a separate shortcut for each instance to be run and using a command line option to start the Baler. The syntax is shown in the following command line:

```
c:\Program Files\...\Baler.exe directory pathname
```

where "directory" is a keyword indicating that the next parameter "*pathname*" is a valid directory path on the computer file system. Each instance of the Baler started in this manner will save its settings in a separate \*.ini file. This initialization file is saved to the directory specified by the "*pathname*" command line argument.

For example a shortcut with the following as the command line in the “Target” window would start the Baler using the initialization file stored in the directory “c:\Campbellsci\LoggerNet\Baler1”.

c:\Program Files\Campbellsci\LoggerNet\Baler.exe directory c:\Campbellsci\LoggerNet\Baler1

## 8.5 Advanced Settings

The Baler includes some advanced settings that allow the user to control how data is retrieved from the communication server and saved to files. These settings are accessed by selecting Options | Advanced Settings from the Baler menu.

### 8.5.1 Collection Order

The Collection Order setting has two options. When Wait for Uncollected Data is selected, if there are holes in the data (Section 5.5.2), data will not be baled until these holes have been filled. If Bale Without Uncollected Data is selected, data will be baled on the defined schedule, whether or not any data is missing from the LoggerNet server's data cache. Any data in holes that is collected after the baling event occurs is placed in the next baled file. Therefore, when using this option data could be collected out of order, but no data will be lost.

### 8.5.2 Select the Baling Starting Point (Retrieval Options)

If the Baler application is closed or the Baling schedule is paused, there may be data that should have been baled when the application was not active. There are two options to determine where baling will begin when program operation is resumed.

When Start From the Last Record Collected is selected, any data records that have been missed since the last baling event will be retrieved to file.

---

**NOTE**

The first time the Baler is run, if Start From the Last Record Collected is selected, it will retrieve all data available in the data cache.

---

When Start Retrieval from Specified Date is selected, those baling cycles that have been missed since the specified starting time will be retrieved to file.

### 8.5.3 Historical Data Storage Selection

If the LoggerNet server has been retrieving data for some time but the Baler has not been active, there will be historical data in the data cache. There are two options for how this historical data will be saved to file. The user can choose to save all of the data in one large file (Bale Historical Data into One Data File) or save the data in interval-sized files (Bale Historical Data into Interval Sized Files).

### 8.5.4 Launching a Program

The user can specify a program to run after a baling event occurs. This program can be an executable file or a batch file. To set up a program to be run, first enter a file name in the Program to Execute field. You can type in the program name directly or press the Browse button below the field to invoke a standard file dialog box to search for a file. For the execution of the file to occur, the Enable Program Execution check box must be selected.

---

**NOTE**

The program to execute must be just the executable program name. The use of command line arguments is not supported. If a program needs to be run with command line arguments, create a batch file with the executable and arguments, then run the batch file from the Baler.

---

## 8.6 Importing Files into Other Applications

Files saved to disk in the comma separated format can easily be imported into a spreadsheet program for analysis or manipulation. Most spreadsheet programs will allow importing comma separated data and will handle the time and date field as a timestamp. Instructions are given in Appendix G for importing a comma separated file into Microsoft Excel. Similar procedures will work with most other spreadsheet or database applications.



# Section 9. Socket Data Export

---



*The Socket Data Export client provides a way to export the data collected by the LoggerNet communication server to another computer program. In this role the Socket Data Export application acts as a conduit to pass the data as it is obtained by the server to a data client application supplied by the user.*

## 9.1 Functional Overview

The Socket Data Export application functions as both a client and a server. It is a client to the LoggerNet server and gets data from the LoggerNet data cache. It works as a server to provide data to customer supplied client applications.

The data to export is selected by enabling or disabling specific tables for each of the stations in the datalogger network. When a data table is enabled for export, every record that the server collects from dataloggers for that table is sent out by the Socket Data Export. If a table is selected but for some reason there is no data being collected by the server, no data will be sent.

There are four options for determining the operation of the Socket Data Export.

**RTMS Format or Standard Format** - Select the format for the records sent out. RTMS format is provided for compatibility with RTMS Acknowledged Named Pipe export. The protocols for both formats are described in Sections 9.5 and 9.6.

**Get All Data** – this option causes the Socket Data Export to get all of the data available in the server data cache for the selected tables. This is done each time a data client is connected to the Socket Data Export application. When this option is disabled Socket Data Export keeps track of what records have been received and acknowledged. When the data client disconnects and reconnects to the socket, the records to be sent will resume with the next record.

**Wait for Holes** – when this option is specified, the data will be exported in record order. If there are holes in the server's data cache Socket Data Export will wait for the holes to either be filled or abandoned before exporting newer records. If this option is disabled, the data records are sent out as they are collected by the server.

**View | Port Number** – allows you to change the setting for the current IP port ID if no data client is connected. If a data client is attached to Socket Data Export, you can only view this setting.

You can run multiple instances of the Socket Data Export application by specifying a different initialization directory for each instance. This is done by adding the directory information to the command line of the shortcut that starts the application. An example of this command line would be:

c:\Program Files\...\SocketDataExport.exe directory *pathname*

where “directory” is a keyword indicating that the next parameter “*pathname*” is a valid directory path on the computer file system. Each instance of Socket Data Export started in this manner will save it’s setting in a separate \*.ini file. This initialization file is saved to the directory specified by the “*pathname*” command line argument.

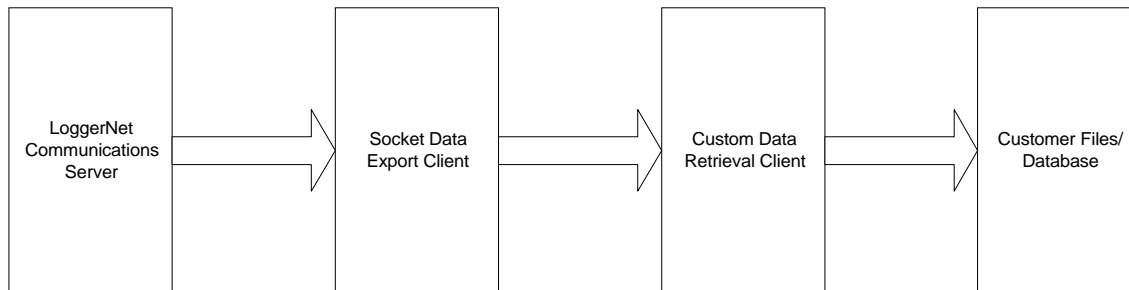
For example a shortcut with the following as the command line in the “Target” window would start Socket Data Export using the initialization file stored in the directory “c:\Campbellsci\LoggerNet\SD1”.

c:\Program Files\Campbellsci\LoggerNet\SocketDataExport.exe directory c:\Campbellsci\LoggerNet\SD1

## 9.2 Theory of Operation

The Socket Data Export client is used in conjunction with TCP/IP Berkeley sockets network transfer protocol to transfer datalogger records from one computer (or process) to another. In this role the Socket Data Export is acting as both a client and a server. The Socket Data Export Client attaches to the LoggerNet communication server and gets the selected data from the data cache. It then makes this data available for retrieval on a TCP/IP socket. The computer program that retrieves the data (the custom data retrieval client application) must connect to the provided socket. The Socket Data Export application acts as a server for the custom data retrieval client.

The most typical use for the Socket Data Export functionality is a situation where the customer has a database or file system that is already integrated with data management procedures. The custom data retrieval client gets the data from the socket provided by the Socket Data Export and writes it to the customer’s database or file.



The LoggerNet server has the responsibility to see that every collectable record is collected from the network of dataloggers. The collected data is stored in the data cache of the server. When the Socket Data Export client is first initialized it sets up the socket and then waits for a data retrieval client to connect. Once the data retrieval client connects, the Socket Data Export client gets the records for the selected tables from the server data cache, and sends them one at a time to the custom data retrieval client.

To ensure that all of these records are transferred to the client, Socket Data Export uses an acknowledgment scheme. The basic idea behind the protocol is that as each record is sent to the client, the client will report the Station Name,

Table Name, and Record Number back to the server after it has secured that record. The server uses the acknowledgment to mark the progress of the transfer. When the session is broken, or if the Socket Data Export doesn't receive the acknowledgment, the unsent records remain in the LoggerNet server's data cache. The Socket Data Export maintains transfer progress information on disk so that if the server goes down or there is another problem with the transfer, it can recover and continue to transfer all collectable records.

The record acknowledgment allows Socket Data Export to ensure that every record it intended to send was successfully received by the client. This capability, coupled with reasonable algorithms that make sure the LoggerNet server receives every record logged by the datalogger, allows for reliable data collection.

## 9.3 Custom Data Retrieval Client

Because there are so many different types of database applications and data handling processes in use, the data retrieval client must be created either by the customer or on contract with Campbell Scientific to the custom specifications of the user's process.

The custom data retrieval client is a software application that connects to the socket provided by the Socket Data Export application. It can be programmed to run on any computer platform that is configured to support TCP/IP as long as there is a computer network connection available to the host computer where the Socket Data Export application is running.

When a connection is established, the Socket Data Export will send one data record as soon as it is available. The first data record sent depends on the Socket Data Export option settings.

When the data retrieval client receives the record, it must parse the data and return the acknowledgment message to the Socket Data Export. The acknowledgment message consists of the name of the datalogger, the name of the table, and the record number of the record received.

If the acknowledgment message is not returned within 60 seconds or if the message is incorrect, the Socket Data Export will re-send the same record again. It will continue sending the same record at 60-second intervals until either the connection is broken or a valid acknowledgment for that record is received.

The custom data retrieval client is programmed to write to the database or file system defined for the user's data handling process.

## 9.4 Custom Client/Socket Data Export Interface Description

This section details the interface for writing a custom data retrieval client that will get data from the Socket Data Export application. The programming

concepts presented assume a familiarity with programming software applications to connect with a TCP/IP socket.

The Socket Data Export application functions as a server providing a TCP/IP socket connection for one remote client. Once the Socket Data Export has connected to the LoggerNet communication server the TCP/IP socket is established and the Socket Data Export application starts “listening” for a client to attach. As soon as a data retrieval client connection is detected the Socket Data Export application sends the first record out over the socket connection. Upon receiving the record the data retrieval client needs to send back an acknowledgment message consisting of the datalogger name, table name and the record number of the received record.

If the Socket Data Export application loses its connection with the LoggerNet communication server, it will need to be re-connected before any records can be obtained and sent out.

There are two record formats used to send the record data:

RTMS format - this format is provided for backward compatibility for customers who had the RTMS system and developed data handling procedures that use the format provided by the RTMS socket export.

Standard format - this format provides an easily interpreted data string containing the data along with format information for each data field.

Details on these formats are provided at the end of this section.

The following illustrations show the state diagrams for the custom client/Socket Data Export interface. (The diagramming notation is by Booch[1] who claims to have adopted it from Harel [2]).

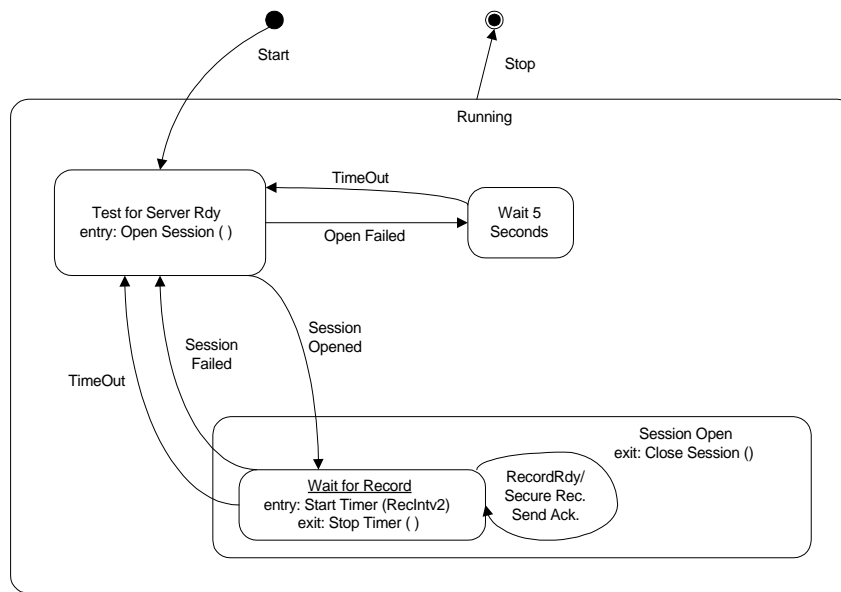


FIGURE 9-1. Client State Diagram

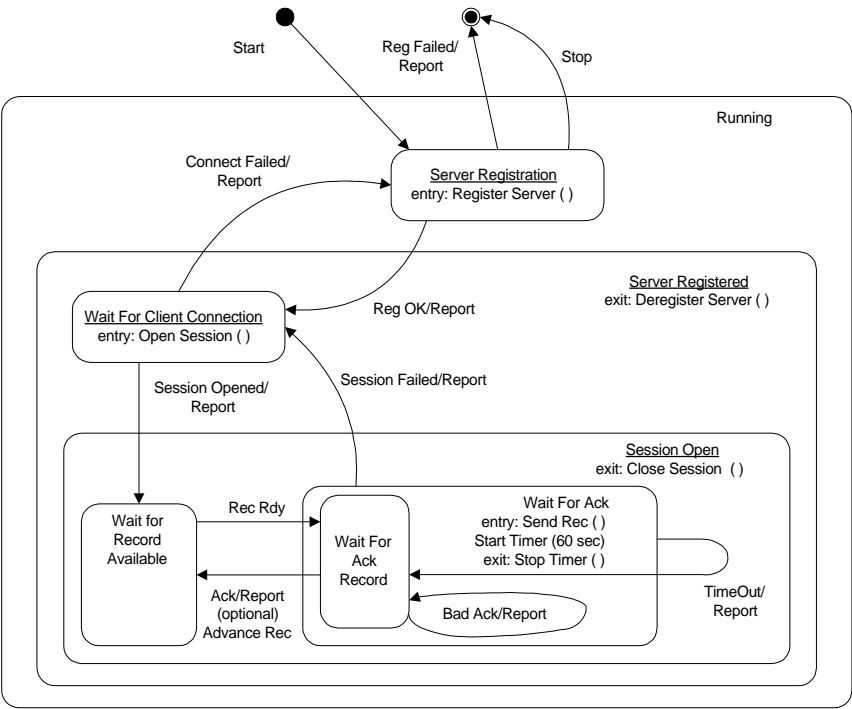


FIGURE 9-2. Socket Data Export Server State Diagram

Key concepts from the state diagrams are shown in the following tables with key words from the diagrams. In these definitions the “server” refers to the Socket Data Export data server and the “client” is the custom data retrieval client application.

Client State Diagram:

Key Word	Description
Test For Server Rdy	With Socket APIs, usually there will be a function used to open the socket. In this state, the client program should attempt to open the socket. If <b>Open Failed</b> , the client should wait 5 seconds and try again.
Wait For Record	In this state the client is waiting for the next data record from the server. When a record is received it should be "secured" (saved to disk or database), then an acknowledgment should be sent back to the server. Once the server has processed the acknowledgment it will not send that record again. The client should use a watchdog timer while waiting for a data record. If the client is in the <b>Wait For Record</b> state for longer than expected ( <b>RecIntv2</b> ) then it should assume that the server has died and close the session. This watchdog operation may be difficult to implement, but it seems that some

implementations of sockets do not properly report a broken socket and so the watchdog is necessary for reliability.

**Rec Intv 2** This is an amount of time greater than 2 times the expected interval between data records. It is just longer than the longest period between records the client would expect to receive from the server. If the client goes longer than this interval without receiving a new record then it should close and reopen the socket, thus allowing the server to recover if it has broken socket connection.

**Secure Rec** The secure record action is taken when a **Record Rdy** event occurs while the client is in the **Wait For Record** state. Before the client sends an acknowledgment to the server it should "secure" the data record sufficiently so that if a power failure or crash occurs the data will be safe.

**Send Ack** The send acknowledgment action is done in response to the **Record Rdy** event, after the record is secured. In **Send Ack** the client forms an acknowledgment record from information taken from the data record and sends it to the server.

**Stop** It is important to note that the Stop event could occur at any time. If it occurs while in the Session Open state then the socket should be closed (**Close Session**) before program termination.

Server State Diagram:

Key Word	Description
<b>Wait For Record Available</b>	In this state the server is waiting for the next record to become available from the server's data record source.
<b>Wait For Ack</b>	In this state the server is waiting for the client to acknowledge that it has secured the record. If an acknowledgment for the wrong record comes in, the server will just continue to wait. After waiting for a minute, the server will re-issue the data record and wait again.
<b>Advance Rec</b>	The advance record action is executed after the server receives a valid acknowledgment record from the client while in the <b>Wait For Ack Record</b> state. This is the point at which the server

recognizes that the client has secured a record and the server relinquishes responsibility for the well being of that record. The server moves on to the next record.

### Stop

Note that in the *Session Open* and *Server Registered* states there are "exit" actions that need to be executed on the *Stop* event.

Communications between the client and server are conducted using ASCII records where each record is terminated by a carriage return - line feed (CRLF) pair. Record length varies quite a bit. For each datalogger record there is exactly one ASCII record. Because of the Block Mode Protocol used to communicate with dataloggers, the maximum size datalogger record is limited to something less than 1024 field values. Assuming 6 characters per value, 13 characters per field name, and 6 characters per field type designation, a single ASCII record could come out to be a little longer than 25K characters.

Typical datalogger programming will produce record sizes of about 150 characters. It would not be unusual to see records that contain one or two hundred values which would come out to a length of 2 to 3K characters in ASCII.

To express the format of ASCII records used for communications between the client and server, we will use Extended Backus Naur - Formalism (EBNF), a notation used to express syntax. This notation was adopted from Wirth [3], and extended here by adding a repetition count preceding some brackets. EBNF is summarized in the following table where A, B and C are syntactic entities of the language being described. Where one of these entities is a literal string it is enclosed in quotes.

### Expression Means

A = BC	The construct A consists of B followed by C.
A = B   C	A consists of B or C.
A = [B]	A consists of B or nothing.
A = {B}	A consists of any number of B's including none.
()	Brackets used to group sections of an expression.

## 9.5 RTMS Format Description

The EBNF description of LDEP syntax is as follows:

```
Record = ( DataRecord | AckRecord ) CRLF.
DataRecord = StationName "," TableName " (" FieldSpecs ") VALUES (" FieldValues ")".
AckRecord = StationName "," TableName "," RecordNumber.
FieldSpecs = FieldName " " FieldType { "," FieldName " " FieldType }.
FieldValues = FieldValue { "," FieldValue }.
StationName = Label.
TableName = Label.
FieldName = Label.
Label = Letter { Letter | Digit }.
FieldType = ( "TIMESTAMP" | Decimal | "FLOAT" | "INTEGER" | VarChar ).
Decimal = "DECIMAL(" Digit [ Digit ] "," Digit [ Digit ] )".
```

```

VarChar = "VARCHAR(" Digit { Digit } ")".
FieldValue = ( TimeStamp | RecordNumber | Number | String ).
TimeStamp = "" Year "-" Month "-" Day " " Hour ":" Minute ":" Second "".
Year = 4( Digit ).
Month = 2( Digit ).
Day = 2( Digit ).
Hour = 2( Digit ).
Minute = 2( Digit ).
Second = 2( Digit ) [ "." { Digit } ].
RecordNumber = 10{ Digit }.
Number = { Digit } [ "." { Digit } ].
String = "" { Character } "".

```

A typical data record might look something like this:

```

Lgr,Sec15 (TMSTAMP TIMESTAMP,RECNBR DECIMAL(10,0),Battery_V
FLOAT,Temp FLOAT) VALUES ('1993-12-08
15:02:00',123456,13.5,72.123)

```

Only without the tabs and carriage return in the middle. One with strings might look like this.

```

PC1,StatMsg (TMSTAMP TIMESTAMP,RECNBR DECIMAL(10,0),SrcStn
VARCHAR(256),AbtStn VARCHAR(256),Hop DECIMAL(3,0),Message
VARCHAR(256)) VALUES ('1993-12-08
15:02:02.25',13355,'PC1','StatMsg',0,'DBSelect End Pipe Queue Dump')

```

The acknowledgment records to be sent back to the server for the two records shown above would be:

```
Lgr,Sec15,123456
```

and

```
PC1,StatMsg,13355
```

## 9.6 Standard Format Description

The following is an EBNF syntax of a new record format that we have developed that we believe is more digestible than the pseudo-SQL syntax that is in the original protocol:

```

outputRec    = recordHeader { "," fieldName "," fieldType "," fieldValue } "\r\n".
recordHeader = stationName "," tableName "," timeStamp "," recNo.
fieldName    = string.
fieldType    = ("TIMESTAMP" | decimalType | "FLOAT" | "INTEGER" | varCharType ).
FieldValue    = string.
StationName  = string.
TableName    = string.
TimeStamp    = "\" year \"-\" month \"-\" day \" \" hour \":\" minute \":\" second "\"".
RecNo        = "\" digit {digit} \"\"".
Year         = 4(digit).
Month        = 2(digit).           ; 0 < month <= 12
day          = 2(digit).           ; 0 < day <= 31

```



```
hour      = 2(digit).           ; 0 <= hour < 60
minute    = 2(digit).           ; 0 <= minute < 60
second    = 2(digit) ["."] {digit}. ; 0.0 <= second < 60.0
string     = "\"" {ascii_character} "\".
DecimalType = "DECIMAL(" digit [digit] ", " digit [digit] ")".
VarCharType = "VARCHAR(" digit {digit} ")".
```

Within a string, quotation marks and back slash characters will be quoted with a backslash character.

The sample record from the original protocol would have the following format under this new syntax:

```
"Lgr","Sec15","1993-12-08 15:02:00","123456","Battery_V","FLOAT",
"13.5","Temp","FLOAT","72.123" CRLF
```

The acknowledgment message is the same as for the RTMS format. The acknowledgment for the above record would be:

```
Lgr,Sec15,123456
```



# Section 10. Security



*Security can restrict access to certain functions in the LoggerNet server. Different levels of access are set up for groups within the Security client, and then users are assigned to groups. Security is an optional feature and does not have to be enabled for the LoggerNet system to operate.*

## 10.1 Security Setup and Operation

When the Security client is first opened, you are prompted to select a communication server. Security settings are server specific, and if you have more than one server system, different configurations can be set up for each.

If security is enabled, to bring up the Security client you will have to enter a user name and password that has administrative access.

Once the name of the LoggerNet server is entered, the main security window appears. The server to which you are connected is displayed in the Server field.



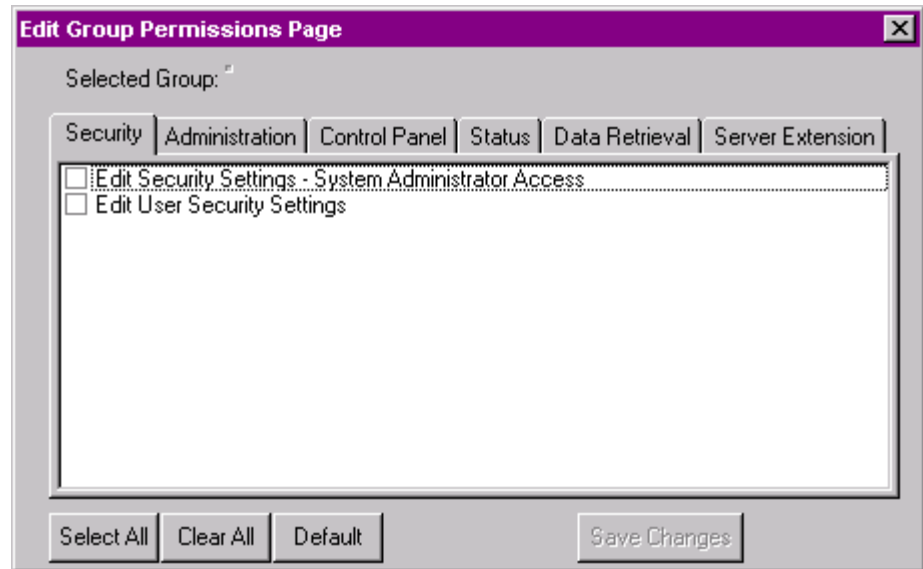
To begin, create a new group by selecting the Add button. A dialog box appears; enter a name for the group and select OK. The group name will be listed in the Defined Groups field. If you want to delete a group, select it with the click of a mouse and press the Delete button.

One of the first groups that should be created is an Administrative Group. This group should have Edit Security privileges as described below. Otherwise, with

security enabled, you will not be able to access the Security application to make changes.

### 10.1.1 Group Permissions

To set up the security configuration for a group, select the group and then press the Edit Group Permissions button to bring up another dialog box.



The Edit Group Permissions dialog box has six different tabs for security settings. Each setting has a check box to the left of it. When the check box has a mark in it, the setting is enabled for the active group. These settings are explained below.

#### 10.1.1.1 Security

The settings on this tab are used to assign privileges to groups for the Security client.

- Edit Security Settings - System Administrator Access - Group members have full administrative rights when using Security.
- Edit User Security Settings - Group members have rights to edit user settings only, but cannot set up or configure groups.

#### 10.1.1.2 Administration

The settings on this tab are used to control access to functions in the Network Administration client.

- Enable Client Communications - Allows the user to communicate with the datalogger network via one or more clients. If this is not selected, no user in the group will be able to connect clients to the LoggerNet server.

- Network Configuration Editing - Users can edit any device in the datalogger network including adding or deleting dataloggers (see Section 5.1).
- View Network Map - Users can view the network map, but they may not have access to edit it (see Network Configuration Editing above).
- Edit Device Settings - Settings for each device in the datalogger network can be edited using the Network Administration client.
- View Device Settings - Users can view device settings, but they may not have access to edit them.
- Communications Testing - Communications tests for each device can be performed using the utility provided in the Network Administration client (see Section 5.6).
- Override Device Settings - Using this function in the Control Panel, the user can increase the frequency of data collection from a datalogger. This may keep other users from accessing other dataloggers that share part of the same communications link.
- Device Connection Management - Allows operation of the Control Panel to connect to a datalogger.

#### **10.1.1.3 Control Panel**

The entries in this tab affect functionality found in the Control Panel client.

- Manage Datalogger Programs - The user is allowed to send programs to and retrieve programs from the datalogger using the Control Panel client (see Section 7.2).
- Set Datalogger Clock - Access is given to the datalogger clock set function in the Control Panel client (see Section 7.3).
- View Datalogger Clock - The user is given rights to the Get Date/Time function in the Control Panel client (see Section 7.3).
- Edit Numeric Display/Set Ports and Flags - The user is allowed to set the values of Input Locations or Public Variables in the Numeric Display. Flags and ports can be toggled high or low (see Section 7.5).
- Terminal Emulation Mode - The user is given the right to communicate with the datalogger in Terminal Emulation mode using the Control Panel client (see Section 7.4.5).

#### **10.1.1.4 Status**

The settings on this tab affect some of the features in the Communications Status Monitor client.

- **Reset Statistics** - The user can reset the data collection and communications statistics that are displayed in the Communications Status Monitor client.
- **View Low Level I/O** - The low-level input/output log file is used mostly for troubleshooting communications problems and is available in the Communications Status Monitor client (see Section 6.5).
- **View Server Messages** - The Server Event Messages can be viewed in the Server Transactions Log displays (View | Server Logs). (See Section 6.4)

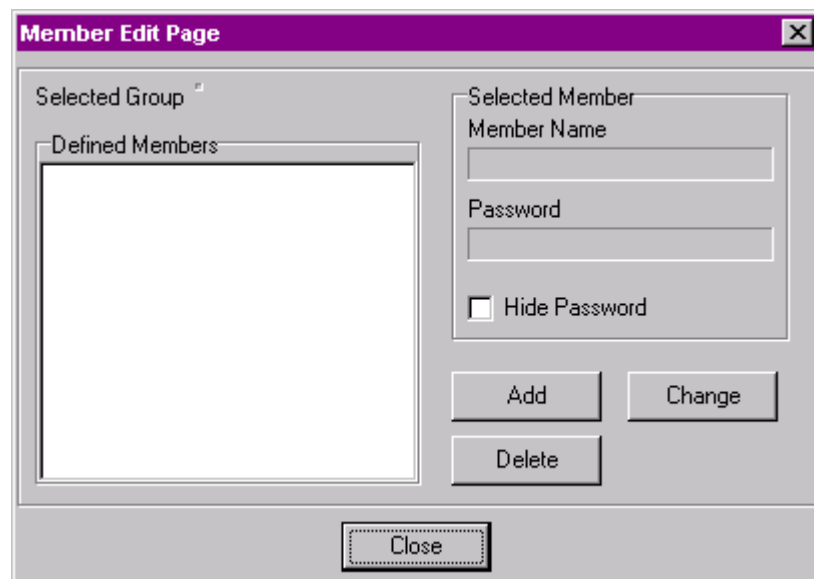
#### 10.1.1.5 Data Retrieval

The settings on this tab determine which methods the user has available to collect data.

- **Data Retrieval from Server** - The user has rights to collect data from the LoggerNet server's data cache (see Section 5.5.1).
- **Data Retrieval From Datalogger** - The user has the right to perform a manual poll to the datalogger. This is used to force an update to the data cache and return datalogger status.

#### 10.1.2 Group Members

Once security access has been defined for one or more groups, individual members can be assigned to the groups. From the Security application's main screen, highlight a group with a mouse click and press the Edit Group Members button to invoke the Member Edit Page. Every group member has the security access identified for the group.



The screenshot shows a dialog box titled "Member Edit Page" with a purple header bar. Inside the dialog, there are two main sections. On the left, under the label "Selected Group", there is a list box labeled "Defined Members" which is currently empty. On the right, under the label "Selected Member", there are three input fields: "Member Name", "Password", and a checkbox labeled "Hide Password". Below these fields are three buttons: "Add", "Change", and "Delete". At the bottom center of the dialog is a "Close" button.

To add a new member, press the Add button. In this dialog box, enter the individual's name and a unique password. Each new member added will be listed in the Defined Members field. An individual can be deleted from the list

by highlighting the member's name and pressing the Delete button. The Change button will allow you change the password for a current member. Passwords are case sensitive so make sure and note the use of capitalization.

While in the Member Edit Page, the individual highlighted in the Defined Members field will appear in the Member Name field. The member's password will be displayed in the Password field, unless the Hide Password check box is selected.

### **10.1.3 Enabling Security**

Once the Groups have been defined and individuals have been assigned to the groups, security must be enabled for the security restrictions to take affect. To do this, select the Security Enabled check box on the main window. If this check box does not have a check or X in it, then security is not enabled and all users will have unlimited access to all of the LoggerNet functions.

As a confirmation that security should be enabled, an Edit dialog box appears when you try to enable security. A user name must be entered for a user that is part of a group with Security Edit privileges for security to be enabled.

The security settings will apply to all client applications connecting to the LoggerNet server after security is enabled.

## **10.2 Resetting Security**

If access cannot be gained to the Security application so that changes can be made, the only way to reset security is by deleting the csilgrnet.sec file in the c:\LoggerNet\SYS\Cora working directory.

Removing this file will disable security and clear the security settings. The security groups and settings can then be rebuilt.

For proper protection of security, a Windows NT platform must be used with file access restrictions to the LoggerNet working directories.





# Section 11. Hole Monitor



*The Hole Monitor provides a way to track the progress of data hole collection for the network of dataloggers attached to the LoggerNet server. Holes are created primarily due to communication problems with the datalogger. When Data Advise data collection is used, the most recent data is collected first leaving any missed records that are still in datalogger final storage as a collectible hole. The LoggerNet server then starts hole collection to collect the missing records.*

*A list of collectible holes is kept for each datalogger in the network. This list is called a hole collection queue. Hole monitor allows the user to see the holes that are waiting to be collected and watch as they are collected.*

## 11.1 Tracking Data Hole Collection

When the Hole Monitor utility is first connected to the LoggerNet server it will get from the server a list of all the outstanding collectible holes for the datalogger network.

A hole is a discontinuity of data in the LoggerNet server's data cache. If the data record received from a datalogger has a record number larger than the next record expected by the server, a hole is identified. The server will then compare the record number with the type of datalogger and the size of the table to see if the data in the hole can be collected from the logger. If the hole appears collectible, it will be entered in the list of holes to be collected. Collectible holes are only created when the Data Advise method of data collection is used.

DeviceName.ColumnName	Begin	End	Hole Count
MyLogger.15min	305734	306127	394
MyLogger.60MIN	75886	76529	644
Officelgr.15min	302422	306127	3706
Officelgr.60MIN	75165	76531	1367

Exit

Current Server:localhost

A data hole is identified by datalogger name, table name, and a range of record numbers. The display on the Hole Monitor utility shows all of the holes waiting for collection. The information is shown on the display in four columns.

- **DeviceName.ColumnName** – DeviceName is the name of the datalogger as it appears in the network map using NetAdmin. The ColumnName is the name of the output table for which data is missing in the data cache. When the hole is actively being collected a shovel icon will appear in this first column to the left of the datalogger name.
- **Begin** – This is the beginning record number for the set of missing data.
- **End** – This is the ending record number for the set of missing data.
- **Hole Count** – The hole count is the number of records that exist in the hole.

As the data records are collected or identified as uncollectible the number of records in the hole count will decrease until the hole has been collected. The hole will then be removed from the list.

## 11.2 Refresh the List of Holes

If communication with the LoggerNet server is lost for some reason and then re-established, the list of holes displayed in hole monitor will not automatically be cleared when the new list of holes is provided by the server. File | Refresh will clear the list of holes and get the current list from the LoggerNet server.

# Section 12. Datalogger Program Creation with Edlog

---



*This section provides information on memory allocation and programming for Campbell Scientific's CR7, CR10, 21X, CR500, CR510, CR10X, CR23X, CR510-TD, CR10X-TD, CR23X-TD and CR10T dataloggers. See Section 12 for information about the CR5000 and CR9000 CRBasic program editor.*

## 12.1 Overview

Edlog is a tool for creating, editing, and documenting programs for Campbell Scientific's array based dataloggers: CR500, CR510, CR10, CR10X, 21X, CR23X, and CR7, and table based dataloggers: CR510-TD, CR10T, CR10X-TD, and the CR23X-TD. It provides a pull-down menu from which to select instructions, with pick-lists and detailed help for completing the instructions' options (or parameters). Edlog checks for errors and potential problems in the program when pre-compiling the program. Some highlights of Edlog's features are listed below.

**Precompiler** - Edlog precompiles the program to check for errors and to create the file that is downloaded to the datalogger. The precompiler will catch most of the errors. The errors that the precompiler misses will be caught by the datalogger when the program is compiled. The download file (\*.DLD) is stripped of comments to make it more compact. During the precompile step, a Program Trace Information file (\*.PTI), which provides an estimate of program execution time, is also created (Section 12.1.1.1). For array based dataloggers the precompiler also creates a Final Storage Label file (\*.FSL) to supply labels for final storage values to be used by other software applications.

**Context-sensitive Help** - Pressing the right mouse button with the cursor on a parameter will provide a pick-list of options or pop-up help for that parameter. More help is available by pressing <F1> at any time or the Help button in various dialog boxes. Help, pick lists, and edit functions are also available from the menus or toolbar.

**Cut and Paste** - Several datalogger programs can be opened simultaneously, and instructions can be copied from one program into another. This simplifies writing different programs for the same sensor set, or similar programs for different sites. Edlog will also allow you to save sections of code as "Library Files" which can then be imported into other programs.

---

### NOTE

Be careful when copying instructions from a program written for one datalogger to a program for a different type of datalogger. Instructions may differ between dataloggers.

---

**Input Location Labels** - Though the datalogger uses a number to address input locations, Edlog allows you to assign labels to these locations for ease of use when programming and later when reviewing the data on-line. Edlog has

several features which aid in the management of these labels. A new Input Location label is automatically assigned the next available Input Location number (address). That Input Location can be picked from a list when needed later in the program for further calculations or output. The Input Location Editor (Section 12.3.3) allows the Input Locations to be edited (moved, inserted, or deleted); the Input Location numbers are then automatically updated wherever the labels appear in the program. When a section of code is pasted into a program, Edlog will automatically use existing locations for matching labels and assign new locations to new labels. All location numbers in the pasted code are updated accordingly.

Final Storage Label Editor – New to this version of Edlog is the ability to view and edit the labels for the data values in final storage records. The labels are stored in the Final Storage Label file for array dataloggers or as part of the datalogger program for table based dataloggers.

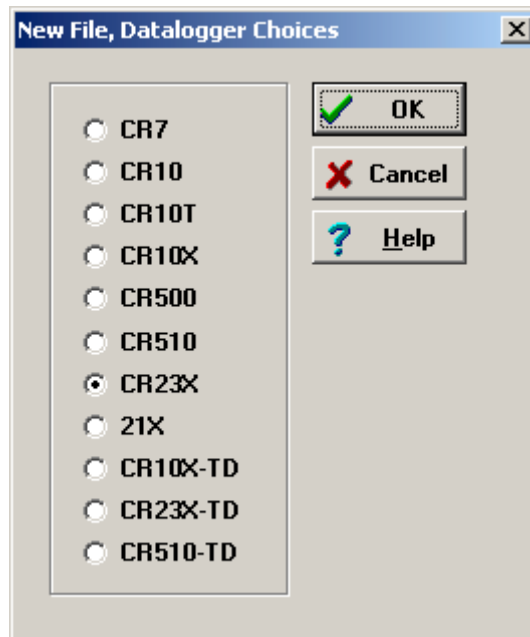
Expression Compiler - Mathematical calculations can be written algebraically using Input Location labels as variables. When the program is compiled, Edlog will convert the expressions to datalogger instructions.

For example, the following expression could be used to create a new input location for temperature in degrees Fahrenheit from an existing input location for temperatures in degrees Celsius.

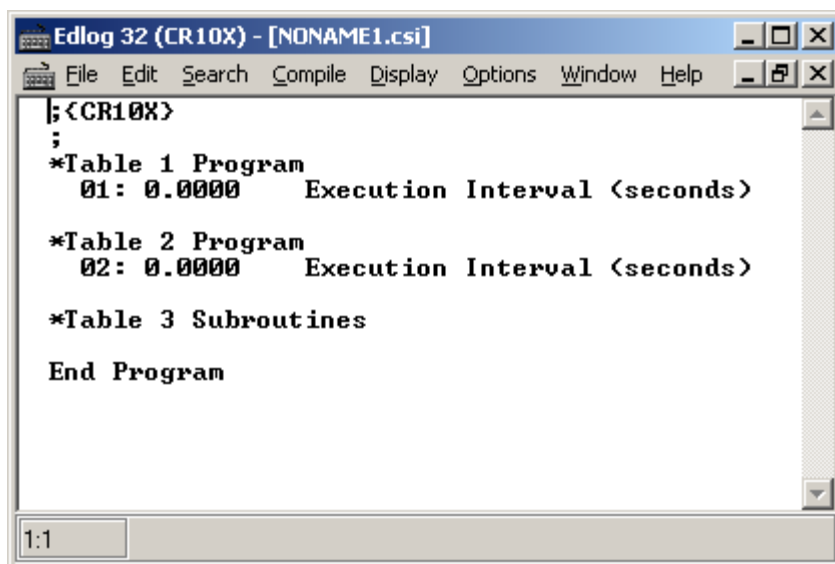
$$\text{TempF} = \text{TempC} * 1.8 + 32$$

### 12.1.1 Creating a New Edlog Program

To create a new datalogger program, choose File | New from the Edlog menu and select the datalogger type from the dialog box. A template similar to the one shown below appears.



Select the datalogger you are using from the list and click OK. A blank program template will come up as shown below for a CR10X.



The first line of text identifies the type of datalogger program to be written. This is followed by a comment line and the Program Table Headers and Execution Interval fields. The Program Table Headers and Execution Interval fields are protected text that cannot be deleted or commented out. When the cursor is moved to the Execution Interval line, the field for the execution interval is highlighted. A numeric value must be entered or the instructions in the table will never be executed.

Instructions inserted under the Program Table 1 header will be run based on the execution interval for that table. Likewise, instructions inserted under the Program Table 2 header will be run based on the execution interval for Program Table 2. Program Table 3 is reserved for subroutines that are called by either of the other tables. Most users find they can write the entire program in Program Table 1, avoiding complications associated with synchronizing two tables. Program Table 2 is normally used only when portions of the program require a different execution interval (placed in Program Table 2).

#### NOTE

Program tables in this section refer strictly to sections of the datalogger program. Do not confuse these program sections with the data tables created in table based dataloggers using P84 to store output data.

When the program is complete, select File | Save from the Edlog menu. A standard file dialog box will appear in which to type a file name. Edlog supports long file names for the datalogger programs. Use descriptive names to help document the program's function. After saving the file, you will be prompted to compile the program. When a program is precompiled the code will be checked for errors. After compiling, the datalogger program can be sent to the datalogger using the Control Panel (Section 7.2).

### 12.1.1.1 Program Structure

While Edlog is not a structured programming language there are some standard programming practices that will help you and others understand what the datalogger program is intended to do.

**Comments** – Edlog provides the ability to add comments on any blank line and to the right of all instructions. Liberal use of descriptive comments makes the program clearer and will help you remember what you were doing when you come back to it a year or two later. Especially useful are descriptions of what sensors are connected and how they are wired to the datalogger.

**Program Flow** – It is easier to follow a program that is written in distinct sections, each of which handles a specific type of instruction. The recommended sequence is:

- Measure Sensors – In this first section put all the instructions that get data from the sensors attached to the datalogger. The sensor readings are stored in input locations, ready for the next section.
- Process Measurements – In this section do all the calculations and data processing to prepare the data for output.
- Control – Do any control of external hardware or devices.
- Output Data – Check to see if it is time, or a condition exists, to trigger output data to be saved in final storage.

**Descriptive Labels** – Use input location and final storage labels that are meaningful for the data they contain.

### 12.1.1.2 Edlog File Types

When a program is saved and compiled, the following files are created:

- \*.CSI - The CSI file is what the user actually edits. When an Edlog program is saved, Edlog automatically adds a CSI extension to the program's name. Existing CSI files can be edited by selecting File | Open. Long file names are supported. Although CSI files are ASCII files they require a particular format, so editing the files with some other text editor can corrupt the Edlog programs so that they no longer compile.
- \*.DLD - When Edlog compiles a program (\*.CSI), a DLD file is created. This is the file that is downloaded to the datalogger, (and also the type of file that is retrieved from the datalogger). If an existing program file is edited and compiled, the old DLD file will be overwritten by the new file. A CSI file can be created from a DLD by choosing File | Document DLD File.
- \*.PTI - PTI files show the execution times for each instruction, block (e.g., subroutine), and program table, as well as the estimated number of final storage locations used per day. The execution times are estimates. PTI files do not account for If commands, Else commands, or repetitions of loops. For some instructions, the execution times are listed as 0. This

occurs when the execution time is unknown (e.g., P23 – Burst Measurement).

- \*.FSL – FSL files are not created for table based dataloggers. Table based datalogger program files contain the final storage labels.

There are a couple of other files that are used in Edlog, but are generated by other means than compiling the file:

- \*.LBR - Library files (\*.LBR) are parts of a program that can be retrieved and used in other Edlog programs. If a programmer often uses an instruction set in his/her datalogger programs, this partial file can be saved to disk and inserted into a new program. For information about creating a library file, see the help on Creating a Library File.

---

**NOTE**

Library files that are created for one type of datalogger should not be used in a different type of datalogger (e.g., do not use an LBR file created for a CR10X-TD in a CR10X or CR510-TD program). Instructions differ among dataloggers, and bringing in an invalid instruction to a datalogger could result in errors.

---

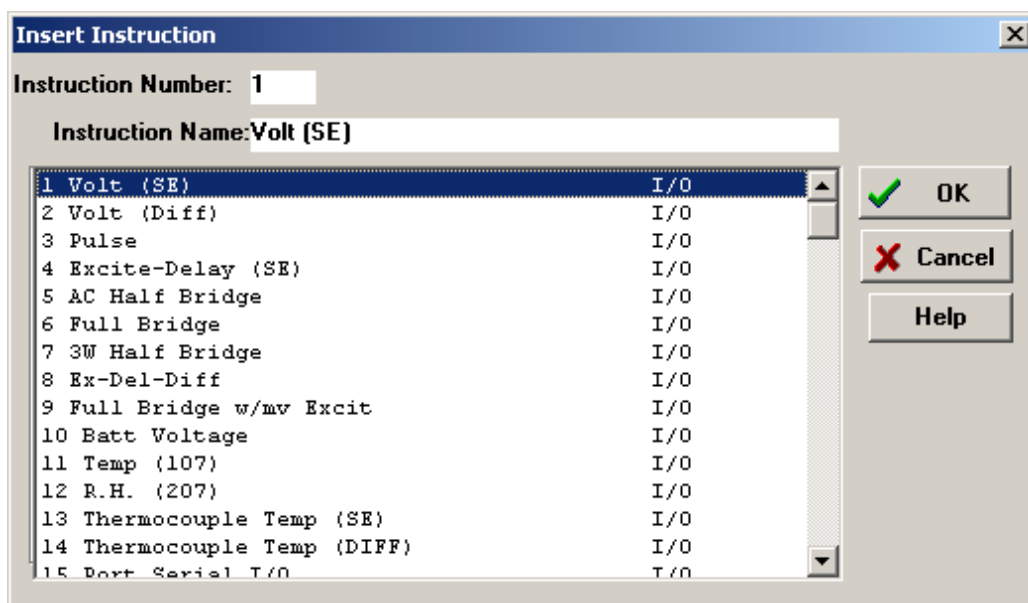
- \*.TXT - Printer output files created by Edlog are saved with a TXT extension. These files can be sent to a printer or viewed with a text editor. A TXT file is created by selecting File | Print to File.

### 12.1.1.3 Inserting Instructions into the Program

Instructions are entered into the program table in the order that they should be executed in the program. There are four ways to insert an instruction:

- Select File | Insert Instruction from the Edlog menu.
- Press Shift+Insert on the keyboard.
- Right click on a blank line and select Insert Instruction from the menu.
- Type the instruction number onto a blank line and press enter.

The first three options will invoke the Insert Instruction dialog box.



To insert an instruction into the program, select and then choose OK, or double click the entry in the list. If you need more information on an instruction, select the instruction and click on the Help button.

Note that to the right of each instruction name is a code for the instruction type: I/O for input/output, Process for instructions that calculate new values, Output for instructions that write to final storage, or Control for instructions that affect program flow.

#### 12.1.1.4 Entering Parameters for the Instructions

When an instruction is inserted, the cursor moves to the first parameter. Type the parameter's value and press <enter> to move to the next parameter. There are two ways to get help on a parameter:

- Select the parameter with your mouse and press the right button. This brings up a dialog box from which to select a value or a pop up description of what should be entered.
- With your cursor anywhere within the instruction, press <F1>. This opens the help system to a detailed description of the instruction and parameters.

Edlog provides hints for each parameter at the very bottom of the Edlog screen. These hints often display the valid entries for a field.

#### NOTE

Many instructions are datalogger specific; refer to the specific datalogger manual for details on a particular instruction.

#### Data Entry Warnings

Edlog has a Data Entry Warning function that is accessed from the Options | Editor menu item. By default, the Data Entry Warning is enabled. When the Data Entry Warning is active, a warning is displayed immediately after an



invalid input has been entered for an instruction's parameter. The warning lists the valid inputs. A valid input must be entered before advancing to the next parameter.

#### 12.1.1.5 Program Comments

Comments can be entered to document the program for the programmer or future users. Comments are ignored by the compiler; they can be entered on any blank line or at the right of instruction or parameter text. A semicolon (;) is used to mark comments. Comments can also be used to temporarily remove instructions from a program for testing purposes.

In addition to typing a semicolon at the beginning of each line while entering comments, there are several ways to comment (or uncomment) lines, instructions, or blocks of code:

- Select a block of text, press the right mouse button, and select “comment” or “uncomment” from the right button menu.
- Select File | Comment or File | Uncomment from the Edlog toolbar.
- Select a block of text and press <ctrl>+n to comment text (or <shift><ctrl>+n to uncomment text).
- Press <End> to automatically insert a semi-colon to the right of the protected text of an instruction or parameter, and type the desired comment.

Edlog will not allow a portion of an instruction or the table execution intervals to be commented out.

#### 12.1.1.6 Expressions

Algebraic expressions can be used in a program to easily perform processing on input locations. When a datalogger program that contains an expression is compiled, the appropriate instructions are automatically incorporated into the DLD file. As an example, the following expression could be used to convert temperature in degrees Celsius to temperatures in degrees Fahrenheit:

$$\text{TempF} = \text{TempC} * 1.8 + 32$$

Following are rules for creating expressions:

- Expressions must be set equal to the label of the Input Location that will store the result. The result label must be to the left of the expression.
- Expressions can have both fixed numbers and Input Location labels. Input Locations can only be referenced by their label; each number in an expression is assumed to be a constant.
- Floating-point numbers are limited to six digits plus the decimal point and sign.

- The operator(s) and/or function(s) used in the expression are limited to those in the Operator and Function list (Table 12.2-1 below).
- Numbers and labels that appear immediately after a function must be enclosed in parentheses.
- Several operators and/or functions can be used in one expression. Operations and functions that are enclosed in parentheses are calculated first; the inner most parentheses are evaluated first.
- To continue an expression to the next line, end the first line with an underscore ( \_ ).

TABLE 12.1-1. Operators and Functions	
Operators	
*	Multiply
/	Divide
+	add
-	subtract
^	Raise to the power of; enclose negative values in parentheses
@	modulo divide
E	scientific notation; 6e-1=0.6
Functions	
COS	cosine; angle in degrees
SIN	sine; angle in degrees
TAN	Tangent; angle in degrees
COTAN	cotangent; angle in degrees
ARCTAN	arctangent; angle in degrees
ARCSIN	arcsine; angle in degrees
ARCCOS	arccosine; angle in degrees
ARCCOT	arccotangent; angle in degrees
SQRT	square root
LN	natural logarithm
EXP	exponent of e; $EXP(2) = e^2$
RCP	reciprocal; $RCP(4) = 1/4 = 0.25$
ABS	absolute value
FRAC	takes the fraction portion; $FRAC(2.78)=.78$
INT	takes the integer portion; $INT(2.78)=2$

Below are examples of valid expressions:

```

Zee      =  Vee+Ex
es       =  tee^(-2)
Root     =  SQRT(ABS(data))
avg      =  (data1+data2+data3+data4+data5)/5
length   =  SQRT((adj^2)+(opp^2))
TempF    =  (TempC*1.8)+32

```

The following section of an Edlog program uses an expression to convert temperature from Celsius to Fahrenheit:

Execution Interval = 10 sec

*;this instruction reads the temperature probe  
;the output is in degrees C*

1: Temperature (107) (P11)

01: 1	REPS
02: 2	Channel
03: 1	Excitation Channel
04: 2	Loc [TempC]
05: 1	Mult
06: 0	Offset

*;the following expression converts TempC to  
;a temperature in degrees Fahrenheit*

TempF = (TempC\*1.8)+32

When this program is compiled, the DLD file contains the following instructions. The last 5 instructions calculate the expression.

1: Temperature, 107 (P11)

01: 1
02: 2
03: 1
04: 2
05: 1.0
06: 0.0

2: Z=X (P31)

01: 2
02: 5

3: Z=F (P30)

01: 1.8
02: 0
03: 3

4: Z=X\*Y (P36)

01: 3
02: 5
03: 5

5: Z=F (P30)

01: 32
02: 0
03: 3

6: Z=X+Y (P33)

01: 3
02: 5
03: 6

### **Errors That Can Occur With Expressions**

The following error codes and messages may occur when using expressions.

<u>Code</u>	<u>Error Message</u>
100	Missing left parenthesis
101	Missing right parenthesis
102	Variable name expected
103	Number expected
104	Floating point numbers limited to 5 digits
107	Function expected
110	New line expected
111	Equal sign expected

#### **Variable Name Expected**

This message occurs when the expression is not set equal to an Input Location label. The label must be to the left of the expression and not enclosed in parentheses. An expression that contains no equal sign causes compiler error 202, "unrecognized text".

For Example:

"Variable name expected" is displayed when a program contains any of these expressions:

$$\begin{aligned} 5 &= eI * (Vee + en) \\ (\lambda) &= \cos(\theta) \\ 10 - (zee/2) &= bee \end{aligned}$$

These are correct ways of entering the above expressions:

$$\begin{aligned} five &= eI * (Vee + en) \\ \lambda &= \cos(\theta) \\ bee &= 10 - (zee/2) \end{aligned}$$

#### **Number Expected**

Indicates one of the following situations:

- (1) An expression with a /, \*, or ^ operator is missing a number or label before and/or after the operator.
- (2) An expression with a + or - operator does not have a number or label after the operator.
- (3) An expression with an @ operator does not have a number after the @; only a fixed number is allowed immediately after the @ operator.
- (4) An expression with an @ operator does not have either a number or label before the @.
- (5) There is nothing between a pair of parentheses (e.g., the expression contains this "()").

- (6) A number is immediately followed by a label or function without an operator (e.g., an expression containing “8label” gets this error message).

### ***Floating Point Numbers Limited to 5 Digits***

All fixed numbers are limited to five digits not including negative signs and decimal points.

### ***Function Expected***

Letters that are immediately followed by parentheses are assumed to be a function. If the letters are not on the function list (see section 2.3.3.2), this error message occurs.

### ***New Line Expected***

Indicates one of the following situations:

- (1) An expression contains more than one equal sign.
- (2) There is no operator between two sets of parentheses.

For Example:

This error message is displayed when a program contains any of these expressions:

```
zee=(label1)(label2)
ex=(5)(ARCTAN(data))
eee=(em)(see^2)
```

These are correct ways of entering the above expressions:

```
zee=(label1)*(label2)
ex=(5)*(ARCTAN(data))
eee=(em)*(see^2)
```

- (3) There is no operator between a set of parentheses and a number.

For Example:

This error message is displayed when a program contains any of these expressions:

```
tee=5(2)
mu=(nu)103
bee=10.52(ef/2)
sigma=-17(RCP(alpha))
```

These are correct ways of entering the above expressions:

```
tee=5*(2)
mu=(nu)*103
bee=10.52*(ef/2)
sigma=-17*(RCP(alpha))
```

- (4) A label or function is immediately after a set of parentheses without an operator.

For Example:

This error message is displayed when a program contains any of these expressions:

```
result=(ex^2)data
gamma=(10-omega)SIN(psi)
dee=(17)number
```

These are correct ways of entering the above expressions:

```
result=(ex^2)*data
gamma=(10-omega)*SIN(psi)
dee=(17)*number
```

### ***Equal Sign Expected***

An equal sign MUST immediately follow the label of the Input Location that stores the results (e.g., label = expression). An expression that contains no equal sign causes compiler error 202, “unrecognized text”.

For Example:

“Equal sign expected” is displayed when a program contains any of these expressions:

```
zee/2=bee
data+number=volt1+volt2
```

These are correct ways of entering the above expressions:

```
bee=zee/2
data=volt1+volt2-number
```

## **12.1.2 Editing an Existing Program**

To edit an existing file, load it into Edlog by choosing File | Open from the Edlog menu. Changes can be made as desired and then the file can be saved and compiled under the same (File | Save) or a new name (File | Save As). Table 12.2-2 provides a list of keystrokes that can be used in editing programs and moving around in Edlog.

TABLE 12.1-2. Editor Keystrokes

PgUp	Page Up
PgDn	Page Down
Up Arrow	Move Up One Line
Down Arrow	Move Down One Line
Right Arrow	Move One Character Right
Left Arrow	Move One Character Left
<Ctrl> Home	Move Cursor to Beginning of File
<Ctrl> End	Move Cursor to End of File
<Ctrl> PgUp	Move Cursor to Top of Screen
<Ctrl> PgDn	Move Cursor to Bottom of Screen
<Enter>	Move to Next Field or Create New Line
<Shift> Ins	Select an Instruction from a Dialog Box
<Ctrl> Right Arrow	Move Instruction 1 Tab Right (Cursor on Parameter)
<Ctrl> Left Arrow	Move Instruction 1 Tab left (Cursor on Parameter) or Move from Input Location label to Input Location number.
<Ctrl> n	Comment out a Line or Instruction
<Shift> <ctrl> n	Uncomment a Line or Instruction
<End>	Move to end of line, Add a comment if on an Instruction
<Ctrl>C	Copy selected text
<Ctrl> X	Cut selected text
<Ctrl>V	Paste clipboard
Del	Delete character to right or selected text
<Shift> Del	Delete the Instruction or Line Under the Cursor
<Esc>	Close Dialog Box

### 12.1.2.1 Editing Comments, Instructions, and Expressions

To edit Comments, Expressions, and Instruction parameters, move the cursor to the appropriate text and retype it. To delete an instruction when the cursor is somewhere within the instruction, select Edit | Delete Instruction or press <Shift> Del. An instruction or block of instructions can also be selected and deleted with the delete key. The entire instruction must be selected or an error message will be returned.

### 12.1.2.2 Cut, Copy, Paste, and Clipboard Options

Edit | Cut, Edit | Copy, and Edit | Paste allow sections of the program to be moved or copied to another area of the program or between programs. Edit | Show Clipboard shows the contents of the clipboard.

#### NOTE

You cannot move, copy, delete or comment out protected text (Tables, Execution Intervals) or partial instructions. To move, copy or delete an Instruction, the entire instruction, including all of the parameters, must be selected.

Cutting and pasting between datalogger programs should only be between programs for the same datalogger type. Instructions and parameters may differ

between dataloggers. The compiler will catch many of these errors; however, this may be at the expense of much time and confusion.

### 12.1.3 Library Files

Library files can be created to store portions of programs, which can then be inserted into a different program. Library files are useful if you want to write different programs for the same sensor set, or if you have several stations that have similar, but not identical, sensor sets.

To create a library file, select the text to be stored and then select Edit | Save To Library File. When the window appears, type in the library file name. To insert a library file in a program, move the cursor to the desired insertion point and select Edit | Insert Library File.

---

**NOTE**

Library files created for one type of datalogger type should not be used in programs for a different datalogger type; i.e., a file for a CR10X-TD should not be used in a program for a CR10X or a CR510-TD. Instructions differ among dataloggers, and bringing in an invalid instruction to a datalogger could result in errors.

---

### 12.1.4 Documenting a DLD File

As noted in Section 12.1.1.2, the CSI file is the file created by Edlog that is used to generate the DLD code and other files. If for some reason your CSI file is missing, you can import the DLD file into Edlog to create another editable CSI file. From the Edlog menu select File | Document DLD. Select the DLD file to be imported and remember to save the file to create a new CSI file.

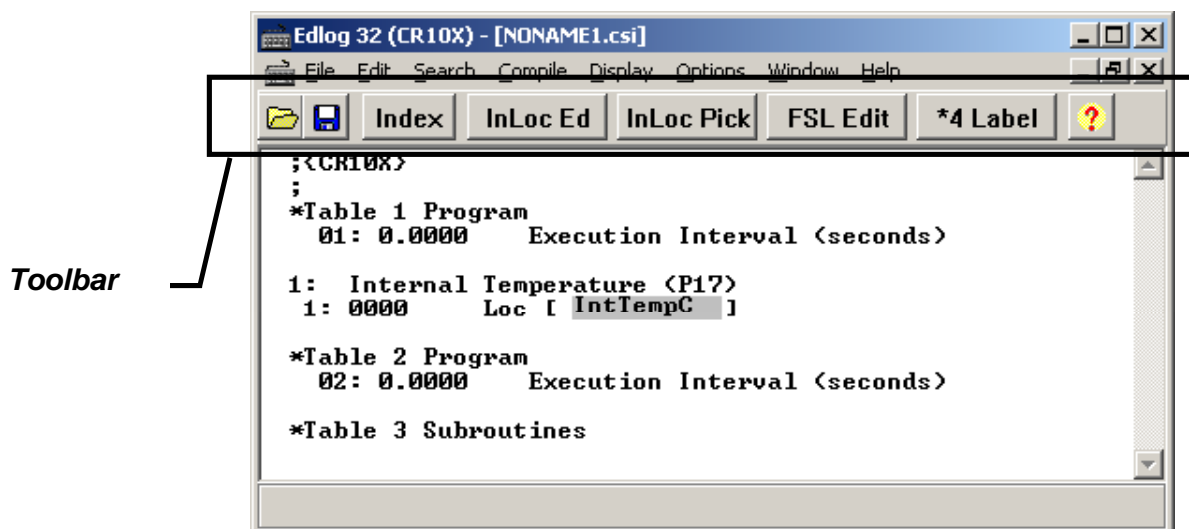
Programs created with the DOS versions of Edlog earlier than 6.0 were stored with the instruction description and comments in a \*.DOC file instead of a \*.CSI file. These programs can be imported into current versions of Edlog by using this Document DLD feature, though any comments will be lost.

### 12.1.5 Display Options

#### 12.1.5.1 Graphical Toolbar

A graphical toolbar provides buttons for some of the more frequently used menu items in Edlog. The toolbar is made visible by choosing Options | Show Toolbar from the Edlog menu. Conversely, it is removed from the screen by choosing Options | Hide Toolbar.





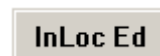
Open a new file.



Save the current file to disk and optionally precompile the program.



Index the parameter that is selected (for information on indexing, refer to your datalogger operator's manual).



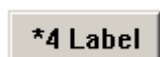
Invoke the input location editor (for a discussion on input locations, see Section 12.3).



Display the input location list; allows the user to select and insert an input location automatically into a parameter.



Invoke the final storage label editor (for more information on editing final storage labels see Section 12.4).



Assign a parameter a star 4 value (for information on star 4 values, refer to your datalogger operator's manual).



Open the on-line help system.

### 12.1.5.2 Renumbering the Instructions

When Automatic Renumbering is enabled, the instructions are automatically renumbered whenever instructions are inserted or deleted. By default, Automatic Renumbering is enabled. Automatic renumbering can be turned off if you have a very large program and auto renumbering is slowing down editing.

### 12.1.5.3 Compress VIEW

When Display | Compress View is selected, only the first line of each instruction is displayed. The compressed view makes it easier to see the program structure and to move around in the program.

Instructions cannot be edited in the compress view mode. Use Display | Uncompress to switch back to the full view or use the <F7> function key to toggle between the compressed and full views.

#### 12.1.5.4 Indention

Indention is typically used with If Then/Else sequences and loops to provide a visual key to program flow. Indention is a visual aid; it has no meaning to the datalogger. If the programmer chooses to use indention, it can be done automatically or manually.

The settings for indention are found under Options | Editor. Turn on Automatic Indention by checking the box next to it. The distance for each indention (in spaces) is set on the same dialog box. To manually indent an instruction, place the cursor on one of the instruction's parameters and press either <Ctrl>+right arrow or <Ctrl>+left arrow; the instruction is indented the direction the arrow is pointing.

The Display | Rebuild Indention menu item resets all existing indentions and rebuilds automatic indentions. Automatic indentions may need to be rebuilt when editing instructions causes the indentions to misalign.

## 12.2 Input Locations

An input location is the space in datalogger memory where the most recent value is stored for each sensor. Each time a sensor is scanned, the input location is overwritten with a new value. Input locations are referenced in the datalogger by number.

In an Edlog program, each Input Location has an Input Location number and a label that appear whenever the Input Location is referenced in the program. Edlog automatically assigns Input Location numbers as labels are entered.

### 12.2.1 Entering Input Locations

When a parameter requires an Input Location, the cursor automatically advances to where the label is keyed in. When a new label is entered, the next available Input Location number is automatically assigned to that label. To select an existing label from a list, press the right mouse button or <F6>.

You may prefer to enter all input locations into the Edlog program before writing the program. This makes all the labels available from the input location pick list, and can help reduce programming errors because of typos. See section 12.2.3.

Labels can have up to 9 characters for array based dataloggers and 14 characters for table based dataloggers. The first character must be a letter. The allowed characters are letters, numbers, and the underscore character ( \_ ). The following labels are reserved for expressions and should not be entered by the user: CSI\_R, CSI\_2, CSI\_3,... CSI\_95.

To enter the Input Location number instead of the label, use the mouse or press <ctrl> left arrow.

### 12.2.2 Repetitions

Many input/output and output processing instructions have a repetitions parameter. Repetitions (REPS) allow one programming instruction to measure several identical sensors or to process data from several Input Locations. When REPS are greater than 1, the Input Locations are assigned consecutive numbers (e.g., with REPS of 2 and LOC of 5, the Input Locations are 5 and 6). Each rep label is the initial label with a “\_” and the next consecutive number (i.e., with 3 REPS and a label of “data” the labels for each REP are: data\_1, data\_2, and data\_3).

Only the first location of an output processing instruction is linked to the label. Reps of input/output instructions and output processing instructions are not linked, so use care if altering their sequence in the Input Locations Editor.

As an example, in the following section of an Edlog program, the TempC and BatteryV Input Locations are sampled with one sample (P70) instruction, with the REPS parameter of 2.

10: Temperature (107) (P11)	
01: 1	REPS
02: 2	Channel
03: 1	Excitation Channel
04: 1	Loc [TempC]
05: 1	Mult
06: 0	Offset
11: Battery, Volt (P10)	
01: 2	Loc [BatteryV]
12: If time is (P92)	
01: 0	minutes into interval
02: 60	minute interval
03: 10	Set high Flag 0(output)
13: Sample (P70)	
01: 2	Reps
02: 1	Loc [TempC]

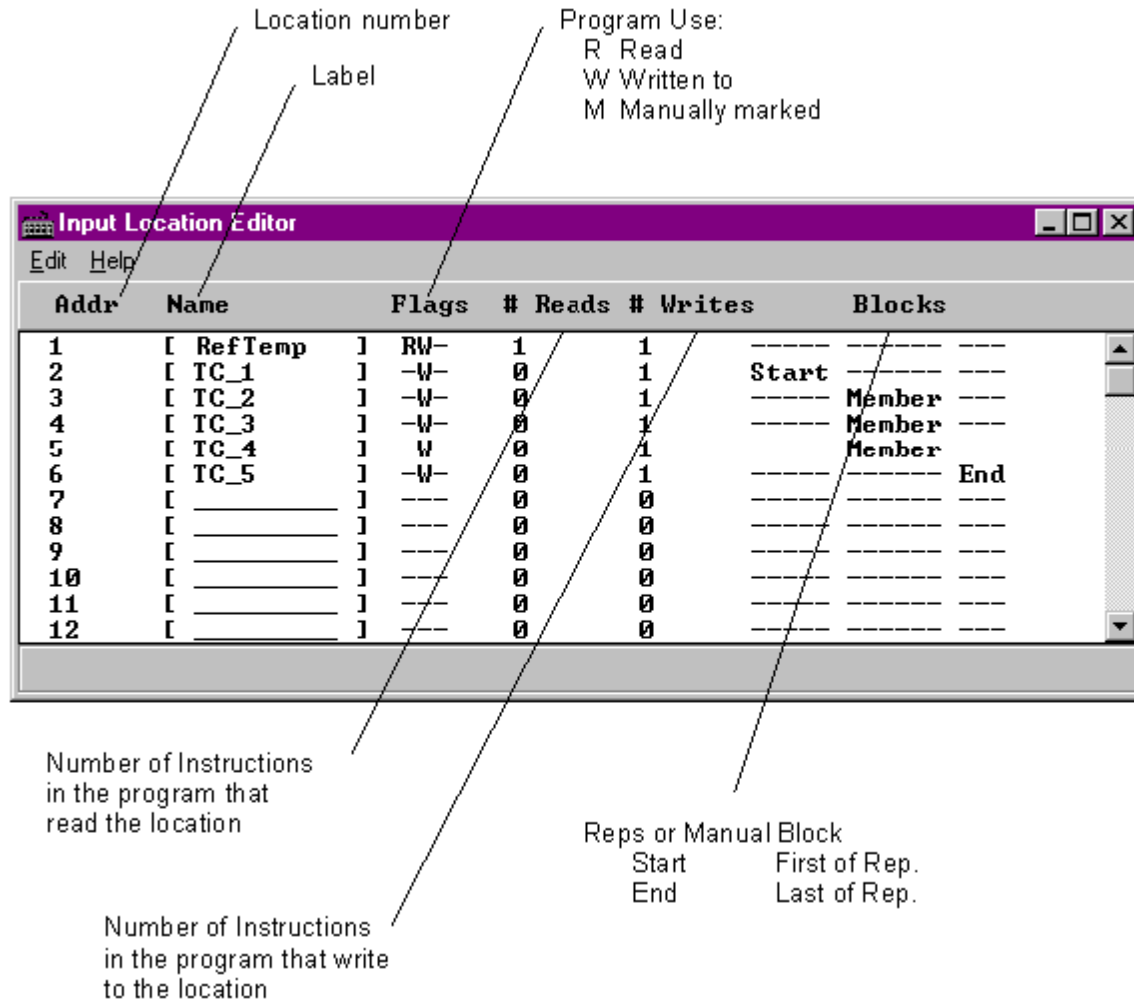
When the program is executed, the datalogger will perform the same instruction twice. The first time, it will sample the value stored in the TempC location. The second time, it will sample the value stored in the battery location.

**NOTE**

If an Input Location is inserted between the TempC and BatteryV location, the inserted location will be sampled instead of BatteryV.

### 12.2.3 Input Location Editor

Input Location labels can be entered and edited by using the Input Location Editor. To access the Input Location Editor, select Edit | Input Labels.



Editing functions are available from the Input Location Editor's Edit menu and a hot key:

**Insert (<F2>)** - Inserts blank Input Locations. This is used to provide space for new input labels between existing labels. This automatically changes the Input Location numbers for all of the labels that are after the inserted location.

**Delete (<F3>)** - Deletes the Input Location label, flags, number of reads and writes, and block information for a designated location number. Wherever the datalogger program references a deleted location label, the Input Location's number automatically becomes 0.

**Move (<F4>)** - Moves the Input Location to a different number. This may change several Input Location numbers.

**Toggle Manual (<F5>)** - Allows the programmer to manually toggle a location as "in use". This is used for burst mode, indexed loops, or other situations where it's not clear to Edlog that the locations are being written to. Input Locations not marked as read, write, or manual are deleted by the optimize command.

Optimize (<F6>) - Deletes Input Locations that aren't read, written to, or marked as Manual. Optimize tries to reduce the total number of locations used by moving Input Location labels to fill in unused locations. This might change several Input Location numbers. Any changes in location number made by the optimize command are reflected in the Edlog program.

Insert Block (<F7>) - Inserts and labels a block of Input Locations and marks them as "Manual". The locations are labeled in the same manner as reps.

Esc - The escape key closes the Input Location Editor and updates the label assignments in the program.

### 12.2.4 Input Location Anomalies

In most instances, Edlog will automatically assign Input Locations for locations which are generated by the datalogger program. An example of this is Edlog's handling of Input Locations for the REPS parameter. Though only one Input Location is specified, if REPS is greater than 1, additional Input Locations are created by Edlog.

There are certain instructions that generate multiple Input Locations for which Edlog does not automatically allocate Input Locations. The user should manually allocate these locations in the Input Location Editor. These are:

- Instruction 15, Serial I/O with Control Port
- Instruction 23, Burst Measurement
- Instruction 49, Spatial Maximum
- Instruction 50, Spatial Minimum
- Instruction 54, Block Move
- Instruction 75, Histogram
- Instruction 80, Store Area
- Instruction 81, Rainflow Histogram
- Instruction 100, TDR Measurement
- Instruction 101, SDM-INT8
- Instruction 105, SDI-12 Recorder
- Instruction 106, SDI-12 Sensor
- Instruction 113, SDM-SIO4
- Instruction 118, SDM CAN
- Instruction 119, TDR100
- Instruction 120, Data Transfer to TGT
- Instruction 127, HDR Goes Status and Diagnostics
- Instruction 128, SHEF Data Transfer to TGT
- Instruction 189,
- Instructions P190-199 PakBus control
- Indexed input locations in a loop

See Edlog Help for each instruction to get a detailed description of input location usage. You can also refer to the datalogger user's manual for more information on these instructions.

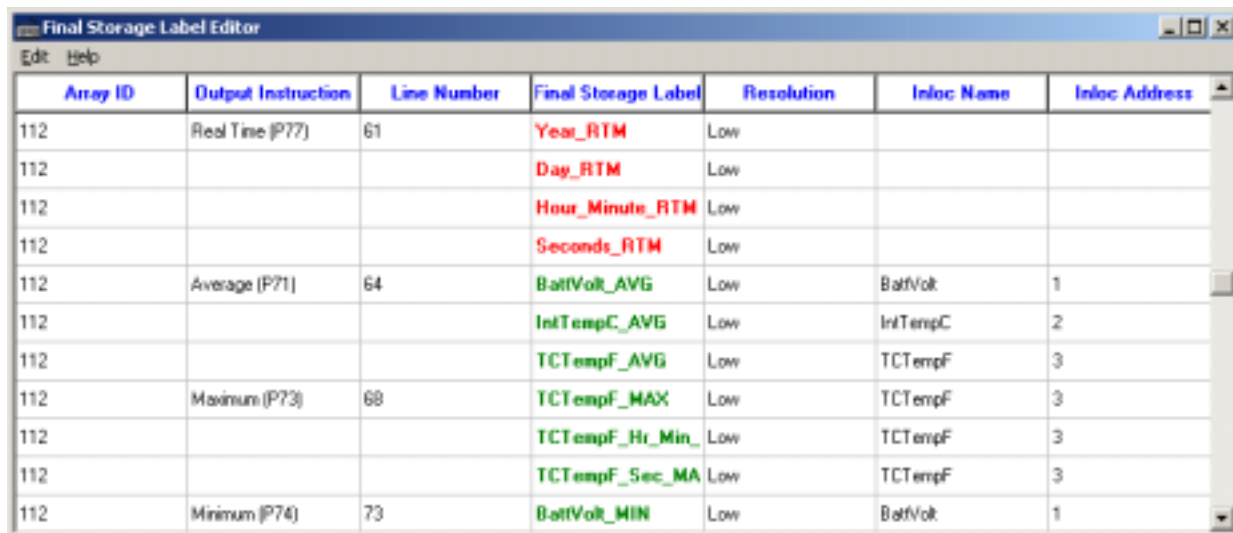
When these instructions are used in a program, the Toggle Manual feature can be used to manually mark Input Locations for use by the program.

## 12.3 Final Storage Labels

When output processing instructions are added to the datalogger program, Edlog creates final storage labels for each of the values that will be stored. The default labels are normally the input location label with a suffix indicating the type of output process instruction that created it. In the example below BattVolt\_AVG is the average battery voltage that is stored as part of array 112.

For table based dataloggers the final storage labels are included as part of the datalogger program in the \*.DLD file. LoggerNet gets the final storage labels as part of the table definitions from the datalogger. Graph and Numeric Display as well as RTMC use the final storage labels.

The user can create a custom label to reflect the meaning of the value that is being stored. Click on the FSL Edit button on the toolbar or press F9 to bring up the Final Storage Label Editor as shown below.



Array ID	Output Instruction	Line Number	Final Storage Label	Resolution	Inloc Name	Inloc Address
112	Real Time (P77)	61	Year_RTM	Low		
112			Day_RTM	Low		
112			Hour_Minute_RTM	Low		
112			Seconds_RTM	Low		
112	Average (P71)	64	BattVolt_AVG	Low	BattVolt	1
112			IntTempC_AVG	Low	IntTempC	2
112			TCTempF_AVG	Low	TCTempF	3
112	Maximum (P73)	68	TCTempF_MAX	Low	TCTempF	3
112			TCTempF_Hr_Min_	Low	TCTempF	3
112			TCTempF_Sec_MA	Low	TCTempF	3
112	Minimum (P74)	73	BattVolt_MIN	Low	BattVolt	1

In this example from an array based datalogger, the final storage output data for Array ID 112 is shown. Each of the columns indicate the essential characteristics of the data value being stored.

- Table Name identifies the set of output data instructions the data is associated with. For array based dataloggers the array ID is at the beginning of each output record. In table based dataloggers, the table name shows the name of the table where the data values will be stored.
- Output Instruction lists the output instruction that was used to output the data value.

- Line Number is the line number in the Edlog program for the output instruction.
- Final Storage Label is the label that is associated with this final storage value. Red labels are associated with automatically created data entries such as time stamps and record numbers. The red labels cannot be changed with the Final Storage Label Editor. The green labels are associated with user programmed sensor data. To change the label, click in the box and type in the new label.
- Resolution shows whether the data will be stored in low or high resolution. (High resolution stores data as a 4-byte floating point number, Low resolution uses a 2-byte number)
- Inloc Name is the label of the input location that the final storage data is based on.
- Inloc Address is the numeric label for the input location used for the final storage data value.

---

**NOTE**

If changes are made to measurement or output instructions after custom final storage labels have been created, you should review the custom final storage labels to make sure the correct labels are still assigned to the desired output values. Some program changes involving an increase or decrease in input locations or output values could cause a label to no longer correspond with the value being output.

---

The final storage labels created by Edlog can be restored by selecting the menu item Edit | Restore Default Labels.





# Section 13. Datalogger Program Creation with CRBasic Editor

---



*This section provides information on the CRBasic Editor used to program the Campbell Scientific CR5000 and CR9000 dataloggers. CRBasic is a full featured programming language providing the power and flexibility to set up complex datalogger programs to support demanding measurement tasks.*

*(Datalogger programs can also be created using the PC9000 program generator software that is included with the CR5000 and CR9000. The program generator is not included as part of LoggerNet and will not be covered in this manual.)*

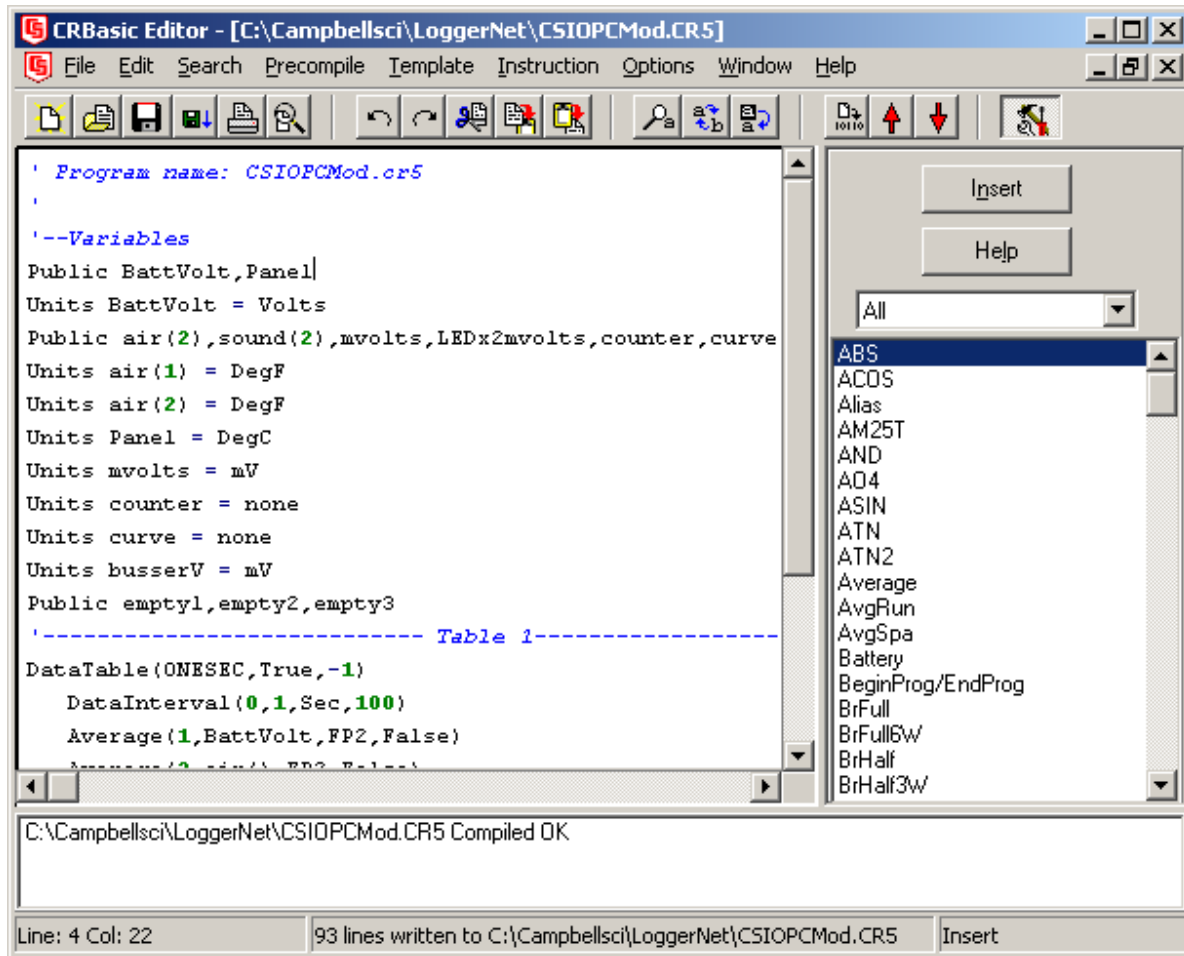
*See Section 8 for information about the program editor for other Campbell Scientific dataloggers, Edlog.*

## 13.1 Overview

The CRBasic Editor is a programming tool for the CR5000 and CR9000 dataloggers. It is intended for use by experienced datalogger programmers who need more flexibility and control over the datalogger operation. This programming language is similar in syntax, program flow, and logic to the Structured BASIC programming language.

As shown below, the CRBasic Editor's main window is divided into three parts: the program entry window, the Instruction Panel, and the message area. The Instruction Panel on the right side is a list that comprises the instructions for a particular datalogger in the CRBasic language. Instructions can be selected from this list or entered directly into the program entry window on the left. The message area at the bottom shows results of the compile and any errors detected.

The CRBasic Editor has been designed with several features to help make creating and editing programs easier.



### 13.1.1 Inserting Instructions

An instruction can be easily inserted into the program by highlighting it in the Instruction Panel list and pressing the Insert button or by double clicking the instruction name. If an instruction has one or more parameters, a parameter dialog box will be displayed. Complete the information in the parameter fields and press Insert to paste the instruction into the program.

You can filter the list of instructions available in the Instruction Panel by clicking the drop down arrow to the right of the text box above the list. This will allow you to display only instructions of a specific type such as Measurement/Control or Declarations. This provides a smaller list to select from and makes it easier to find the instruction you want. Switch back to All to see all of the instructions available. You can create custom instruction filter lists as described in Section 13.1.7.

Where the instruction is inserted depends upon the settings in the Instruction Panel Options box (accessed via Options | Instruction Panel Preferences). If At Cursor is selected, the instruction will be inserted directly at the cursor location. Care should be taken when using this option, so that new instructions are not inserted in the middle of existing ones. If Whole Line is selected, the

instruction will be placed on the line where the cursor is located and the existing line will be moved down to the next line.

#### NOTE

Be careful using the At Cursor option since it is possible to insert the new instruction in the middle of an existing instruction.

If the On Inserting, Edit Parameters check box is cleared, the Parameter dialog box will not be displayed when an instruction is inserted.

### 13.1.2 Parameter Dialog Box

The Parameter dialog box will appear when an instruction is added that has one or more parameters or when the cursor is placed on an existing instruction and the right mouse button is pressed. This dialog box contains a field for each of the parameters in the instruction. Edit these fields as necessary and then press the Insert button to paste the instruction into the program.

Below is an example of the Parameter dialog box for the differential voltage instruction (VoltDiff).

Parameter Type	Value	Comment
Destination	Volt()	
Repetitions	1	
Voltage Range	mV5000	
DiffChan	1	
RevDiff	True	Measure second time Low referenced to High
SettlingTime	0	
Integration	250	250 us integration
Multiplier	1.0	
Offset	0	

Variables: Volt() [dropdown]

[Insert] [Cancel] [Help]

#### Short Cuts for Editing the Parameters

Rightclicking or pressing F2 on a parameter that uses a Variable as an input type will invoke a list of variables that have been defined in the program. A sample list is shown below.

The variable list is sorted by variable type and then alphabetically by name. In the list below, the first green A denotes that the variable AIRCOOL is set up as an Alias.

Constants are listed with a blue C, Dimensioned variables are listed with a red D, and Public variables are listed with a black P.



At any time you can press F10 to bring up the list of variables, regardless of the input type for the selected parameter. Also, defined variables can be selected from the Variables drop-down list box at the upper right of the Parameter dialog box.

Right clicking or pressing F2 on a parameter that has a finite number of valid entries will bring up a list of those available options.

Right clicking or pressing F2 on a parameter that does not fall within the two categories above will bring up help for that parameter.

Pressing F1 with any parameter selected will bring up help for that parameter along with a list of possible options where appropriate.

#### Changing Default Parameters Values for an Instruction

Each instruction offers default values for each parameter. For instance, in the Parameter box above, the default for the Range is mV5000. If you wanted to edit this so that each time you inserted the VoltDiff instruction the Range value defaulted to mV1000, you would highlight the instruction in the Instruction Panel, select Instruction | Edit Instruction Defaults from the menu, and make the change in the resulting dialog box.

### 13.1.3 Right Click Functionality

The result of a right click action varies, depending upon your cursor location.

Right click on an instruction name to show the Parameter dialog box to edit the instruction parameters.

Right click on a parameter that uses a Variable as an input type to bring up a list of variables that have been defined in the program as described in the previous section.

Right click on a parameter that has a finite number of valid entries to bring up a list of those available options. You can change the option by clicking on the desired option.

Right click on another type of parameter to bring up help for that parameter.

Right click on a block of text that is highlighted to bring up a short cut menu with the following options:

- **Comment/Uncomment Block:** Only one of these options will be available, depending upon the status of the highlighted text. If the text has been marked as a Comment, you can choose to uncomment it. If the text is not commented, you can chose to make it into a comment. Commented text has a single quote ( ' ) at the beginning of the line. Comments are ignored by the datalogger's compiler.
- **Decrease/Increase Indent:** You can increase or decrease the indention of the selected text. The spacing is increased or decreased by one.
- **Cut/Copy/Paste/Delete:** Standard editing functions can be accessed through this menu.

### 13.1.4 Toolbar

The toolbar of the CRBasic Editor provides easy access to frequently used operations.



**New** – Creates a new program window to start writing a new program. If you have defined a default template, the new program will start with the defined template instructions.



**Open** – Brings up a File Open dialog to select a program file to open. File extension filters are provided to list only files of a certain type such as .cr5 files for CR5000 programs. Data files (\*.dat) can also be opened.



**Save** – Saves any changes to the currently opened program. If this a new program and has not been saved yet, a Save As dialog will prompt you for the file name and location to save the file.



**Save and Download** – Saves any changes to the currently opened program and opens PC9000 to send the file to the datalogger. If the CRBasic editor can't find PC9000 you will be prompted to browse for it.



**Print** – Prints the program listing of the currently open program to the printer.



**Print Preview** – Opens a Print Preview screen that will show what the program will look like when printed. You can check and set the margins and printer options.



**Undo** – Each time the Undo button is clicked it will step back through the last changes made to the program.



**Redo** – Cancels the undo and steps forward restoring the changes.



Cut – Removes the selected part of the program and puts it on the clipboard to be pasted elsewhere.



Copy – Places a copy of the selected part of the program on the clipboard to be pasted elsewhere.



Paste – Inserts a copy of the contents of the clipboard into the program at the cursor location.



Find – Brings up a Find dialog to specify a text string to search for in the program listing. Click on the Find Next button or press F3 to go to successive occurrences of the text.



Replace – Brings up a Find and Replace dialog that allows you to specify a text string to search for and a text string to replace it with. You can replace all occurrences of the text or check them one at a time to make sure they should be replaced.



Find Next – Finds the next occurrence of the text string specified in the Find dialog.



Compile – Starts the compiler to check the current program for errors and consistency. Compile results and errors will be displayed in the message area at the bottom of the screen.



Previous Error – Moves the cursor to the part of the program where the previous error was identified.



Next Error – Moves the cursor to the part of the program where the next error was identified.



Instruction Panel – Controls whether the Instruction Panel is displayed. Hiding the Instruction Panel can allow more room in the window to view the program listing.

### 13.1.5 Compile

Compile is a function provided by the CRBasic Editor to help the programmer catch problems with the datalogger program. Compile is available from the toolbar and the Compile menu.

When the Compile function is invoked, the CRBasic Editor checks the program for syntax errors and other inconsistencies. The results of the check will be displayed in a message window at the bottom of the main window. If an error can be traced to a specific line in the program, the line number will be listed before the error. You can double-click an error preceded by a line number and that line will be highlighted in the program editing window. To move the highlight to the next error in the program, press the Next Error button or choose Next Error from the Compile menu. To move the highlight to the previous error in the program, press the Previous Error button or choose Previous Error from the Compile menu.

The error window can be closed by selecting the Close Message Window menu item.

**NOTE**


---

This function only verifies the integrity of the program. Actual compilation of the program takes place in the datalogger.

---

## 13.1.6 Templates

The use of templates can be a powerful way to quickly create a set of similar datalogger programs. All or part of a program can be saved so that it can be used when creating new programs. These files are called templates. The Template menu provides access to create and use templates.

**Save File as Template** - Saves the comments and instructions in the active file as a template. To save part of a program as a template, copy the selected part to a new program file and then Save File as Template.

**Save as Default Template** - Saves the comments and instructions in the active file as a template that will be used each time File | New is selected for that type of datalogger.

**Save as Initial Template** - Saves the comments and instructions in the active file as a template that will be used each time the CRBasic Editor is started. If Display Last Window Used option is selected, the initial template will not be used.

**Delete** - When selected, a list of all dataloggers is displayed. Select a datalogger to invoke a dialog box containing a list of saved templates. A template can then be highlighted and deleted from disk.

**(Datalogger Types)** - When a datalogger type is selected, a list of all templates is displayed.

**NOTE**

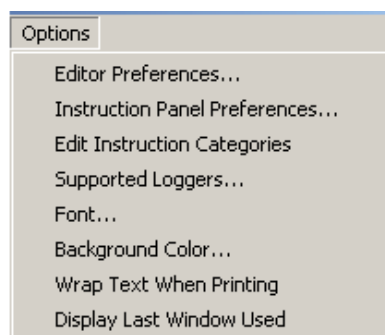

---

Template files are associated with a specific datalogger type. Templates for a CR5000 cannot be used for CR9000 programming and vice versa. Each datalogger has its own set of instructions that may be different than the other.

---

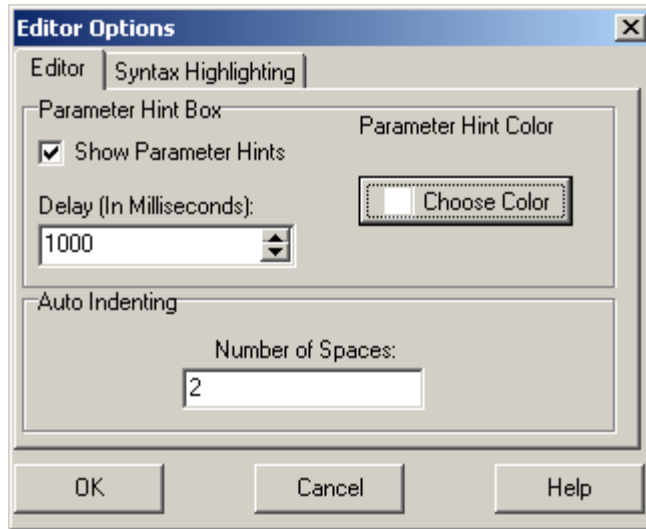
## 13.1.7 CRBasic Editor Options

This menu item allows you to specify the files used in the CRBasic Editor and customize its look and syntax highlighting.

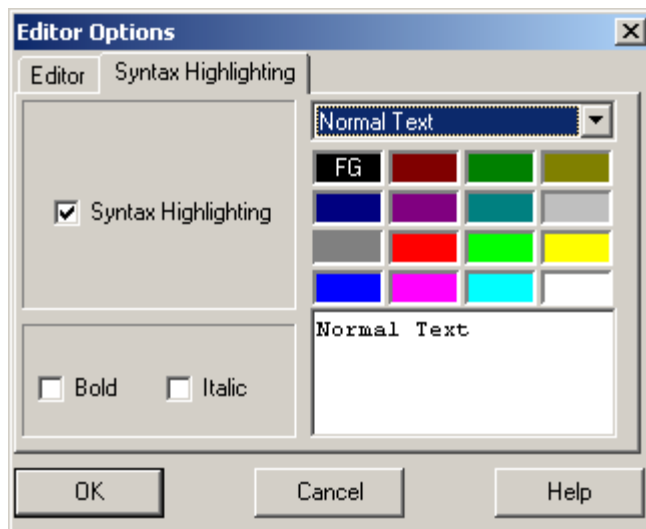


**Editor Preferences** sets up the appearance options for the text instructions and the behavior of popup hints.

The **Editor** tab allows the user to toggle the popup hints for instructions on or off, set the amount of time the cursor must hover over the instruction before the popup hint occurs, and the background color of the popup hint. This is also used to set the number of spaces used when the CRBasic Editor automatically indents an instruction.

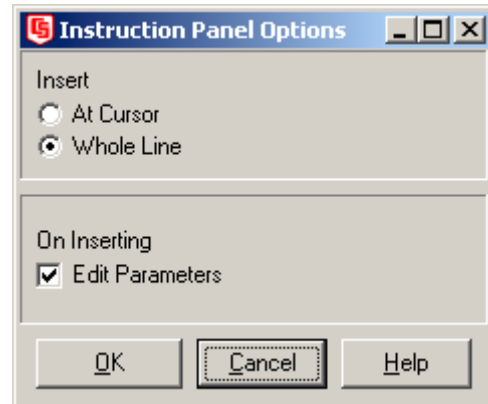


The **Syntax Highlighting** tab sets up the appearance of different text elements in the program using different font styles and colors. Comments, Instruction Names, and other text elements can each have a different appearance. You can customize the appearance of the text by giving normal text, keywords, comments, operators, and numbers each a different font style and color to make the program easier to read and edit. Text colors and styles can be disabled by clearing the Syntax Highlighting check box.



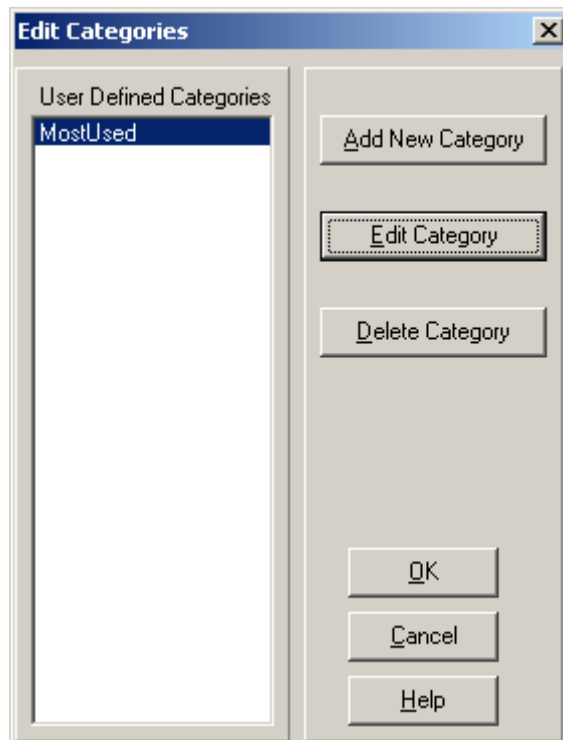


**Instruction Panel Preferences** allows the user to define how the insert function will behave in the program.

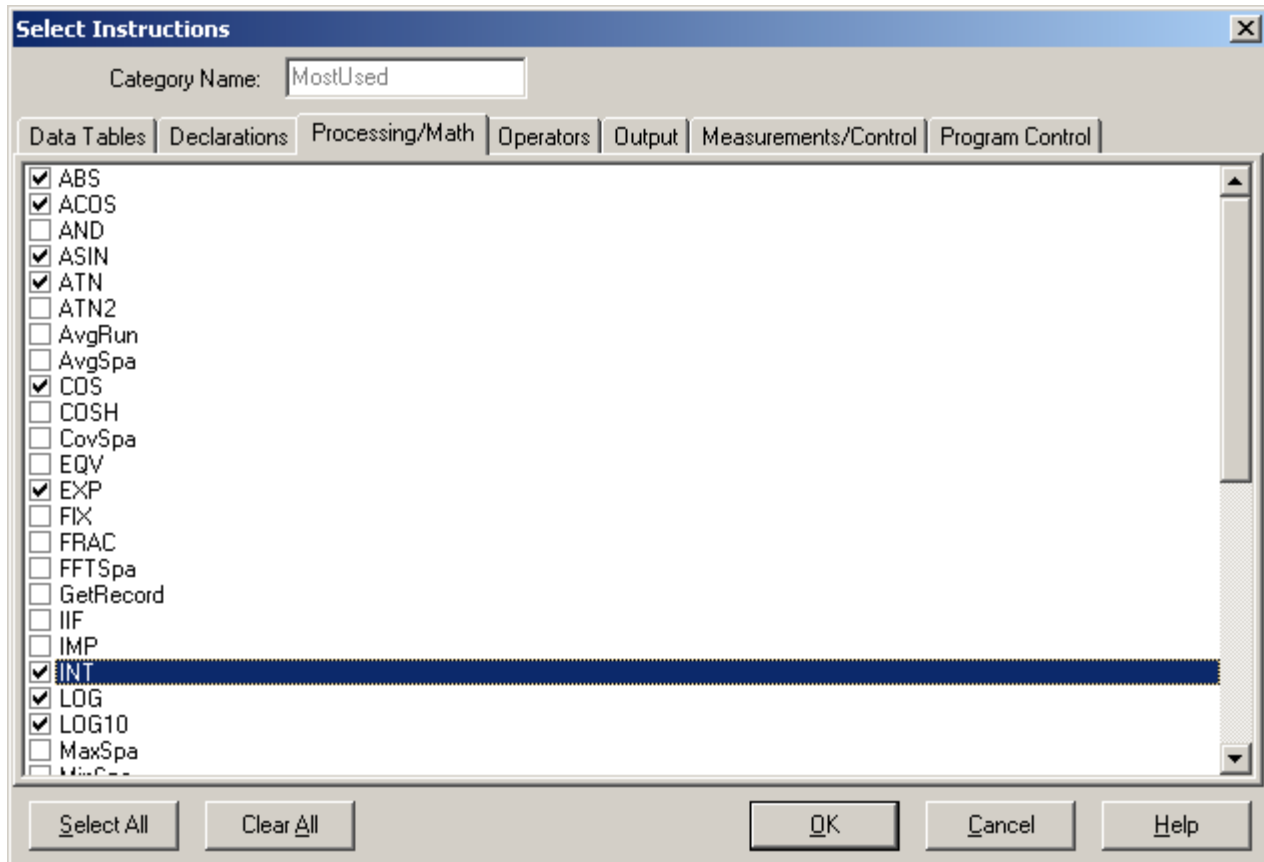


- **At Cursor/Whole Line:** When At Cursor is selected, inserted instructions will appear at the cursor location. When Whole Line is selected, a new line will be created for the instruction. (Note that the At Cursor option may cause you to unintentionally insert an instruction in the middle of another instruction.)
- **On Inserting:** When Edit Parameters is selected, the Edit Parameters dialog box will appear when an instruction is added to the program. When it is not selected, the instruction will be inserted directly, using the default parameters.

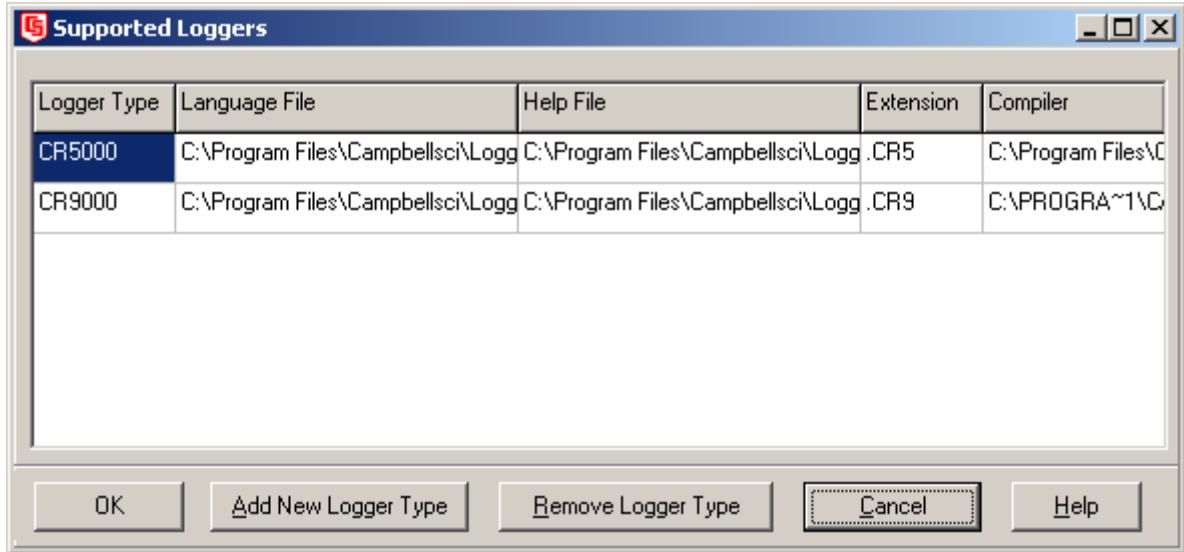
**Edit Instruction Categories** allows the user to create custom categories. (Note: The default categories cannot be edited or deleted.)



In the Edit Categories dialog shown above, a new category MostUsed has been added. Clicking on the Edit Category button will bring up the Select Instructions dialog that is shown below. This will allow the user to display a filtered instruction list containing only those instructions most often used. The instructions to be displayed are selected in the Select Instruction screen shown below. Select the check box next to the instructions you want in the user category from each list of instruction types. Press OK to save the list.



**Supported Loggers** allows you to set up additional dataloggers to be programmed using the CRBasic Editor. In the dialog window shown below the editor is set up for CR5000 and CR9000 dataloggers. When new dataloggers are added the language, help and compiler files can be linked to the CRBasic editor in this window.



**Font** brings up a font selection dialog to select the font typeface and size for the text in the CRBasic Editor. Font style and color are set under Editor Preferences.

**Background Color** brings up the color selection dialog to set the color of the CRBasic program window.

**Wrap Text When Printing** - When this option is selected, long lines that extend past the right margin will be wrapped to the next line. This option affects printing, as well as the Print Preview mode. A check mark will appear next to the option in the menu when it is selected.

**Display Last Window Used** - When this option is enabled, the program that was active in the CRBasic Editor when it was last closed will be visible when the Editor is reopened. If this option is disabled, no program will be loaded when the Editor is opened. A check mark will appear next to the option in the menu when it is selected.

### 13.1.8 Available Help Information

Pressing the Help button at the bottom right of the Parameter dialog box will bring up detailed help topic for the instruction being edited. Pressing F1 when your cursor is within a parameter field will bring up help only on that parameter. Some fields also have text in the Comments column, which provides a short description of the option that has been selected for the parameter.

## 13.2 CRBasic Programming

CRBasic is a programming language that has some similarities to a structured BASIC. There are special instructions for making measurements and for creating tables of output data. The results of all measurements are assigned variables (given names). Mathematical operations are written out much as they would be algebraically. This section describes a program, its syntax, structure, and sequence.

### 13.2.1 Programming Sequence

The structure of a datalogger program requires that variables and subroutines be defined before they can be used. The best way to do this is to put all the variable declarations and output table definitions at the beginning, followed by the subroutines, and then the program. Below is the typical layout of a program. Note that the online help has example code for each instruction to demonstrate the use of the instruction in a program.

<b>Declarations</b>	<i>Make a list of what to measure and calculate.</i>
<b>Declare constants</b>	<i>Within this list, include the fixed constants used,</i>
<b>Declare Public variables</b>	<i>Indicate the values that the user is able to view while the program is running,</i>
<b>Dimension variables</b>	<i>the number of each measurement that will be made,</i>
<b>Define Aliases</b>	<i>and specific names for any of the measurements.</i>
<b>Define data tables</b>	<i>Describe, in detail, tables of data that will be saved from the experiment.</i>
<b>Process/store trigger</b>	<i>Set when the data should be stored. Are they stored when some condition is met? Are data stored on a fixed interval? Are they stored on a fixed interval only while some condition is met?</i>
<b>Table size</b>	<i>Set the size of the table in RAM.</i>
<b>Other on-line storage devices</b>	<i>Should the data also be sent to the external storage?</i>
<b>Processing of Data</b>	<i>What data are to be output (current value, average, maximum, minimum, etc.).</i>
<b>Define Subroutines</b>	<i>If there is a process or series of calculations that needs to be repeated several times in the program, it can be packaged in a subroutine and called when needed rather than repeating all the code each time.</i>

<b>Program</b>	<i>The program section defines the action of datalogging.</i>
<b>Set scan interval</b>	<i>The scan sets the interval for a series of measurements.</i>
<b>Measurements</b>	<i>Enter the measurements to make.</i>
<b>Processing</b>	<i>Enter any additional processing with the measurements.</i>
<b>Call Data Table(s)</b>	<i>The Data Table must be called to process output data.</i>
<b>Initiate controls</b>	<i>Check measurements and Initiate controls if necessary.</i>
<b>NextScan</b>	<i>Loop back (and wait if necessary) for the next scan.</i>
<b>End Program</b>	

### 13.2.2 Program Declarations

Variables must be declared before they can be used in the program. Variables declared as Public can be monitored by LoggerNet using the Numeric Monitor screen or the Graphical Display. Variables declared using Dim cannot be displayed. Variables assigned to a fixed value are used as constants.

For example, in a CRBasic program there may be more than one temperature (or other) measurements. Rather than have many different names, a *variable array* with one name and many elements may be used. A thermocouple temperature might be called TCTemp. With an array of 20 elements the names of the individual temperatures are TCTemp(1), TCTemp(2), TCTemp(3), ... TCTemp(20). The array notation allows compact code to perform operations on all the variables. For example, to convert ten temperatures in a variable array from C to F:

```
For I=1 to 10
    TCTemp(I)=TCTemp(I)*1.8+32
Next I
```

Aliases can also be created that will allow an element of an array or another data result to be referred to by a different name.

### 13.2.3 Mathematical Expressions

Mathematical expressions can be entered algebraically into program code to perform processing on measurements, to be used for logical evaluation, or to be used in place of some parameters.

As an example of **Measurement Processing**, to convert a thermocouple measurement from degrees Celsius to degrees Fahrenheit, you could use the following expression:

`TCTempF=TCTemp(1)*1.8+32`

Logical Evaluation expressions could be used to determine the flow of a program:

```
If TCTemp(1) > 100 Then
Call Subroutine1
Else
'enter code for main program
End If
```

Many parameters will allow the entry of expressions. In the following example, the DataTable will be triggered, and therefore data stored, if `TCTemp(1)>100`.

`DataTable(TempTable, TCTemp(1)>100, 5000)`

## 13.2.4 Measurement and Output Processing Instructions

Measurement instructions are procedures that set up the measurement hardware to make a measurement and place the results in a variable or a variable array. Output processing instructions are procedures that store the results of measurements or calculated values. Output processing includes averaging, saving maximum or minimum, standard deviation, FFT, etc.

The instructions for making measurements and outputting data are not found in a standard basic language. The instructions Campbell Scientific has created for these operations are in the form of procedures. The procedure has a keyword name and a series of parameters that contain the information needed to complete the procedure. For example, the instruction for measuring the temperature of the CR5000 input panel is:

`PanelTemp (Dest, Integ)`

`PanelTemp` is the keyword name of the instruction. The two parameters associated with `PanelTemp` are: *Destination*, the name of the variable in which to put the temperature; and *Integration*, the length of time to integrate the measurement. To place the panel temperature in the variable `RefTemp` (using a 250 microsecond measurement integration time) the code is:

`PanelTemp(RefTemp, 250)`

The use of these instructions should become clearer as we go through an introductory example in Section 13.3.

### 13.2.5 Inserting Comments Into Program

It is often useful to provide comments in your datalogger program so that when you review the program at a later date, you will know what each section of code does. Comments can be inserted into the program by preceding the text with a single quote. When the program is compiled, the datalogger compiler will ignore any text that is preceded by a single quote. A comment can be placed at the beginning of a line or it can be placed after program code. If Syntax Highlighting is enabled (Options | Editor Preferences | Syntax Highlighting), commented text will appear formatted differently than other lines of code.

```
'CR5000

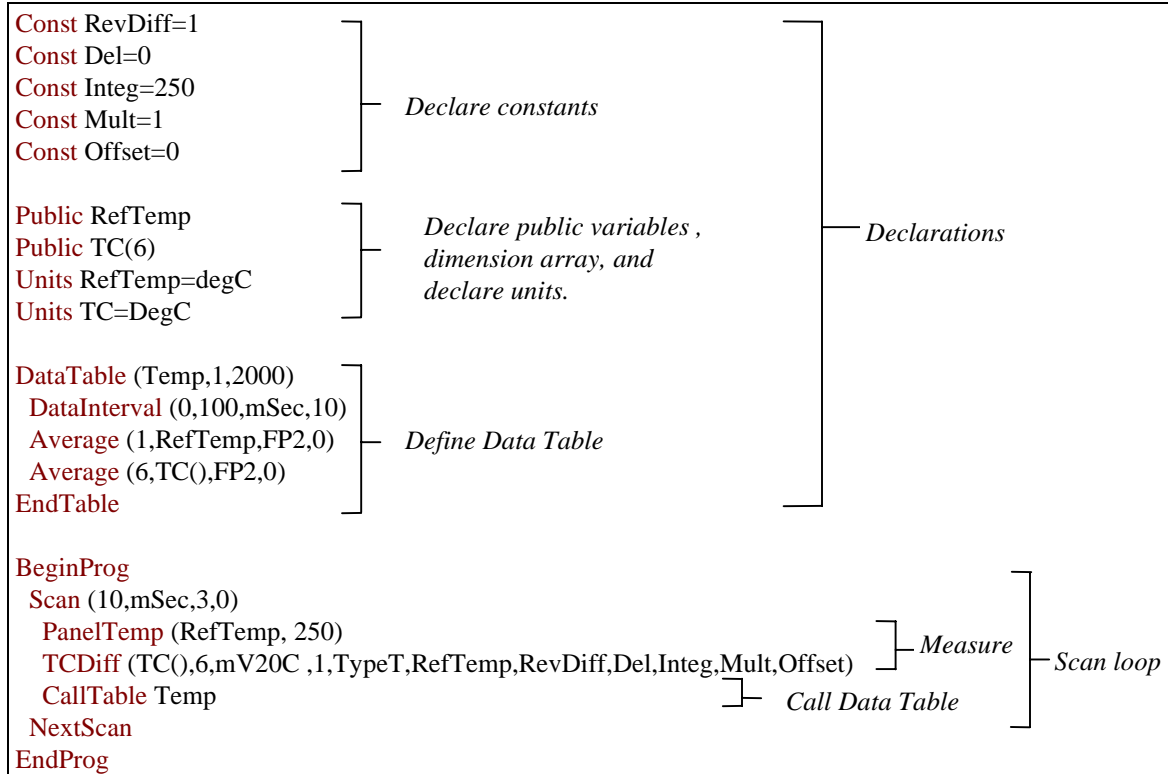
'The following program is used to measure
'4 thermocouples

'VARIABLE DECLARATION
Dim TCTemp(4)           'Dimension TC measurement variable
Alias TCTemp(1)=EngineCoolantT  'Rename variables
Alias TCTemp(2)=BrakeFluidT
Alias TCTemp(3)=ManifoldT
Alias TCTemp(4)=CabinT
```

In the sample code above, the commented text will be ignored by the datalogger compiler.

## 13.3 Example Program

The following program will serve as a programming example in this section to illustrate the concepts and program structure.



### 13.3.1 Data Tables

Data storage follows a fixed structure in the datalogger in order to optimize the time and space required. Data are stored in tables such as:

TOA5 TIMESTAMP TS	StnName RECORD RN	Temp RefTemp_Avg degC Avg	TC_Avg(1) DegC Avg	TC_Avg(2) degC Avg	TC_Avg(3) degC Avg	TC_Avg(4) degC Avg	TC_Avg(5) degC Avg	TC_Avg(6) degC Avg
1995-02-16 15:15:04.61	278822	31.08	24.23	25.12	26.8	24.14	24.47	23.76
1995-02-16 15:15:04.62	278823	31.07	24.23	25.13	26.82	24.15	24.45	23.8
1995-02-16 15:15:04.63	278824	31.07	24.2	25.09	26.8	24.11	24.45	23.75
1995-02-16 15:15:04.64	278825	31.07	24.21	25.1	26.77	24.13	24.39	23.76

The user's program determines the values that are output and their sequence. The datalogger automatically assigns names to each field in the data table. In the above table, `TIMESTAMP`, `RECORD`, `RefTemp_Avg`, and `TC_Avg(1)` are fieldnames. The fieldnames are a combination of the variable name (or alias if one exists) and a three letter mnemonic for the processing instruction that outputs the data. Alternatively, the `FieldNames` instruction can be used to override the default names.



The data table header may also have a row that lists units for the output values. The units must be declared for the datalogger to fill this row out (e.g., Units RefTemp = degC). The units are strictly for the user's documentation; the datalogger makes no checks on their accuracy.

The above table is the result of the data table description in the example program:

```
DataTable (Temp,1,2000)
    DataInterval(0,10,msec,10)
    Average(1,RefTemp,fp2,0)
    Average(6,TC(1),fp2,0)
EndTable
```

All data table descriptions begin with **DataTable** and end with **EndTable**. Within the description are instructions that tell what to output and that can modify the conditions under which output occurs.

*DataTable(Name, Trigger, Size)*  
 DataTable (Temp,1,2000)

The DataTable instruction has three parameters: a user specified name for the table, a trigger condition, and the size to make the table in CR5000 RAM. The trigger condition may be a variable, expression, or constant. The trigger is true if it is not equal to 0. Data are output if the trigger is true and there are no other conditions to be met. No output occurs if the trigger is false (=0). The example creates a table name Temp, outputs any time other conditions are met, and retains 2000 records in RAM.

*DataInterval(TintoInt, Interval, Units, Lapses)*  
 DataInterval(0,10,msec,10)

DataInterval is an instruction that modifies the conditions under which data are stored. The four parameters are the time into the interval, the interval on which data are stored, the units for time, and the number of lapses or gaps in the interval to track. The example outputs at 0 time into (on) the interval relative to real time, the interval is 10 milliseconds, and the table will keep track of 10 lapses. The DataInterval instruction reduces the memory required for the data table because the time of each record can be calculated from the interval and the time of the most recent record stored.

#### NOTE

Event driven tables should have a fixed size rather than allowing them to be allocated automatically. Event driven tables that are automatically allocated are assumed to have one record stored per second in calculating the length. Since the datalogger tries to make the tables fill up at the same time, these event driven tables will take up most of the memory leaving very little for the other, longer interval, automatically allocated data tables.

The output processing instructions included in a data table declaration determine the values output in the table. The table must be called by the program in order for the output processing to take place. That is, each time a

new measurement is made, the data table is called. When the table is called, the output processing instructions within the table process the current inputs. If the trigger conditions for the data table are true, the processed values are output to the data table. In the example below, several averages are output.

*Average(Reps, Source, DataType, DisableVar)*

Average(1,RefTemp,fp2,0)

Average(6,TC(1),fp2,0)

Average is an output processing instruction that will output the average of a variable over the output interval. The parameters are repetitions (the number of elements in an array to calculate averages for), the Source variable or array to average, the data format to store the result in (Table 13.3-1), and a disable variable that allows excluding readings from the average if conditions are not met. A reading will not be included in the average if the disable variable is not equal to 0; the example has 0 entered for the disable variable so all readings are included in the average.

**TABLE 13.3-1 Formats for Output Data**

Code	Data Format	Size	Range	Resolution
FP2	Campbell Scientific floating point	2 bytes	±7999	13 bits (about 4 digits)
IEEE4	IEEE four byte floating point	4 bytes	1.8 E -38 to 1.7 E 38	24 bits (about 7 digits)
LONG	4 byte Signed Integer	4 bytes	-2,147,483,648 to +2,147,483,647	1 bit (1)

### 13.3.2 The Scan -- Measurement Timing and Processing

Once the measurements and calculations have been listed and the output tables defined, the program itself may be relatively short. The executable program begins with BeginProg and ends with EndProg. The measurements, processing, and calls to output tables bracketed by the Scan and NextScan instructions determine the sequence and timing of the datalogging.

```

BeginProg
Scan(1,MSEC,3,0)
    PanelTemp(RefTemp, 250)
    TCDiff(TC(),6,mV50,4,1,TypeT,RefTemp,RevDiff,Del,Integ,Mult,Offset)
    CallTable Temp
NextScan
EndProg

```

The Scan instruction determines how frequently the measurements within the scan are made:

*Scan(Interval, Units, BufferOption, Count)*

Scan(1,MSEC,3,0)

The Scan instruction has four parameters. The Interval is the interval between scans. Units are the time units for the interval. The maximum scan interval is one minute. The BufferSize is the size (in the number of scans) of a buffer in

RAM that holds the raw results of measurements. Using a buffer allows the processing in the scan to at times lag behind the measurements without affecting the measurement timing (see the scan instruction in the help for more details). *Count* is the number of scans to make before proceeding to the instruction following NextScan. A count of 0 means to continue looping forever (or until ExitScan). In the example the scan is 1 millisecond, three scans are buffered, and the measurements and output continue indefinitely.

## 13.4 Numerical Entries

In addition to entering regular base 10 numbers there are 3 additional ways to represent numbers in a program: scientific notation, binary, and hexadecimal (Table 13.4-1).

TABLE 13.4-1 Formats for Entering Numbers in CRBasic		
Format	Example	Value
Standard	6.832	6.832
Scientific notation	5.67E-8	5.67X10 <sup>-8</sup>
Binary:	&B1101	13
Hexadecimal	&HFF	255

The binary format makes it easy to visualize operations where the ones and zeros translate into specific commands. For example, a block of ports can be set with a number, the binary form of which represents the status of the ports (1= high, 0=low). To set ports 1, 3, 4, and 6 high and 2, 5, 7, and 8 low; the number is &B00101101. The least significant bit is on the right and represents port 1. This is much easier to visualize than entering 72, the decimal equivalent.

## 13.5 Logical Expression Evaluation

### 13.5.1 What is True?

Several different words get used to describe a condition or the result of a test. The expression,  $X > 5$ , is either **true** or **false**. However, when describing the state of a port or flag, **on** or **off** or **high** or **low** is more intuitive. In CRBasic there are a number of conditional tests or instruction parameters, the result of which may be described with one of the words in Table 13.5-1. The datalogger evaluates the test or parameter as a number; 0 is false, not equal to 0 is true.

**TABLE 13.5-1. Synonyms for True and False**

Predefined Constant	True (-1)	False (0)
Synonym	High	Low
Synonym	On	Off
Synonym	Yes	No
Synonym	Trigger	Do Not Trigger
Number	≠0	0
Digital port	5 Volts	0 Volts

### 13.5.2 Expression Evaluation

Conditional tests require the datalogger to evaluate an expression and take one path if the expression is true and another if the expression is false. For example:

**If X>=5 then Y=0**

will set the variable Y to 0 if X is greater than or equal to 5.

The datalogger will also evaluate multiple expressions linked with **and** or **or**. For example:

**If X>=5 and Z=2 then Y=0**

will set Y=0 only if both X>=5 and Z=2 are true.

**If X>=5 or Z=2 then Y=0**

will set Y=0 if either X>=5 or Z=2 is true (see And and Or in the help). A condition can include multiple **and** and **or** links.

### 13.5.3 Numeric Results of Expression Evaluation

The datalogger's expression evaluator evaluates an expression and returns a number. A conditional statement uses the number to decide which way to branch. The conditional statement is false if the number is 0 and true if the number is not 0. For example:

**If 6 then Y=0,**

is always true, Y will be set to 0 any time the conditional statement is executed.

**If 0 then Y=0**

is always false, Y will never be set to 0 by this conditional statement.

The expression evaluator evaluates the expression, X>=5, and returns -1, if the expression is true, and 0, if the expression is false.

**W=(X>Y)**

will set W equal to -1 if X>Y or will set W equal to 0 if X<=Y.

The datalogger uses -1 rather than some other non-zero number because the **and** and **or** operators are the same for logical statements and binary bitwise comparisons. The number -1 is expressed in binary with all bits equal to 1, the number 0 has all bits equal to 0. When -1 is anded with any other number the result is the other number, ensuring that if the other number is non-zero (true), the result will be non-zero.

## 13.6 Flags

Any variable can be used as a flag as far as logical tests in CRBasic are concerned. If the value of the variable is non-zero the flag is high. If the value of the variable is 0 the flag is low. LoggerNet looks for the variable array with the name **Flag** when the option to display flag status is used in one of the real-time screens.

## 13.7 Parameter Types

Instruction parameters allow different types of inputs. These types are listed below and specifically identified in the description of the parameter in the following sections or in CRBasic help.

Constant  
Variable  
Variable or Array  
Constant, Variable, or Expression  
Constant, Variable, Array, or Expression  
Name  
Name or list of Names  
Variable, or Expression  
Variable, Array, or Expression

Table 13.7-1 lists the maximum length and allowed characters for the names for Variables, Arrays, Constants, etc.

TABLE 13.7-1. Rules for Names		
Name for	Maximum Length (number of characters)	Allowed characters
Variable or Array	16	Letters A-Z, upper or lower case, underscore “_”, and numbers 0-9. The name must start with a letter. CRBasic is not case sensitive.
Constant	16	
Alias	16	
Data Table Name	8	
Field name	16	

### 13.7.1 Expressions in Parameters

Many parameters allow the entry of expressions. If an expression is a comparison, it will return -1 if the comparison is true and 0 if it is false (Section 13.5.3). An example of the use of this is in the DataTable instruction where the trigger condition can be entered as an expression. Suppose the variable TC(1) is a thermocouple temperature:

*DataTable(Name, TrigVar, Size)*

DataTable(Temp, TC(1)>100, 5000)

Entering the trigger as the expression,  $TC(1) > 100$ , will cause the trigger to be true and data to be stored whenever the temperature  $TC(1)$  is greater than 100.

### 13.7.2 Arrays of Multipliers & Offsets for Sensor Calibration

If variable arrays are used for the multiplier and offset parameters in measurements that use repetitions, the instruction will automatically step through the multiplier and offset arrays as it steps through the channels. This allows a single measurement instruction to measure a series of individually calibrated sensors, applying the correct calibration to each sensor. If the multiplier and offset are not arrays, the same multiplier and offset are used for each repetition.

*VoltSE(Dest,Reps,Range,SEChan,Delay,  
Integ,Mult,Offset)*

'Calibration factors:  
 Mult(1)=0.123 : Offset(1)= 0.23  
 Mult(2)=0.115 : Offset(2)= 0.234  
 Mult(3)=0.114 : Offset(3)= 0.224  
 VoltSE(Pressure(),3,mV1000,6,1,1,100,Mult(),Offset())

## 13.8 Program Access to Data Tables

Data stored in a table can be accessed from within the program. The format used is:

*Tablename.Fieldname(fieldname index,records back)*

Where *Tablename* is the name of the table in which the desired value is stored. *Fieldname* is the name of the field in the table. The fieldname is always an array even if it consists of only one variable; the *fieldname index* must always be specified. *Records back* is the number of records back in the data table from the current time (1 is the most recent record stored, 2 is the record stored prior to the most recent). For example, the expression:

$Tdiff = Temp.TC\_Avg(1,1) - Temp.TC\_Avg(1,101)$

could be used in the example program (Section 4.3) to calculate the change in the 10 ms average temperature of the first thermocouple between the most recent average and the one that occurred a second (100 x 10 ms) earlier.

In addition to accessing the data actually output in a table, there are some pseudo fields related to the data table that can be retrieved:

*Tablename.record(1,n)* = the record number of the record output n records ago.

*Tablename.output(1,1)* = 1 if data were output to the table the last time the table was called, = 0 if data were not output.

*Tablename.timestamp(m,n)* = element m of the timestamp output n records ago where:

timestamp(1,n) = microseconds since 1990  
timestamp(2,n) = microseconds into the current year  
timestamp(3,n) = microseconds into the current month  
timestamp(4,n) = microseconds into the current day  
timestamp(5,n) = microseconds into the current hour  
timestamp(6,n) = microseconds into the current minute  
timestamp(7,n) = microseconds into the current second

*Tablename.eventend(1,1)* is only valid for a data table using the DataEvent instruction, *Tablename.eventend(1,1)* = 1 if the last record of an event occurred the last time the table was called, = 0 if the data table did not store a record or if it is in the middle of an event.

---

**NOTE**

The values of *Tablename.output(1,1)* and *Tablename.eventend(1,1)* are only updated when the tables are called.

---





# Section 14. Real-Time Monitor and Control



*The Real-Time Monitor and Control (RTMC) software provides the ability to create and run graphical screens to display real-time data as LoggerNet collects it from the dataloggers. Controls are also provided to view and set datalogger ports and flags along with the data in settable variables. For graphical displays that can show historical data, such as line charts, RTMC will bring in and display the historical data available in LoggerNet's data storage. RTMC also provides expressions to condition the collected data for display (e.g., convert temperature in Celsius to Fahrenheit).*

*RTMC creates files that can be used with RTMC Run-Time on another computer that is connected to the LoggerNet computer by a local area network (LAN) or the Internet.*

## 14.1 Overview

Real-Time Monitor and Control is an application that allows the user to quickly create graphical display screens that monitor real-time data, control set points, and toggle ports and flags. RTMC can combine data from multiple dataloggers on a single display. As LoggerNet collects data from the dataloggers, the displays in RTMC are automatically updated.

RTMC has two operating modes: Development and Run-Time. The Development mode allows you to create and edit a real-time graphic display screen to display the data collected from the dataloggers. Once the screen is built and saved as a file, the screen can be displayed using RTMC Run-Time. This allows graphic display screens to run on other computers with just the RTMC Run-Time program.

## 14.2 Development Mode

RTMC Development is a graphic display editor that allows the user to easily place graphical components on the display screen and connect them to data values.

The RTMC Development window, as shown below, has three sections.

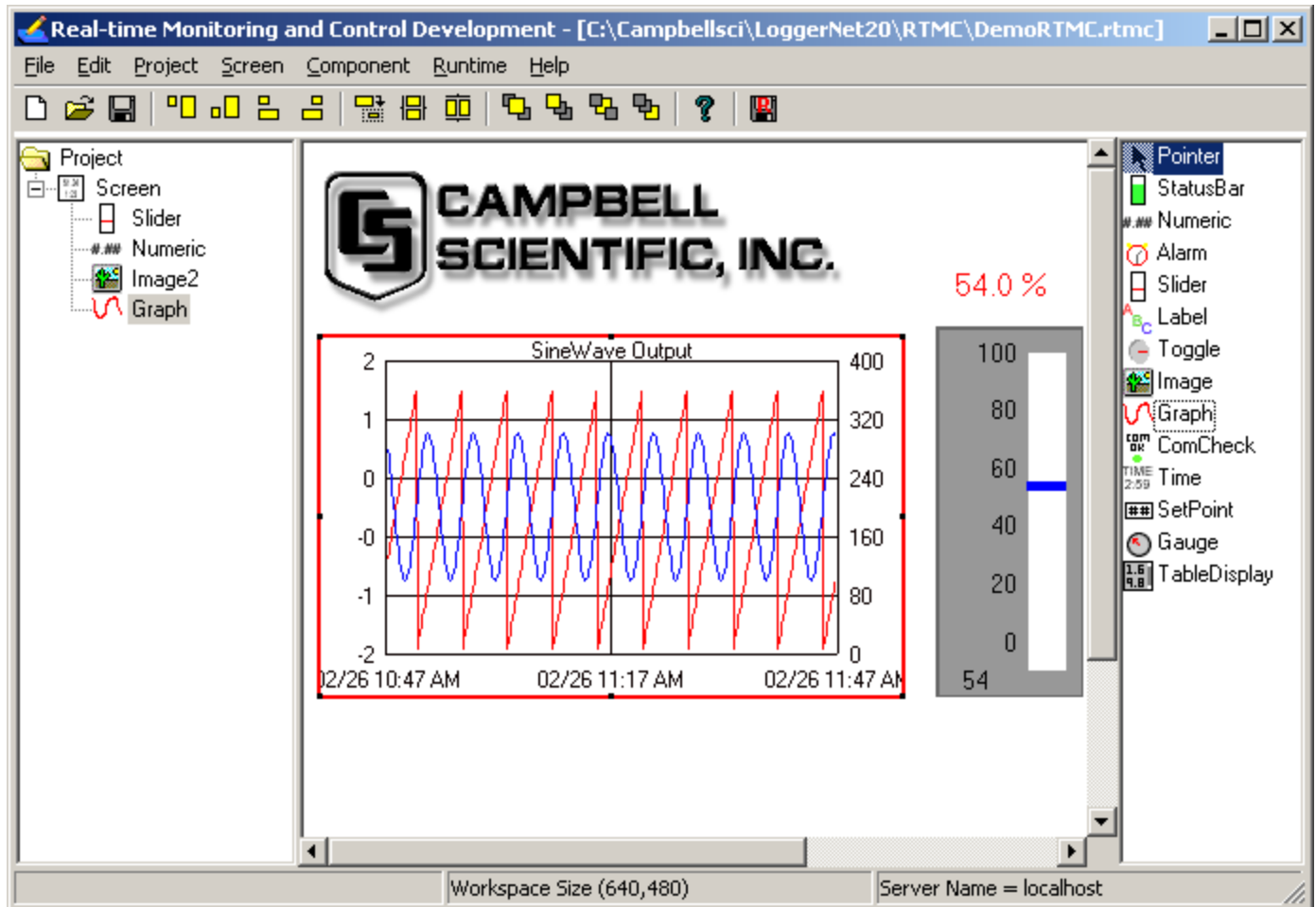
**Project Component List** - The panel on the left shows the hierarchy of the display components and how they are associated with each other. Every component of the display screen is shown in this list and it provides a shortcut to get to any graphical component.

**Project Workspace** - The middle panel is the display screen workspace. The graphic components are placed in the workspace, as they should appear on the final display.

**Display Components List** - The panel on the right contains the display screen components that can be placed in the workspace. Selecting a component and

clicking in the workspace places the component and brings up the Properties window for that component.

RTMC was designed to be easy and straightforward to use. Experiment with different combinations and options to get the display results you are looking for.



As seen in the example screen above, different types of graphic components can be combined to create an attractive real-time display. Company logos, maps, or any image stored in a standard graphic file format can be placed on the screen. Many images have been included with RTMC and others can be added as needed.

### 14.2.1 The RTMC Workspace

The RTMC workspace is a fixed size container for holding one or more display screens. As new display screens are added they appear as tabs in the project. The display screen is a container for the various display components that make up each tab of the real-time display.

## 14.2.2 Display Components

Display components are the objects that are used to display data. To add a component to the workspace, click on an item in the Display Components List in the RTMC window and then click anywhere in the workspace. The component's Properties window is automatically displayed when the object is first placed in the work area. The Properties window is used to set colors, scale values, text, etc., and to assign the data value to be displayed by the component.

### NOTE

When a display component is linked to a data value, the value will be automatically updated on the display when data is collected by LoggerNet. If data collection is not set up in LoggerNet or the selected data value is excluded from collection, the values will not update.

After a component's properties have been set, select Apply to enable the changes, or Apply & Close to enable the changes and close the Properties window. Once the link to the data value has been applied, if there is data available from LoggerNet for the component, the value on the display will update.

To make changes in display component settings, the Properties window can be invoked by double clicking the component. If you make changes to a component's properties but then decide to reject those changes, press the Cancel button to return the properties to the last applied state. If Cancel is selected when a component is first placed in the work area (and Apply or Apply & Close has not been pressed), the display component will be removed from the screen.

## Available Components

The following is an overview of the display components available. The online help has detailed information about each of the components and their properties.



Pointer

**Pointer** returns the cursor to a normal selection tool. Several components can be moved as a group, by selecting each with the pointer while holding the Ctrl key, and dragging them to a new position. A group of components can also be selected by dragging a box around them.



StatusBar

**Status Bar** is used for displays such as a thermometer where the height of the colored bar is based on the data value. You can set high and low values, bar color and placement on a background graphic.



### Numeric

**Numeric** displays the selected data value as a numeric value. You can set the font size and type along with the number of decimal points and a units label.



Alarm

**Alarm** is used to set up a display component that changes appearance when an alarm condition occurs. You can set the normal and alarm graphics, the alarm set point, and a sound file to play when the alarm occurs. In run-time mode, right clicking the component with your mouse, and selecting Acknowledge Alarm will disable an audible alarm.



Slider

**Slider** is used to display a value as a point on a scale. It can also be used to set the value of an input location or Public variable. You can set the color, background graphic and the upper and lower limits.



Label

**Label** is text that can be used to put titles or labels on the graphic components.



Toggle

**Toggle** is used to display the value of flags, ports, or other boolean data that can take one of two values. In run-time mode, it can be used to set a port, flag or boolean value. Any non-zero number is On or True and zero (0) is Off or False. In run-time mode, right click a toggle to change its state. The option to change the state of a toggle with a double click can be enabled in the Properties window. You can set the two graphic images and separate locations to set and read the value.



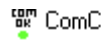
Image

**Image** is used to place graphic images on the display. Most types of graphic image file formats are supported including GIF, JPG, TIF, PNG, and BMP. You can set the angle of rotation if you want the graphic at an angle or turned sideways.



Graph

**Graph** is used to display a line graph of one or more data values. The time stamp on the X axis reflects the PC clock. Note that a difference in the PC clock and the datalogger clock, coupled with a small time window for the graph, could result in no data being displayed. You can add multiple traces to the graph, set the background, set right and left axis scales, and colors for traces.



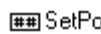
ComCheck

**ComCheck** is used to provide a visual and/or audible alarm when data collection has failed a sufficient number of times to put the datalogger into a Primary or Secondary Retry mode. In run-time mode, right clicking the component with your mouse will disable the audible alarm.



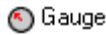
Time

**Time** is used to display time and date. You can set options to display the server time, server time at last data collection, station time, station time of last record stored, or PC time.



SetPoint

**SetPoint** is used like the Numeric component to display the selected data value as a number. When linked with an input location or Public variable in run-time mode, the data value can be set to a new value by double clicking the component and entering a new value in the resulting dialog box.



Gauge

**Gauge** is used to display the selected data value on a circular gauge. You can set the maximum and minimum scale along with the maximum and minimum pointer position.



TableDisplay

**TableDisplay** is used to display the data from a datalogger in a row and column format. For table based dataloggers, the data from one table is displayed. You can set the time format for the timestamp displayed next to each record.

**NOTE**

A popup description of each field in a component's Properties window can be displayed by pressing F1 while that field has the focus. Use the tab key or click with the mouse to select a field.

### 14.2.3 RTMC Operations

All of the RTMC operations are available from the menus at the top of the Development window. Many of the options are also available as buttons on the toolbar, or by right clicking on the components or other parts of the window. Operations that have toolbar buttons will have the button icon shown next to the description below.

#### File Menu



**New Project** starts a new RTMC project. The currently opened project will be closed. If there are changes that have not been saved the user will be prompted to save changes.



**Open** brings up the File Open dialog to open a previously saved project.



**Save** will save the changes in the current project to the RTMC project file. If this is the first time the project has been saved, a Save As dialog will open to select the file name and directory for the project file.

**Save As** brings up the Save As dialog to save the current project with another name or in a different directory.

**Exit** closes RTMC. If there are unsaved changes, the user will be prompted to save changes before exiting.

#### Edit Menu

**Undo** cancels the last change made to the project.

**Redo** repeats the change that was just undone.

**Cut/Copy/Paste** are standard editing operations to take selected objects to an internal clipboard and paste back into RTMC.

**NOTE**

Cut/Copy does not go to the Windows clipboard so these objects are not available to paste into other applications.

**Project Menu** options work with the whole project or workspace.

**Change Workspace Size** allows you to specify the size in pixels of the run-time display screen. Default is 640 x 480.

**Change Server Connection** allows you to connect to a LoggerNet server on the same or a remote PC. The server name is the network name or IP address of the computer where LoggerNet is running. If you are connecting to a version of LoggerNet that supports server security, and security is enabled, you will need to enter the username and password. (LoggerNet 2.0 does not support security.) The port number should always be 6789.

**Enable Connection Management** tells LoggerNet to try and maintain an active communications link with the dataloggers for which data is being displayed by RTMC. This might be beneficial if you are trying to display data in RTMC with a fast update rate, over a communications link that takes time to establish (such as a phone modem).

**Rename Project** changes the name of the project as shown on the component tree.

**Screen Menu** options work with the tabbed screens in the project. The Screen Menu is also available by right clicking on any blank area of the workspace.

**Screen Properties** brings up the dialog to choose the background image for the current screen.

**Add Screen** adds a new screen to the project. Each screen appears as a tabbed page on the display. When the project is run the user can click on the tab to bring each screen to the front.

**Delete Screen** removes the current screen from the project. If there are components on the screen, they will also be removed.

**Rename Screen** brings up a dialog to change the name of the current screen. This is the name that appears on the screen tab in run-time mode.

The **Component Menu** is used to set the component properties, placement and alignment. The Component Menu is also available by right clicking on any of the components in the workspace.

**Component Properties** brings up the Properties window for the selected component.

**Rename Component** lets you change the name of the component in the list tree.

**Delete Component Selection** removes the selected component from the workspace.

**Insert** brings up a submenu allowing you select one of the components to insert on the screen. When the component is added to the screen the Properties window for the new component will come up.

**Align** provides some options for lining up a group of components with the first component selected. Select two or more components by using the cursor to click and drag a bounding box around the desired components. Components can also be selected by selecting the first component and then selecting the other components while holding down the <ctrl> key. With the components selected choose one of the alignment options. The components will be aligned based on the first component selected. The first component is identified by the dark red boundary. The other selected components have a light red boundary.

**NOTE**

Be careful about the alignment you choose. Selecting Top Align for a group of components that are arranged vertically will cause all the components to end up on top of each other.



**Top Align** lines up the selected components to the top of the first component selected.



**Bottom Align** lines up the selected components to the bottom of the first component selected.



**Left Align** lines up the selected components with the left side of the first component selected.



**Right Align** lines up the selected components with the right side of the first component selected.

**Size** allows you to set two or more objects to the same overall size, width or height as the first object selected. Select one or more components by using the cursor to click and drag a bounding box around the desired components. The components can also be selected by selecting the first component and then selecting the other components while holding down the <ctrl> key. The first component is identified by the dark red boundary. The other selected components have a light red boundary.



**Copy Size** makes all the selected components the same size as the first component.



**Copy Width** makes the width of the selected components the same as the first component.



**Copy Height** makes the width of the selected components the same as the first component.

**Ordering** is used to manage the position of graphic objects on the workspace. Displays are often a combination of a background graphic and data display objects in front. Objects added to the workspace are, by default, placed on top of any existing objects. These operations are used to determine the order in which objects are displayed.



**Bring to Front** brings the selected component in front of all other display objects.



**Send to Back** places the selected component behind all of the other display objects.



**Move Forward** in areas where multiple objects are layered, brings the selected object forward ahead of display objects in the next layer up.



**Move Backward** in areas where multiple objects are layered, moves the selected object in back of display objects in the next layer down.

**Run-Time Menu** is used to Save and Run Current Project. This will save the project and start RTMC Run-Time to display the project.



Help Menu provides access to help for all of the features of RTMC. In Contents you can find an introduction and overview of RTMC as well as detail descriptions of all of the display properties and operations. The Index allows you to look up help topics based on a key word.

## 14.2.4 Expressions

RTMC has a built-in expression interpreter that allows the user to condition the data or create displays based on calculations on a data point.

The components that display data values (Status Bar, Numeric, Graph, and Gauge) can be processed using mathematical expressions. For instance, a temperature reading in degrees Celsius can be processed to display in degrees Fahrenheit using a mathematical expression.

An expression is entered for a component by first selecting the data value in the Select Data field, and then entering the mathematical expression after the defined data value. Using the above example, if the data value is defined as CR5000.TempData.Temp1 (datalogger.table.variable), you would enter

“CR5000.TempData.Temp1”\*1.8+32

to convert the temperature reading from degrees Celsius to degrees Fahrenheit.

The data values from different dataloggers can be combined to form the expression. Data values are referenced as in the above example by datalogger name as it appears in NetAdmin, followed by the table name, followed by the data label.

---

### NOTE

Spaces must be used before and after the predefined constants and functions. Operators do not require spaces.

---



#### 14.2.4.1 Operators

Operator	Description
( )	Prioritizes parts of an expression within the larger expression.
*	Multiply by
/	Divide by
^	Raised to the power of
+	Add
-	Subtract
=	Equal
<>	Not equal
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to

#### 14.2.4.2 Predefined Constants

Constant	Description
e	2.718282
PI	3.141593
True	-1
False	0

#### 14.2.4.3 Functions

The following functions show the use and placement of the numbers the function operates on. The parentheses are not required unless there are two or more parameter values. (e.g., ATN2(y,x))

Function	Description
ABS(x)	Returns the absolute value of a number.
ACOS(x)	Returns the arc cosine of a number.
(x)AND(y)	Performs a logical conjunction on two numbers.
ASIN(x)	Returns the arc sine of a number.
ATN(x)	Returns the arc tangent of a number.
ATN2(y,x)	Returns the arctangent of y/x.
COS(x)	Returns the cosine of a number.
COSH(x)	Returns the hyperbolic cosine of a number.
CSGN(x)	Changes the sign of a number by multiplying by -1.0.

(x)EQV(y)	Performs a logical equivalence on two numbers.
EXP(x)	Returns e raised to the x power.
FIX(x)	Returns the integer portion of a number. If the number is a negative, the first negative integer greater than or equal to the number is returned.
FRAC(x)	Returns the fraction part of a number.
IIF(x,y,z)	Evaluates an expression (x) and returns one value if true (y), a different value if false (z).
(x)IMP(y)	Performs a logical implication on two numbers.
INT(x)	Returns the integer portion of a number. If the number is a negative, the first negative integer less than or equal to the number is returned.
LOG(x)	Returns the natural log of a number.
LOG10(x)	Returns the logarithm base 10 of a number.
(x)MOD(y)	Performs a modulo divide of two numbers.
NOT(x)	Performs a logical negation on a number.
(x)OR(y)	Performs a logical disjunction on two numbers.
PWR(x,y)	Raises constant x to the power of y.
RND	Generates a random number.
SGN(x)	Used to find the sign value of a number (-1, 0, or 1).
SIN(x)	Returns the sine of an angle.
SINH(x)	Returns the hyperbolic sine of a number.
SQR(x)	Returns the square root of a number.
TAN(x)	Returns the tangent of an angle.
TANH(x)	Returns the hyperbolic tangent of a number.
(x)XOR(y)	Performs a logical exclusion on two numbers.

#### 14.2.4.4 Order of Precedence

Anything inside parentheses ( )

Exponentiation ^

Negation (unary) -

Multiplication \*, division /

Modulo (remainder) MOD

Addition +, subtraction -

When consecutive operators have the same priority, the expression evaluates from left to right. This means that an expression such as a-b-c is evaluated as (a-b)-c.

### 14.2.5 Remote Connection

RTMC has the ability to connect to LoggerNet software running on another computer. By default RTMC connects to the computer where it is running.



To set a connection to LoggerNet on another computer bring up the server connection dialog from the Project | Change Server Connection menu. The Server IP is the network computer name or IP address. The computer name is defined by the network administrator. A connection can also be made over the Internet or local area network using the 4 number Internet Protocol (IP) address. This is a number that will have four digits between 0 and 255 separated by decimal points. An example would be 192.168.4.32. Do not put leading zeros with the numbers.

The User Name and Password are only used if you are connecting to a LoggerNet server where the network administrator has implemented security. The Port Number should always be 6789.

Clicking the Save Login Info will save the computer address, user name and password as part of the RTMC file so the screen can be run without requiring the user to know the address.

## 14.3 Run-Time

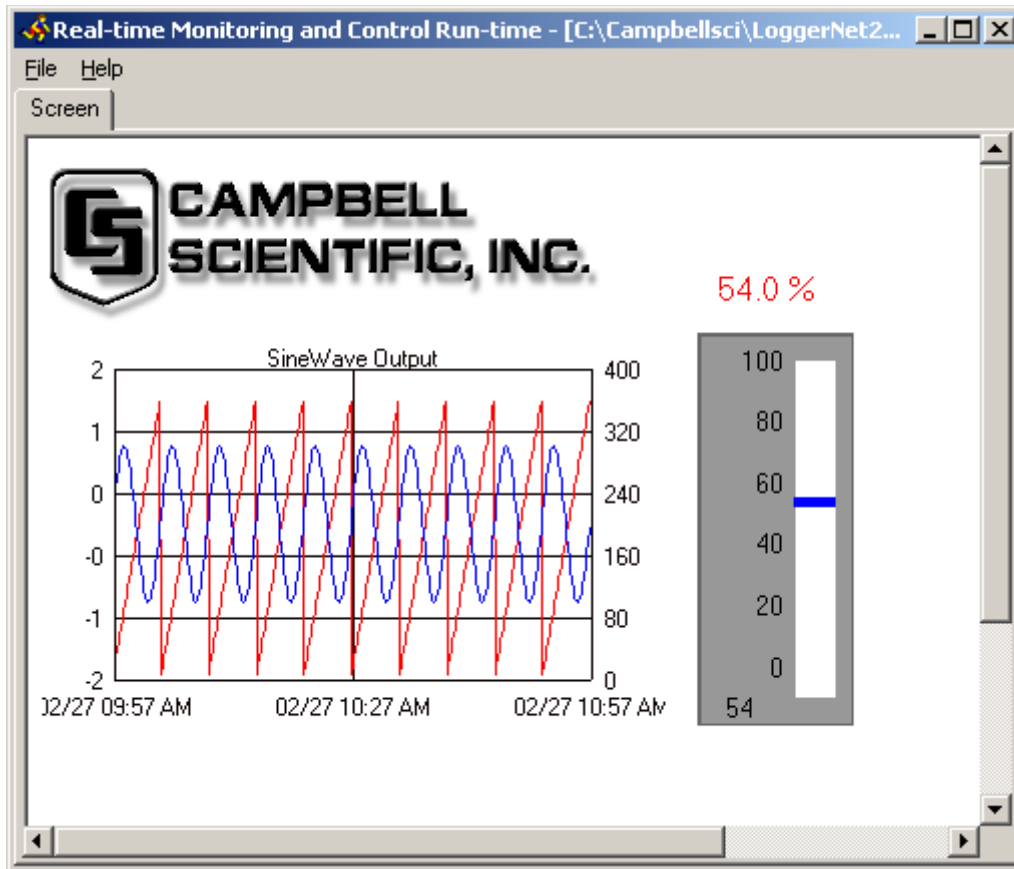


The run-time operation allows you to run the real-time graphic display screen that was created in the developer mode. From the RTMC Development window you can test the operation of the display screen using the Runtime | Save and Run Current Project menu or clicking on the Run-Time icon on the toolbar. This will close the development window and start the project window with RTMC Run-Time as shown in the window below.

When the run-time display screen is started, the display components will be highlighted in reverse video until data is received from LoggerNet. If data is not displayed, check to see that the data is being collected by LoggerNet.

Once a project file has been created, the display screen can be run without starting the development mode window. From the Windows Start Menu under Programs | LoggerNet click on RTMC Run-Time. In the Run-Time window select File | Open Project to select the RTMC project screen to run.

RTMC Run-Time will print an image of the display screen by selecting File | Print Screen.



# Section 15. Troubleshooting Guide

---

*This section is provided as an aid to solving some of the common problems that might be encountered using the LoggerNet software. This list is not comprehensive but should provide some insight and ability to correct simple errors without a call to Campbell Scientific technical support.*

*This section also includes descriptions of some of the tools such as Ping and Data Table Monitor that can be useful in troubleshooting LoggerNet problems.*

## 15.1 LoggerNet Server Problems

The following sections identify problems that have been observed with operation of the server. If you are experiencing problems with the server look through the following conditions to see if any of these match the problem you are having. If you find the problem listed, try the suggested remedies. If your problem is not listed or the remedies don't fix the problem, contact Campbell Scientific for technical assistance.

### 15.1.1 Running and Connecting to the Server

**Problem:** No data is being collected on the defined schedule. However, active connection and clock check via the Control Panel are successful.

**Remedy:** Ensure "Allow Automated Operations" is enabled on the Server Settings of the Network Administration application.

**Problem:** When trying to start a client, the Server Selection box appears repeatedly.

**Remedy:** Make sure the LoggerNet server software is running on the named computer.

**Problem:** The LoggerNet server is running but the server selection box keeps asking for user name and password.

**Remedy 1:** If security is enabled, get a user name and password from the LoggerNet administrator.

**Remedy 2:** If security is not enabled, check to make sure TCP/IP services are enabled and properly configured.

**Remedy 3:** Check to see if there are multiple TCP/IP devices set up on the computer. It is possible that the server is using one of the services and the client applications are trying to connect using another.

**Remedy 4:** Try reaching the server machine with other utilities such as Ping or Telnet. Both are available from the standard DOS

command box. See the sections below for details on using Ping or Telnet

**Problem:** Cannot start the LoggerNet server on a laptop computer.

**Remedy 1:** If the computer is sometimes used on a computer network and no network card is installed, TCP-IP services may not be starting. Try starting the computer with the network adapter card installed.

**Remedy 2:** If remedy 1 doesn't work you will need to configure a phony dialup service with a fixed IP address. There is a description of this procedure in Section 2.2.

### 15.1.2 Data Collection Issues

**Problem:** Scheduled data collection is enabled but no data is available in the LoggerNet server data cache.

**Remedy 1:** Make sure tables are included for collection and the table definitions indicator shows Current.

**Remedy 2:** Check communication state on CommStatus.

1 = normal collection

2 = primary retry

3 = secondary retry

4 - collection disabled

You may need to update the data cache to get to a normal state.

**Remedy 3:** Go to the Control Panel application and on the Retrieve tab click the Update Data Cache button. This will force collection from the datalogger and will return the datalogger to the normal collection state.

## 15.2 Client Application Problems

**Problem:** The error message "Not Connected, cannot post messages" keeps popping up and nothing on the application will work.

**Remedy:** Some problem has occurred with the connection between the client application and the server. Try to close the application by clicking on the 'X' in the upper right hand corner of the application. If the application will not close by normal means, you will need to go to the Windows Task Manager and manually end the task.

**Problem:** Cannot set the clock from the Control Panel tools tab. Other communications with the datalogger succeed.

**Remedy:** Check the Maximum Deviation Allowed on the Automated Clock Check tab for the datalogger in NetAdmin. If the difference in the datalogger and system clock is less than the maximum deviation specified, the clock will not be set.

**Problem:** Datalogger program won't compile and returns an E43 error.

**Remedy:** There is an interval mismatch between the P84 or P92 instructions and the program scan rate. If the P84 or P92 intervals are not multiples of the scan rate, the program won't compile and will return this error.

## 15.3 General Communication Link Problems

**Problem:** Communications are not solid and difficulty is experienced sending programs to the datalogger.

**Remedy 1:** If there are slow serial devices in the communication path, such as older modems, the server might be overrunning the buffers. Set the maximum packet size to a smaller number.

**Remedy 2:** On RF networks, make sure that the extra response time is sufficient for the reply to come back, especially if there are repeaters in the network.

## 15.4 RF Communication Link Issues

There are two sets of problems that can degrade RF communications. The first is using combinations of RF components that do not work well together. The second is deterioration or failure of the RF components in the system. There are also situations where the equipment is performing as it should but marginal communications are due to line-of-sight issues or other environmental factors. There are a number of ways to test the operation of an RF system. The three sections following illustrate things to look for and tests to perform to troubleshoot RF operations.

### 15.4.1 Checking RF Components and Connections

Before testing RF signal strength, there are several things that should be done to verify that the right RF components are in place.

1. Check that the RF modem has the correct switch ID set on the DIP switches and that the right PROM is installed. For table-based RF the PROM part number should be 6873 with varying versions. (This is a common problem and should be checked first.)
2. Check the type and brand of the radio. In general, the radios in a network should be the same type wherever possible.
3. Check that the radio is set for the right frequency. With a radio that can be programmed from a computer, verify that the correct frequency and other settings are set properly. If the radio is crystal based there should be an identifying label showing the frequency. If not, you will have to test the radio with a programmable scanner or frequency analyzer.
4. Check the cable connecting the radio to the RF modem. Different combinations of radios and RF modems require specific cables to make the

right connections. For questions in this area, contact the network installer or Campbell Scientific.

5. Check that the antenna is the right type (directional or omnidirectional) and is designed for the frequency being used. Most antennas will have labels identifying the frequency range. Make sure the antenna is mounted so that clear line-of-sight is provided and that directional antennas are properly oriented.

6. Make sure the antenna is the right impedance to match the system. This is almost always 50  $\Omega$ . This should match the cable connecting the antenna to the radio and the radio connection.

7. Check that the cable connecting the radio to the antenna matches the impedance of the antenna and the radio. This is almost always 50  $\Omega$ .

### 15.4.2 RF Signal Strength Testing

Once you have verified that the right equipment is in place, make sure that all of the components have power. Then you are ready to proceed with performance testing.

To test a station's radio/cable/antenna transmission capabilities, a directional wattmeter is needed such as the Bird Electronic Corporation's Model 4304A Wattmeter. Proper connectors are also needed to place the wattmeter in series between the radio and antenna cable. A voltmeter is required to measure the battery voltage of the datalogger with and without radio transmission.

---

#### NOTE

If you are using a data radio that does not have a transmit button built in, you can easily build a push to transmit button from the documentation of the radio/RF modem interface connector. There will be one pin that when pulled high or pulled low will initiate radio communication. See the radio documentation to identify this pin. Connect a momentary pushbutton to either raise or ground that pin. **Always make sure that the antenna is connected to the radio before attempting to transmit.** Serious damage to the radio can occur if transmitting without an antenna.

---

Place the wattmeter in series between the radio and antenna cable. Set the wattmeter to the 15-Watt range, or the next highest wattmeter setting, and point the directional arrow first toward the antenna cable to measure forward power ( $W_f$ ). Initiate radio communication, let the wattmeter stabilize, and record the wattmeter reading. Reverse the directional arrow so it is pointing back toward the radio, initiate radio communication, let the wattmeter stabilize, and record the wattmeter reading. This second reading is the reflected power ( $W_r$ ). Take the square root of the reflected power divided by the forward power to arrive at the square root ratio ( $R$ ). Calculate the Voltage Standing Wave Ratio (VSWR) with the following equation:

$$\text{VSWR} = [(1+R)/(1-R)]$$

$$\text{Where, } R = (W_r/W_f)^{1/2}$$



The impedance of the RF transmission cable (usually RG-8A/U) and antenna combination should match the impedance ( $50\ \Omega$ ) of the radio output circuit. When the transmission cable or antenna does not match the impedance of the output circuit of the radio, not all of the energy supplied to the cable will flow into the antenna. Some of the energy supplied will be reflected back to the radio, causing standing waves on the cable. The ratio of voltage across the line at the high voltage points to that at the low voltage points is known as the Voltage Standing Wave Ratio, or VSWR. The VSWR should be less than 1.5:1 for error-free radiotelemetry.

For example, if the forward power ( $W_f$ ) is 5 Watts and the reflected power ( $W_r$ ) is 0.2 Watts, the VSWR is 1.5:1.

A problem has been found if the VSWR is greater than 1.5:1. The VSWR will increase when:

- There is a problem with the connectors. Check for loose, corroded or damaged connectors. (Connector problems are the most common source of RF communications failures.)
- The antenna is used in proximity of metal, which is reflecting the signal back to the radio.
- Transmitting inside a building.
- The cable is worn, cut or damaged so that not all of the radio energy can travel through to the antenna.
- The antenna design frequency does not match the radio frequency.

If the VSWR is below 1.5:1, then that radio/cable/antenna link is good. However, be sure the antenna is oriented properly.

While at the station, check the voltage on the 12V port of the datalogger both with and without the radio transmitting. Regardless of the battery type, the datalogger requires a minimum of 9.6 Volts.

### 15.4.3 Troubleshooting With Attenuation Pads

This test is used to measure the signal strength of the radio signal between two radios. There are situations where the signal from one radio can be heard by the other, but the signal is not strong enough to establish communications. In general a signal strength of greater than -95 dBm must be maintained for good communications.

There are many factors than can contribute to a loss of power in an RF system.

- Line of sight may be marginal or poor. (Vegetation on trees or other obstacles.)
- Corroded connectors or connections not made properly.
- Inadequate antenna gain.

- The antenna is not properly oriented.
- Outside interference on the channel frequency.

Testing the radio transmission quality between radios requires the use of a programmable scanner and a set of attenuators or attenuation pads. You will need someone at each end of the radio link and a way to talk to each other.

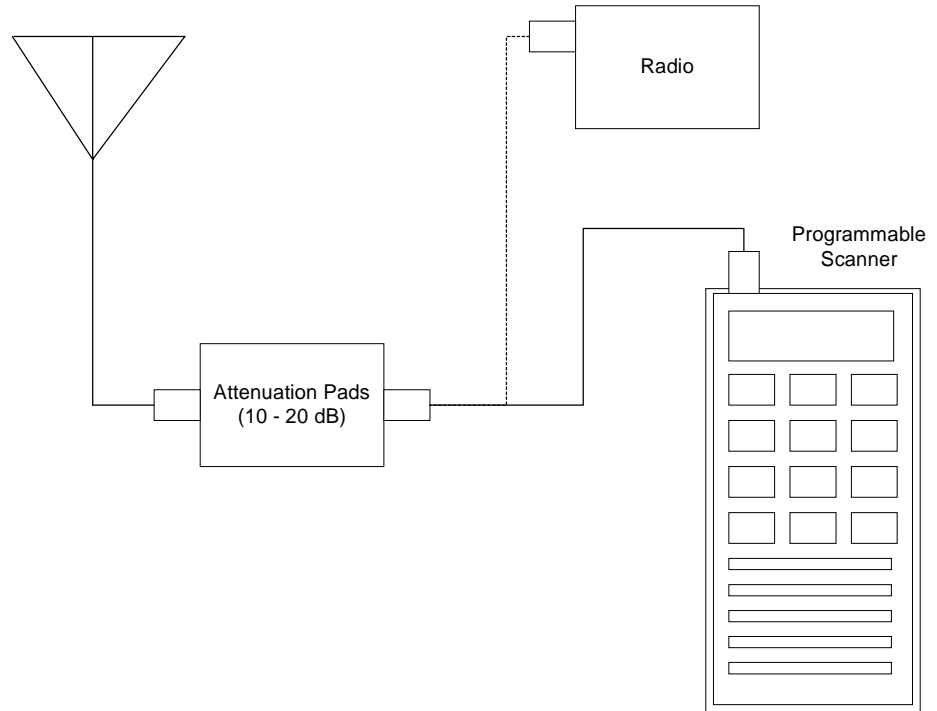
If the carrier detect light is coming on at the RF base station radio, but communication quality is poor or not being set up properly, there may be a marginal or low signal power inherent in the RF link. In this case, it is a good idea to do a signal power check with attenuation pads for each sub-link in a complete RF link. Every RF link has one or more sub-links. For example, if there is one repeater in an RF link then there is a sub-link between the base station and the repeater and a sub-link between the repeater and the field station. The sub-links should be checked in both directions of communication.

Before proceeding, it is a good idea to calculate the theoretical signal power for each of the RF links. Appendix C of Campbell Scientific's RF Telemetry manual outlines the calculations.

For proper radio communications the signal power must be greater than  $-95$  dBm at the standard transmission rate. However, squelch will break on the radios with a signal power as low as  $-115$  dBm. Therefore, there is a 20 dBm range in which the radios are not working, but may "sound" proper.

An attenuation pad inserted into the link increases the power loss of the system. If a 20 dB attenuation pad (or two 10dB pads in series) is inserted into the link and subsequently the radio will not break squelch, the signal power is between  $-95$  and  $-115$  dBm which is below the power limit for good data transmission.

Similarly, if a 10 dBm attenuation pad is inserted in the link and the radio subsequently will not break squelch, the actual signal power is between  $-105$  and  $-115$  dBm. In this case, the signal power is far below the power limit.



To test the power being received by a radio over an RF link, disconnect the radio from the antenna and insert the programmable scanner as shown in the figure above. Program the scanner to the radio frequency and adjust the squelch control until it is just about the ambient RF noise. This level will normally be around -110 to -115 dBm. The scanner is now ready to conduct the test.

#### NOTE

If you are using a data radio that does not have a transmit button built in, you can easily build a push to transmit button from the documentation of the radio/RF modem interface connector. Campbell Scientific offers a Push-To-Talk button that can be used for this purpose. **Always make sure that the antenna is connected to the radio before attempting to transmit.** Serious damage to the radio can occur if transmitting without an antenna.

First, test the sub-link of the base station to the first repeater or field station. Initially treat the base station as the transmitting station and the first field or repeater station as the receiving station. Disconnect the radio's multicolored cable from the RF modem. To start the test have the person at the base station initiate a radio transmission. When the radio transmission is received, if squelch is broken, you will hear it on the speaker of the scanner. If you don't hear the radio transmission, the signal is getting lost in the ambient noise and will not be picked up. If squelch is not broken, then either the signal power is less than -115 dBm, or something is wrong with the power supply, antenna orientation, or cable connections. If squelch is broken on the receiving radio, the site can be tested with the attenuation pads to determine the approximate signal power if it is between -115 and -95 dBm.

Insert the attenuation pad(s) (20 dB) between the scanner and antenna of the receiving station ONLY (most attenuation pads have a limited current capacity). Initiate radio transmission from the base station transceiver. If squelch is broken at the receiving station, this sub-link is good in this direction. If squelch is not broken this sub-link has signal power between -95 and -115 dBm which should be corrected. Corrections can involve shortening the distance between radios, reorienting antennas, fixing connectors or cables, providing a better power supply, or shortening coaxial cable lengths.

If it did not break squelch with the 20 dBm attenuation pad, it is possible to decrease the attenuation to 10 dBm to determine if signal power is between -95 and -105 dBm, or between -105 and -115 dBm. This will identify if the signal power is close to or far away from -95 dBm.

If it did break squelch with the 20 dBm attenuation pad, then that sub-link is good in that direction. The next sub-link can now be tested. Remember to place the attenuation pads at the receiving station only! If all of the sub-links were good, the same sub-links can be tested in the opposite direction. If reversing directions in a sub-link gives bad results while the other direction is good, be suspicious of the transmitting radio in the bad direction and the radio's power supply.

## 15.5 Ping

Ping is a utility often used by computer network administrators to check whether a computer is accessible over the network. This utility can be used to verify that the computer where the LoggerNet server is running is available over the network for communications. If you can't get a response from the computer with Ping you will not be able to use any of the client applications to connect to the server.

To use Ping, bring up a Command or DOS window. You should see a prompt that looks like: C:\>. At the prompt type in the word "ping" followed by the name or numerical IP address of the computer you are trying to reach. For example if the computer name is Thor you would type in: *ping Thor*. If you have the IP address you can type in: *ping 197.144.3.120*.

If the command succeeds you will see four lines indicating the amount of time the computer took to respond to the request.

```
C:\>ping 127.0.0.1
```

Pinging 127.0.0.1 with 32 bytes of data:

```
Reply from 127.0.0.1: bytes=32 time<10ms TTL=128
Reply from 127.0.0.1: bytes=32 time<10ms TTL=128
Reply from 127.0.0.1: bytes=32 time<10ms TTL=128
Reply from 127.0.0.1: bytes=32 time<10ms TTL=128
```

If the command failed you will see four lines indicating that the request timed out or that the computer could not be reached. No response was received.

```
C:\>ping 192.145.23.2
```

Pinging 192.145.23.2 with 32 bytes of data:

```
Reply from 216.126.193.69: Destination host unreachable.  
Reply from 216.126.193.69: Destination host unreachable.  
Reply from 216.126.193.69: Destination host unreachable.  
Reply from 216.126.193.69: Destination host unreachable.
```

```
C:\>ping 192.168.4.34
```

Pinging 192.168.4.34 with 32 bytes of data:

```
Request timed out.  
Request timed out.  
Request timed out.  
Request timed out.
```

Ping can also be used to see if TCP/IP communications are available on your own computer by typing: *ping localhost* or *ping 127.0.0.1*. This will talk to the local computer. If Ping does not work you will not be able to run clients on the same machine where the LoggerNet server is running.

## 15.6 Telnet

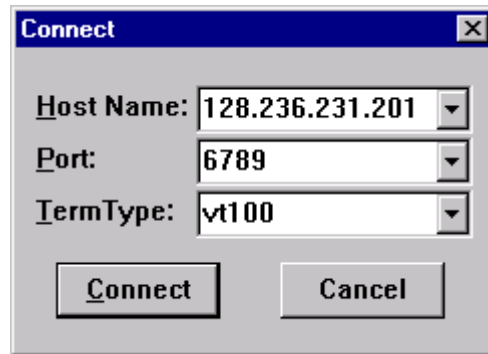
Telnet is an application provided as part of the computer operating system to allow terminal connections to other computers. With LoggerNet Telnet can be used as a second step to troubleshoot problems with connecting client applications to the LoggerNet server. If Ping (described above) will communicate with the computer and Telnet will not, there is a configuration problem with the TCP/IP setup. The computer network administrator can help sort out the settings.

If Telnet and Ping are both able to communicate with the LoggerNet server, there is a problem with the client application or security is set on the server.

To run Telnet go to Start -> Programs -> Accessories -> Telnet

If Telnet does not appear you will have to go back to Windows set-up and install the communications options.

When the Telnet window comes up click on the Connect | Remote System menu item. This will bring up a dialog box similar to that shown below.



**Host Name** - enter the name or IP address of the computer where the LoggerNet server is running.

**Port** - enter the number 6789 which is the port ID for connections to LoggerNet.

**Term Type** - leave this entry set to VT100.

If the connection fails, Telnet will display an error message box with the reason for the failure. If the connection succeeds, the name of the host computer will appear on the title bar of the Telnet window.

## 15.7 Using Data Table Monitor

Data Table Monitor is a utility that was created to retrieve data from the LoggerNet server data cache and display it on the screen. It also has the option to export it to a file. Once the utility has been started, as new records are collected to the server, the new records will be displayed and sent to the file.

The most important use of Data Table Monitor is to see what records are being stored in the data cache and to diagnose suspected data cache problems.

Data Table Monitor gets all the data available from the data cache that matches the export conditions. As the server collects new records from the datalogger, they are automatically displayed and sent to the data file. This continues until Data Table Monitor is closed or data export is stopped.

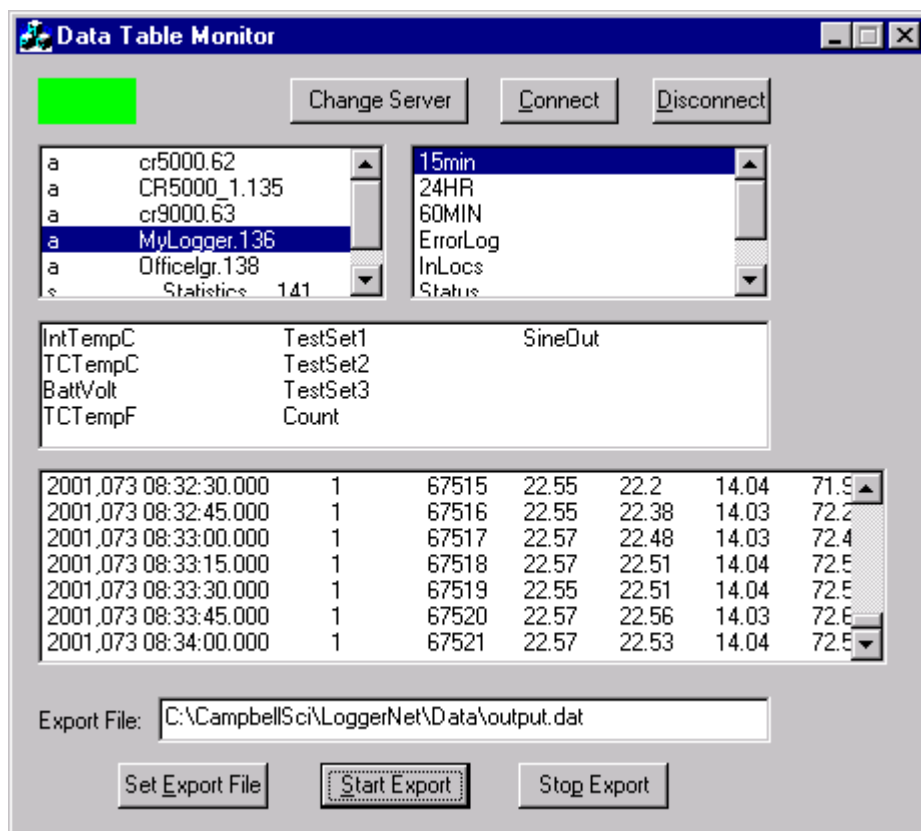
---

### CAUTION

One caution about the data file created by Data Table Monitor—there are no limits to size or longevity. If you plan to use the export to file feature on a regular basis, make sure to either restart Data Table Monitor (which overwrites the exported file) or delete the files periodically. The data export can easily be restarted by clicking on the Start Export button. This will delete the old file and start a new one.

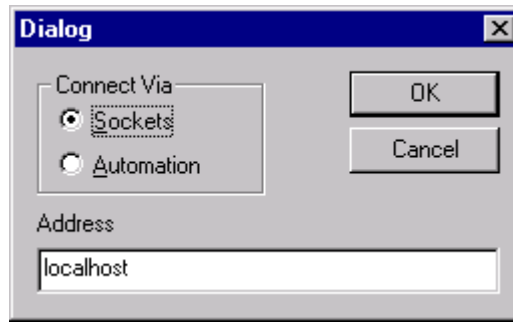
---

To start Data Table Monitor go to Start -> Programs -> LoggerNet -> Tools -> Data Table Monitor. The utility will start with a screen similar to the one shown below. The red rectangle in the upper left corner indicates that it is not connected to a LoggerNet server.



To set up Data Table Monitor, select the name of the computer where the LoggerNet server is running. Do this by clicking on the Change Server button. When the dialog comes up, make sure that Connect Via Sockets is selected. For a host computer address, type in the name or the IP address of the computer where the LoggerNet server is running. If you are running the Data Table Monitor application on the same computer as the LoggerNet server you can type in "localhost" without the quotes. When you have entered the name of the computer click on OK.

You now need to establish a connection with the communication server by clicking on the Connect button. When a connection is established the red box in the upper left corner will turn green and the names of the dataloggers attached to the LoggerNet server are listed.



Selecting a datalogger will list the names of the data tables in the datalogger. Note that if collection from a specific table has not been enabled in the NetAdmin client, no data will be coming into the data cache. Data Table Monitor can only display and output data from the data cache.

In the list of tables, selecting a table name will present a list of the data values that are in the table. You can select as many of these as you want by clicking on them to select the desired values. Clicking on a selected value will deselect it.

Once you have selected the data values to export, you can specify the datafile for export by clicking on the Set Export File button. This will bring up the Windows File | Open dialog. Choose a filename and location for the file to be created. (This file should be located on the same machine where Data Table Monitor is running. There have been problems writing to some network drives.)

Click on the Start Export button to bring up the choices for which records to export.



Set up the Data Advise Start Properties dialog as follows.

**Ordering Option:**

- 1 - Collection Order: displays and writes the data to the file in the order it was collected by the server. This setting is useful to look at the actual data record storage in the data cache. Data collected as holes will appear as out of order records. Do not use for output needing sequential records.
- 2 - Logged Order With Holes: The output will include only complete data sequences. If the Data Table Monitor comes to a hole that has not yet been filled, it will wait for the hole to fill before displaying or writing the next record to the file.
- 3 - Logged Order Without Holes: The data output will be displayed and written to file as quickly as it is collected, without waiting for holes to be filled. Any data in holes will be skipped in the output.
- 4 - Real Time: the most recent data is always sent out starting with the last record stored. This will not provide a complete data set.

**Start Option:** This selects the starting point for the data to be output to the file.

- 1 - RecordNo/FileMarkNo: This option allows a selection of starting position based on the file mark and record number. An entry of 0 in both fields will get all of the data in the data cache.
- 2 - At or After Date: This option allows a selection of the starting position based on the timestamp in the data. The time and date are set in the Beginning Date field at the bottom of the dialog. All of the records available after this timestamp are output.

- 3 - At Newest: This option will set the starting position to the last record stored in the data cache. This last record and any future records stored will be output.
- 4 - After Newest: This option will set the starting position to be the next record stored in the data cache. Output begins with the next record stored in the data cache. No historical records will be output.
- 5 - Relative to Newest Stamp: This option starts from the most recent record collected. The Starting Position specifies how many records back from the write index should be collected. For example, a starting position of 10 will get the 10 most recent records.

The Starting Position, RecNo, FileMarkNo, and Beginning Date edit boxes are used only with the corresponding start options above.

Once the start options have been set, click on the OK button to start writing to the file. The records are always displayed in the list box on the bottom of the screen. If you have set up an output file they are also sent to the output file.

# Section 16. Implementing Advanced Communications Links

---

*This section describes the configuration and operation of a variety of communications links. The communications links included here require special setup or configuration, or require special consideration in the implementation to work properly.*

---

**NOTE**

Refer to Section 5.1 if you need general information on adding devices to the device map.

---

## 16.1 Table-based Dataloggers via RF

Communications over RF with table-based dataloggers works differently than RF with array based dataloggers. Table-based datalogger communications are based on messages that are sent from the LoggerNet server to the datalogger. The datalogger performs the requested action and responds with a message. In RF networks the LoggerNet server never establishes a direct connection with the datalogger but relies on the RF base to route the message through the RF modems to the datalogger. The return message is routed by the RF modems to the RF base where the message is kept until the server next talks to the RF base.

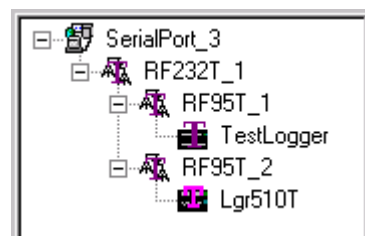
The RF base has the ability to handle several messages at a time. Once a message has been sent to the datalogger, the RF base goes to the next message in line and sends that one. This leads to greater efficiency in working with large networks because several operations can be in process at the same time.

### 16.1.1 Setup

Setting up a network of table-based dataloggers using RF communications requires some special consideration, and careful attention must be paid to the configuration settings used for data collection.

In a table-based RF system, the RF base handles all message communications to the datalogger and stores the collected data and message responses until the computer requests them.

The device map for a simple RF network would look similar to the following network map.



The following sections identify configurations and settings that are either unique to RF network implementation, or require different considerations when used with an RF network.

### **Network map**

The RF network map is set up in NetAdmin. When communications are enabled with the RF base, the network map is sent to the RF base containing all the information for the RF network, including the switch IDs of the remote RF modems and the layout of repeaters. The interval at which the remotes will be polled for data collection by Data Advise is also included as part of this configuration.

### **Operation of Repeaters**

Repeaters are used to extend the range of the RF network where line of sight or distance prevents a direct RF link. Each RF modem used as a repeater contains the portion of the network map for all of the RF modems it communicates with. Be sure that the switch ID set on the network map in NetAdmin matches the switch ID set in the RF modem hardware.

### **Extra Response Time**

The Extra Response time settings should only be used when exceptional delays are encountered in RF communications. Under normal situations with good RF links the default settings should be adequate. If communications failures occur in the system due to operations timing out, these times can be extended to allow extra time for a response.

### **Max Time on Line**

The Max Time On Line setting has little practical application in RF networks since the only connection maintained by the server is with the RF base. Since an RF connection must have a dedicated serial port, there is no reason to limit the time it can use the port. In practice this setting doesn't affect RF operations since exceeding the time would just cause the server to disconnect momentarily and then reconnect immediately. The only situation where the Max Time On Line might be used is with a Phone to RF system.

### **RF Poll Interval**

The RF Poll Interval is used with Data Advise data collection to determine how often the RF base will contact all of the dataloggers in the network for new data. This setting only affects data collection on those dataloggers that have Data Advise enabled. Standard data collection is not affected by this setting.

### **RF Base Real-time Clock**

The RF base has its own real time clock. If there is a difference between the RF base clock and the server clock you may notice delays between the Data Advise RF poll and the time the computer gets the data from the base. The clock in the RF base can be set using the Cora Script application (see Appendix F). Automated clock sets can be set up for the RF base in NetAdmin.

## 16.1.2 Data Collection Options

There are two types of data collection available for table-data dataloggers such as the CR10X-TD. Standard data collection is under control of the LoggerNet server and the collection schedule set up in NetAdmin for the datalogger. Data Advise data collection is handled by the RF base and uses the RF Poll Interval to determine how often data is collected. Each type of data collection has advantages and drawbacks. The choice of data collection mode depends on the data needs and size of the network.

Datalogger network administrators should carefully consider the following sections to determine which type of data collection best fits their needs.

### 16.1.2.1 Standard Data Collection

Data collection uses a message from the LoggerNet server to collect data from each datalogger. The timing of these data collection messages is set by the scheduled collection configuration set up in NetAdmin for the datalogger. To use standard collection, the Collection Via Data Advise check box on the Data Collection tab for the datalogger in NetAdmin must be disabled. The RF poll interval on the Scheduled Collection tab in NetAdmin for the RF base has no effect on this type of data collection. For more details on standard data collection see Section B.4.

In the RF system, the data collection messages are routed by the RF base through the RF modems to the datalogger. The datalogger sends a message response containing the data back through the RF network. This response message with data is stored in the RF base until contacted by the LoggerNet server.

Standard data collection gets data from the datalogger starting with the oldest uncollected records. Real-time data is available as long data collection has caught up with the current data in the datalogger. If communications are lost for a time, real-time data will not be available until data collection catches up with current data.

There is no constraint on the maximum collection interval. The minimum collection interval should be set such that there is enough time to do all of the data collection from the dataloggers in the network and allow time for the manual operations as described in the section on operational considerations below.

Standard data collection should be considered for RF networks in the following situations.

- Real-time data is not as important as a complete set of data records.
- Data does not need to be collected more often than every 20 minutes.
- The number of dataloggers served by one RF base is relatively small.

### 16.1.2.2 Data Advise Collection Mode

The table-based RF system was optimized for efficient data collection using Data Advise operations. These optimizations allow the RF base to handle data collection from large networks of dataloggers while polling quickly enough to bring in real-time data. In LoggerNet, Data Advise is enabled on the Data Collection tab for each datalogger using the NetAdmin application. For details on the operation of Data Advise data collection see Section B.4.

When Data Advise data collection is used with an RF network, there are some differences from the standard operation. When the RF network is configured in NetAdmin and communication is enabled, a map of the devices in the RF network is sent to the RF base. This map includes the RF modem switch IDs and datalogger station IDs for the network served by the RF base. The network map also indicates which dataloggers have Collect Via Data Advise enabled. Once the network map has been loaded into the RF base the base will start polling the dataloggers using Data Advise data collection at the interval specified by the RF Poll Interval on the Scheduled Collection tab for the RF base in NetAdmin.

When there are repeaters in the RF network, each repeater polls for data from the RF modems in its sub-network and stores the data until the RF base polls for the data. In this way the RF base polls for data from all of the first level RF modems without concern for the repeater sub-networks.

Once the RF network map has been loaded into the RF base by LoggerNet and the Data Advise operations have been started, polling of the dataloggers in the network will continue without intervention by the communication server. This can present a problem if you are trying to disable the RF network. If you disable communications with the RF base, the server will attempt to send an empty network map to the RF base to suspend the polling. If for some reason that operation is not successful, the only way to disable the RF polling is to disable communications with each of the remote RF modems (RF95T) and then disable the base RF modem communications.

Data Advise data collection should be considered for the following situations.

- The most current data available needs to be brought in first.
- There are a large number of dataloggers in the network served by one RF base so that data needs to be collected as efficiently as possible.

#### 16.1.2.2.1 RF Poll Interval for Data Advise Data Collection

When Data Advise data collection is used, the RF poll interval is the fundamental time factor for the RF network. This number, which is set in NetAdmin for the RF base, actually sets two intervals. This is the interval used by the RF base to poll the RF modems in the network for data and this is also the interval at which the server computer checks the RF base for data. If the RF base ever detects that its internal data buffer is getting larger without data being gathered by the computer, it will suspend polling until communications are restored with the server computer.

Data Advise data collection will never occur at a rate faster than the RF poll interval. Some thought should go into the setup of the RF poll interval including the following considerations:

- How often the data is needed.
- How much data is being collected with each poll.
- The number of stations in the network.

Table-based RF networks have a maximum polling interval of 20 minutes. The poll interval should be set to minimize the amount of data for each poll while allowing time to contact all of the stations.

Offsets can be set for both the RF polling interval and the computer interval. The RF polling interval refers to how often the RF base checks the stations in the network for new data. The computer interval is always set to be the same as the RF polling interval, and refers to how often the computer contacts the RF base for stored data. If the dataloggers are saving data on 5-minute intervals, the RF poll could be set up at 5 minutes with an offset of 30 seconds to always get the most recent data available. If the computer interval is then given an offset of 2 minutes, depending on the time required to poll all of the dataloggers, the computer also gets the most recent data.

Another factor that should be considered is that some operations such as sending a datalogger program, hole collection, clock check/set, and manual data polling, are all done in the time left over after the scheduled collection is done. There can be problems if the collection schedule is very tight and there is a large hole to be collected.

Summary: As a general rule try to minimize the poll interval only to the point that it can collect the current data from all of the stations in about half of the time between polls.

#### **16.1.2.2.2 Scheduled Data Collection Interval with Data Advise**

In a table-based RF system using Data Advise data collection, the server does not have control over data collection. The Scheduled Collection interval only determines the amount of time the LoggerNet server will wait for data to come in. The timeout period is determined by multiplying the collection interval by three. For example if the collection interval is set at 5 minutes then the server will wait 15 minutes for data to come in. This is significant because, depending on RF communications, data may not be collected every poll. This means that if the collection interval is set to be the same as the poll interval, there is a risk of the server timing out and discarding the data when it comes. This leads to recollection of the data as a hole.

The *recommendation* is to set the scheduled collection interval to be longer than the poll interval. Setting a longer scheduled collection interval does not affect the rate of data collection since the RF modem polls the datalogger at the RF poll rate, and gets any records stored by the datalogger regardless of the collection interval setting.

#### 16.1.2.2.3 Primary and Secondary Retry Intervals

The primary and secondary retry intervals are used when communications fail with the datalogger or some part of the communications path. These have the same effect in RF communications as the scheduled collection interval in setting the time the server will wait for data. Therefore, these intervals should never be set to a value less than the interval used for scheduled collection. If there are problems with the network communications being delayed it may be helpful to set these times longer than the collection interval.

### 16.1.3 Operational Considerations

The following sections address the function and implications of user or datalogger network administrator operations. These manual operations are not typically part of the normal automated data collection but can impact data collection.

Manual operations refer to functions other than the scheduled data collection. These operations occur independently of the RF poll interval and generally are messages meant to be passed directly to the datalogger through the RF system.

Because the server computer cannot determine when the response to these manual operations will come, while a manual operation is active, the server will continually exchange handshaking messages with the RF base about 5 times a second until the response is received or the operation times out. This causes a great deal of serial communications traffic between the computer and the RF base while the manual operations are pending.

---

**NOTE**

The `bmp1LowLevelDelay` (device setting #53) on the RF base can be used to control the frequency of the handshaking messages. This sets the number of milliseconds delay between messages. Using this setting can greatly reduce the serial line traffic between the RF base and computer. This setting is only available through Cora Script (Appendix F)

---

Manual operations are also significant because they must share resources not used by scheduled data collection. In systems where there is a lot of data or the data collection doesn't leave time for other messages, these manual operations can be delayed or cause delays in the scheduled data collection.

#### 16.1.3.1 Check/Set Clock

The most basic manual operation is checking or setting the datalogger clock. This message is passed to the datalogger, and it responds with either the current datalogger time or a confirmation message that the clock was set. This operation normally takes 5-15 seconds, depending on the number of repeaters in the RF path.

#### 16.1.3.2 Update Data Cache

The Update Data Cache operation is also known as a *Manual Poll* and is similar to standard data collection. For manual polling to work, if a Data



Advise operation is active, it must be cancelled, so the server will stop the Data Advise before starting the manual poll. The manual poll will first collect all of the pending holes in the data. Then it will continue collecting data until all of the data from the datalogger has been collected. After the manual poll has finished, if Data Advise was active before, the server will restart it.

One consideration with the Update Data Cache is that real time or current data is not being collected from that station while the operation is active. If there are a lot of holes or a lot of data to collect it can take a long time (minutes or even hours) for this operation to complete. The actual amount of time to complete the Update Data Cache depends on the RF communications quality, how many stations are in the network, the data poll interval, and how much data is in the datalogger to collect.

### **16.1.3.3 Hole Collection**

Hole collection is only done in connection with the Data Advise data collection. (There is no hole collection for standard data collection, because retrieving the oldest uncollected data is always retrieved.)

Hole collection is done by the same mechanism used for the manual poll, except it does not require that Data Advise be stopped for it to work. A hole is a set of missing records that did not get collected into the server data cache. The Data Advise operation is set up to always send the most recent data. If there have been communications problems or for some other reason all of the data was not collected, when the newer records are collected the server identifies the missing data as a hole to be collected.

One consideration with hole collection is that hole data can only be collected from one station at a time and only in blocks of up to 2048 bytes. This means that if there are a lot of holes or very large holes the system may be doing hole collection for a long time.

With the increased communication traffic, marginal RF links will show up as very long periods to collect data.

Using CoraScript there are provisions to manage the holes that have been identified, including removing them from the collection list or clearing all of the pending holes. See the documentation for CoraScript (Appendix F) for more information.

If the primary need for the data from the RF network is for real time monitoring and a complete archive of the data is not needed, hole collection should be turned off.

### **16.1.3.4 Send Program**

The Send Program manual operation puts a datalogger program into the datalogger. This process can take significant time with large programs. Sending large programs over the RF network is very susceptible to noise and may require several attempts. In some environments it may not be possible to send large programs so these will need to be downloaded directly at the site using a laptop computer directly connected to the station.

### 16.1.3.5 Connection Management Operation

Connection Management is used to keep communications active with a specified device so that re-establishing the connection is not needed for every operation. The primary use is with phone modems to prevent hanging up and redialing. In most situations with an RF network this operation is not useful because it does not have the ability to establish a connection to the station. The only connection maintained is the communication between the computer and the RF base.

### 16.1.3.6 Datalogger Program Considerations

When setting up an RF network there are some special considerations for the program running in the datalogger.

#### Quantity of Data

Each data packet from the datalogger through to the computer has to be less than 2048 bytes of data. If the amount of data to be collected with each RF poll interval is greater than the maximum packet size, the RF base collects that data in a separate polling cycle after the normal poll has completed. The greatest efficiency in data transfer will be when the amount of data collected per RF poll interval is less than 2K bytes. One way to reduce the amount of data per poll interval is to speed up the poll rate.

#### Interval Based Tables

Interval based tables are the standard method for data storage and are best handled by scheduled data collection. The primary consideration is to make sure that the amount of data being stored by the datalogger is consistent with the RF data collection rate so that the data can be passed through the system. This is very important in an RF network that is using repeaters to funnel data from multiple dataloggers.

#### Use of Final Storage vs. Inlocs

The amount of data being collected can be reduced by storing desired input locations to final storage tables instead of bringing back all the input location. Some of the dataloggers default to a minimum of 28 input locations no matter how few are used in the datalogger program. Datalogger programs created in Edlog will have blank input locations added up to 28. Input locations are also often used to store temporary or intermediate values used in other calculations and these values may not be of interest. All of the input locations -including blank locations- are collected every time the Inlocs table is collected. This causes unnecessary traffic and can delay other operations.

Another consideration with collecting the Inlocs table is that an input location record is generated by the datalogger based on how it is contacted. Because of this the amount of data coming in is not very predictable. This may occur as often as two or three times per Data Advise poll or as infrequently as every other poll.

The general *recommendation* is that, unless there is a need to set data values or see the ports and flags provided by the Inlocs table, all the data requested

during RF data collection should be in final storage tables. The functionality of the Inlocs table can be simulated by a small, fast, final storage table. This could be done with a P84 instruction set to store data every program execution in a table that is one or two records long, and which samples the input locations of interest.

The fast final storage table allows the option of selecting only those values which need to be available. This helps reduce the amount of data being transferred over the RF collection system.

#### **Avoid Program Overruns**

A program overrun occurs when the instructions in the datalogger program take longer to execute than the specified program execution rate. This is also referred to as a “skipped scan” and means that the interval between executions of the program may not be as the programmer desired. Program overruns have another side effect that is amplified when using an RF network for data collection. When the datalogger is busy with program execution, the external communications are impaired causing delays or missed data when the RF modem is trying to get it. An occasional overrun won’t cause a problem, but consistent overruns or a program that always waits for the completion of one or two instructions that take longer than the data collection interval may cause data collection problems.

### **16.1.4 Special Considerations**

#### **16.1.4.1 Status Table**

The Status table is a special purpose table that is always present in the datalogger. The Status table on CR10T loggers displays the program signature every time a status record is created, but this must be calculated by the logger each time it is requested. This takes about 4 seconds and slows down the execution of the datalogger program. Therefore on CR10T loggers the Status table should only be requested as needed and not part of the scheduled data collection. This affects only the CR10T. The other dataloggers such as the CR10X-TD keep the value of the program signature stored in memory and only recalculate periodically.

### **16.1.5 Error Messages**

There are a number of error messages which appear in the server log files that can help determine how the RF communications are operating. To view these logs use the Communication Status Monitor application. Under the View menu, choose Server Logs. Enable the transaction log and resize the window for better viewing. These logs can be saved to the computer hard drive either by clicking the check box on the view window or in the NetAdmin application choosing Server Settings under the Options menu.

#### **16.1.5.1 RF Broadcast Failed and RF Poll Failure**

When the RF base does its scheduled poll each of the RF modems in the network is supposed to respond. If for some reason the RF modem does not

respond the base notes a warning which is sent to the server computer. The base will attempt the poll four times. If unsuccessful in getting a response from the RF modem in four attempts the base sends an “RF Poll failure” message to the server. These messages indicate a communications problem between the RF base and the RF modems.

An example of how these messages would appear in the log is included below.

```
"2000-04-10 22:39:00.440","Logger1","40","Datalogger  
message","W"," RF Broadcast failure."
```

```
"2000-04-10 22:39:25.360","Logger1","40","Datalogger  
message","W"," RF Broadcast failure."
```

```
"2000-04-10 22:39:48.210","Logger1","40","Datalogger  
message","W"," RF Broadcast failure."
```

```
"2000-04-10 22:40:55.400","Logger1","40","Datalogger  
message","F"," RF Poll failure."
```

### 16.1.5.2 RF Device Failure (routing error)

This failure is identified in a message from the RF base and shown as an “RF95T packet delivery failed” message with a failure type of “2”. This means the RF base is unable to send the message to the intended device. When the server gets this message it is interpreted to mean the RF base has lost the network map and the server sends the network map to the RF base again. An example of this sequence of messages is shown below, including the error message and the messages showing the update of the RF base network map.

```
"2000-04-10 07:49:31.100","RF232T_1","68","RF95T Packet delivery  
failed","2"
```

```
"2000-04-10 07:49:31.140","RF232T_1","37","Updating BMP1 network  
description"
```

```
"2000-04-10 07:49:57.240","RF232T_1","38","BMP1 network  
description update complete"
```

### 16.1.5.3 RF Packet Delivery Failure “4”

The Packet delivery failure type “4” is an indication that the message intended for the station was not received. The basis for this message is the lack of a response from the station within a timeout period in the RF base.

An example of this type of message is shown. In the logs this message will normally be followed by a message from the server indicating the type of transaction that failed.

```
"2000-04-10 22:38:00.350","RF232T_1","68","RF95T Packet  
delivery failed","4"
```

## 16.1.6 RF Communications Test

The RF communications links can be tested using the RF communications test available under the Options menu of the LoggerNet NetAdmin application. Either select an RF modem from the network map shown or enter the switch

IDs for the RF modems in the path to test in the (Repeater) Switch setting ID field. Switch IDs must be numbers separated by spaces. For details on the RF communications test see Section 5.6.2.

The RF communications test is useful because it tests only the connection to the RF modem without involving the datalogger.

If the communications test does not succeed or the results of the test indicate a problem in the system you will need to do more extensive evaluation of the RF communications link. See section 13.4 for details on RF communications link testing.

## 16.2 Phone to RF

Phone to RF is used in situations where the RF network is far away from where the LoggerNet server computer is located and phone access is available to the RF base site. Before implementing this type of network, consideration needs to be given to the polling intervals and the communication time required between the computer and the RF base. With a table-based RF network using Data Advise collection, polling must be done at least every 20 minutes. The computer polls the RF base at the same rate so there will be at least one phone call every 20 minutes. The duration of the call depends on the amount of data buffered in the RF base.

Standard data collection is another option. The LoggerNet server will make a call each time that it does data collection for a station. It will stay on-line until a response is received - either the data, or an error indicating that the data collection failed. Standard data collection would work for networks where the desired collection interval is longer than 20 minutes and there are a small number of stations in the RF network.

The server will also initiate a call for any datalogger operations such as clock check/set, hole collection, get table definitions or program send. The server maintains communication with the RF base until a response is received from the datalogger or the message fails. Depending on the network and the operating conditions, the server may be maintaining a continuous connection to the RF base.

### 16.2.1 Setup

The device map set up in the Network Administration client for a Phone to RF link would look similar to the communications network below.



To begin, add a Serial Port to the device map if one does not exist. Add a Phone Modem to the Serial Port. To this Phone Modem, add a Remote Phone Modem. Next, add an RF base modem and then an RF remote modem. To complete the network, add your datalogger to the remote RF modem. Review all of the settings for each device, and make any changes to customize the settings for your network configuration.

## 16.2.2 Operational Considerations

### 16.2.2.1 Extra Response Time

LoggerNet is pre-programmed to expect certain response times from devices depending on the type of intermediate devices and the communication rates chosen. If the datalogger network communications link is marginal, it may take longer to negotiate communication between the devices. Extra response time added to one or more of the devices may help to prevent the software from timing out. Note that the extra response times added for each device are cumulative for the entire communications link, and the total response time includes the default times plus any extra response time that has been added.

#### 16.2.2.2 RF Address

The hardware settings for the address of the RF base must be 255 for phone to RF operation. Additionally, ensure that the addresses set for the RF remote hardware match the settings defined in NetAdmin.

## 16.3 Phone to MD9

### 16.3.1 Setup

The device map for a phone to MD9 link would look similar to the communications network below.



To begin, add a Serial Port to the device map if one does not exist. Add a Phone Modem to the Serial Port. To this Phone Modem, add a Remote Phone Modem. Next, add an MD9 base and then a remote MD9. To complete the network, add your datalogger to the remote MD9. Review all of the settings for each device, and make any changes to customize the settings for your network configuration.

## 16.3.2 Operational Considerations

### 16.3.2.1 MD9 Addresses

The address for an MD9 base device is set in the LoggerNet communications software at 255. The hardware configuration in the MD9 base modem must match for successful communications (refer to your MD9 users manual for information on setting the hardware switches within the device). In addition, the address specified in NetAdmin for the MD9 remote modem must match its hardware configuration.

### 16.3.2.2 Extra Response Time

LoggerNet is pre-programmed to expect certain response times from devices depending on the type of intermediate devices and the communication rates chosen. If the datalogger network communications link is marginal, it may take longer to negotiate communication between the devices. Extra response time added to one or more of the devices may help to prevent the software from timing out. Note that the extra response times added for each device are cumulative for the entire communications link, and the total response time includes the default times plus any extra response time that has been added.

### 16.3.2.3 Grounding

Depending on the configuration and distance of the MD9 network, be sure to follow the grounding guidelines provided in the MD9 hardware manual. Grounding issues have been known to prevent reliable communications and data collection.

## 16.4 TCP/IP to RF

The development of Serial Server devices that allow serial communications devices to be connected to TCP/IP networks now allows an RF network to be connected to the LoggerNet server over the Internet or across a Local Area

Network. A Serial Server has a standard TCP/IP connection on one side and one or more serial ports, typically RS232, on the other. This type of network setup is typically used for organizations that have field offices or stations that are connected together by a TCP/IP network. This allows the LoggerNet server computer to be located in a central area for administration while providing communications to remote RF networks.

### 16.4.1 Setup

The device map set up in the Network Administration application for a TCP/IP to RF link would look similar to the communications network below.



To begin, add an Internet Com Port to the device map if one does not exist. Add an RF base modem to the Internet Com port, and to this, add the remote RF modem. To complete the network, add your datalogger to the remote RF modem. Review all of the settings for each device, and make any changes to customize the settings for your network configuration.

### 16.4.2 Operational Considerations

There are several settings that should be configured to optimize the TCP/IP to RF network.

The **Low Level Packet Delay** is configured using CoraScript (see Appendix F). This is setting number 53 for the RF Base. This governs how rapidly handshaking packets are exchanged by the server and the RF base while a datalogger transaction is pending. By default there is no delay so these packets pass back and forth about 5 or 6 times a second. For TCP/IP communications this should be slowed down by setting the number of milliseconds to wait to at least 1000 (1 second delay).

**Max Time online** – This should be set to zero (disabled) for all of the devices in the RF network. Otherwise, when the communications link is dropped because this value is exceeded, communication will be re-attempted immediately. Forcing the connection offline and back on quickly causes errors because not enough time is allowed for the serial server to reset the TCP/IP socket.

**Connection Management** – If the RF network is being polled rapidly and there are a large number of “Serial Sync byte not found” messages in the communications log, it may be necessary to use Connection Management. This is done by using the Control Panel application, selecting one of the stations in the RF network and selecting Options | Connect to Station, to set up



persistent communication. This will keep the connection between the server and the RF base active.

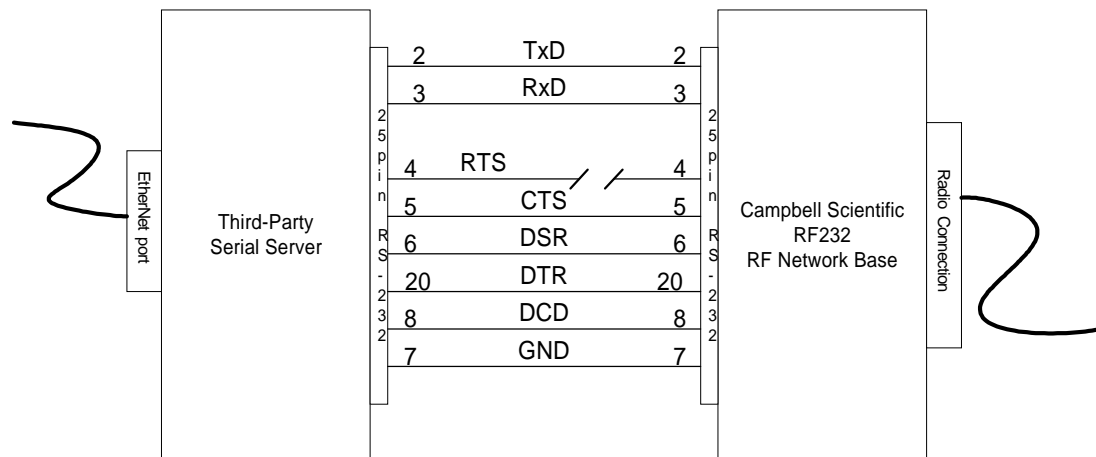
### 16.4.3 Special Considerations

To implement TCP/IP to RF communications a serial server has to be provided as the interface between TCP/IP and the serial connection on the RF base. There are a number of Serial Server devices available including the NL100 Network Link Interface manufactured by Campbell Scientific.

The RF base has a CSI I/O serial port instead of the RS-232 standard for communications. The NL-100 interface has a built-in CSI I/O port so no special cabling is required.

When connecting the RF Base to a third-party Internet Serial Server you will need to build or modify a serial cable to either cut or remove the RTS line. The other standard serial communication lines need to be in place.

This special cable is needed to allow an RF base to work with the standard RS-232 Port on other Internet serial devices. The drawing below depicts the cable needed.



The serial server must be configured for this application before operation. The basic settings that must be configured are listed below. Depending on the specific serial server, there may be other settings that must be configured for proper operation.

**IP address** – This is the Internet Protocol address that is used by LoggerNet to communicate with the serial server. This address must be unique on the network where it is running and is typically assigned by a network administrator. An IP address is typically entered as four numbers separated by periods. As an example 198.199.32.45 would be an IP address.

**Subnet Mask** - This setting is used to limit the search applicability area for IP addresses. If both the server and the serial server are in the same low level subnet this would be set to 255.255.255.0. Consult with the network administrator for the proper setting.

**Default Gateway** - This specifies the IP address of the router for the local computer network. Consult with the computer network administrator for the proper setting.

**Baud Rate** – This specifies the baud rate used by the serial server to communicate with the serial device attached to the COM port. The RF base communicates at a baud rate of 9600.

**IP Port ID** – This specifies the port ID used by the serial server to direct serial communications. This must be set even on devices with only one port. This number is entered as part of the IP address in NetAdmin for the InternetComPort device. For example, if the port ID was specified to be 3201, using the IP address above the entry in NetAdmin would appear as follows:  
198.199.32.45:3201

**Inactivity Timeout** – This timer resets the TCP/IP socket port if there has not been any activity on the port for the specified period of time. The time is usually specified in minutes. This prevents a situation where the socket gets left open after a call and blocks other incoming calls.

# ***Appendix A. Glossary of Terms***

---

## **A**

**Advise** – See Data Advise

**ASCII File** - A computer file containing letters, numbers, and other characters using the ASCII character encoding.

**Asynchronous** - The transmission of data between a transmitting and a receiving device occurs as a series of zeros and ones. For the data to be "read" correctly, the receiving device must begin reading at the proper point in the series. In asynchronous communications, this coordination is accomplished by having each character surrounded by one or more start and stop bits that designate the beginning and ending points of the information (see Synchronous). The transfer of information is not otherwise coordinated between the sender and receiver.

**Analog Channel** - A terminal on the datalogger's wiring panel where leads for analog signals are connected. The analog channels are designated single-ended (SE) or differential (DIFF) on the wiring panel. Many sensors, such as thermistor temperature probes and wind vanes, output analog signals.

**Array Based Datalogger** - Array based dataloggers save all output in the datalogger's final storage memory. When data is directed to final storage, a unique array ID number is stored, followed by other values as determined by the datalogger program. These are called "elements". Array based dataloggers save all information that is directed to output storage to the same area of datalogger memory (as opposed to table-based dataloggers that store different output processing to separate tables in datalogger memory). Data retrieved by PC software is separated based on the array IDs.

## **B**

**Baling** – The process by which the Baler produces data files containing the information collected by the LoggerNet Server during a user specified interval.

**Batch Files** - An ASCII text file that contains one or more DOS commands or executable file commands. When the batch file is run, the commands in the file are executed sequentially.

**Battery** - This entry in the status table returns the datalogger battery voltage.

**Baud** - The rate at which a communication signal travels between two devices.

**Binary File** - A file based on software defined formatting. A binary file can only be interpreted by the software programmed to decode the formatting. This format is used for more efficient data storage than is provided by ASCII.

**BMP (Block Mode Protocol)** – The communications protocol used by the server to communicate with table-based dataloggers and RF modems.

**Broadcast** – Part of the radio (RF) technique of polling remote radio modem datalogger sites. A single modem sends a message (broadcast) that all affected remotes hear and respond to.

## C

**Call-back** - When a datalogger is programmed for Call-back, it will automatically call the host computer when a specified condition is met. The computer must be set up to look for such an incoming call.

**Call-back ID Number** - A three-digit number that is used to identify what datalogger has called the host computer. (Not available for Table-based dataloggers.)

**Cancel** - Choosing Cancel from a dialog box will ignore any changes made and close the box.

**Carrier** - An electrical signal used to convey data or other information. For example, radio and phone modems use carrier signals. Phone modems attempt to detect carrier when the call is placed. The red LED on the RF95T lights when the modem detects a carrier.

**Child Node** - See Node. A node that is accessed through another device (parent node). For example a remote radio (RF) site is accessed through the base RF232T. All nodes are child nodes of the PC.

**Client** - a software application designed to connect to a server. Usually provides some type of user interface or data acquisition.

**Coaxial cable** - Special type of cable with two conductors (center conductor and outer shield conductor). Classified by size, impedance, and loss characteristics. Used to connect MD9 modems and to connect radios to antennas.

**Collection** - (see Data Collection)

**COM Port** - A computer's serial communications port. Cables and other interface devices are connected between the computer's COM port and the datalogger.

**Control Port** - Dataloggers have digital output ports that can be used to switch power to sensors such as the HMP35C relative humidity circuit or to control relays. These digital outputs are called Control Ports and are labeled C1, C2, etc., on the wiring panel. Control ports on some dataloggers can also be used as inputs to sense the digital (high or low) state of a signal, or used as data input/output connections for SDI-12 sensors.

**CoraScript** - This is a command line interpreter that allows the user access to many of the capabilities of the LoggerNet server using direct commands or programmed script files.

**CR10X-TD Family of Dataloggers** - This includes the table-based dataloggers, specifically the CR10T, CR510-TD, CR10X-TD, and CR23X-TD.

**CRBasic** - The programming language used for CR5000/9000 dataloggers. The CRBasic editor is used to create the program files for these dataloggers.

## D

**Data Advise (Datalogger)** – A mutual agreement between the server and the datalogger about which tables are to be collected every time the datalogger is contacted. Based on the dataloggers table definitions.

**Data Advise (Server)** – an agreement between a client application and the server to provide specified data as it is collected by the server.

**Data Advise Notification** – The packet of data sent by the datalogger based on the Data Advise agreement.

**Data Cache** – The storage for data collected from the datalogger by the LoggerNet server. This data is stored in binary files on the hard disk of the computer where the server is running.

**Data Collection** - Getting stored data from the datalogger and saving it in the communication server's data cache (compare to Data Retrieval). Unless specified as data collection by Data Advise, this refers to standard data collection with each message requesting specified records from the datalogger.

**Data Point** - A data value which is sent to Final Storage as the result of an Output Instruction. A group of data points output at the same time make up a record in a data table.

**Data Retrieval** - Extracting data from the communication server's data cache to export to a file, network, or data display (compare to Data Collection).

**Data Storage Table, Data Table** - A portion of the datalogger's Final Storage allocated for a particular output. Each time output for a given data table occurs, a new record is written to the table. The size of the table (in number of records) and when records are written to the data table are determined by the datalogger's Data Table Instruction (P84). The fields (columns) of the table are determined by the Output Processing Instructions that follow the Data Table Instruction.

**Data Table Instruction** - Instruction 84. Used to create a Data Table and to cause records to be written to the Data Table.

**DaysFull** - This field in the status table shows the number of days remaining before any of the tables using automatic record allocation are filled.

**Differential Analog Input** - Some sensors have two signal wires and the measurement is reflected in the voltage difference between them. This type of sensor requires two analog connections. The channels marked DIFF on the datalogger wiring panel are used to connect differential sensors.

**DLD File** - The DLD file is an ASCII file that can be sent to program the datalogger. Dataloggers must be programmed to perform measurements, convert data to final units, and to save data for retrieval. Edlog is used to create these files that are saved to disk with a DLD filename extension. A program must be sent to the datalogger before the datalogger will begin to collect data.

## E

**Edlog** - Campbell Scientific's editor application to create new or edit existing datalogger programs.

**EEPROM** - Electrically erasable read only memory. The memory CR10X-TD, CR510-TD, and CR23X-TD dataloggers use to store their operating system. A new operating system can be transferred to the datalogger using a special software package (see PROM).

**Execution Interval** - The periodic interval on which the datalogger program is run. The execution interval is sometimes referred to as the Scan Interval. For example, when an execution interval of 60 seconds is set, the datalogger will execute its program table every 60 seconds. Between executions the datalogger enters a sleep (quiescent) mode. This conserves battery power and creates predictable measurement intervals. The execution interval is synchronized with the datalogger's real-time clock.

**Execution Time** - The time required to execute an instruction or group of instructions. If the total execution time of a Program Table exceeds the table's Execution Interval, the Program Table will be executed less frequently than programmed. Each time this occurs, a Table Overrun occurs. Table Overruns are considered to be "errors" and are reported in the datalogger status information table.

**Excitation Channel** - Many sensors require a precise electrical voltage to be applied. The excitation channels, marked as E1, E2, etc., on the datalogger wiring panel, provide this required voltage.

## F

**Fault** – Message relating to network activity where repeated problems or errors have occurred. Repeated faults usually indicate a failure of some kind.

**F1** - In most instances, pressing the F1 key will provide context sensitive help for the highlighted object on the screen.

**Final Storage** - Final Storage is an area in the datalogger's memory where data is stored for collection to a PC. When you collect data from the datalogger you are collecting data from a Final Storage table.

**Flag** - Memory locations where the program can store a logical high or low value. These locations, called User Flags, are typically used to signal a state to another part of the program.

## G

**Ground Connection** - Most sensors require one or more ground connections in addition to excitation or signal inputs. Ground connections may serve any of several purposes:

- a reference for a single-ended (SE) analog voltage
- a power return path
- a connection for cable shield wire to help reduce electrical noise

## H

**Highlight** – Text or objects can be highlighted, by positioning the cursor where you want the highlight to begin, holding the left mouse button, and dragging it across the words or group of objects to be highlighted. A single object can be highlighted, by clicking it once with the left mouse button. Highlighted items can then be edited or activated.

**Holes** – Because Data Advise always get the most recent data records, there can be sequences of older data available from the datalogger that have not yet been collected to the data cache. The server tracks and optionally collects these holes. This entry in the status table shows the number of data points in missed records for the data storage tables in that station.

**Hole Collection** – The process used by the server to collect from the datalogger, data records missing from the data cache. If Hole Collection is delayed or disabled, the memory in the datalogger can ring around and overwrite the missing data records resulting in an Uncollectable Hole.

**Host Computer** - The machine where the LoggerNet communication server software is running.

## I

**INI Files** - Configuration files that are used to preserve the last known setups of a program or device.

**Initialization String** - A string of alphanumeric characters that are sent to a device, such as a modem, to prepare that device for communications.

**InLocs** - Abbreviation for “Input Locations”. This entry in the status table shows the number of input locations allocated for the program.

**Input Location Storage** - Each time a measurement or calculation is performed the resultant value is stored in an Input (memory) Location, sometimes abbreviated as "InLoc."

**Input/Output Instructions** - Datalogger program instructions used to make measurements or send data automatically to other devices.

**Intermediate Storage** - Datalogger memory used to temporarily store values, typically to be used for output calculations. The datalogger uses Intermediate Storage to accumulate sensor readings until output.

## L

**Link** – Communications route between two devices, for example the phone link between two phone modems.

**Log Files** - Log files are text files that are stored on the host computer's hard drive. They contain information about communications between the LoggerNet server and other devices in the datalogger network. Log files are typically used for troubleshooting purposes. LoggerNet has four types of log

files: Communications Status, Object State, Transaction, and Low Level I/O. Refer to Appendix D for information on these log files.

## M

**MD9** - An MD9, or multi-drop modem, is a communications device that uses twisted pair cable for connection. Typically, the system consists of one MD9 base modem that is attached to the user's computer, with one or more remote modems at the datalogger field site. One remote modem is needed for each datalogger at the field site.

**Measurements** - Measurements are what the datalogger stores in an Input Location after reading an electronic signal from a sensor and converting the raw signal into meaningful units.

**Modem** - A device used to transmit and receive digital data over normally analog communications lines, usually as an audio signal on telephone circuits. A modem attached to a computer performs a digital-to-analog conversion of data and transmits them to another modem which performs an analog-to-digital conversion that permits its attached computer to use the data.

**Monitor** - A computer's CRT, or display, is referred to as a monitor.

## N

**NetAdmin** – Network Administration program, used to create and edit network descriptions, and to display/modify the device settings.

**Net Description** – Description of dataloggers and communications devices that form the datalogger network. Entered using NetAdmin and used by the server to communicate with the various dataloggers.

**Node** – Part of the description of a datalogger network. Each node represents a device that the server will dial through or communicate with directly. Nodes are organized as a hierarchy with all nodes accessed by the same device (parent node) entered as child nodes. A node can be both a parent and a child node.

## O

**ObjSrlNo** - This entry in the status table provides the revision number of the datalogger PROM.

**Output Interval** - The output interval is the interval in which the datalogger writes data to Final Storage. The output interval is defined by Instruction 84 in Edlog (for table-based dataloggers).

**Output Processing** - Writing to final storage memory a sample or summary statistic of data measurements. Output processing options include sending a sample, average, maximum, minimum, total, or wind vector of data to Final Storage. Each Output Processing data value is kept in a separate location within the datalogger. This allows multiple output processing for each



measurement. For example, you can average air temperature over a 60-second interval, a one-hour interval, and a 24-hour interval.

**Overflow Errors** - Overflow errors occur when the actual program execution time exceeds the execution interval. This causes program executions to be skipped. When an overflow error occurs, the Table Overflow parameter in the datalogger's status table is incremented by 1.

**Overruns** - This entry in the status table provides the number of table overruns that have occurred. A table overrun occurs when the datalogger has insufficient time between execution intervals to complete one pass through the program. This counter is incremented with each table overrun.

## P

**Packet** – a unit of information sent between two BMP devices that are communicating. Each packet can contain data, messages, programming, etc. Usually contains addressing and routing information.

**Parameter** - Numbers or codes which are entered to specify exactly what a given datalogger instruction is to do.

**Path** – The modems, or other devices that make up a link to communicate with a remote site datalogger.

**Polling** – Process where a datalogger or other communications device is periodically checked for any packets it needs to send. The server polls dataloggers for most communications links. With Radio (RF) the RF232T base or repeaters can poll datalogger sites.

**Polling Interval** – The user-specified interval that determines when to poll a given device.

**PrgmFree** - An entry in the status table that shows the amount of remaining program memory, in bytes.

**PrgmSig** - An entry in the status table that shows the signature of the datalogger program. The signature is a unique number derived from the size and format of the datalogger program

**PromID** - An entry in the status table that shows the version number of the datalogger PROM or OS.

**PromSig** - An entry in the status table that shows the signature of the datalogger PROM or OS. As with the PrgmSig, if this signature changes, the datalogger instruction set has somehow been changed.

**Processing Instructions** - Datalogger instructions that further process input location data values and return the result to Input Storage where it can be accessed for output processing. Arithmetic and transcendental functions are included in these instructions.

**Program Control Instructions** - Datalogger instructions that modify the sequence of execution of other instructions in the datalogger program; also used to set or clear user flags.

**Program Signature** - A program signature is a unique value calculated by the datalogger based on program structure. Record this signature in a daily output to document when the datalogger program is changed.

**Program Table** - The area where a datalogger program is stored. Programming can be separated into two tables, each having its own execution interval. A third table is available for programming subroutines which may be called by instructions in Tables 1 or 2. The length of the tables is constrained only by the total memory available for programming.

**PROM** - Programmable Read-Only Memory. PROM integrated circuit chips are used to store the datalogger Operating System (OS) in the CR10T datalogger. The PROM can be replaced to install a new operating system (see EEPROM).

**Pulse Channel** - Some sensors output voltage pulse signals. Such sensors can be connected to Pulse Channels for measurement (labeled as P1, P2, etc., on the datalogger's wiring panel).

## Q

**Quiescent Mode** - Often referred to as "sleep mode". The datalogger is in a low power state between program execution intervals.

## R

**Real-Time Clock** - All dataloggers have an internal clock. The date and time information from this clock are used in the time stamp for stored data. The datalogger's execution interval and timer are synchronized with the clock. The CR10X-TD, CR510-TD, and CR23X-TD have battery backups which maintain the clock even when 12 V power is not available.

**Record** - A group of data values output at the same time to the same data table. Records are written in response to the Data Table Instruction (84). The individual fields within each record are determined by the Output Processing instructions following the Data Table Instruction that created the data table.

**RecNbr** - An entry in the status table that shows the record number in the table.

**Remote Site** - Usually refers to the site where datalogger is located at the other end of a communications link. Also can refer to the site where a radio (RF) repeater is located.

**Repeater** - a radio (RF) site that relays packets of information to a remote site. Used to extend the range of radio transmissions. Any remote datalogger site with radio can act as a repeater.

**Retries** - When a transaction or communication between two devices or programs fail, the transaction or communication is usually repeated until it succeeds.

**Retrieval** - (see Data Retrieval)

**RF** - Dealing with radio telemetry. Stands for Radio Frequency.

**RTMS** - Real Time Monitoring Software. A software application designed by Campbell Scientific for fast real time data acquisition. RTMS was designed for IBM's OS/2 PC operating system and replaced by LoggerNet.

## S

**Scan Interval** - See Execution Interval.

**SDI-12** - SDI-12 stands for Serial Digital Interface at 1200 baud. It is an electrical interface standard and communications protocol that was originally developed by Campbell Scientific and other manufacturers for the U.S. Geological Survey for hydrologic and environmental sensors. SDI-12 was designed to be a simple interface (ground, 12 volts, and signal) that improves compatibility between dataloggers and "smart" microprocessor-based sensors.

Other goals of the SDI-12 standard are:

- low power consumption for battery powered operation via the datalogger
- low system cost
- use of multiple sensors on one cable connected to one datalogger
- allow up to 200 feet of cable between a sensor and a datalogger

**Security Code** - A four-digit code entered into the datalogger to prevent unauthorized access to datalogger settings, programs, and data.

**Server** – Refers to a software application that accepts connections from client applications and provides data or other information as requested. The LoggerNet server manages all the communications and data collection for a network of dataloggers. The collected data is made available for client applications.

**Signature** – Number calculated to verify both sequence and validity of bytes within a packet.

**Single-ended Analog Input** - Some analog sensors have only one signal wire. (They will also have another wire that can be grounded and that is used as the reference for the signal wire.) With this type of sensor, only one analog connection is required. Hence, it needs a "single-ended" or SE analog input. The single ended channels are marked as SE on the datalogger wiring panel.

**Socket Data Export** – a software application that connects to the LoggerNet server and provides a TCP/IP socket for a user created application to receive data records from the server data cache.

**Standard Data Collection** - The normal means of collecting data where the server specifies in a message which records from which tables to collect. Compare Data Advise.

**Station** - A datalogger site is often referred to as a station.

**Station Number** – The LoggerNet server assigns and uses station numbers for routing packets to the dataloggers. These numbers can be modified using CoraScript.

**Storage** - An entry in the status table that shows the number of final storage locations available.

**Synchronous** - The transmission of data between a transmitting and a receiving device occurs as a series of zeros and ones. For the data to be "read" correctly, the receiving device must begin reading at the proper point in the series. In synchronous communications, this coordination is accomplished by synchronizing the transmitting and receiving devices to a common clock signal (see Asynchronous).

## T

**Tab Windows** - Some screens depict a series of related windows in a multi-tabbed notebook format. When you click on the file folder tab, the information on the tab you chose will be displayed.

**Tables** - An entry in the status table that shows the number of user-created data tables. (See also Data Table.)

**Table-based Dataloggers** - Table-based dataloggers store each record of data that follows an output instruction in a table. Each separate occurrence of an output instruction directs the datalogger to store the data in a separate table. Refer to Appendix B for additional information on table-based dataloggers.

**Table Definitions** - List of data available from a datalogger. The datalogger supplies this list on request. The tables are based on the datalogger programming. The LoggerNet server must have a current version of the table definitions to collect data from the datalogger.

**Throughput** - The rate at which a measurement can be made, scaled to engineering units, and the reading stored in Final Storage.

**Time Stamp** - The date and time when data are stored in the datalogger.

**TMStamp** - An entry in the status table that shows the date and time the status information was recorded.

**Transaction** - The exchange of data or information between two devices or programs. For example setting of the clock in a datalogger is a transaction between the server and the datalogger.

## U

**Uncollectable Hole** - Occurs when a hole in the data cache cannot be collected from the datalogger before the data table wraps around and the records are overwritten.

## V

**Variable Name** - Edlog uses variable names in expressions. Variables are another name for input location labels. For instance, in the equation  $\text{TempF} = (\text{TempC} * 1.8) + 32$ , TempC is an input location label and TempF is a new location calculated from TempC.

## W

**Wiring Panel** - The set of terminals and underlying circuits that enable connections of sensors, control and power supply wiring to the datalogger itself. Some dataloggers such as the CR23X-TD have built-in wiring panels. Others, such as the CR10X-TD, have removable wiring panels.

**Watchdog** - An entry in the status table that shows the number of watchdog errors that have occurred. The watchdog checks the processor state and resets it if necessary. If an error occurs, the watchdog error counter is incremented.



# ***Appendix B. Table-Based Dataloggers***

---

*This section describes some of the characteristics and features of the CR10X-TD family and CRx000 family of table-based dataloggers. The dataloggers included in these families are CR510-TD, CR10T, CR10X-TD, CR23X-TD, CR5000 and CR9000.*

## **B.1 Memory Allocation for Final Storage**

The datalogger memory includes four important areas: the datalogger program storage, input storage, intermediate storage, and final storage. When a program is downloaded to the datalogger and compiled, datalogger memory is allocated for each of these areas.

The CR10X family of array-based and table-based dataloggers are identical in hardware and differ only in the operating system. The primary distinction between array-based and table-based dataloggers is how final storage is allocated and filled. CRx000 family of dataloggers are based on CRbasic programs and have a different memory allocation structure.

### **B.1.1 CR10X-TD Family Table-Based Dataloggers**

CR510-TD, CR10T, CR10X-TD, and CR23X-TD table-based dataloggers store data from different intervals in different final storage tables. Final storage tables are made up of records and fields. Each row in a table represents a record and each column represents a field. The number of fields in a record is determined by the output processing instructions in the datalogger program that follow the Data Table output instruction (P84 Output Table; refer to your datalogger user's manual for more information). The total number of fields for each table will be the number of output processing instructions multiplied by the number of values stored by each of the output instructions.

The number of records to be kept in a table before the oldest data is overwritten can be fixed by the user, or left for the datalogger to determine automatically. With automatic allocation the datalogger tries to set the sizes of automatically allocated tables such that all of the tables will fill up at about the same time. Once the sizes of the tables are determined, the datalogger allocates available final storage to these tables.

Note that the tables are allocated by size with the smallest tables first. For dataloggers with extended flash memory, any tables that will not fit in SRAM memory are allocated to flash memory. Flash memory is allocated in 64 K blocks; therefore, even a very small table will take 64 K of flash memory. If the table sizes specified by the user exceed the amount of memory available for that purpose in the datalogger, an error will occur when the program is compiled by the datalogger.

**NOTE**

---

Event driven tables should have a fixed size rather than allowing them to be allocated automatically. Event driven tables in CR10X-TD type dataloggers that are automatically allocated are assumed to have one record stored per second in calculating the length. Since the datalogger tries to make the tables fill up at the same time, these event driven tables will take up most of the memory leaving very little for the other, longer interval, automatically allocated data tables.

---

## **B.1.2 CR5000/CR9000 Memory for Programs and Data Storage**

The datalogger memory for the CR5000 and CR9000 is divided between Random Access Memory (RAM) and Electrically Erasable Programmable Read Only Memory (EEPROM). The EEPROM, or flash memory, is used to store the operating system and the user programs that have been saved in the datalogger. When the datalogger powers up, the program marked as “Run on Power-up” is transferred to RAM and executes from there. Additional storage is available using PCMCIA cards.

When a datalogger program is sent to the datalogger, it is divided into two tasks that run simultaneously. All of the program instructions that deal with measuring sensors, controlling outputs, or are time sensitive, are placed in the **measurement task**. The instructions that deal with data processing, including calculations, data storage, averaging, minimum and maximum tracking, and data I/O operations, are placed in the **processing task**.

The measurement task is executed at the precise specified scan rate and stores the raw data into a memory buffer. As soon as the measurement task has completed filling the buffer for the current scan, the processing task starts the data processing on the buffered data. There are at least two memory buffers, allowing the measurement task to fill one buffer while the processing task is working with the data in the other.

The data processing task stores data as records in final storage data tables. LoggerNet can collect the records from these data tables either manually or with scheduled data collection. The datalogger program can also make some or all of the variables used for measurement storage or calculations available to LoggerNet. These variables are found in the Public table, which is similar to the Input Location table in CR10X or CR10X-TD type dataloggers.

Final storage tables are made up of records and fields. Each row in a table represents a record and each column represents a field. The number of fields in a record is determined by the number and configuration of output processing instructions that are included as part of the Data Table definition.

The number of records to be kept in a table before the oldest data is overwritten can be limited by the user, or left for the datalogger to determine automatically. The datalogger tries to set the sizes of automatically allocated tables such that all of the tables will fill up at about the same time. Once the sizes of the tables are determined, the datalogger allocates the available memory to these tables.



If the amount of memory requested for the data tables exceeds the available memory, the program will not run.

#### NOTE

Event driven tables should have a fixed size rather than allowing them to be allocated automatically. Event driven tables in CR5000/CR9000 dataloggers that are automatically allocated are assumed to have one record stored per execution interval in calculating the length. Since the datalogger tries to make the tables fill up at the same time, these event driven tables will take up most of the memory leaving very little for other, longer interval, automatically allocated data tables.

## B.2 Converting an Array-Based Program to a CR10X-TD Table-Based Program using Edlog

The following information is provided for those users familiar with writing programs for array-based dataloggers or users who have existing array-based datalogger programs that need to be changed to table-based programs.

### B.2.1 Steps for Program Conversion

If you are converting a program for the same series of datalogger (e.g., a CR10X program to a CR10X-TD program) you can edit the existing program in Edlog. If you are converting a program from one datalogger series to another (e.g., CR10X to CR23X-TD), you may need to start the program from scratch.

To convert a program for the same series of datalogger:

1. Open the CSI file in Edlog.
2. The first line of the file will read

```
;{CR10X}
```

Change this line to

```
;{CR10X-TD}
```

3. **Review all of the instructions provided in the section below.** If any of these are included in your program, format them as a comment or delete them from the program.
4. Save the file to a new file name, but **do not compile the file when prompted.**
5. Open the newly created file in Edlog. It will be opened using the CR10X-TD datalogger template instead of the CR10X. Make any changes necessary to replace the commented or deleted instructions.

6. Save and compile the program, correcting any errors that may be found by the compiler.

## B.2.2 Program Instruction Changes

Several programming instructions have changed or are not used in table-based datalogger programs. Make sure you “comment out” any of these instructions before you try to convert the array-based program. These are listed below:

- Check any instructions that may set the Output Flag (Flag 0) high or low by using the Command Code Options. The output flag is not used in table-based programming. Instructions that may include reference to the output flag are: P83, If Case; P86, Do; P88, If (X< = > Y); P89, If (X< = > F); P91, If Port/Flag; and P92, If Time.

If any of these instructions set the output flag high, the instruction can be replaced with Instruction 84, Table Data. Instruction 84 is used to define a table of final storage data. New records of data are stored in the table-based on time (interval data) or when a user flag is set (event data). Time based output intervals are specified in seconds.

- **Instruction 18, Time** - Instruction 18 is used to store the current time into an input location. Parameter 1 designates what format will be used when storing the time. There are differences in this instruction's Parameter 1 for the two datalogger types.
- **Instructions 73 and 74, Maximum and Minimum** - These instructions are used to store the maximum or minimum for a value over a period of time. Parameter 2 in these instructions is used to designate a time option. There are differences in the instructions' Parameter 2 for the two datalogger types.
- **Instruction 77, Real Time** - Instruction 77 is used to store the current time in final storage for array-based dataloggers. This instruction is not included in table-based dataloggers, since the time is assigned to records automatically when data is retrieved (see Section 5.3 for information on how data is time stamped).
- **Instruction 80, Set Active Storage Area** - Instruction 80 is used to direct output processing to final storage area 1, final storage area 2, or an input location. This instruction is not included in the table-based programming instructions. Output processing can be redirected to input locations in a table-based datalogger using Instruction P84, Table Data (see Edlog's help).
- **Instruction 92, If Time** - Instruction 92 is used to perform one or more actions based on time. The interval for table-based dataloggers is in seconds only; array-based dataloggers offer the options of seconds or minutes. The instruction for array-based dataloggers defaults to minutes, so if you are using this instruction it may need to be changed.

Also, check any Instruction 92s for Command Codes that may affect the output flag (see discussion above on output flag instructions).

- **Instruction 96, Serial Output** - Instruction 96 is used to send data in the active Final Storage area to a storage module, computer, printer, or alternate final storage area. This instruction is not included in the table-based programming instructions.
- **Instruction 97, Initiate Telecommunication** (“Call-back”) - Instruction 97 is used to program the datalogger to call a remote computer, voice modem, or other datalogger when a user flag is set low. This instruction is not included in the CR10T datalogger. Newer operating systems for the other dataloggers do support instruction 97.
- **Instruction 98, Send Printer Character** - Instruction 98 is used to send characters to either an addressed or pin-enabled printer. This instruction is not included in the table-based programming instructions.
- **Conditional Data Output** – check to make sure that the output data is not being output conditionally. Table-based dataloggers require that the size of the output record is constant. Any instructions that dynamically change the number of data values in a record or the size of the record need to be removed. (e.g., don’t change data resolution from low to high based on a conditional. )

## B.3 Table Data Overview

In the datalogger all data is organized into tables with fixed data records. Each of these tables has a definite number of records that is either fixed by the datalogger program or allocated when the program is compiled by the datalogger. Once the maximum number of records for a table have been stored, the next record stored will overwrite the oldest record in the table.

Tables that are automatically allocated in the datalogger program are allocated a number of records based on the time interval for the records. The datalogger attempts to allocate these tables so that all of the automatically allocated tables fill up at the same time. For example a tables with records stored every 30 minutes and 60 minutes would have twice as many records allocated for the 30-minute table.

---

**NOTE**

Event driven tables should have a fixed size rather than allowing them to be allocated automatically. Event driven tables in the CR10X-TD type dataloggers that are automatically allocated are assumed to have one record stored per second in calculating the length.

In CR5000/CR9000 dataloggers event tables are assumed to have one record stored per execution interval.

Since the datalogger tries to make the tables fill up at the same time, these event driven tables will take up most of the memory leaving very little for the other, longer interval, automatically allocated data tables.

---

Within a data table, data is organized in records and fields. Each row in a table represents a record and each column represents a field. To understand the concept of records it may be helpful to consider an example.

**Example:**

A CR10X-TD is to be used to monitor three thermocouples (TC). Each hour a temperature for each of the three thermocouples is to be stored. The table has five fields: DATE\_TIME, RECORD #, TEMP1, TEMP2, TEMP3.

The program is written so that each hour an Instruction 84, Table Data, generates a new "record" in the data table. This hourly table would then be organized as follows:

DATE_TIME	RECORD #	TEMP1	TEMP2	TEMP3
2002-01-27 10:00:00	14	23.5	24.6	28.2
2002-01-27 11:00:00	15	24.2	22.4	23.4

Only the hourly data triggered by the Instruction 84 above would be written to this table. If other table data instructions existed, the output for these tables would be written to their own tables.

Data tables can also be event driven rather than interval driven. That is, a new record is stored when a specified event occurs rather than based on time.

Each table is completely independent of any other tables and all records in a given table have the same number of fields.

## B.4 Default Tables

Each table-based datalogger has a set of default tables plus the tables created by the datalogger program. The four default tables in the CR10X-TD family of dataloggers, are Timeset, Errorlog, Inlocs, and Status. The default tables in CR5000/9000 dataloggers are Status and Public.

- **Timeset Table** - The Timeset table contains a history of clock sets for the datalogger. It includes three fields: TimeStamp, RecordNumber, and OldTime. TimeStamp is the time and date the clock was set. RecordNumber is incremented each time the clock is set. When the datalogger is reset or a new program is loaded, RecordNumber is reset to 1. OldTime is the datalogger's clock value before the time was set (CR10X-TD family dataloggers only).
- **Errorlog Table** - The Errorlog table contains any errors that occur in the datalogger. It includes three fields: TimeStamp, RecordNumber, and ErrorCode. TimeStamp is the time and date the error occurred. RecordNumber is incremented each time an error occurs. When the datalogger is reset or a new program is loaded, RecordNumber is reset to 1. ErrorCode is the code returned by the datalogger when an error occurs. Refer to the datalogger user's manual for a list of all error codes (CR10X-TD family dataloggers only).
- **Inlocs Table** (CR10X-TD family dataloggers) or **Public Table** (CR-x000 dataloggers) - When a datalogger measures a sensor, the sensor reading is

stored in a temporary register called an input location or variable. With each new measurement, the old value is overwritten by the new value. The Inlocs or Public table contains a time stamp, record number, flag status, port status, and the reading from each sensor scanned or user created input locations.

- **Status Table** - The Status table contains information on the datalogger. Data is written to the table with each datalogger program execution. Note that the actual fields contained in the table are datalogger-specific. Table B-1 below describes each of the fields in the table for a CR10T. CR10X-TD, CR510TD and CR23X-TD status tables are similar. See the datalogger operator's manual for specifics. Table B-2 describes the fields for the CR5000 datalogger. CR9000 is similar.

Table B-1. Example of Status Table Entries (CR10T)	
<b>TMStamp</b>	Date and time the status information was recorded.
<b>RecNBR</b>	The record number in the table.
<b>Battery</b>	Datalogger battery voltage.
<b>Watchdog</b>	The watchdog checks the processor state, software timers, and program related counters. If an error occurs, the watchdog counter is incremented.
<b>Overruns</b>	A table overrun occurs when the datalogger has insufficient time between execution intervals to complete one pass through the program. This counter is incremented with each table overrun.
<b>InLocs</b>	Number of input locations allocated for the program.
<b>PrgmFree</b>	Amount of remaining program memory, in bytes.
<b>Storage</b>	Number of final storage locations available.
<b>Tables</b>	Number of user-created data tables.
<b>DaysFull</b>	Estimated number of days of data the tables using automatic record allocation can hold. (NOTE: this number is only based on tables stored at intervals. Automatically allocating an event based table will often result in very small interval tables.)
<b>Holes</b>	Number of missed records in all data storage tables.
<b>PrgmSig</b>	Signature of the datalogger program. The signature is a unique number derived from the size and format of the datalogger program. If this signature changes, the program has been altered.
<b>PromSig</b>	Signature of the datalogger PROM. As with the PrgmSig, if this signature changes, the datalogger instruction set has somehow been changed.
<b>PromID</b>	Version number of the datalogger PROM.
<b>ObjSrlNo</b>	Revision number of the datalogger PROM.

Table B-2. CR5000 Status Table Entries	
<b>ROMVersion</b>	Version of the ROM code. This value is stored in the ROM and read by the OS at compile time.
<b>OSVersion</b>	Current version of the operating system.
<b>OSItem</b>	The CSI item number for the operating system.
<b>OSDate</b>	Date that the Operating System was compiled.
<b>StationName</b>	String stored as the Station Name of the CR5000.
<b>ProgName</b>	The Name of the currently running program.
<b>StartTime</b>	Time that the program began running.
<b>Battery</b>	Current value of the battery voltage. This measurement is made in the background calibration.
<b>PanelTemp</b>	Current Panel temperature measurement.
<b>LithiumBattery</b>	A Boolean variable signaling "True" (-1) if the lithium battery is OK and "False" (0) if not. The lithium battery is loaded and a comparator checked every 4 seconds to verify that the battery is charged.
<b>CPUSignature</b>	The Operating System signature. The value should match the value obtained by running the CSI sig program on the <i>name.obj</i> operating system file.
<b>DLDSignature</b>	Signature of the current running program file.
<b>ProgSignature</b>	Signature of the compiled binary data structure for the current program. This value is independent of comments added or non functional changes to the program file.
<b>PC-CardBytesFree</b>	Gives the number of bytes free on the PC-Card.
<b>MemoryFree</b>	Amount (in bytes) of unallocated memory on the CPU (SRAM). The user may not be able to allocate all of free memory for data tables as final storage must be contiguous. As memory is allocated and freed there may be holes that are unusable for final storage, but that will show up as free bytes.
<b>DLDBytesFree</b>	Amount of free space in the CPU RAM disk that is used to store program files.
<b>ProcessTime</b>	Time in microseconds that it took to run through processing on the last scan. Time is measured from the end of the EndScan instruction (after the measurement event is set) to the beginning of the EndScan (before the wait for the measurement event begins) for the subsequent scan.
<b>MaxProcTime</b>	The maximum time required to run through processing for the current scan. This value is reset when the scan exits.

Table B-2. CR5000 Status Table Entries

<b>MeasureTime</b>	The time required by the hardware to make the measurements in this scan. The sum of all integration times and settling times. Processing will occur concurrent with this time so the sum of measure time and process time is not the time required in the scan instruction.
<b>SkippedScan</b>	Number of skipped scans that have occurred while running the current program.
<b>SlowProcTime</b>	Time required to process the current slow scan. If the user has slow scans then this variable becomes an array with a value for the system slow scan and each of the user's scans.
<b>MaxSlowProcTime</b>	The maximum Time required to process the current slow scan. If the user has slow scans then this variable becomes an array with a value for the system slow scan and each of the user's scans.
<b>LastSlowScan</b>	The last time that this slow scan executed. If the user has slow scans then this variable becomes an array with a value for the system slow scan and each of the user's scans.
<b>SkippedSlowScan</b>	The number of scans that have been skipped in this slow sequence. If the user has slow scans then this variable becomes an array with a value for the system slow scan and each of the user's scans.
<b>MeasureOps</b>	This is the number of task sequencer opcodes required to do all measurements in the system. This value includes the Calibration opcodes (compile time) and the system slow sequence opcodes.
<b>WatchdogErrors</b>	The number of Watchdog errors that have occurred while running this program. This value can be reset from the keyboard by going to status and scrolling down to the variable and pressing the DEL key. It is also reset upon compiling a new program.
<b>Low12VCount</b>	Keeps a running count of the number of occurrences of the 12VLow signal being asserted. When this condition is detected the logger ceases making measurements and goes into a low power mode until the system voltage is up to a safe level.
<b>StartUpCode</b>	A code variable that allows the user to know how the system woke up from poweroff.
<b>CommActive</b>	A variable signaling whether or not communications is currently active (increments each time the autobaud detect code is executed).

Table B-2. CR5000 Status Table Entries	
<b>ProgErrors</b>	The number of compile (or runtime) errors for the current program.
<b>ErrorCalib</b>	A counter that is incremented each time a bad calibration value is measured. The value is discarded (not included in the filter update) and this variable is incremented.
<b>VarOutOfBound</b>	Flags whether a variable array was accessed out of bounds.
<b>SkippedRecord</b>	Variable that tells how many records have been skipped for a given table. Each table has its own entry in this array.
<b>SecsPerRecord</b>	Output interval for a given table. Each table has its own entry in this array.
<b>SrlNbr</b>	Machine specific serial number. Stored in FLASH memory.
<b>Rev</b>	Hardware revision number. Stored in FLASH memory.
<b>CalVolts</b>	Factory calibration numbers. This array contains twenty values corresponding to the 20 integration / range combinations. These numbers are loaded by the Factory Calibration and are stored in FLASH.
<b>CalGain</b>	Calibration table Gain values. Each integration / range combination has a gain associated with it. These numbers are updated by the background slow sequence if the running program uses the integration / range.
<b>CalSeOffset</b>	Calibration table single ended offset values. Each integration / range combination has a single ended offset associated with it. These numbers are updated by the background slow sequence if the running program uses the integration / range.
<b>CalDiffOffset</b>	Calibration table differential offset values. Each integration / range combination has a differential offset associated with it. These numbers are updated by the background slow sequence if the running program uses the integration / range.
<b>CardStatus</b>	Contains a string with the most recent card status information.
<b>CompileResults</b>	Contains any error messages that were generated by compilation or during run time.



# ***Appendix C. Software Organization***

---

## **C.1 LoggerNet/Client Architecture**

The LoggerNet communication server provides the interface to all of the dataloggers and the support for the different communications mediums. It runs in the background and provides an attachment for the clients that provide the user interface. The server handles all communications with the dataloggers.

The LoggerNet server can support the attachment of multiple clients simultaneously, allowing many different views and ways to access the data collected from the dataloggers. The clients can run on the same computer as the LoggerNet server application, or can attach over a computer network such as a corporate Local Area Network (LAN) or the Internet. Using this network capability, users can both control and access information from the datalogger network from anywhere on the computer network.

The LoggerNet server can automatically collect data from the dataloggers on a schedule as well as on request from the user. It can automatically check and update the clocks in the dataloggers and handle administration support functions.

## **C.2 LoggerNet Server Data Cache**

The LoggerNet server data cache is a set of files kept on the hard disk of the computer where the server is running. These data files are in binary format and can only be used or interpreted by the LoggerNet server.

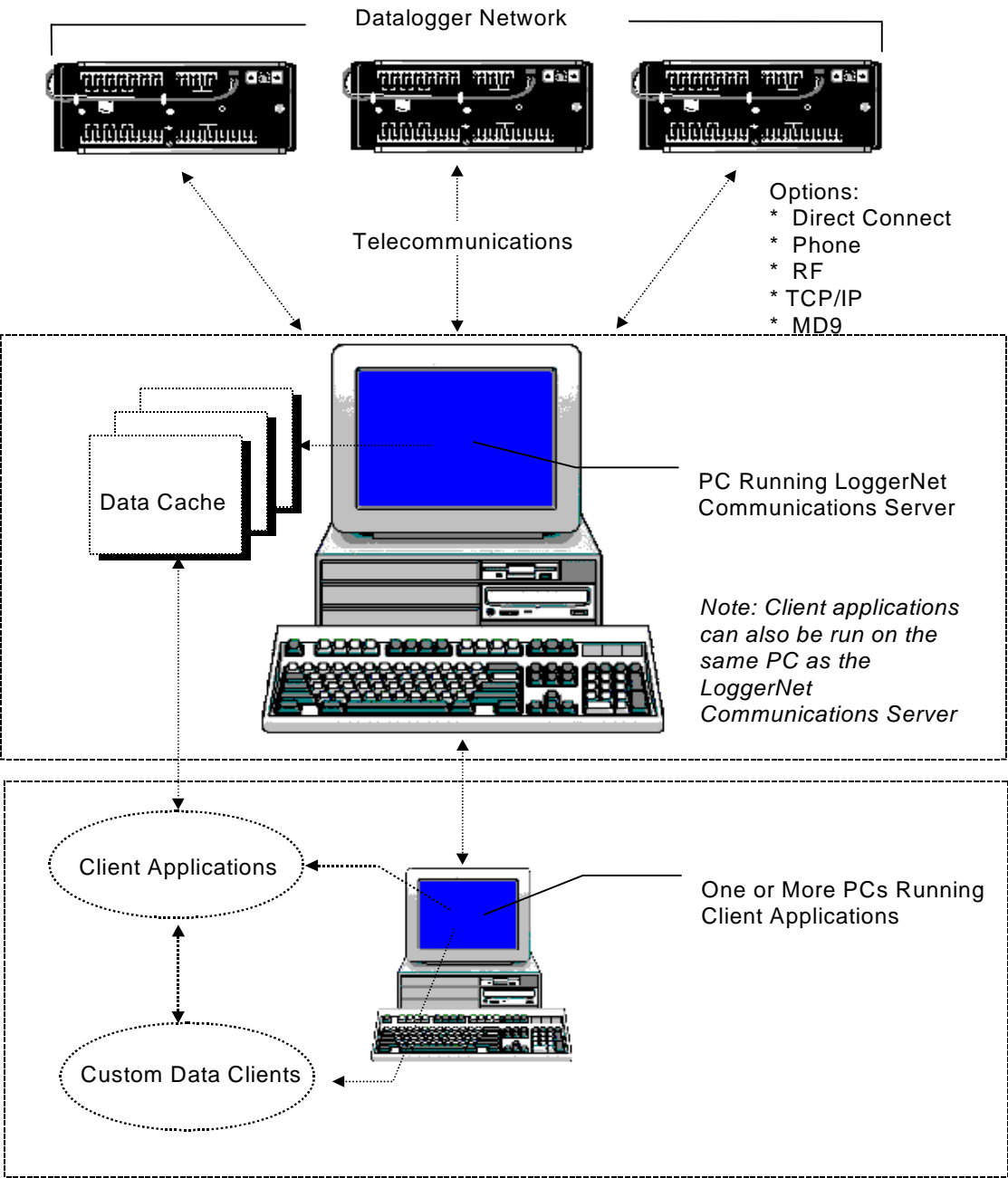
### **C.2.1 Organization**

The data cache is set up to emulate the way data is stored in the datalogger. When a new datalogger station is defined for the network and communication is established with the station, the server requests the table definitions from the datalogger. This table information is used to set up equivalent tables for data storage in the data cache. The size of the areas set up in the data cache is dependent on the size of the tables in the datalogger and the Database Table Size Factor as set on the Data Collection tab for the datalogger station.

For example, if the Database Table Size Factor is set at 2 (the default), enough space will be set up in the data cache to hold two times the number of data records the datalogger has in that table. Therefore, if there is a data table in the datalogger that stores 2000 records before overwriting the oldest, the data cache will hold 4000 records. Datalogger tables that hold only one record, such as the Input Locations table and the Status table, would have only two records assigned in the data cache.

The storage in the data cache is designed to operate with “ring memory” just like the datalogger. This means that records will be stored in the data cache area for that table until it has reached the

maximum number of records, the next data record will replace the oldest record in the storage table, and so on.



## C.2.2 Operation

Normal data collection from the datalogger is done with data requests based on the scheduled collection interval set up by the user. This is the most efficient means of data collection for networks with rapid direct communications links. When it is time for a scheduled data collection the server sends a data request message to the datalogger to get all of the data stored in the selected tables since the last collection. The tables to be collected are specified by the user in NetAdmin.

Data collection from the datalogger over RF networks is accomplished using a Data Advise operation (refer to Section B.4). Data Advise sets up a list in the datalogger for the tables to be collected. When it is time for the data to be collected the RF base broadcasts a message to the remote RF modems that prompt the dataloggers. The dataloggers respond with the new data for the pre-selected data tables. The Data Advise operation gets the most current set of data records. This is the most efficient way to get data over RF since it doesn't have to establish a link to the datalogger or specify the data to collect every time.

As data is collected from the datalogger and placed in the data cache, the server monitors the timestamps and record numbers for the records being received from the datalogger. If there is a gap in the record numbers the server identifies this as a hole, or missing records, in the data. A hole collection operation is started automatically to try and collect the missing records from the datalogger. Only Data Advise operations create holes that can be collected from the datalogger.

As each record is written to the data cache, the server adds a filemark number to the record as it is stored. This filemark number is used to identify discontinuities in the data. The filemark number starts out as zero when the table for the data cache is created or re-initialized. This number is incremented each time a discontinuity is seen in the data records. Such a discontinuity can occur when a hole becomes "uncollectable" because the data table in the datalogger has filled up and then overwritten the missing data. This also can occur if the record number rolls over from the maximum to start back at zero or an identical program is loaded into the datalogger without going through the server.

Data can also be collected from the datalogger using a manual poll operation. This is achieved by selecting Update Data Cache from the Control Panel's Retrieve to File tab. When a Manual Poll is done the data from the datalogger is also put into the data cache. No holes are created with Manual Polling since it gets all of the records it needs rather than the most recent as done by the Data Advise.

## C.2.3 Retrieving Data from the Cache

Once the data has been stored in the data cache it is retrieved by clients that request the data by datalogger, table, and data field. The data can be requested by a query where the request specifies the starting and ending timestamp or record number along with the data to retrieve. A server Data Advise can also be set up that will automatically send the selected data out to the client

applications as it is collected from the dataloggers to the data cache. This improves efficiency since the client doesn't have to keep checking to see if new data has been collected.

## C.2.4 Updating Table Definitions

When the table definitions are obtained from the datalogger they are kept in the server and used to identify the data available in the data cache. Every time new data is collected from a datalogger, a table definition signature is sent that should match the signature stored in the server. If this table definition signature doesn't match it indicates that the table definitions in the datalogger have changed.

There are a number of things that could cause datalogger table definitions to change. A new program may have been downloaded to the datalogger, or the keyboard display may have been used to manually make changes to the datalogger program.

---

**NOTE**

If the datalogger program is re-compiled without changing table definitions, the record numbers will reset to zero causing the server to assume the datalogger record numbers have wrapped around. This will result in the re-collection of all of the data in the datalogger.

---

When a change in table definitions is detected, the server stops data collection and indicates in NetAdmin and the Control Panel that the table definitions have been changed. Data collection cannot be restarted until either a new datalogger program is loaded into the datalogger by the server, or updated table definitions are received from the datalogger. Either of these actions causes the data in the data cache for that datalogger to be removed and new data cache tables set up based on the new table definitions for that datalogger.

---

**NOTE**

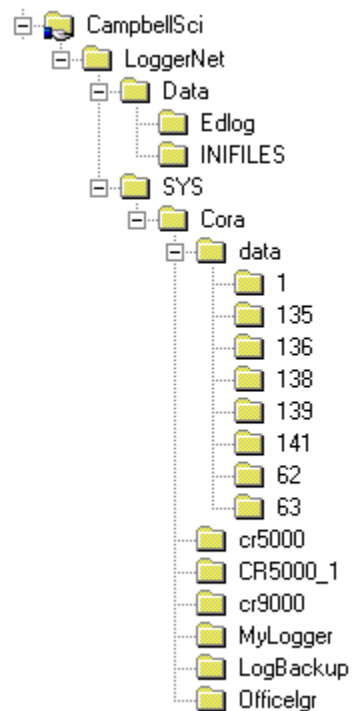
All of the important data from the data cache should be retrieved before loading a new program or updating the table definitions. If not, the data will be lost.

---

## C.3 Directory Organization

The default installation of the LoggerNet software installs two directories: the C:\CampbellSci\LoggerNet working directory and C:\Program Files\CampbellSci\LoggerNet program directory.

### C.3.1 C:\CampbellSci\LoggerNet Directory (Working Directory)



The C:\CampbellSci\LoggerNet working directory includes two subdirectories: Data and SYS. The c:\CampbellSci\LoggerNet directory is used for storing user-created files, data files, and status information files from the LoggerNet server. Any files collected using the Control Panel are stored by default in the C:\CampbellSci\LoggerNet\Data subdirectory. The Data subdirectory may also include a Baled\_Files subdirectory, and when Baling starts other subdirectories are created under Baled\_Files that reflect the datalogger names. When files are collected from the data cache using the Baler, they are stored in this C:\CampbellSci\LoggerNet\Data\Baled\_Files\(*datalogger\_name*) subdirectory. The Inifiles subdirectory under C:\CampbellSci\LoggerNet\Data contains all configuration files for the Control Panel, Baler, and other clients.

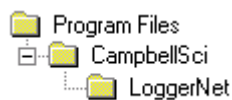
The C:\CampbellSci\LoggerNet\Data\Edlog subdirectory is used to store all of the files created by the Edlog program editor. These files include: \*.CSI, \*.DLD, \*.PTI, and \*.FSL.

The C:\CampbellSci\LoggerNet\SYS\Cora subdirectory contains configuration files for the devices in your datalogger network (\*\_CONF), the modem configuration file (modem.ini), and the log files generated by the LoggerNet server (\*.log). This subdirectory has a C:\CampbellSci\LoggerNet\SYS\Cora\Data subdirectory that is used to store the Data Cache for the devices in the network.. In the Data directory there is a numbered subdirectory that is used to store the data tables for each datalogger. The directory number corresponds to the device ID used by the server to identify that device in the network. The subdirectories that match the names of the

dataloggers in the network are used to store the last program sent to the datalogger by the server.

System administrators concerned about security and system integrity should use Windows NT and its directory access tools to control access to the working directories.

### **C.3.2 C:\Program Files\CampbellSci\LoggerNet Directory (Program File Directory)**



All files necessary for running the LoggerNet programs are stored in the C:\Program Files\CampbellSci\LoggerNet subdirectory.

No other files are saved to these program file subdirectories in order to protect the integrity of the LoggerNet communication server and the clients.

# Appendix D. Log Files

---

## D.1 Event Logging

As LoggerNet performs its work, it will create records of various kinds of events. The log files are very useful for troubleshooting problems and monitoring the operation of the datalogger network. See Section 7.2.10 for information on saving the logs to disk.

### D.1.1 Log Categories

The LoggerNet server logs four different kinds of events as follows:

**Transaction Status (TranX.log)** — This log file documents the state of the various transactions that occur between the LoggerNet server and devices in the datalogger network. Examples of these events are:

- Datalogger clock check/set
- Datalogger program downloads
- Data collection

The format and type of records in this log are strictly defined to make it possible for a software program to parse the log records.

**Communications Status (CommsX.log)** — This log file documents the quality of communications in the datalogger network.

**Object State (StateX.log)** — This log file documents the state of an object. This is primarily for troubleshooting and the messages are relatively free in form.

**Low Level I/O (IOXSerial Port\_1.log)** — A low level log file is associated with each root device in the datalogger network to record incoming and outgoing communications. While the entire network can be monitored from a single messaging session of the transaction, communications status, or object state logs, monitoring of the low-level log is performed on a session with the root device for that log.

### D.1.2 Enabling Log Files

Use NetAdmin (Options | Server Settings) to enable logging of events to files. If enabled, the server will write log records to text files in the server's working directory using the following file names (depending on the log type):

- Transaction Log - TranX.log
- Communications Log - CommsX.log
- Object State Log - StateX.log
- Low Level Log - IOXSerial Port\_1.log

where “X” is “\$” for the currently active file and 0, 1, 2, etc. for archived files.

The server stores the most recent log records in a file that has a \$ character in the place of the version number. When this file grows to the point that it will exceed the threshold set by the File Size setting for that log, the server will rename the log file by replacing the dollar sign with a new version number. At the same time that the server rolls over to a new log file, the File Count parameter for that log will also be evaluated. If there are more saved files for that log than is allowed by the File Count parameter, the server will delete the oldest of these files until the count is less than or equal to the File Count.

## D.1.3 Log File Message Formats

### D.1.3.1 General File Format Information

The communications status, transaction, and object state logs all share the same basic file format. Each record in a log file ends with a carriage return and line feed. A single record will consist of two or more fields where each field is surrounded by quotation marks and separated by commas.

The two fields that will be present in all records are:

**Time Stamp** - The server time when the record was generated. It will have the following format:

YYYY-MM-DD HH:MM:SS.mmm

where "YYYY" is the 4-digit year, "MM" is the month number, "DD" is the day of the month, "HH" is the hour in the day (24 hour format), "MM" is the minutes into the hour, "SS" is the seconds into the minute, and "mmm" is the milliseconds into the second.

**Device Name** - The name of the device associated with the message. If the message is associated with the LoggerNet server, this will be an empty string.

### D.1.3.2 Communications Status Log Format

Each record in the communications status log includes two fields in addition to the time stamp and device name:

**Severity** - A single character code that indicates the type of message. The following values are legal:

- "S" (Status) Indicates that the identified operation has successfully completed.
- "W" (Warning) Indicates that the server has attempted to retry the operation with the identified device.
- "F" (Fault) Indicates that the identified operation has failed and that the server has stopped retrying.



**Description** - text providing more details about the event.

#### **D.1.3.2.1 Communications Status Log Example**

```

3/6/2002 12:06:00 PM | "IPPort_2","S","opening","192.168.8.129,3001"
3/6/2002 12:06:00 PM | "IPPort_2","S","Provider opened"
3/6/2002 12:06:00 PM | "IPPort_2","S","Open succeeded"
3/6/2002 12:06:00 PM | "IPPort_2","S","Device dialed"
3/6/2002 12:06:00 PM | "ComPort_2","S","opening comm port","com2","38400"
3/6/2002 12:06:00 PM | "ComPort_2","S","Provider opened","38400"
3/6/2002 12:06:00 PM | "ComPort_2","S","Open succeeded"
3/6/2002 12:06:00 PM | "ComPort_2","S","Device dialed"
3/6/2002 12:06:05 PM | "CR23X_2","S","Classic::CmdA"
3/6/2002 12:06:05 PM | "CR23X_2","S","Classic::CmdClockSet"
3/6/2002 12:06:07 PM | "CR510TD","W","Unable to locate serial synch byte"
3/6/2002 12:06:07 PM | "CR510TD","S","Serial packet 2 exchanged"
3/6/2002 12:06:08 PM | "CR510TD","S","Serial packet 3 exchanged"
3/6/2002 12:06:08 PM | "CR510TD","S","Serial packet 0 exchanged"
3/6/2002 12:06:08 PM | "CR510TD","S","BMP1 packet received"

```

Message Text	Message Meaning	User Response to Message
Serial packet X exchanged	The low level serial communication framing packet was sent and the response received from the device.	
BMP1 packet received	A BMP1 packet was received from the device.	
RPC packet exchanged	A BMP3 packet	
Datalogger did not respond to end command	The computer tried to terminate the connection but the datalogger did not acknowledge the shutdown.	This is an indication that there is a communications problem between the computer and the datalogger. Check the cables and connectors and make sure the datalogger has power.
Invalid low level signature	The packet received from the device got corrupted and the packet signature doesn't match the packet contents.	Check to find out where in the communications link noise or signal corruption is causing the data to be disrupted.
Provider opened	The serial communications port has been initialized.	
Device dialed	The communications link has been initialized to transfer data packets.	
Provider closed	The serial communications port has been closed.	

Message Text	Message Meaning	User Response to Message
Serial synch byte not found	The low level communications synchronization byte was not received after the computer sent out a serial packet.	This indicates that the device is either not responding or responding with an invalid communications protocol. This message would appear if trying to talk to an array based datalogger that is setup as a table based datalogger in the network map.

### D.1.3.3 Transaction Log Format

Each record in the transaction log includes at least two fields in addition to the time stamp and device name:

**Message Type Code** - Identifies the type of event that has occurred.

**Message Type Description** - A piece of text that describes the message type code.

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
1	Network device added	Device Name	A new device was added to the network map.	
2	Network branch deleted	Device Name	A branch of the network map was deleted (this may consist of a single device)	
3	Network branch moved	Device Name	A branch of the network map was moved from one parent device to another (not supported in 1.2)	
5	Network logon succeeded	Logon Name	A client application successfully attached to the server	
6	Network logon failed	Logon Name	A client application failed to attach to the server	If unsuccessful logon messages occur frequently, use a network monitor to determine who is trying to connect. If security is enabled this message will appear for someone trying to connect with the wrong user name or password.

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
7	Security session opened		The security configuration utility has attached to the server.	
8	Security database read failed		When the server started up it could not read the security settings file.	This is a normal message on server startup if security has not been set up. If security should be set the file needs to be removed and security re-configured.
9	Modem default database read failed		When the server started up it could not read the default modem file wmodem.ini.	This file should exist in the working directory on the server computer (c:\campbellsci\loggernet\sys\cora). May indicate a permissions or configuration problem on the computer.
10	Modem custom database read failed		When the server started up it could not read the user customized modem settings file wmodem.cust.	If the user has not set up custom modem configurations, this file will not exist.
11	Clock check started		A clock check has been initiated. This clock check is not sent out to the station until the transaction is sent.	
12	Clock set	Device time before set; Server time;	The device clock has been set.	
13	Clock checked	Datalogger time	The datalogger clock has been checked.	
14	Clock check failed	Reason code: 3. Communication failure 4. Invalid datalogger security clearance 5. Invalid transaction number specified (already in use) 6. Communications are disabled for this device 7. The transaction was aborted by client request 8. The device is busy with another transaction	The clock check/set failed for the reason specified in the reason code.	Check the connections of the communication path to the datalogger, make sure the datalogger is connected and has power, check the security setting in the datalogger and in NetAdmin, check that communications are enabled in NetAdmin for all the devices in the path.

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
15	Starting BMP data advise transaction		A start data advise operation has been initiated. Data advise is not in place until the datalogger responds.	
16	Stopping BMP data advise transaction		A stop data advise operation has been initiated.	
17	BMP data advise transaction started		The message from the datalogger confirming the start of data advise has been received.	
18	BMP data advise transaction stopped		The message from the datalogger confirming the suspension of data advise has been received.	
19	BMP data advise transaction failed		The attempt to start or stop a data advise with the datalogger has failed or the operation has timed out waiting for a response.	Check communications with the datalogger by trying to check the clock. If that fails follow the steps for message 14.
20	Hole detected	Table name; Beginning record number; Ending record number	A hole or missed records has been detected in the data coming from the datalogger.	The server will automatically try to collect the data if hole collection is enabled.
21	Hole collected	Table name; Beginning record number; Ending record number	The missing records specified have been collected from the datalogger.	
22	Hole lost	Table name; Beginning record number; Ending record number	The missing records have been overwritten in the datalogger.	
23	Hole collect start	Table name; Beginning record number; Ending record number	The hole collect request has been started. This message won't go to the datalogger until the BMP1 message is sent. (see message 104)	

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
24	Hole collect response received		The datalogger has returned the response to the hole collect request. This will contain either the data or state that the hole is lost.	
25	Hole collect failed		The hole collection request either timed out or a communication failure occurred.	Check communications with the datalogger by trying to check the clock. If that fails follow the steps for message 14.
26	Data polling started		Data collection by polling started.	
27	Data polling complete		Data collection by polling completed	
28	Data polling failed		Data collection by polling failed due to communication failure or a timeout.	Check communications with the datalogger by trying to check the clock. If that fails, follow the steps for message 14.
29	Directed data query start		A user initiated query has been started.	
30	Directed data query continue		The requested data in the directed query could not fit in one block and the next part is being requested.	
31	Directed data query complete		The user requested data has been received by the server.	
32	Directed data query failed		The directed query request failed.	
33	Getting logger table definitions		The server is getting the table definitions from the datalogger.	Getting the datalogger table definitions will erase any data in the data cache.
34	Received logger table definitions		The server has received the datalogger table definitions.	
35	Failed to get logger table definitions		The request to get table definitions has failed.	

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
36	Logger table definitions have changed		The server has detected a change in the table definitions in the datalogger.	A change in table definitions indicates that the datalogger program may have changed. Before updating table definitions make sure the needed data in the data cache has been saved to a file if desired.
37	Updating BMP1 network description		The network description in the RF base is being updated to reflect changes in collection schedule or stations to collect.	
38	BMP1 network description update complete		The RF base has acknowledged the network description update.	
39	BMP1 network description update failed		The network description update to the RF base has either timed out or communication has failed.	Check the connections from the PC to the RF base.
40	Datalogger message	Severity (S for Status, W for Warning, F for Fault); Message text.	This is a message that has been generated by the datalogger (or in some cases the RF base on behalf of the datalogger).	Datalogger warning and fault messages should be investigated using the datalogger operators manual or contacting an applications engineer at Campbell Scientific.
41	Records received	Table name; Beginning record number; Ending record number	Datalogger records have been received and stored in the data cache.	
42	A datalogger transaction has timed out	Time out period in milliseconds	The server has waited longer than the allotted time for the expected response to a transaction.	Determine the reason for the timeout. This is usually due to a problem with the communications path between the PC and the datalogger.
43	Terminal emulation transaction started		Terminal emulation message has been sent to the datalogger.	

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
44	Terminal emulation transaction complete		Terminal emulation response message has been received from the datalogger.	
45	Terminal emulation transaction failed		The expected terminal emulation response from the datalogger was not received.	
46	Set variable started		The message to set an input location, flag or port has been sent to the datalogger.	
47	Set variable complete		The datalogger has acknowledged the set of an input location, flag or port.	
48	Set variable failed		The datalogger failed to acknowledge the set variable message.	
49	Table resized		The size of the table storage area in the data cache has been changed.	If the table is made smaller the oldest data will be lost.
50	Program file send start		The server is sending a program to the datalogger. The actual program segments will appear as BMP1 message type 4.	
51	Program file send status		The datalogger has received the program segment.	
52	Program file send complete		The datalogger has compiled the program.	
53	Program file send failed		The datalogger did not acknowledge the receipt of the program, the program did not compile, or communications failed with the datalogger.	If the program did not compile check the error messages. Otherwise, check communications with the datalogger by trying to check the clock. If that fails, follow the steps for message 14.

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
54	Program file receive start		The server is requesting the datalogger program. The actual program segments will appear as BMP1 message type 5.	
55	Program file receive status		A program segment has been received.	
56	Program file receive complete		The datalogger program has been received from the datalogger.	
57	Program file receive failed		The datalogger failed to send the program or communications with the datalogger failed.	Check communications with the datalogger by trying to check the clock. If that fails, follow the steps for message 14.
58	Collection schedule: normal		This is an advisory message that the normal data collection schedule is active.	
59	Collection schedule: primary retry		A normal data collection has failed and data collection will be attempted at the primary retry interval.	Determine the reason for communication failure. Temporary communication problems may cause the collection state to change between normal and primary.
60	Collection schedule: secondary retry		The number of primary retries specified has passed and data collection will be attempted at the secondary retry interval.	
61	Collection schedule suspended		The scheduled data collection has been turned off or suspended because communication is disabled or table definitions have changed.	



Code	Message Text	Message Parameters	Message Meaning	User Response to Message
62	Primary retry collection attempt failed		Data collection on the primary data collection interval failed.	Check communications with the datalogger by trying to check the clock. If that fails, follow the steps for message 14.
63	Secondary retry collection attempt failed		Data collection on the secondary data collection interval failed.	Check communications with the datalogger by trying to check the clock. If that fails, follow the steps for message 14.
64	Device restore from file succeeded		On server startup a device previously entered in the network map has been restored.	
65	Device restore from file failed		On server startup a device in the network map could not be restored.	This is an indication that the configuration file has been corrupted. Check the network map and the computer file system.
66	Device save to file succeeded		The update to the device configuration file was successful.	
67	Device save to file failed		The update to the device configuration file failed.	This may be due to a problem with directory permissions or a corrupted directory.
68	Packet delivery failed	Fault code: 1. Incompatible BMP1 device or malformed packet 2. Routing failure {unrecognized station number} 3. Temporarily out of resources 4. Link failure	This is a message from the RF base indicating that a BMP1 message didn't make it to the data logger.	Codes 1 and 3 are rare. If ever seen contact an application engineer at Campbell Scientific. Code 2 indicates that the RF base has lost the network map and doesn't know how to route the message. The server automatically resends the network map. Code 4 is an indication that the RF base was not able to communicate with the RF modem attached to the datalogger. These will happen occasionally as part of

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
				normal operations. Frequent occurrences indicate that the radio, antenna, connectors and RF link be reviewed.
69	Unexpected change in datalogger table definitions		As part of data collection the server has detected a change in the datalogger's table definitions.	A change in table definitions indicates that the datalogger program may have changed. This will suspend data collection and warnings will be shown in the Control Panel and NetAdmin. Data Collection can only be restored by updating table definitions. Before updating table definitions make sure the needed data in the data cache has been saved to a file if desired. See section 7.4.3
70	A device setting value has changed	Setting Identifier; Client's logon name; New value of the setting	A client has changed one of the device configuration settings.	
71	A LgrNet setting value has changed	Setting Identifier; Client's logon name;	A client has changed one of the server configuration settings.	
72	Client defined message	Client defined message	These messages are placed in the transaction log by client applications. The message should indicate which client entered the message.	
73	Socket listen failed		Indicates an error in the computer system that prevents the server from listening for client connections on a socket.	This is a rare error and results in a problem with the computer operating system. If rebooting the computer does not clear the error, contact an application engineer.

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
74	Device renamed		The name of a device in the network was changed.	
75	Logger locked		This message indicates the start of a transaction such as terminal emulation that will tie up the datalogger preventing other operations.	
76	Logger unlocked		The transaction blocking datalogger access has completed.	
77	Null program sent		The server has sent a null program to get an older datalogger (CR7X or 21X) out of keyboard emulation mode.	
78	Server started	The server version	The server has been started.	
79	Server shut down		The server is being shut down	If a new “server started” message is seen without the shut down message before it, this is an indication that the server or the PC crashed without exiting properly.
80	Collect area initialized	Collect area name	A data cache collect area has been created.	
82	Collect area removed		A data cache collect area has been removed	
83	LgrNet restore failed		On server startup the network description file, csilgrnet.dnd, could not be read.	The network setup and configuration will have to be restored from a backup or re-entered. Try to determine what corrupted or removed the network description file.
84	Security manager restore failed		On server startup the security manager database could not be restored.	There is a problem with the computer or operating system. If rebooting the machine does not get it working get help from someone who can troubleshoot computer problems.

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
85	Data restore failed		On server startup the data broker data storage area could not be created.	This is a computer problem. The files are either not present or are corrupted. See notes for message 83.
86	Manual poll transaction started	Client logon name	The listed client is starting a manual poll operation according to the scheduled collection settings. A manual poll is initiated from the “Update Data Cache” button on the Control Panel data retrieval tab.	
87	Manual poll transaction complete		The manual poll operation has received the data from the datalogger.	
88	Manual poll aborted		The manual poll operation was stopped or failed to complete due to communications failure or a timeout.	Check communications with the datalogger by trying to check the clock. If that fails, follow the steps for message 14.
89	Selective manual poll begun	Collect area name	A user specified poll has been started for one of the datalogger collect areas.	
90	Selective manual poll complete	Collect area name	The user specified manual poll has completed.	
91	Selective manual poll aborted	Collect area name	The user specified manual poll failed.	Check communications with the datalogger by trying to check the clock. If that fails, follow the steps for message 14.
92	Polling started on collect area	Collect area name	Data has been requested for the specified collect area. This message is always associated with another message indicating whether this is scheduled, manual or selective manual polling.	Collect areas can be table for table mode dataloggers, final storage areas, ports and flags, or input locations.

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
93	Collect area poll data	Collect area name	Data has been received from an array based datalogger for the specified collect area.	
94	Collect area polling complete	Collect area name	Data collection for the specified collect area has successfully completed.	
95	Collect area polling failed	Collect area name	Data collection for the specified collect area failed.	Check communications with the datalogger by trying to check the clock. If that fails, follow the steps for message 14.
96	Scheduled polling begun		Scheduled data collection has started.	
97	Scheduled polling succeeded		Scheduled data collection has completed.	
98	Scheduled polling failed		Scheduled data collection failed.	Check communications with the datalogger by trying to check the clock. If that fails, follow the steps for message 14.
99	Collect area first poll		This message is posted either the first time data is collected for a collect area, or holes were lost for the datalogger.	If this is not the first poll for the collect area, this message indicates that data that had been stored in the datalogger was lost before it could be collected.
100	Table mount failed	Table name; Operating system information regarding the failure	The server was not able to create a data collection area from the stored table configuration file or new table definitions. This could be the result of trying to create table files that are too large for the computer system.	Check the computer operating system integrity. Verify that the LoggerNet system configuration files exist and the directory has not been corrupted.
101	Add record failed	Table name; Beginning record number; End record number; A reason for the failure	The server was not able to write data records to the data storage area.	This indicates a problem writing to files on the computer hard disk. Verify write permissions are set and that there is sufficient space left on the disk.

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
102	Collect area skipped warning	Collect area name	The specified collect area was skipped because the associated table has not been initialized by the server yet.	During system startup this is a normal message. If it occurs at other times contact an application engineer.
103	Collect area skipped error	Collect area name	The specified collect area was skipped because the server could not initialize the associated table.	See message 100
104	BMP1 packet sent	The packet message type code: 0 Packet Delivery Fault Notification 1 Status/Warning/Fault Notification 2 Network Description Transaction 3 Clock Check/Set Transaction 4 Program Down-load Transaction 5 Program Up-load Transaction 7 Data Advise Command Transaction 8 Data Advise Notification Packet 9 Hole Collection Command Transaction 10 Control Command (Set Variable) Transaction 11 User I/O Transaction (Terminal Mode) 12 Memory Image Down-load Transaction 13 Memory Image Up-load Transaction 14 Get Table Definitions Transaction 15 RF Test Transaction 16 Communication Status Notification	The specified BMP1 packet was sent to the serial communication interface. The number specifies the type of message that was sent.	

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
105	BMP1 packet received	<p>The packet message type code:</p> <p>0 Packet Delivery Fault Notification</p> <p>1 Status/Warning/Fault Notification</p> <p>2 Network Description Transaction</p> <p>3 Clock Check/Set Transaction</p> <p>4 Program Down-load Transaction</p> <p>5 Program Up-load Transaction</p> <p>7 Data Advise Command Transaction</p> <p>8 Data Advise Notification Packet</p> <p>9 Hole Collection Command Transaction</p> <p>10 Control Command (Set Variable) Transaction</p> <p>11 User I/O Transaction (Terminal Mode)</p> <p>12 Memory Image Down-load Transaction</p> <p>13 Memory Image Up-load Transaction</p> <p>14 Get Table Definitions Transaction</p> <p>15 RF Test Transaction</p> <p>16 Communication Status Notification</p>	The specified BMP1 packet was received over the serial communications link. The number indicates the type of message received.	
106	Data file output failed		Data collected from a datalogger could not be written to the data output file.	Check that there is space available on the hard disk and that write permissions allow the server to write the data output files.
107	Max time on-line exceeded	The amount of time the device was connected, in milliseconds	A client kept the communication link on-line longer than the specified max time on-line.	

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
108	Table reset	The name of the table that was reset; The account name of the logged in client	The name of a table was changed at the request of a client. On CR5000 and CR9000 loggers this is a reset for the table in the datalogger and on the PC.	
109	Collect schedule reset	The account name of the logged in client	The collection schedule was reset by the indicated client.	
110	Collect area setting changed	The name of the collection area; The setting identifier for the setting that was changed; The new value of the setting; The account name of the logged in client	One of the settings for the specified collect area was changed. The identifiers for the setting can be found in Cora Script help.	
111	PakBus route added		A new PakBus route has been added to the routing table.	
112	PakBus route lost		A PakBus route has been lost and will be removed from the routing table.	
113	PakBus station added		A new PakBus station was added to the network.	
114	Datalogger callback begin		A datalogger has called in to the server starting the call-back response.	

#### D.1.3.4 Object State Log Format

The object state log includes two fields in addition to the time stamp and device name:

**Object Name** - The name of the object from which the message is being generated. Typically this will be the name of an object method.

**Description** - Any extra information associated with the event.



**Object State Log Example**

```

"2002-02-07 12:04:29.003","CR10T","Logger::on_selective_manual_poll_cmd","66","9557"
"2002-02-07 12:04:29.003","CR10T","Dev::cmdAdd","BMP1 Low Level Serial","3"
"2002-02-07 12:04:29.003","CR10T","Dev::onNextCommand","Executing command","BMP1 Low
Level Serial","3"
"2002-02-07 12:04:29.003","CR10T","Bmp1::Bmp1Tran::get_next_out_message","hole collection"
"2002-02-07 12:04:29.233","CR10T","Dev::cmdFinished","BMP1 Low Level Serial"
"2002-02-07 12:04:30.004","CR10T","Logger::on_selective_manual_poll_cmd","66","9558"
"2002-02-07 12:04:30.004","CR10T","Dev::cmdAdd","BMP1 Low Level Serial","3"
"2002-02-07 12:04:30.004","CR10T","Dev::onNextCommand","Executing command","BMP1 Low
Level Serial","3"
"2002-02-07 12:04:30.004","CR10T","Bmp1::Bmp1Tran::get_next_out_message","hole collection"
"2002-02-07 12:04:30.234","CR10T","Bmp1::Bmp1Tran::get_next_out_message","hole collection"
"2002-02-07 12:04:30.325","CR10T","Bmp1::Bmp1Tran::get_next_out_message","hole collection"
"2002-02-07 12:04:30.495","CR10T","Bmp1::Bmp1Tran::get_next_out_message","hole collection"
"2002-02-07 12:04:30.685","CR10T","Bmp1::Bmp1Tran::get_next_out_message","hole collection"
"2002-02-07 12:04:31.006","CR10T","Logger::on_selective_manual_poll_cmd","66","9559"
"2002-02-07 12:04:31.446","CR10T","Bmp1::Bmp1Tran::get_next_out_message","hole collection"
"2002-02-07 12:04:31.807","CR10T","Dev::shouldGetOffLine","communication disabled"
"2002-02-07 12:04:31.807","CR10T","Dev::cmdFinished","BMP1 Low Level Serial"
"2002-02-07 12:04:31.807","CR10T","Dev::onNextCommand","Ending"
"2002-02-07 12:04:31.837","ComPort_1","Dev::relDevice","CR10T"
"2002-02-07 12:04:31.837","CR10T","Dev::cmdFinished","BMP1 low level serial end"
"2002-02-07 12:04:31.837","CR10T","Dev::goingOffLine","Reset off-line counter"
"2002-02-07 12:04:31.837","ComPort_1","Dev::onNextCommand","Ending"
"2002-02-07 12:04:31.867","ComPort_1","Dev::cmdFinished","Comm::Root::end_command_type"
"2002-02-07 12:04:31.867","ComPort_1","Dev::goingOffLine","Reset off-line counter"
"2002-02-07 12:04:31.867","ComPort_1","SerialProvider::execute","thread stopping"
"2002-02-07 12:04:31.907","CR10T","Dev::shutDown","Shutting down"
"2002-02-07 12:04:31.997","CR10T","Dev::shutDown","Shutting down"
"2002-02-07 12:04:32.047","ComPort_1","Dev::shutDown","Shutting down"
"2002-02-07 12:04:32.047","ComPort_1","Dev::shutDown","Shutting down"

```



# Appendix E. Using RTDM with LoggerNet

---

## E.1 RTDM

Real Time Data Monitoring (RTDM) software provides graphical displays of data from data files and automatically updates those displays as the data file is updated. RTDM requires file formats similar to those generated by Campbell Scientific's PC208W software. LoggerNet users can generate files in those formats using Data Table Monitor, a utility accessed from Start | Programs | LoggerNet | Tools | Data Table Monitor. RTDM expects files in an array-based format, an example of which appears below:

```
101,1999,120,1724,19.000,0,0,75.46,75.02,532.5,12,0.668
101,1999,120,1724,20.000,0,1,75.45,75.03,531.7,12,1.336
101,1999,120,1724,21.000,0,2,75.48,75.03,531.7,12,1.002
```

The fields in this format represent:

Array ID, Year, Day of Year, HourMinutes, Seconds, Filemark #, Record #, Data values...

RTDM checks a data file periodically and updates its display with new data if any has been added. Display options include graphs, charts, digital values, and other graphic components to present the data.

Data Table Monitor can only output the data from a single table on one datalogger to an output file. If you need to monitor the data from more than one table you can run multiple copies of the Data Table Monitor application. Each will create its own file. RTDM can read multiple files for the same display. See the section 13.7 for directions to start and set up Data Table Monitor.

---

### CAUTION

One caution about the data file created by Data Table Monitor—there are no limits to size or longevity. If you plan to use this export feature on a regular basis, make sure to either restart Data Table Monitor or delete the files periodically. The data file can be easily deleted and restarted by clicking on the Start Export button.

---

Data Table Monitor should be running on the same computer where the RTDM Application is running to avoid trying to keep the output file open across the network connection.

## E.2 Setting up RTDM

For detailed descriptions of the setup and configuration of RTDM refer to the Getting Started section in RTDM's Help and documentation included with RTDM.

This section is included to help with setup considerations that are different than the normal RTDM operation.

On the Data Source Tab in the Data Source Editor on the RTDM form, enter the filename and path for the datafile being created by TableMon. Select an update rate that is appropriate for the data being monitored.

Table data files include final storage labels in the table definition kept in the datalogger. The FSL file created by Edlog for table-based dataloggers does not contain array ID information and cannot be used by RTDM.

To see the sample data on the bottom you need to add an array and change the selected array ID to 101. The only array ID provided by Data Table Monitor is 101. You should now be able to see a sample of the data if the selected file has data in it. This will be very helpful in the next step.

On the Data Values tab you will need to create a new Item for each data value you are monitoring. When you click on the New Item button an item with the name FIELD is created. Change the name of the item to a meaningful label and change the selected array index to point to the data. As the Index is changed the related position in the record is highlighted on the display at the bottom.

Select the Output Series to link the data to the graph object.

To get the timestamp information from the record go to the Date/Time Definitions tab. Set up the year, day, hours-minutes, and seconds to be read from the array, in each case setting the Index value to point to the appropriate field.

# ***Appendix F. CoraScript***

---

## **F.1 CoraScript Fundamentals**

CoraScript is a command line interpreter that reads its commands as text from its standard input device and writes the results of those commands as text to its standard output device. This style of input and output makes it possible to externally control the LoggerNet server operation using input and output redirection. It also makes it possible to string together commands in scripts that can be executed from the command line.

The CoraScript command interpreter is started by executing the program file Cora\_Cmd.exe in a command prompt environment. CoraScript is also available through the LoggerNet program files start menu under “Tools”. When the script processor starts up it will output a response:

CoraScript 1,1,1,30

The numbers indicate the version number of the current CoraScript.

CoraScript is a batch processing interpreter. It treats its input (from the standard input device) as a sequence of commands that are processed serially from the first to the last. As a command is processed, the results are written to the standard output device. A command is defined as the text up to a semicolon (;). The semicolon tells Cora Script that the command is complete and ready to execute.

The flexibility of the commands available within CoraScript and the independence from user interface considerations make CoraScript a valuable tool for testing, troubleshooting, and automating LoggerNet server operations.

---

**NOTE**

Because the commands available in CoraScript operate directly on the LoggerNet server and not through a user interface, there are no confirmation prompts for critical operations. Care should be exercised in using the commands to avoid interrupting normal server operations.

---

There is an extensive on-line help file available for CoraScript. To bring up the help file, type “help;” on the command line. (Make sure and include the semicolon ‘;’ at the end and leave off the quotes.) Read through the directions and try some examples.

## F.2 Useful CoraScript Operations

The following sections provide an overview of some common and very useful commands available with CoraScript. Some rules about formatting input and interpreting the responses:

- Always end the command with the semicolon (;) character. CoraScript uses the semicolon to mark the end of the command input and will not process anything until it is detected.
- Command parameters are often set using a combination of the parameter name and the value in this format: `--name=login`. Be sure to read the help for the command you are using.
- A response preceded by a plus sign (+) indicates that the command was successfully processed.
- A response preceded by a minus sign (-) indicates that the command failed and will usually be accompanied by a reason for the failure.

### F.2.1 Connecting to the LoggerNet Server

Before you can execute any commands a connection to the LoggerNet server must be established. The connect command sets up the server context in which subsequent commands will operate until the end of the script is reached or another connect command is processed. The following segment shows a sample use of the connect command:

```
connect      # the name of the command
localhost    # specifies the server's host address
--name="bilbo" # specifies the logon name (optional)
--password={baggins} # specifies the password (optional)
;            # marks the end of the command
```

This command would normally appear on one line as follows:

```
connect localhost --name="bilbo" --password={baggins} ;
```

For a more detailed explanation of the interpretation of the symbols and syntax refer to the CoraScript help.

### F.2.2 Checking and Setting Device Settings

The current value of any of the configuration settings for a device is available using the *get-device-setting* command. Devices are referred to by the name as shown in the NetAdmin network map and settings are referenced by number. The setting numbers and their meanings are described in the CoraScript help.

To set the configuration setting for any device, use the *set-device-setting* command. As with *get-device-setting* the device is referred to by name and the setting by number.

### F.2.3 Creating and using a Network Backup Script

One of the most useful commands for network maintenance is *create-backup-script*. Executing this command will create a script file of CoraScript commands that can be used to rebuild the existing datalogger network and configure all of the device settings. It will not save or restore any data that has been collected but can serve as a means to restore the network map and settings.

To create the script enter the command along with a filename and path for the script file:

```
create-backup-script c:\temp\mynetwork.script;
```

This will write all of the commands necessary to build and configure the network map into the file *mynetwork.script*. To restore the network map, remove all of the devices in the current map and then from a command prompt enter:

```
Cora_cmd.exe < c:\temp\mynetwork.script
```

The less than symbol ( < ) is redirection that tells the CoraScript executable to read commands from the specified file. For more information about this command see the CoraScript help.

### F.2.4 Hole Management

There are several commands available with CoraScript to manage the hole collection process. These functions are not available through the standard user interface applications. See the CoraScript help for details about these commands.

List-holes

Purge-holes

Delete-holes

### F.2.5 Scripting CoraScript Commands

To automate network processes, scripts can be created with other scripting language tools that would call the CoraScript interpreter, and send commands to the LoggerNet server. This provides an alternate means of controlling data collection, hole collection and maintenance functions such as clock check and set.





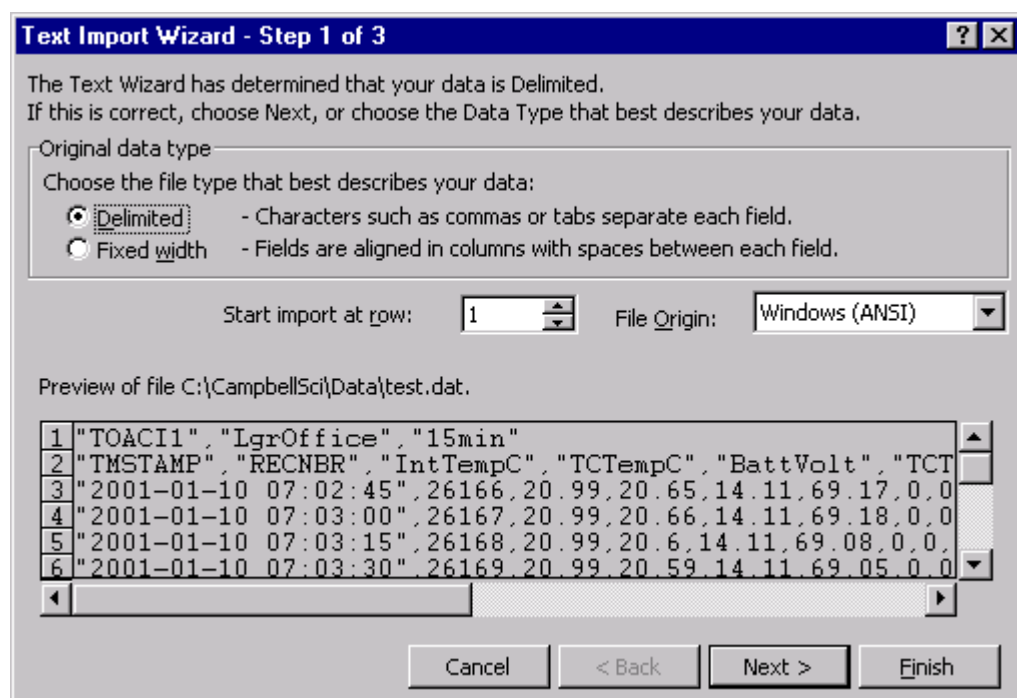
# ***Appendix G. Importing Files into Excel***

Files saved to disk using the Baler or the Control Panel Retrieve to File, can be imported into a spreadsheet program for analysis or manipulation. Instructions are given below for importing a comma separated file into Microsoft Excel.

From the Excel menu, select File | Open. Browse for the \*.dat file that you want to import. Excel will recognize the file as not being in an xls format, and will invoke the Text Import Wizard. The Text Import Wizard consists of three steps, each having its own window.

## **Step 1 of 3**

Select the Delimited option from the Original Data Type group box. Using the arrow buttons to the right of the Start Import at Row field, select the number of the first row of data to be imported. If your data file has headers included, you can import those or start the import at the first row of data (typically row 3). Select the Next button.



## **Step 2 of 3**

From the Delimiters group box, select Comma. The Comma option directs Excel to place each data value, which is separated by a comma, into a separate column.

From the Text Qualifiers list box, select " (quotes). Select the Next button.

**Text Import Wizard - Step 2 of 3**

This screen lets you set the delimiters your data contains. You can see how your text is affected in the preview below.

**Delimiters**

☐ Tab    ☐ Semicolon    ☒ Comma  
☐ Space    ☐ Other:

☐ Treat consecutive delimiters as one

Text Qualifier:

**Data preview**

TOACI1	LgrOffice	15min	TCTempC	BattVolt	T
TMSTAMP	RECNR	IntTempC			
2001-01-10 07:02:45	26166	20.99	20.65	14.11	6
2001-01-10 07:03:00	26167	20.99	20.66	14.11	6
2001-01-10 07:03:15	26168	20.99	20.6	14.11	6
2001-01-10 07:03:30	26169	20.99	20.59	14.11	6

Cancel    < Back    Next >    Finish

Step 3 of 3

A quick look at the columns of data is provided in the Data Preview group box.

To complete the import, select the Finish button.

**Text Import Wizard - Step 3 of 3**

This screen lets you select each column and set the Data Format.

'General' converts numeric values to numbers, date values to dates, and all remaining values to text.

**Column data format**

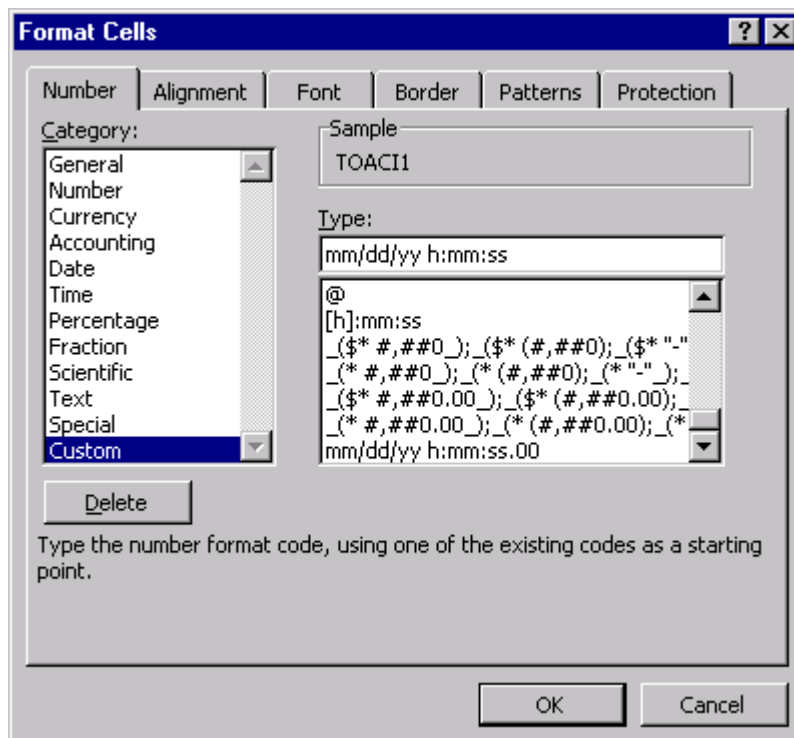
☒ General  
☐ Text  
☐ Date: MDY  
☐ Do not import column (Skip)

**Data preview**

General	General	General	General	General	G
TOACI1	LgrOffice	15min	TCTempC	BattVolt	T
TMSTAMP	RECNR	IntTempC			
2001-01-10 07:02:45	26166	20.99	20.65	14.11	6
2001-01-10 07:03:00	26167	20.99	20.66	14.11	6
2001-01-10 07:03:15	26168	20.99	20.6	14.11	6
2001-01-10 07:03:30	26169	20.99	20.59	14.11	6

Cancel    < Back    Next >    Finish

Excel will display the date in a default format showing the year, month and day along with the hour and minute. To display the seconds select the column containing the time stamp and select Format | Cells from the menu. On the following form click on custom and enter the type as shown. Then click on OK.



A sample of the imported data appears below.

TOACI1	LqrOffice	15min				
TMSTAMP	RECNBR	IntTempC	TCTempC	BattVolt	TCTempF	TestSet1
01/10/01 7:02:45	26166	20.99	20.65	14.11	69.17	0
01/10/01 7:03:00	26167	20.99	20.66	14.11	69.18	0
01/10/01 7:03:15	26168	20.99	20.6	14.11	69.08	0
01/10/01 7:03:30	26169	20.99	20.59	14.11	69.05	0
01/10/01 7:03:45	26170	20.99	20.61	14.11	69.1	0
01/10/01 7:04:00	26171	20.99	20.61	14.11	69.1	0
01/10/01 7:04:15	26172	20.99	20.62	14.11	69.11	0
01/10/01 7:04:30	26173	20.99	20.65	14.11	69.17	0
01/10/01 7:04:45	26174	20.99	20.65	14.11	69.17	0

