

# INSTRUCTION MANUAL



**LoggerNet**

Version 3.1

Revision: 12/04

Copyright (c) 1999-2004  
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 make a copy of this software on a second computer 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.



# ***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. The recommended minimum hardware for LoggerNet is a 300 MHz Pentium II processor with 64 megabytes of RAM and a screen area of at least 800x600. LoggerNet uses the features of Windows NT, 2000, or XP that maximize the reliability of unattended scheduled data collection and multitasking application programs. LoggerNet may be run successfully on Windows 95, 98, or ME if the user limits the number of screens open at any one time.

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.

# LoggerNet Table of Contents

---

PDF viewers note: These page numbers refer to the printed version of this document. Use the Adobe Acrobat® bookmarks tab for links to specific sections.

<b>Preface — What’s New in LoggerNet 3.1?</b> .....	<b>xi</b>
<b>1. System Requirements</b> .....	<b>1-1</b>
1.1 Hardware and Software.....	1-1
1.2 Configuration of TCP/IP Services .....	1-1
<b>2. Installation, Operation and Backup Procedures</b> ....	<b>2-1</b>
2.1 CD-ROM Installation.....	2-1
2.2 Upgrade Notes .....	2-1
2.3 LoggerNet Operations and Backup Procedures .....	2-2
2.3.1 LoggerNet Directory Structure and File Descriptions.....	2-2
2.3.1.1 Program Directory .....	2-2
2.3.1.2 Working Directories .....	2-2
2.3.2 Backing up Data.....	2-4
2.3.3 Loss of Computer Power.....	2-5
2.3.4 Program Crashes .....	2-5
2.3.5 Restoring from Backup .....	2-6
2.4 Special Note on Windows XP Service Pack 2 .....	2-6
<b>3. Introduction</b> .....	<b>3-1</b>
3.1 What is LoggerNet? .....	3-1
3.1.1 What Next? .....	3-1
3.2 Overview of Major LoggerNet Functions and Associated Software Applications.....	3-2
3.2.1 The Heart of it All — LoggerNet Toolbar .....	3-2
3.2.1.1 Toolbar Customization .....	3-2
3.2.1.2 Toolbar Menus .....	3-3
3.2.1.3 Command Line Arguments .....	3-3
3.2.2 Setting Up Datalogger Communication Networks.....	3-4
3.2.3 Real Time Tools.....	3-4
3.2.4 Network Status and Problem Solving .....	3-5
3.2.5 Creating and Editing Datalogger Programs.....	3-5
3.2.6 Working with Data Files .....	3-6
3.2.7 Automating Tasks with Task Master.....	3-7
3.2.8 Managing External Data Storage Devices.....	3-7
3.2.9 Optional Client Products Compatible with LoggerNet .....	3-7
3.2.9.1 LoggerNetData .....	3-7
3.2.9.2 Data Display Clients.....	3-8
3.2.9.3 Baler .....	3-8
3.2.9.4 PC-OPC, CsiOPC Server.....	3-8
3.2.9.5 Software Development Kit .....	3-8
3.3 Getting Help for LoggerNet Applications.....	3-9

<b>4. Setting up Datalogger Networks .....</b>	<b>4-1</b>
4.1 EZSetup Wizard .....	4-1
4.2 Setup Screen .....	4-3
4.2.1 Adding Devices to the Network .....	4-4
4.2.2 Applying Changes, Undo and Redo .....	4-5
4.2.3 Renaming Network Devices .....	4-6
4.2.4 Device Settings .....	4-6
4.2.4.1 ComPort .....	4-6
4.2.4.2 IPPort (Internet Protocol Serial Port) .....	4-7
4.2.4.3 TAPIPort (Telephony API) .....	4-8
4.2.4.4 Datalogger .....	4-9
4.2.4.5 PhoneBase .....	4-17
4.2.4.6 PhoneRemote .....	4-18
4.2.4.7 RFBase .....	4-19
4.2.4.8 RFRemote .....	4-20
4.2.4.9 RF95T .....	4-20
4.2.4.10 RF95TPass .....	4-21
4.2.4.11 MD9 Base .....	4-22
4.2.4.12 MD9 Remote .....	4-23
4.2.4.13 RF400 .....	4-23
4.2.4.14 RF400 Remote .....	4-24
4.2.4.15 Generic Modem .....	4-25
4.2.4.16 PakBusPort .....	4-26
4.2.5 Setting Up Scheduled Data Collection .....	4-28
4.2.5.1 Data Collection Scheduling Considerations .....	4-28
4.2.5.2 Intervals .....	4-28
4.2.5.3 Setting Up Scheduled Data Collection .....	4-30
4.2.6 Setting the Clock .....	4-31
4.2.7 Sending a Program to the Datalogger from Setup .....	4-32
4.2.8 Setup's Options Menu .....	4-33
4.2.8.1 Convert from PC208W Network .....	4-33
4.2.8.2 LoggerNet PakBus Settings .....	4-34
4.2.8.3 LoggerNet Clock Settings .....	4-35
4.2.8.4 LoggerNet IP Manager Settings .....	4-35
4.3 Device Configuration Utility .....	4-36
<b>5. Real-Time Tools .....</b>	<b>5-1</b>
5.1 The Connect Screen Toolbox .....	5-1
5.1.1 Connecting to the Datalogger -- or Not .....	5-1
5.1.2 Data Collection .....	5-
5.1.2.1 Collect Now .....	5-3
5.1.2.2 Custom Collection .....	5-3
5.1.3 Ports and Flags .....	5-8
5.1.4 Datalogger Clock .....	5-8
5.1.5 Program Management .....	5-9
5.1.5.1 Sending a Datalogger Program .....	5-9
5.1.5.2 CR200 Series Programs .....	5-10
5.1.5.3 Retrieving Datalogger Programs .....	5-10
5.1.6 Program Association .....	5-11
5.1.7 Data Displays .....	5-11
5.1.7.1 Data Display Limitations .....	5-12
5.1.7.2 Numeric Display Screens .....	5-12
5.1.7.3 Graphical Display Screens .....	5-16

5.1.8	File Control for CR5000, CR1000, and CR9000 Dataloggers ..	5-22
5.1.9	Terminal Emulator .....	5-24
5.1.10	Status Table Information (Table Dataloggers Only) .....	5-24
5.2	Real-Time Monitor and Control .....	5-25
5.2.1	Development Mode .....	5-25
5.2.1.1	The RTMC Workspace .....	5-27
5.2.1.2	Display Components .....	5-27
5.2.1.3	Functions Available from the RTMC Menus .....	5-29
5.2.1.4	Expressions .....	5-33
5.2.1.5	Remote Connection .....	5-36
5.2.2	RTMC Run-Time .....	5-37

## 6. Network Status and Resolving Communication

### Problems .....6-1

6.1	Status Monitor.....	6-1
6.1.1	Visual Status Indicators.....	6-2
6.1.2	Status Monitor Functions .....	6-2
6.1.2.1	Selecting Columns.....	6-2
6.1.2.2	Toggle Collection On/Off.....	6-7
6.1.2.3	Reset Device.....	6-7
6.1.2.4	Collect Now/Stop Collection.....	6-7
6.2	LogTool .....	6-7
6.2.1	Log Types .....	6-7
6.2.2	Using LogTool.....	6-8
6.2.3	Saving Logs to File .....	6-9
6.3	Comm Test.....	6-9
6.4	PakBus Graph .....	6-10
6.4.1	Selecting the PakBus Network to View .....	6-11
6.4.2	Dynamic and Static Links .....	6-12
6.4.3	Viewing/Changing Settings in a PakBus Datalogger.....	6-12
6.4.4	Right-Click Functionality.....	6-12
6.4.5	Discovering Probable Routes Between Devices .....	6-13
6.5	Troubleshooter.....	6-13
6.5.1	Status Information.....	6-14
6.5.2	Retrieving Information for a Datalogger.....	6-15
6.5.3	Archiving Troubleshooter Results .....	6-16
6.5.4	Other Tools in Troubleshooter .....	6-16

## 7. Creating and Editing Datalogger Programs .....7-1

7.1	Review of CSI Datalogger Models .....	7-1
7.2	Short Cut .....	7-2
7.2.1	Overview.....	7-2
7.2.2	Creating a Program Using Short Cut.....	7-3
7.2.2.1	Step 1 – Create a New File or Open Existing File.....	7-4
7.2.2.2	Step 2 – Choose Sensors to Monitor.....	7-5
7.2.2.3	Step 3 – Setup Output Tables .....	7-10
7.2.2.4	Step 4 – Convert the Program into the Format Required by the Datalogger .....	7-12
7.2.3	Short Cut Settings .....	7-13
7.2.3.1	Scan Interval.....	7-13
7.2.3.2	Program Security .....	7-13
7.2.3.3	Datalogger ID .....	7-14
7.2.3.4	Power-up Settings.....	7-14
7.2.3.5	Select CR200 Compiler .....	7-14
7.2.3.6	AC Noise Rejection.....	7-14

7.2.3.7	Font.....	7-15
7.2.3.8	Set Working Directory.....	7-15
7.2.4	Editing Programs Created by Short Cut .....	7-15
7.2.5	New Sensor Files.....	7-15
7.3	Edlog.....	7-16
7.3.1	Overview.....	7-16
7.3.1.1	Precompiler.....	7-16
7.3.1.2	Context-sensitive Help.....	7-16
7.3.1.3	Programming Efficiency.....	7-16
7.3.1.4	Input Location Labels.....	7-16
7.3.1.5	Final Storage Label Editor.....	7-17
7.3.1.6	Expression Compiler.....	7-17
7.3.2	Creating a New Edlog Program.....	7-17
7.3.2.1	Program Structure.....	7-19
7.3.2.2	Edlog File Types.....	7-20
7.3.2.3	Inserting Instructions into the Program.....	7-21
7.3.2.4	Entering Parameters for the Instructions.....	7-21
7.3.2.5	Program Comments.....	7-22
7.3.2.6	Expressions.....	7-23
7.3.2.7	Editing an Existing Program.....	7-28
7.3.2.8	Editing Comments, Instructions, and Expressions.....	7-29
7.3.2.9	Cut, Copy, Paste, and Clipboard Options .....	7-29
7.3.3	Library Files.....	7-29
7.3.4	Documenting a DLD File.....	7-29
7.3.5	Display Options.....	7-30
7.3.5.1	Graphical Toolbar.....	7-30
7.3.5.2	Renumbering the Instructions.....	7-31
7.3.5.3	Compress VIEW.....	7-31
7.3.5.4	Indentation.....	7-31
7.3.6	Input Locations.....	7-31
7.3.7	Entering Input Locations.....	7-32
7.3.8	Repetitions.....	7-32
7.3.9	Input Location Editor.....	7-33
7.3.10	Input Location Anomalies.....	7-34
7.3.11	Final Storage Labels.....	7-35
7.3.12	Datalogger Settings Stored in the DLD File.....	7-37
7.3.13	Program Security.....	7-37
7.3.13.1	Setting Passwords in the DLD.....	7-37
7.3.13.2	Disabling Passwords.....	7-37
7.3.14	Final Storage Area 2.....	7-38
7.3.15	DLD File Labels.....	7-38
7.3.15.1	Mixed-array Dataloggers.....	7-38
7.3.15.2	Table-Based Dataloggers.....	7-38
7.3.16	Power Up Settings/Compile Settings.....	7-39
7.3.17	Datalogger Serial Port Settings.....	7-39
7.3.18	PakBus Settings.....	7-40
7.3.18.1	Network.....	7-40
7.3.18.2	Beacon Intervals.....	7-40
7.3.18.3	Neighbor Filter.....	7-41
7.3.18.4	Allocate General Purpose File Memory.....	7-41
7.4	CRBasic Editor.....	7-41
7.4.1	Overview.....	7-41
7.4.2	Inserting Instructions.....	7-42
7.4.2.1	Parameter Dialog Box.....	7-42
7.4.2.2	Right Click Functionality.....	7-44
7.4.3	Toolbar.....	7-44
7.4.3.1	Compile.....	7-46
7.4.3.2	Templates.....	7-47

7.4.3.3	CRBasic Editor Options .....	7-47
7.4.3.4	Available Help Information .....	7-50
7.4.4	CRBasic Programming .....	7-51
7.4.4.1	Programming Sequence .....	7-51
7.4.4.2	Program Declarations .....	7-52
7.4.4.3	Mathematical Expressions .....	7-53
7.4.4.4	Measurement and Output Processing Instructions .....	7-53
7.4.4.5	Inserting Comments Into Program .....	7-54
7.4.4.6	Example Program .....	7-54
7.4.4.7	Data Tables .....	7-55
7.4.4.8	The Scan -- Measurement Timing and Processing .....	7-57
7.4.4.9	Numerical Entries .....	7-58
7.4.4.10	Logical Expression Evaluation .....	7-58
7.4.4.11	Flags .....	7-60
7.4.4.12	Parameter Types .....	7-60
7.4.4.13	Program Access to Data Tables .....	7-61
7.5	Transformer Utility .....	7-62
7.5.1	Transforming a File .....	7-62
7.5.2	Controls .....	7-65

## **8. Working with Data Files on the PC .....8-1**

8.1	View .....	8-1
8.1.1	Opening a File .....	8-1
8.1.1.1	Opening a Data File .....	8-1
8.1.1.2	Opening Other Types of Files .....	8-2
8.1.2	Final Storage Label (FSL) Files .....	8-2
8.1.3	Data View .....	8-3
8.1.3.1	Array Selection .....	8-4
8.1.3.2	Text View Options .....	8-4
8.1.3.3	Changing the Font .....	8-4
8.1.4	Graph Panel .....	8-5
8.1.4.1	Graph Options .....	8-6
8.1.5	Printing Options .....	8-7
8.1.5.1	Printing Text .....	8-7
8.1.5.2	Printing Graphs .....	8-7
8.1.6	Advanced Topics .....	8-7
8.1.6.1	Assigning Data Files to View .....	8-7
8.1.6.2	Array Definitions (Mixed-array dataloggers only) .....	8-8
8.2	Split .....	8-9
8.2.1	Functional Overview .....	8-9
8.2.2	Getting Started .....	8-10
8.2.3	Split Parameter File Entries .....	8-16
8.2.3.1	Input Files .....	8-16
8.2.3.2	Output Files .....	8-48
8.2.4	Help Option .....	8-54
8.2.5	Editing Commands .....	8-54
8.2.6	Running Split From a Command Line .....	8-54
8.2.6.1	Processing Alternate Files .....	8-54
8.2.6.2	Processing Multiple Parameter Files with One Command Line .....	8-55
8.2.6.3	Using Splitr.exe in Batch Files .....	8-55
8.2.6.4	Command Line Switches .....	8-55

<b>9. Automating Tasks with Task Master .....</b>	<b>9-1</b>
9.1 Task Master.....	9-1
9.1.1 Setup Tab .....	9-2
9.1.1.1 Adding Tasks.....	9-2
9.1.1.2 Logger Event Tasks .....	9-3
9.1.1.3 Scheduled Event Tasks .....	9-4
9.1.1.4 Define What the Task Does .....	9-4
9.1.2 Status Tab.....	9-5
9.1.3 Messages Tab .....	9-6
<b>10. Storage Module Software (SMS) .....</b>	<b>10-1</b>
10.1 Overview.....	10-1
10.2 Getting Started .....	10-2
10.2.1 The Setup Screen.....	10-3
10.2.2 Establishing Communications .....	10-4
10.2.2.1 Selecting a COM Port.....	10-4
10.2.2.2 Baud Rate .....	10-4
10.2.2.3 Via Datalogger/Target Address (SM192/716 and SM4M/SM16M Modules Only) .....	10-5
10.2.2.4 Connect/Disconnect.....	10-6
10.3 Status Information.....	10-7
10.3.1 Status Box/Update Status .....	10-7
10.3.2 Advanced Status Information .....	10-7
10.3.2.1 CSM1/MCR1 .....	10-7
10.3.2.2 SM192/SM716.....	10-8
10.3.2.3 SM4M/SM16M .....	10-10
10.4 Programs .....	10-10
10.4.1 Program Location .....	10-11
10.4.2 Clear .....	10-11
10.4.3 Store .....	10-12
10.4.4 Read.....	10-12
10.5 Data.....	10-12
10.5.1 File Format .....	10-13
10.5.1.1 Comma Separated.....	10-13
10.5.1.2 ASCII with Array IDs.....	10-13
10.5.1.3 As Stored .....	10-13
10.5.2 File Naming Options .....	10-14
10.5.2.1 File name .....	10-14
10.5.2.2 Auto Name Control.....	10-14
10.5.3 Show Card/Module Directory .....	10-15
10.5.4 Get All.....	10-16
10.5.5 Get New.....	10-16
10.5.6 Get One .....	10-16
10.6 Erase.....	10-16
10.6.1 Erase Data .....	10-16
10.6.2 Erase Data and Programs.....	10-7
10.6.3 Erase and Test Card/Module .....	10-17
10.7 The Menu Bar .....	10-17
10.7.1 File.....	10-17
10.7.2 Options .....	10-18
10.7.3 <u>D</u> ata .....	10-18
10.7.4 <u>T</u> ools.....	10-19
10.7.5 <u>H</u> elp.....	10-19
10.8 Abort.....	10-19

**11. Clients Installed with LoggerNet.....11-1**

11.1 CoraScript..... 11-1  
 11.1.1 CoraScript Fundamentals ..... 11-1  
 11.1.2 Useful CoraScript Operations ..... 11-2  
     11.1.2.1 Connecting to the LoggerNet Server ..... 11-2  
     11.1.2.2 Checking and Setting Device Settings..... 11-3  
     11.1.2.3 Creating and using a Network Backup Script..... 11-3  
     11.1.2.4 Hole Management ..... 11-3  
     11.1.2.5 Scripting CoraScript Commands..... 11-4

**12. Utilities Installed with LoggerNet.....12-1**

12.1 Device Configurator ..... 12-1  
 12.1.1 Overview..... 12-1  
 12.1.2 Main DevConfig Screen..... 12-2  
 12.1.3 Screens Applicable to All (or most) Devices ..... 12-3  
     12.1.3.1 Send OS Tab - Downloading an Operating System..... 12-3  
     12.1.3.2 Logger Control Tab ..... 12-5  
     12.1.3.3 Terminal Tab ..... 12-6  
     12.1.3.4 PakBus Dataloggers – Settings Editor Tab..... 12-7  
     12.1.3.5 Device-specific Screens and Deployment Tab ..... 12-8  
 12.2 TOB File Converter ..... 12-20  
 12.3 Resource Monitor ..... 12-20

**13. Optional Client Applications Available for  
 LoggerNet.....13-1**

13.1 Allowing Remote Connections to the LoggerNet Server..... 13-1  
 13.2 LoggerNet Data..... 13-2  
     13.2.1 Data Filer ..... 13-2  
         13.2.1.1 DataFiler Requirements..... 13-2  
         13.2.1.2 Using the DataFiler..... 13-2  
         13.2.1.3 The Collected Data ..... 13-5  
         13.2.1.4 Determining the Data Available in the Data Cache..... 13-5  
         13.2.1.5 Record Number Anomalies ..... 13-5  
         13.2.1.6 Communication Status..... 13-6  
 13.3 RTMC Run-time ..... 13-6  
 13.4 RTDM..... 13-6  
 13.5 Baler..... 13-6  
 13.6 OPC Server ..... 13-7  
 13.7 Software Development Kit ..... 13-7

**14. Implementing Advanced Communications  
 Links .....14-1**

14.1 Phone to RF ..... 14-1  
 14.1.1 Setup ..... 14-1  
 14.1.2 Operational Considerations ..... 14-2  
     14.1.2.1 Scheduled Data Collection ..... 14-2  
     14.1.2.2 Extra Response Time..... 14-2  
     14.1.2.3 RF Address ..... 14-2  
     14.1.2.4 Max Time Online ..... 14-2  
 14.1.3 Attaching a Datalogger to the RF Base ..... 14-2  
     14.1.3.1 Hardware Setup ..... 14-3

14.1.3.2 Network Setup in LoggerNet.....	14-3
14.2 Phone to MD9 .....	14-3
14.2.1 Setup.....	14-3
14.2.2 Operational Considerations .....	14-4
14.2.2.1 Scheduled Data Collection.....	14-4
14.2.2.2 MD9 Addresses .....	14-4
14.2.2.3 Extra Response Time .....	14-4
14.2.2.4 Max Time Online.....	14-4
14.2.2.5 Grounding.....	14-5
14.3 TCP/IP to RF.....	14-5
14.3.1 Setup.....	14-5
14.3.2 Operational Considerations .....	14-5
14.3.3 Special Considerations .....	14-6

## **15. Troubleshooting Guide.....15-1**

15.1 What's Changed?.....	15-1
15.2 LoggerNet Server Problems .....	15-1
15.2.1 Starting LoggerNet and Connecting to the Server.....	15-1
15.2.2 Socket Errors .....	15-2
15.2.3 Data Collection Issues .....	15-4
15.3 Application Screen Problems .....	15-4
15.4 General Communication Link Problems .....	15-4
15.5 Terminal Emulator to Test Communications .....	15-5
15.6 RF Communication Link Issues .....	15-9
15.6.1 Checking RF Components and Connections .....	15-9
15.6.2 RF Signal Strength Testing .....	15-10
15.6.3 Troubleshooting with Attenuation Pads .....	15-11
15.7 Using Data Table Monitor .....	15-14
15.8 Troubleshooting PakBus Communications .....	15-18

## ***Appendices***

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

### **B. Campbell Scientific File Formats ..... B-1**

B.1 PC File Data Formats .....	B-1
B.1.1 Comma Separated .....	B-1
B.1.2 ASCII Printable .....	B-2
B.1.3 TOAC11 .....	B-2
B.1.4 TOA5 .....	B-3
B.1.5 TOB1 .....	B-3
B.2 Datalogger Data Formats.....	B-5
B.2.1 TOB2 or TOB3 .....	B-5
B.3 Binary Data Value Types .....	B-6
B.3.1 FP2 (2 Byte Low Resolution Format).....	B-7
B.3.2 FP4 (4 Byte High Resolution Format).....	B-7
B.3.3 IEEE4 .....	B-7
B.3.4 IEEE8 .....	B-7
B.4 Converting Binary File Formats .....	B-7
B.4.1 Split.....	B-7
B.4.2 View.....	B-7
B.4.3 TOB Convert .....	B-8
B.4.4 TOB32.EXE .....	B-8

<b>C. Table-Based Dataloggers .....</b>	<b>C-1</b>
C.1 Memory Allocation for Final Storage .....	C-1
C.1.1 CR10X-TD Family Table-Based Dataloggers.....	C-1
C.1.2 CR5000/CR1000/CR9000 Memory for Programs and Data Storage.....	C-2
C.1.3 CR200 Series Dataloggers.....	C-3
C.2 Converting an Array-Based Program to a CR10X-TD Table-Based Program using Edlog .....	C-3
C.2.1 Steps for Program Conversion.....	C-4
C.2.2 Program Instruction Changes .....	C-4
C.3 Table Data Overview .....	C-5
C.4 Default Tables.....	C-7
<b>D. Software Organization .....</b>	<b>D-1</b>
D.1 LoggerNet/Client Architecture.....	D-1
D.2 LoggerNet Server Data Cache .....	D-1
D.2.1 Organization.....	D-1
D.2.2 Operation.....	D-2
D.2.3 Retrieving Data from the Cache.....	D-2
D.2.4 Updating Table Definitions.....	D-2
D.3 Directory Organization .....	D-3
D.3.1 C:\CampbellSci\LoggerNet Directory (Working Directory).....	D-3
D.3.2 C:\Program Files\CampbellSci\LoggerNet Directory (Program File Directory).....	D-4
D.3.3 Backing Up Critical Information.....	D-4
<b>E. Log Files.....</b>	<b>E-1</b>
E.1 Event Logging .....	E-1
E.1.1 Log Categories.....	E-1
E.1.2 Enabling Log Files.....	E-1
E.1.3 Log File Message Formats.....	E-2
E.1.3.1 General File Format Information .....	E-2
E.1.3.2 Transaction Log Format.....	E-2
E.1.3.3 Communications Status Log Format.....	E-18
E.1.3.4 Object State Log Format.....	E-20
<b>F. Importing Files into Excel .....</b>	<b>F-1</b>
F.1 Array-Based Data File Import .....	F-1
F.2 Table-Based Data File Import .....	F-4



# ***Preface — What's New in LoggerNet 3.1?***

---

## **Product History**

This release of LoggerNet continues the original design of client-server functionality that first appeared when Version 1.0 was released for Windows to replace Real Time Monitoring Software (RTMS) that ran on OS/2 operating systems. Versions in the 1.x series supported only table-based dataloggers and provided large network users with sophisticated capabilities to develop clients to the server to move data without having to store it in interim files.

Version 2.0 added support for dataloggers with mixed-array operating systems, the CRBasic dataloggers, and additional communications devices. It also supported client applications' requests for data via TCP/IP, but also automatically created files on the PC for final storage data. Subsequent revisions in the 2.x series added support for hardware as it was released and refined the client-server architecture to make it more robust and flexible. Software development kits and standalone clients were released to provide additional functionality.

One of the main efforts in the development of LoggerNet 3.1 has been to incorporate support for the CR1000 datalogger. This includes datalogger management (connect, collect data, set clock, send program, etc.) in LoggerNet, as well as programming support in CRBasic and Short Cut. To help with creating CR1000 programs, a Transformer utility has been developed to convert existing CR10X Edlog programs to CR1000 CRBasic programs.

## **LoggerNet Products**

Campbell Scientific offers two LoggerNet software packages, *LoggerNet* and *LoggerNetData*, and several standalone client products. Each of these packages is purchased separately. *LoggerNet* is the main software application and comes with all of the applications needed to set up and configure a network of dataloggers including tools to write programs and monitor retrieved data. *LoggerNetData* is a complementary product that includes applications that can be used on a remote computer to monitor data, or retrieve, view, and analyze data that was collected by LoggerNet into a file.

## **New and Improved Applications**

Several new applications, as well as improvements to previously available applications, are included to make data acquisition and troubleshooting easier than ever.

### **EZSetup Wizard**

The EZSetup Wizard, first introduced in PC400 software, was designed to provide step-by-step help to set up and configure a datalogger for communication and data collection. Each screen in the wizard has fields for the pertinent information about your station. After the last step in the Wizard has

been completed, the information is saved and the datalogger and peripheral communication devices are added to LoggerNet's device map. The Setup window is still available for users who may have more complicated communication networks that require a level of detail not provided by the EZSetup Wizard.

## Troubleshooter

The Troubleshooter summarizes the status and possible problems with communications with individual dataloggers. It provides leads for investigating network failures.

## Short Cut for Windows

While Short Cut is not a new application, this is the first version of LoggerNet that includes this easy-to-use datalogger program generator as part of its suite of applications.

## PakBus Graph

Earlier versions of LoggerNet supported the new PakBus dataloggers and devices and provided a textual tool, PakBusAdministrator, to configure PakBus devices. PakBus Graph provides this functionality in an easier to use graphical format. It is, essentially, a graphical display of the LoggerNet server's PakBus routing table, but it also provides the capability to change settings in remote devices.

## Device Configuration Utility (DevConfig)

The Device Configuration Utility (DevConfig) is accessed from a separate shortcut icon in the program folder rather than a button on the Toolbar. This application offers a powerful tool for configuring the various settings in devices – both dataloggers and communications devices such as the RF400 radios and NL100 Network Link Interface – including sending new operating systems to devices with flash memory. You can save a configuration from one device and send it to other devices of the same type as a starting point to make sure settings are unique or uniform as required (such as when setting up a network of PakBus RF400 radios, where most of the settings in the radios should be the same). You may also download a fresh DevConfig from our web site to get the latest version for evolving devices.

## Setup Window

The setup information on each tab for a device has been divided into Standard and Advanced sections. The standard fields must be completed by the user for communication with the datalogger to be successful. The advanced fields can be left at their defaults for most applications.

Clock and Program tabs have been added for dataloggers with which to update the datalogger's clock and running program. LoggerNet will connect to the datalogger, check or set the clock, send the program, wait for the updated clock

values or program compilation results and then terminate the communication link.

You can still right-click on a device in the Network Map to add child devices, but in addition, the Add window has been improved so that only those devices that are valid connections for the highlighted device are displayed. As each device is added, the contents of the Add box will change based on the last device added.

Settings have been added to configure LoggerNet for receiving incoming UDP address updates from an AirLink Raven CDMA digital modem with dynamic IP addressing.

Clock settings have been added to the LoggerNet server to determine how LoggerNet should adjust its time for daylight saving time.

The new LoggerNet server supports independent PakBus addresses for its individual PakBus ports, effectively allowing you to isolate subnetworks from each other. The Setup screen offers a pull-down menu item to set the LoggerNet PakBus settings and bridge or not bridge the individual ports.

## Connect Window

The biggest change to the Connect screen is that you no longer have to stay connected to a datalogger to initiate a data collection, clock check/set, or program send/receive attempts. If you are not yet connected to a datalogger, LoggerNet will initiate a link to the datalogger, perform the desired action, and then terminate communication.

Numeric and Graphical displays will automatically display (but do not collect to a file unless told to do so) the last record in final storage from table-based dataloggers, and they each include a separate field for their own collection interval. The server keeps track of when tables should be written to and avoids collections if no new data has been stored.

## Status Window and LogTool

Enhancements to the Status window include a new Trouble Indicator icon and a Comm Status Image displaying communication quality over the past 12 hours.

Status information for any statistic can now be sorted in ascending or descending order when a column heading is clicked to make troubleshooting easier. The LogTool replaces separate launching buttons for server and low level logs. You can access any of the log displays from this one LogTool screen, and choose to display log messages only for a selected device(s) to let you focus on a particular problem. You can also choose to save or print the current log windows without having to open the individual log files.

A Communications Test window is also now available from the Status window toolbar to let you test communications with one or more devices. This can be very handy when setting up a network to test each device as it's brought on line.

## Split Report Generator

Many new enhancements have been made to the Split report generator, including:

### **Input File Options**

A new option has been added to the Start and Stop conditions, which will start or stop the processing of a data file on or after the specified time is met (Start-Stop On/After Time). Previously, the exact time would have to exist in the data file before processing would start or stop.

You can now access up to 8000 elements on the Select line. Previously, you could access only 1 through 254. If a data file had more than 254 data points in a record or array, Split could not access them for output. Additionally, the select line is no longer limited to 254 characters. The limit is now 4,700 bytes.

This version increased the number of columns that can display headers from 60 to 254.

Two new functions have been added (WDQ and WDQS) to output an alphabetical quadrant (e.g., "N", "SE", "WNW") from wind direction data.

A new date format, "hourarray", has been added that can be used to convert the midnight timestamp for table-data dataloggers (0000) to a 2400 hour format. This function, along with "dayofyear", can be used to convert the date/time strings collected from table-based dataloggers to the year-day-hhmm format native to mixed-array dataloggers so as to make data file formats backwards compatible.

### **Output File Options**

In addition to overwriting or appending to an existing output file, there is now a Create New File option. The new filenames have "\_0", "\_1", "\_2", etc. appended to the original filename to keep them distinct.

When using the final storage label file (FSL) to populate column header fields, the labels are now broken at the underscore ( \_ ) character. Any characters following an underscore will appear on the next column heading line.

A No Dashes option has been added to remove the line of dashes after the header in the output file.

Summary data and/or the summary heading can be suppressed from being added to the end of the report.

# Section 1. System Requirements

---

## 1.1 Hardware and Software

LoggerNet is a collection of 32-bit programs designed to run on Intel-based computers running Microsoft Windows operating systems. The recommended minimum computer configuration for running LoggerNet is Windows NT , Windows 2000, or Windows XP because they offer the most stable operating environment. LoggerNet may also run on Windows 98 if operations such as open screens are limited so as not to exhaust the operating system's resources. All installations require at least a Pentium II or equivalent processor, a minimum of 128 MB of RAM, a minimum of 100 MB free space on the hard disk, and TCP/IP support installed.

## 1.2 Configuration of TCP/IP Services

TCP/IP services must be running on the computer for LoggerNet to run. Following are the procedures for enabling TCP/IP communication on a Windows 98 or NT system. For Windows 2000 or Windows XP the same things need to be set up, but they are accessed in different ways. See the documentation and help for these operating systems 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 CD-ROM (or floppy disks as appropriate) handy.

---

As you install these options you may be prompted to insert various disks or the CD-ROM to complete the installation.

1. Click the Start button and select Settings | Control Panel.
2. When the Control Panel window comes up double click the Add/Remove Programs icon.
3. Select the Windows Setup tab.
4. Select Communications and click the Details button.
5. On the Communications options screen click the box by "Dial-Up Networking" (Win 98) or "Phone Dialer" (NT). If already checked, click cancel and skip to step 9.
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 the Network icon.
10. In the list box on the Configuration tab (Win 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, cancel and skip the next steps.
11. Click the Add button. In the Select Network Component Type window which comes up select Protocol or TCP/IP protocol and click 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 2. Installation, Operation and Backup Procedures***

---

## **2.1 CD-ROM Installation**

Place the installation disk in your computer's CD-ROM drive. If autorun is enabled for the drive, the LoggerNet installation will start automatically. If the installation does not start automatically, select Start | Run from the Windows' system menu. Type D:\Setup.exe (assuming D is your CD-ROM drive letter) in the Open field. Alternately, use the Browse button to access the CD-ROM drive and select the Setup.exe file from the disk. Press OK and 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 Toolbar and some other selected utilities. At the end of installation you also have the option to add a desktop shortcut to LoggerNet.

By default, the installation copies the LoggerNet program files to the C:\Program Files\CampbellSci\LoggerNet directory. Many operating system configurations will require the user name under which the software is installed to have administrative privileges to the computer. After the software is installed, administrative privileges are not required by the user to run the software.

In addition to placing files in the Program Files directory of your computer, the installation also creates working directories for the LoggerNet server and the individual LoggerNet applications under C:\CampbellSci. Section 2.3.1 provides more detail on the directories that are created.

## **2.2 Upgrade Notes**

If you have LoggerNet version 2.0 or 2.1 installed on your computer, the LoggerNet 3.1 installation will detect that version and ask if you want to upgrade to the current version. If you choose Yes, the installation will upgrade the existing software to version 3.1 using the directories chosen in the original installation. If you choose No, you will be prompted for new directories in which to install the LoggerNet 3.1 software.

If you upgrade the existing 2.0 or 2.1 installation, LoggerNet 3.1 will continue to use the network map, data collection schedules, data file locations, etc., of the existing installation. Essentially, you will be able to "pick up where you left off" the last time you used LoggerNet. If you install version 3.1 as a new installation, you will need to recreate your network map, set up new data collection schedules, etc. Note that the first time you collect data from your datalogger(s), LoggerNet will collect all the data in the datalogger's memory.

**CAUTION**

LoggerNet 3.1 can readily use the network map from LoggerNet 2.0 or 2.1. However, network maps are not backwards compatible. If you upgrade your existing version, once LoggerNet 3.1 is opened, the network map will no longer be compatible with version 2.0 or 2.1. If you think there may be any reason that you need to revert back to a previous version, make sure to back up the network map before installation.

---

If you are upgrading from an installation of PC208W version 3.0 or greater, you can convert your current network map from PC208W format to the LoggerNet format. A Convert utility has been provided to bring the old PC208W network description into the new LoggerNet format. You should make sure that data collection is up to date in PC208W before converting the network map. It is also a good idea to make a backup of the PC208W files. The explanation for the use of Convert is in Section 4.2.8.1 on network setup.

## 2.3 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.

### 2.3.1 LoggerNet Directory Structure and File Descriptions

#### 2.3.1.1 Program Directory

As described in the installation procedures, all of the files for program execution are stored in the C:\Program Files\Campbellsci\LoggerNet directory. This includes the executables, DLLs, and most of the application help files. This directory does not need to be included in back up efforts. LoggerNet and its applications rely on registry entries to run correctly; therefore, any restoration of the program should be done by reinstalling the software from the original CD.

#### 2.3.1.2 Working Directories

In this version of LoggerNet, each major application keeps its own working directory. The working directory holds the user files created by the application, as well as configuration and initialization (\*.INI) files. Because of this new working directory scheme (implemented in version 3.0), when you use File | Open in Edlog, CRBasic Editor, Split, etc., you may find yourself in an unfamiliar directory and may have to navigate to a different directory to find existing data files, datalogger programs, etc.

This scheme was implemented because we use the underlying tools and many of the applications (the server itself, library files, datalogger program editors, etc.) in a number of different products. By providing a common working directory for each major application, we hope to make it easier to keep track of files and information as you move from one product to another.

The following figure shows the typical working directories for LoggerNet if the default options were selected during installation.

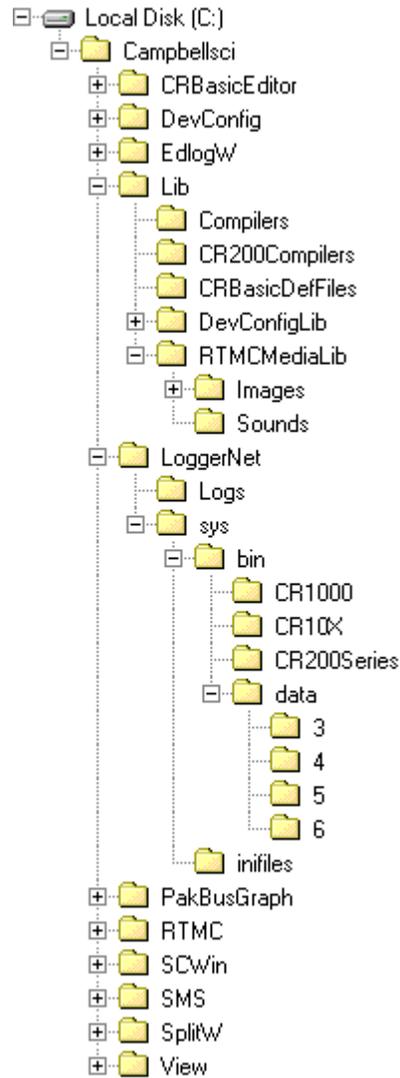


FIGURE 2-1. Typical Working Directories for LoggerNet

By default, the files that you create in each of the applications will be stored in their respective folders in the working directory. You can override that default and store the files in a different location. Each application "remembers" the last directory in which a file was saved and will default to that directory until a different directory is selected.

Note that most all applications have one or more subdirectories in which configuration files are saved.

**Lib directory** - The Lib directory is a library directory for several of the LoggerNet applications. The Compilers folder holds all of the compilers for the CRBasic Editor, except for the CR200 compilers, which are stored in the CR200Compilers directory. The CRBasicDefFiles folder holds the definition files and help files for all dataloggers supported by the CRBasic Editor. The definition files are the files which provide the unique instructions and parameters for each datalogger.

**DevConfig and Library directories** - These directories contain the files for each datalogger that will be used by the Device Configuration Utility.

**RTCMediaLib directory** - This directory contains all of the media files that can be used by RTMC to provide graphics and sound for your RTMC projects. Any custom graphics or sounds that you create and wish to use in your project should be stored in one of these directories.

**LoggerNet directory** - The ASCII data files that are saved to disk as a result of data collection from the dataloggers are stored to the LoggerNet directory with a \*.dat extension. The Logs directory holds the logs that are created when communication takes place between the LoggerNet server and client applications, and the LoggerNet server and the dataloggers. These logs are used to help troubleshoot communication problems.

The Sys directory holds the network map description, configuration files, and the binary data cache. (The data cache is a repository for the data which is collected from the dataloggers by the LoggerNet server, and which each client application accesses when processing that data. In the example above, folders 3, 4, 5, and 6 represent the data caches for four different dataloggers. See Appendix D for additional information.)

## 2.3.2 Backing up Data

As with any computer system that contains important information, the data stored in the LoggerNet working directory should be backed up to a secure archive or transferred to another system on a regular basis. 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 Toolbar 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 (C:\CampbellSci). This will back up all of the working files for the server and the client applications. Note that the working directory cited here assumes that the default directory structure suggested by the install utility was used during installation. If a different working directory was used, then the files will be contained in the same set of subdirectories, under the main working directory. At the very least, we recommend that you back up the

C:\Campbellsci\LoggerNet directory to ensure that your data files, the LoggerNet data cache, and your network map are not lost.

The maximum interval for backing up data files 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 the storage area or table 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.

A command line utility called CoraScript can be used to back up the network map and all of its settings. There is a function available 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 to restore the network after a computer failure causes the network map or configuration files to be corrupted. It will not, however, restore the data cache or the system state. For more information on CoraScript see Section 11.

### 2.3.3 Loss of Computer Power

The LoggerNet communications server writes to several files in the \SYS directory during normal operations. The most critical files are the data cache table files and the network 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 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 98 machines using the FAT32 disk file formatting. Windows NT, 2000, and XP offer the choice of NTFS that provides a greater protection for this type of event. Thus, Windows NT, 2000, and XP offer 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 server's working files need to be restored from backup to maintain the synchronization in the server state.

### 2.3.4 Program Crashes

If the communication server crashes, there is a possibility that files can be corrupted (note, however, that corruption is much less likely with a program crash than during a 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, 2000, and XP than with Windows 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, 2000, and XP systems make sure that the LoggerNet server has not left a process running. You can check this by going to the Windows Task Manager and selecting the Process tab. In the list of processes look for the Toolbar or one of the client applications. If one of these processes exists but the Toolbar is not running, select this process and click “End Process”; you will be asked to confirm the end process.

### **2.3.5 Restoring from Backup**

To restore server operation from a backup copy of the data and configuration files, 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.

## **2.4 Special Note on Windows XP Service Pack 2**

With Service Pack 2, Microsoft has automatically enabled a firewall application on each individual PC. This was done to protect PCs from invasion by outside, unauthorized programs that may try to connect via a socket using TCP/IP.

Remember, however, that LoggerNet is a client-server application that uses TCP/IP as the link between clients and the server. This means that windows such as Setup, Connect, Status, RTMC, etc., get their access to devices and data only through the LoggerNet server. While, in the basic LoggerNet installation most of these clients typically access the server on the same PC (which will already be “behind” the PC’s local firewall), the server is capable of distributing the same information to similar clients connected via TCP/IP from anywhere in the world, revealing the true power of the client-server design. If, for example, you want others to use RTMC or DataFiler (packaged with LoggerNetData) to get data from your LoggerNet PC, you can enable this remote connection by toggling a menu item on the Toolbar (Options | Allow Remote Connections). This causes the server to open a socket on a specific port (default is port 6789) to listen for requests for data from its clients. This is usually quite safe since: 1) no other application should be trying to use this port, and 2) the server will only respond to LoggerNet-specific messages on this port (it will not run viruses or other unauthorized bits of code). If you allow remote connections, however, the firewall in Windows XP SP2 will put up a window telling you that it has blocked “Toolbar” and asking if you wish to unblock Toolbar from Campbell Scientific. At this point, if you click “Unblock”, Windows XP will make an exception for LoggerNet and you should not have to unblock it each time you start it. You can reverse this decision by opening the Windows Control Panel, selecting Windows Firewall, and delete Toolbar from the list of exceptions.

# **Section 3. Introduction**

---

## **3.1 What is LoggerNet?**

LoggerNet is a software application that enables users to set up, configure, and retrieve data from a network of Campbell Scientific dataloggers and share this data over an Ethernet communications network. This software application is designed to run under Windows NT version 4.0, Windows 2000, and Windows XP. The software will also run under Microsoft Windows 98, though there may be some performance issues with these systems (see Section 1, System Requirements).

LoggerNet software supports communication and data collection for Edlog dataloggers including the CR500, CR510, CR10, CR10X, 21X, CR23X, and CR7 in any of their mixed-array, table data, or PakBus operating systems, and the CRBasic dataloggers including the CR200/205, CR1000, CR5000, and CR9000.

The LoggerNet software is written using advanced “client-server” architecture. The server software engine runs in the background handling all of the datalogger communications. The server also takes care of storing the data and providing information to manage the datalogger network. In turn, the client applications connect to the server to access the information collected from the dataloggers.

One significant benefit of the software design is that some of the client applications (RTMC, for instance) can be run on any computer that connects to the main computer by a TCP/IP network connection. Some examples of these networks are Local Area Network (LAN), Wide Area Network (WAN), or the Internet. Another benefit is the efficiency that is gained, since several client applications can simultaneously request and receive information from the software server.

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

### **3.1.1 What Next?**

The ultimate goal with most datalogging applications is to retrieve data to a computer for further analysis and manipulation. Now that you have installed LoggerNet on your computer, how do you reach that goal?

The first step is to set up a communication link between your computer and the datalogger station. This step may also include the configuration of peripheral communication devices. Next you'll need to develop a program for the datalogger, and then send the program to the datalogger and ensure that measurement results are viable. Once the datalogger has been storing data for a period of time, you will want to collect that data and store it to a file on your computer for further analysis.

LoggerNet provides the tools to accomplish these steps, and well as tools to resolve problems along the way if they should arise. The remainder of this section briefly reviews the steps and the various tools that can be used to accomplish them. More detailed information is provided in the subsequent sections of this manual.

## **3.2 Overview of Major LoggerNet Functions and Associated Software Applications**

### **3.2.1 The Heart of it All — LoggerNet Toolbar**

The LoggerNet Toolbar has several functions. The most important is starting the server that handles all communications with the dataloggers in the network. As long as the Toolbar is running, either visible or minimized, the server is working and able to communicate with the dataloggers. Shutting down the Toolbar also shuts down the server and suspends all communications with the dataloggers in the network.

When you run LoggerNet from the Window's Program menu or from a desktop shortcut, you are launching the LoggerNet Toolbar. Some buttons on the Toolbar launch applications that connect to the server and allow you to set up the network or view the collected data. Other buttons launch stand-alone applications to perform other functions, such as program editing.

LoggerNet's client-server design allows client applications to be run on the same computer as LoggerNet, or on remote computers running LoggerNet data applications. Note that the LoggerNet Toolbar must be up and running and Remote Connections must be enabled for remote computers to access a LoggerNet server. For additional information on Remote Connections, refer to Section 13.1.

#### **3.2.1.1 Toolbar Customization**

When LoggerNet is installed, the Toolbar is configured with buttons for the applications that are most likely to be needed by the majority of users -- EZSetup, Setup, Connect, Status, Edlog, Short Cut, CRBasic, Split, View, and RTMC. Additional applications are shipped with LoggerNet that can be accessed from the Toolbar's File | Open menu. One or more of these applications can be added to the Toolbar, or existing applications can be removed from the Toolbar, by selecting Options | Configure Buttons. This will open a list showing all available applications. Select or clear the checkbox beside each application name to add or remove it from the Toolbar, respectively.

In addition, the size and orientation (horizontal or vertical) of the Toolbar can be changed by selecting a corner of the Toolbar and dragging it to a new size.

### 3.2.1.2 Toolbar Menus

As noted above, the File menu has an Open item that can be used to launch a LoggerNet application. The program can also be closed (File | Exit) from this menu, or you can close it by selecting the X in the upper right corner of the Window.

The Tools menu provides access to the network test and troubleshooting tools. These tools, which can also be accessed from other places in the program, include the Troubleshooter, Terminal Emulator, PakBus Graph, and LogTool. The Options menu is used to configure the Toolbar, enable/disable Remote Connections, and enable/disable Tasks. All of these tools and functions are discussed in detail elsewhere in this document.

### 3.2.1.3 Command Line Arguments

Command line arguments allow you to change LoggerNet's default behavior when it is started from a shortcut.

Currently, there are four command line arguments:

**/W** Sets the working directory to something other than the default. Usage:

```
"C:\Program Files\CampbellSci\LoggerNet\ToolBar.exe" /w C:\CampbellSci\test
```

where "**C:\Program Files\CampbellSci\LoggerNet\ToolBar.exe**" is the directory and filename for the LoggerNet Toolbar (which essentially is the LoggerNet server) and **C:\CampbellSci\test** is the working directory to be used.

**/M** Launches LoggerNet in a minimized state. Usage:

```
"C:\Program Files\CampbellSci\LoggerNet\ToolBar.exe" /m
```

**/IPPORT=XXXXX** Causes the server to use port XXXXX for TCP/IP communications with clients. Handy if some other software is using the default port of 6798. Usage:

```
"C:\Program Files\CampbellSci\LoggerNet\ToolBar.exe" /ippport=12345
```

**/TOOLBARPORT=YYYYY** Causes the Toolbar to use port YYYYY for TCP/IP communications with the programs it launches, such as Setup, Connect, Status, Edlog, CRBasicEditor, etc. The Toolbar communicates with these other programs to cause them to minimize, restore, close, etc. Handy when other software is using the default port of 8005. Usage:

```
"C:\Program Files\CampbellSci\LoggerNet\ToolBar.exe" /toolbarport=12346
```

## 3.2.2 Setting Up Datalogger Communication Networks

Network setup tools allow you to define and configure the dataloggers in the network, how they are connected to the computer, and what data should be collected. These tools include the EZSetup Wizard and the Setup window. Either can be used to set up the datalogger communication network.

The EZSetup Wizard provides a simple step-by-step sequence of screens, with on-screen help and many pre-set values that makes it easy to add a new datalogger and communications devices to your LoggerNet network. You start with the type of datalogger you wish to add, and then enter the settings for the communications devices used to reach it, ending with a communications test and an opportunity to set the clock, send a program, and set up an automatic data collection schedule. The relevant devices are added to LoggerNet's network map, enabling you to subsequently edit them in the conventional Setup window, if necessary.

The Setup window accomplishes the same tasks, but allows you a bit more control when setting up your network, and allows for more complex network configurations.

A standalone tool, the Device Configurator, or DevConfig, uses a direct serial port connection to set the settings in a device (such as the PakBus address or routing information for a datalogger, or RF400 radio information) so that LoggerNet can communicate with it. It can also be used to send a new operating system to devices that accept a downloadable OS. DevConfig supports all CSI dataloggers, as well as configurable communication devices such as RF400 radios, MD485s, and NL100s.

LoggerNet also ships with a command line scripting tool, CoraScript, which can be used to configure the datalogger network from a command prompt. It is a useful tool for quickly backing up or restoring LoggerNet's communication network.

## 3.2.3 Real Time Tools

LoggerNet's real-time tools are used to connect to a datalogger, set the clock and send a program to the datalogger, and view the data being collected from the datalogger by the LoggerNet server. These tools include the Connect window and RTMC.

The Connect window is used primarily for initializing or checking operation of a datalogger and manually collecting data. This screen 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, and getting status information from the datalogger. There are windows for displaying data either graphically or in numeric format, as well as setting input locations, ports, and flags. You can also manually retrieve data in various formats, and communicate with a datalogger in terminal emulation mode.

RTMC is used to for real-time data displays of the data collected by the LoggerNet server. You can create customized graphic displays that include graphs, tables, dials, alarms, digital values and other graphic elements. These

displays automatically update when LoggerNet collects new data. Graphical elements are also available for toggling ports or flags, or setting input locations (or variables) in a datalogger. The displays created in RTMC can be distributed to other users who have licenses to run RTMC Run-time software (purchased separately). This allows a remote computer, accessible via TCP/IP, to connect to the LoggerNet server and display the real-time data.

### 3.2.4 Network Status and Problem Solving

Since communications invariably fail at some point, LoggerNet includes several tools to pinpoint where the problem lies.

The Status Monitor is used to monitor the health of datalogger network communications. The integrity of the communications link can be verified quickly from the color of the status icon for each device. A Communications History graphic shows the success of communications over the last 12 hours. Columns can be set up to display detailed statistics on communications quality and data collection. For troubleshooting purposes, The LogTool application is available to view operational log messages for the server as well as the low-level communication between the datalogger and the server. A Comm Test window and PakBus Graph can also be launched from the Status Monitor.

Troubleshooter is used to identify possible problems disrupting communications or data collection. You can narrow the display to one or a few devices, or expand it to view the entire network. Finding a problem, you can launch a separate Communications Test utility, open the server logs, or launch PakBus Graph. Problems are highlighted in different colors to indicate levels of severity.

PakBus Graph provides a graphical display of a PakBus network, and quick access to the PakBus settings in LoggerNet and other PakBus devices. PakBus is a packet-switched protocol developed by Campbell Scientific that facilitates communications between PakBus-capable devices, including dataloggers, some communications peripherals (NL100s, RF400s, etc.), and LoggerNet itself. Among the advantages of PakBus are: more robust communications due to packet-based communications, multi-threading of communications (e.g., you can use a keyboard/display at the same time as a PC is collecting data via telecommunications; or two PCs can request data from a datalogger at the same time), peer-to-peer communications (PakBus dataloggers can send to or request data from other PakBus dataloggers without a PC involved), and dynamic routing (PakBus devices can be configured as “routers” and learn about the presence of other PakBus devices or alternative routes to those devices as they come on line or routes change).

In addition to these tools, a Troubleshooting guide is provided in this manual. Refer to Section 15.

### 3.2.5 Creating and Editing Datalogger Programs

LoggerNet offers two programs editors (one for CRBasic datalogger programs and one for Edlog datalogger programs), and a program generator (which supports both programming language types).

For straightforward “measurement/control/data storage” datalogger programs, the Short Cut program generator is an excellent choice for datalogger program creation. Short Cut prompts you for the type of datalogger, scan interval, sensors to be measured (including those on multiplexers and other peripherals), and desired final storage output. It then creates the program for you, along with a wiring diagram showing how each of the sensors should be connected to the datalogger. Short Cut’s resulting programs can even be imported into Edlog or CRBasic Editor when you want to get a head start on more elaborate programs. Short Cut is also available from the Campbell Scientific website ([www.campbellsci.com](http://www.campbellsci.com)) so you can update it with newer sensor files as they become available.

---

**NOTE**

While Short Cut programs can be imported into Edlog or the CRBasic Editor, once they have been edited in one of these programs, the modified program cannot be imported back into Short Cut.

---

Edlog is the tool to create and edit datalogger programs for all Campbell Scientific dataloggers except the CR200/205, CR1000, CR5000, and CR9000. Instructions are available for sensor measurement, intermediate processing, program and peripheral control, and data storage. The built-in precompiler provides error checking and warns of potential problems in the program. For Edlog dataloggers with PakBus operating systems, you can include settings for PakBus routing in the datalogger program itself.

The CRBasic Editor is a program editor for the CR200/205, CR1000, CR5000, and CR9000 dataloggers. Instructions are included for sensor measurement, program and peripheral control, data storage, and peer-to-peer data transfer. The editor checks for program validity and offers many user-configurable options to make editing long programs easier.

For those users of CR10X dataloggers and Edlog programming who are switching over to CR1000 dataloggers, a Transformer utility has been developed. The Transformer, which is a stand-alone utility launched from the Windows Start menu, reads in an Edlog CSI or DLD file and generates a CR1000 CR1 program file. The two files are displayed side-by-side for comparison purposes; double-click an instruction in the Edlog program, and the associated instruction is highlighted in the CRBasic program. Edlog program instructions that cannot be converted directly to a CRBasic program instruction are listed in a Messages window and are included as commented text in the CR1 file. After conversion, the newly created CR1 file can be opened in the CRBasic Editor for further editing.

### 3.2.6 Working with Data Files

LoggerNet includes applications that allow you to view and process the data collected from the dataloggers. These include the file viewer, View, a report generation tool, Split, and a TOB (binary data file) File Converter.

View is used to inspect data files, from either mixed-array or table data dataloggers. The data is displayed in either comma-separated or tabular format by record or array. A graph can be displayed to show one or two columns of

data, and you can zoom in on individual points for detailed date/time markers or values. You can also print graphs.

Split is used to post process and generate reports from collected data files from either mixed-array or table-based dataloggers. Traditionally it has been used to separate mixed-array data files into individual files based on the array ID, but it can also create files in custom formats for use in reports or as input to other data applications, including converting mixed-array datalogger time stamps (year, Julian day, Hour/Minute) to more conventional date/time stamp formats. Split includes time series function, which can be used to provide summary information from more frequent data (e.g., hourly summaries from one-minute data).

The TOB File Converter is used to convert TOB1, TOB2, and TOB3 files to TOA5 format (TOB2/TOB3 files can also be converted to TOB1 format). TOB files are binary files that are either created by LoggerNet during collection or are collected directly from a PC card installed in a CRX000 datalogger. A command line file converter is also included in LoggerNet. (Refer to Appendix B for additional information on File Formats.)

### **3.2.7 Automating Tasks with Task Master**

The Task Master is used to set up a Task that can be triggered on a defined schedule or upon a data collection event from a datalogger. A Task can be data collection from another datalogger, or anything that can be executed in a computing environment (i.e., a command line operation, a program executable, a batch file, or a script).

### **3.2.8 Managing External Data Storage Devices**

The Storage Module Software (SMS) handles configuration, data retrieval, and datalogger program management with Campbell Scientific-formatted PC cards or Campbell Scientific storage modules. This program supports the SM192/716 and SM4M/16M storage modules along with the CSM1 or MCR card reader, or (for Windows 98) a PCMCIA card slot installed on the computer.

The File Control functionality, accessed from the Connect window, can be used to manage files created by CRX000 dataloggers on other, non proprietary formatted, PC cards.

### **3.2.9 Optional Client Products Compatible with LoggerNet**

#### **3.2.9.1 LoggerNetData**

LoggerNetData consists of client applications that can be installed and run from a remote computer connected by TCP/IP to the computer running LoggerNet. This connection could be a local area network (LAN) or the Internet. LoggerNetData provides you with tools to remotely monitor collected data, retrieve data, and view or process it on the remote PC.

The applications included with LoggerNetData are RTMC, View, Split, and DataFiler. RTMC, View, and Split are described above. DataFiler is a

application used to collect data from LoggerNet's data cache, and store the collected data to a file on a remote computer. DataFiler does not collect data from a datalogger directly, but from the data cache. Therefore, data collection must occur in LoggerNet (either by a scheduled or manual data collection), for the data to be available to the DataFiler.

### 3.2.9.2 Data Display Clients

RTMC Run-Time is available separately from LoggerNet. It allows you to run forms created in RTMC so that data can be displayed on a remote computer.

The RTMC Web Server is used to display RTMC forms in an HTML format. Once the web server is in place, the only thing required to view the data is a browser such as Internet Explorer or Netscape. For security reasons, the displays are set as "read-only", so toggling a port or flag, or setting a value in an input location, is not available.

RTDM is a stand-alone data display client that can display data from collected data files or directly from the LoggerNet server. RTDM provides more controls and greater flexibility than RTMC; however, it also has a somewhat steeper learning curve. RTDM can output images of a display at regular intervals, which can then be incorporated into html pages.

### 3.2.9.3 Baler

The Baler is another remote data retrieval client that creates "bales" of files based on the date and time stamp of the collected data, and stores those files on the remote PC. The individual baled files are saved to disk using part of this date and time stamp for the name. The Baler differs from DataFiler in that it stores files based on the date and time stamp in the data file, where DataFiler collects and stores data based on the clock of the remote PC.

### 3.2.9.4 PC-OPC, CsiOPC Server

PC-OPC is a client that can be run on a remote PC or the same PC as LoggerNet. It makes data available from the LoggerNet server in an OPC format. This allows use of the data by third party software applications that communicate using the OPC protocol.

### 3.2.9.5 Software Development Kit

PC-SDK, the LoggerNet Software Development kit, allows developers to create custom applications to communicate with the LoggerNet server. The kit includes three ActiveX controls that work with Visual Basic or Delphi, along with a beginner's guide, programmer's reference, and examples. Note that the SDK includes the development tools only; LoggerNet must be purchased separately.

---

**NOTE**

Access to the LoggerNet server by remote clients requires that Remote Connections be enabled in LoggerNet. See Section 13.1.

---

## 3.3 Getting Help for LoggerNet Applications

Detailed information on each application is included in subsequent sections of this manual. Each application also 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. Additionally, Edlog and CRBasic program editors have context sensitive help that can be displayed by right-clicking an instruction or parameter.

Popup hints are available for many of the on-screen controls. Let the mouse pointer hover over the control, text box or other screen feature; the hint will appear automatically and remain visible for a few seconds. These hints will often explain the purpose of a control or a suggested action. For text boxes where some of the text is hidden, the full text will appear in the hint.



# Section 4. Setting up Datalogger Networks

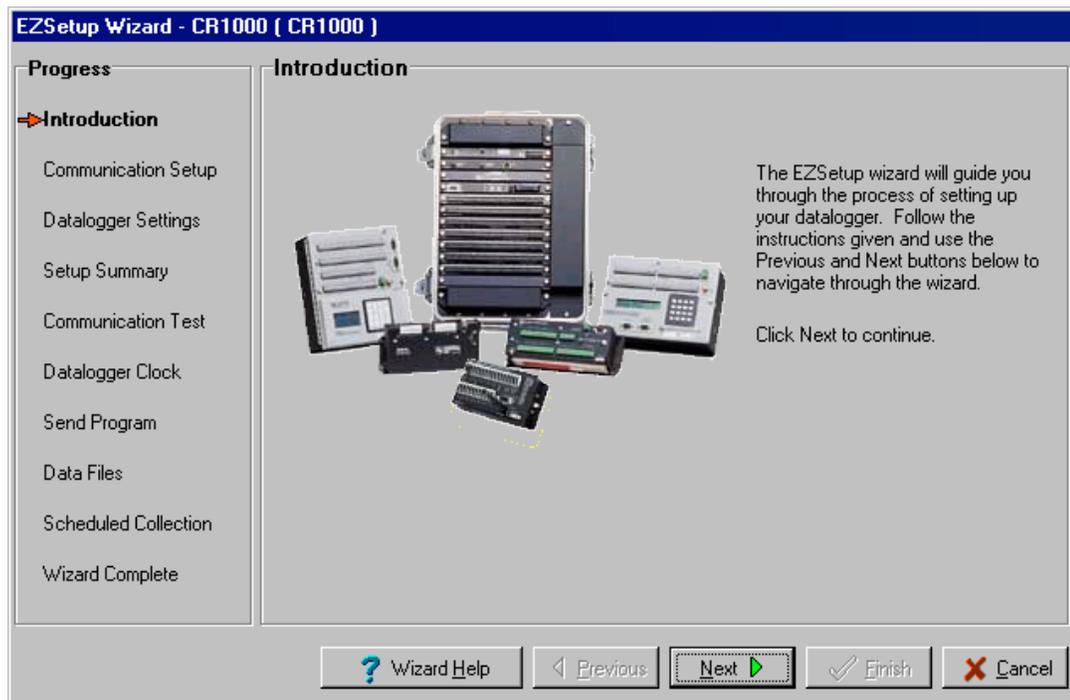
The EZSetup Wizard and the Setup Screen provide ways to create and maintain the communications link and data collection schedules for a network of dataloggers. The EZSetup Wizard is a tool that walks you through the setup step-by-step. In the Setup window, you add devices and configure their settings on your own. Either task will result in a network map showing all of the devices and communications links to reach the datalogger stations.

The Device Configurator, or DevConfig, is a stand-alone tool that can be used to configure settings in the dataloggers themselves, as well as in communication devices such as RF400 radios or NL100s.

## 4.1 EZSetup Wizard

The EZSetup Wizard is automatically displayed when LoggerNet is run for the first time (after the initial launch, it can be opened from the Toolbar). It was designed to walk you through the setup and configuration of your datalogger network. As you work through the steps for the Wizard, each screen has fields that are completed with the pertinent information about your station. In addition to setting up a new station, the EZSetup Wizard can be used to edit an existing station.

The EZSetup Wizard starts with the page shown below.



Subsequent pages are similar. Previous and Next buttons are provided to move through each step of the wizard. Progress is shown by the red arrow next to each step displayed at the left. Field descriptions and helpful tips are displayed on the wizard page. If additional help is needed, the on-line help can be opened by pressing the Help button on the bottom left of each page.

In the Communication Setup step you first select the datalogger type and give it a name. (This name will also become the default file name for data files collected from that datalogger.) Next you choose which of the possible communications media are supported for that datalogger. The EZSetup Wizard will display the serial ports (COM ports) known to your Windows operating system. You then choose whether the connection is a direct RS232 connection or via some communications peripheral. EZSetup Wizard fills in as many communications settings as possible; in many cases you can use the default settings. It also provides fields for user-entered communications settings such as phone numbers and RF radio addresses.

The Datalogger Settings step is provided for fine tuning the connection to the datalogger. The baud rate offered is typically the maximum baud rate supported by that datalogger and communications medium; lower rates may be required for cell phones or noisy telephone links. Enter a Security Code only if the datalogger is configured – via the keyboard/display or settings in the datalogger program – to use it.

The Setup Summary step provides a list of the settings entered. You can use the Previous button to return to a page and change these settings if necessary.

The Communications Test step allows you to test the communications link before going any further. If the datalogger is not installed, you can skip this and the next two steps.

If communication succeeds, you can move to the Datalogger Clock step where you can check or set the datalogger's clock to match the PC's system time. If the datalogger is in a different time zone, you can enter an offset in hour units.

The Send Program step allows you to send a program to the datalogger. This may be a program you created with Short Cut, Edlog or the CRBasic Editor or a program supplied by someone else. If it is a mixed-array datalogger, and the datalogger is already running a program, you should associate the .DLD file so that LoggerNet will use the labels for input locations and final storage. Dataloggers with table-based operating systems (TD, PakBus, and CRx000) will know their program if one is running and will provide table definitions that contain the labels. If you don't have a program for the datalogger you can skip this step and send a program later from the Setup window or Connect window.

The Data Files step is where you define what data tables, or final storage areas, should be collected by LoggerNet and saved to disk. If you used the EZSetup Wizard to send a program to a table-based datalogger, the software will already be aware of the data tables that exist in the datalogger. If the program was already loaded, or for some reason no tables are displayed, press the Get Tables Definition button to retrieve the table names.

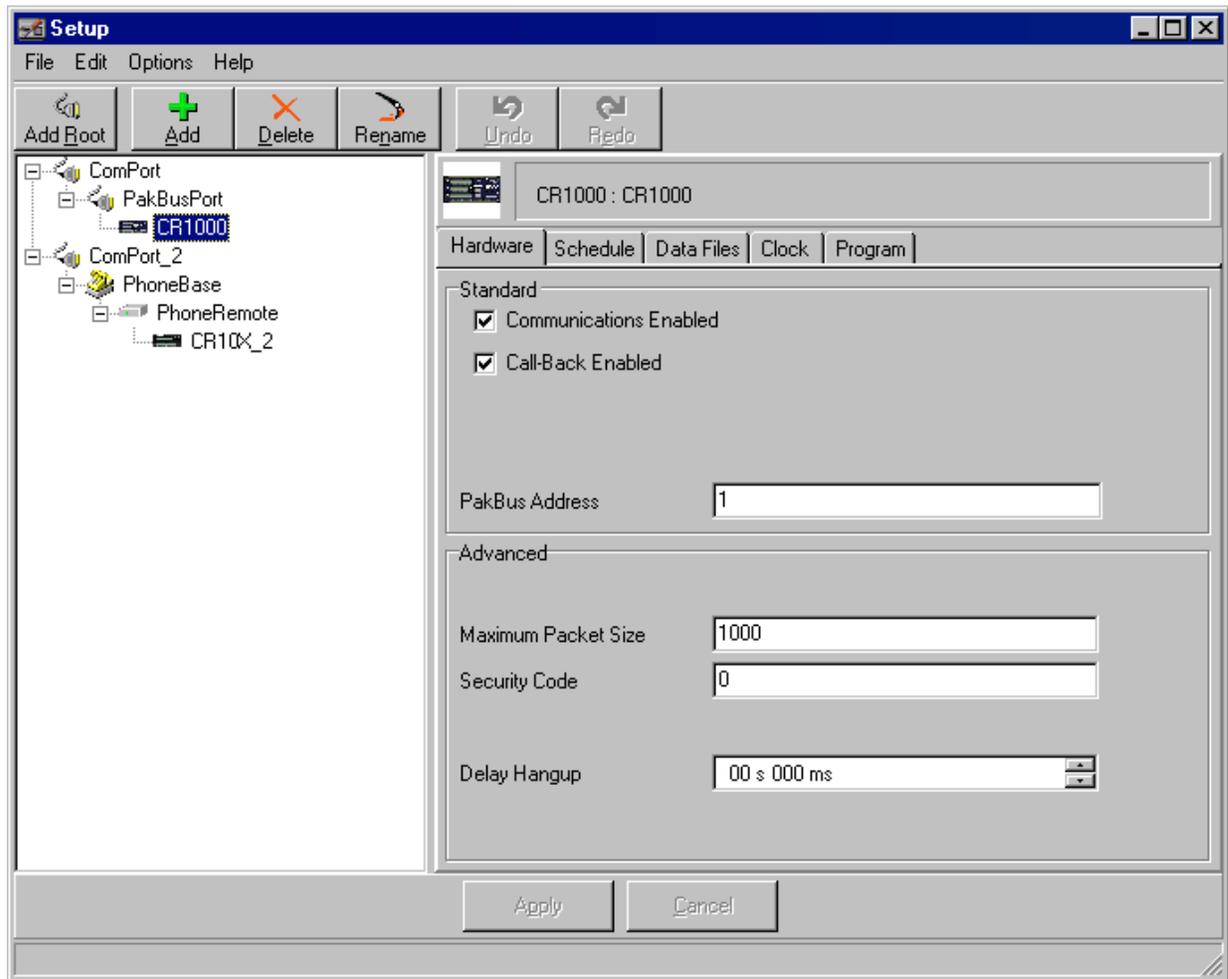
The Data Files step also has an Enabled for Data Collection field. When enabled, LoggerNet will collect that table or final storage area from the datalogger on a manual or scheduled data collection attempt.

The Scheduled Collection step is where you can define a schedule on which LoggerNet will automatically call the datalogger and collect data.

Once a datalogger station has been configured, it can be edited directly in the Setup window, or you can edit it using the EZSetup Wizard. When editing in the EZSetup Wizard, click the step on the left side of the Introduction page to go directly to that step, or walk through each wizard page using the Next button.

## 4.2 Setup Screen

Clicking the Setup button on the LoggerNet toolbar will bring up the Setup Screen. 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).

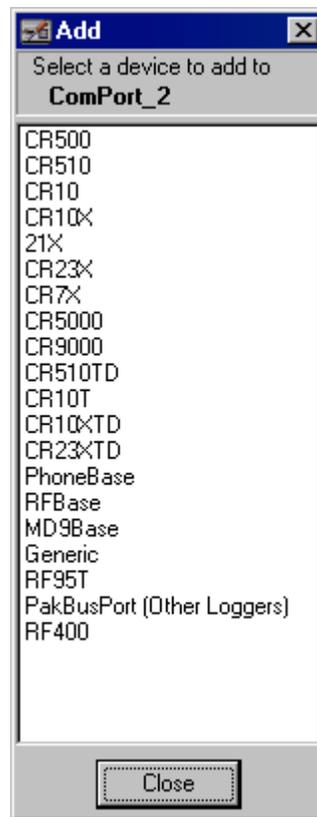


The number of tabs will vary, based upon the type of device that is selected. Some devices may have only a hardware tab, which other devices, such as dataloggers, have multiple tabs.

## 4.2.1 Adding Devices to the Network

Devices are added 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 ComPort, then the telephone modem, the remote phone modem, and the datalogger.

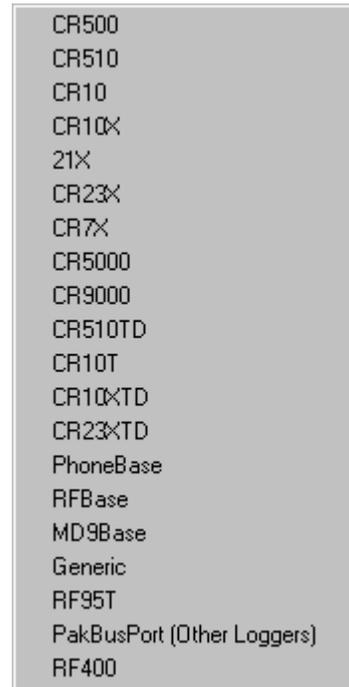
To add a ComPort to the network map either right click in the blank area of the network map, click the Add Root button, or choose Edit | Add Root. Once the ComPort is in place you can click the Add button or choose Edit | Add from the menu to bring up the Add Device window.



The contents of the Add Device window will change as each device is added to the network map. Only those devices that are valid components to add to the last device added will be shown. Continue to add devices in this manner until your network map is complete.

An alternative to the Add Device window is to press the right mouse button while your cursor is on a device within the main device map window. A shortcut menu like the one shown below 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

options for root devices such as ComPorts or IPPorts. When you right click a ComPort, only valid connections for ComPorts will be presented.



To delete a device from the network map select the device and either click the Delete button or select Delete from the Edit menu. This will delete the device and any devices that were connected below it. A keyboard shortcut Ctrl+D will also delete the selected device.

## 4.2.2 Applying Changes, Undo and Redo

The device map is not saved or entered in LoggerNet until you click the Apply button at the bottom of the screen. You can build a complete network and set up the configurations for all of the devices without applying. However, it is a good idea to build the network map in stages and periodically apply changes. If there is a problem with the computer, any changes that have been applied have been saved and will not have to be entered again.

Changing the network map or any of the device settings enables the Undo button. Clicking the Undo button will roll back each change in reverse order to the originally saved network and settings. If you undo a change and really wanted to keep it, you can click the Redo button and restore the change.

Once the changes to the network map and device settings have been applied, they can no longer be rolled back or restored using the Undo or Redo button.

Clicking the Cancel button before changes are applied will undo all of the changes to the network map and settings, and restore the saved configuration.

## 4.2.3 Renaming Network Devices

The names of all of the devices can be changed as desired. Rename a device by selecting the device and either clicking again with the left mouse button on the selected device, clicking the Rename Device button, 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.

Device names can reflect a location, layout, or physical location of network devices. Think carefully when naming the devices since these names are used throughout LoggerNet to refer to the devices.

## 4.2.4 Device Settings

When you highlight any device on the network shown on the left side of the Setup window, configuration tabs appear on the right side with the relevant settings. These settings are different for different devices and are described in detail below. Some of the tabs have a Standard section and an Advanced section. The Standard section contains information that must be reviewed to ensure it matches the settings for the device. The Advanced section contains settings that can be left at the default for most applications.

As with changes to the network map, the changes made to the device settings are not used until they have been applied.

### 4.2.4.1 ComPort

The ComPort (or serial port) has only a Hardware tab to configure.

#### **Standard**

**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.

**ComPort 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 the ComPorts that are set up on your computer.

#### **Advanced**

**Call-back Enabled** - Enabling call-back tells LoggerNet to watch for a call-back from the datalogger on this port. If there is a phone modem attached it will be set to accept incoming calls.

**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 LoggerNet should delay before terminating 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 1 or 2 seconds.

**Delay Hangup** - The amount of time, in milliseconds, that LoggerNet should delay before hanging up the link to the device. If a new command to the device is issued before the delay has expired, communication will not be terminated.

**COM Port Communication Delay** - The amount of time, in milliseconds, that LoggerNet will wait after opening a communication port and before sending data to the device. This can be useful when addressing drivers that require a short delay before accepting data (such as an IRDA port driver).

---

**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.

---

#### 4.2.4.2 IPPort (Internet Protocol Serial Port)

Like the standard serial port, configuration for the IPPort has only the Hardware tab. Following is an explanation of each of the fields on this form.

##### **Standard**

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

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

**Internet IP Address** - In this field, enter the TCP/IP address and port through which LoggerNet 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 123.123.123.123:1024.

##### **Advanced**

**Call-back Enabled** - Enabling call-back tells LoggerNet to watch for a call-back from the datalogger on this port.

**Cache IP Address** - When selected, once the DNS is resolved LoggerNet will cache the IP address for future use. Caching the IP address can reduce the amount of time it takes to establish communication. In most instances, this

check box should be enabled (the default), unless the target address must be resolved through dynamic DNS.

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

**Delay Hangup** - The amount of time, in milliseconds, that LoggerNet should delay before hanging up the link to the device. If a new command to the device is issued before the delay has expired, communication will not be terminated.

**IP Port Used for Call-back** - If call-back is enabled for the IP port, enter the port number that LoggerNet should open and monitor for incoming call-back messages.

**AirLink Device ID** - If an AirLink modem is being used, enter the Device ID set in it. By default, this is the 11-digit Electronic Serial Number of the device.

---

**NOTES**

When entering the IP address, do not use leading zeros for the address numbers. For example use 123.123.2.34 instead of 123.123.002.034.

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.

---

#### 4.2.4.3 TAPIPort (Telephony API)

The TAPI port uses the phone modems that have been installed and configured in Windows. This eliminates the need for LoggerNet to specify the modem type or work with initialization strings. Like the standard serial port, configuration for the TAPI port has only the Hardware tab. Following is an explanation of each of the fields on this form.

##### **Standard**

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

**TAPI Line** – Select the modem you want to use for communication. The modems listed are defined by Windows as part of the computer's Modem Setup. All of the parameters for the modem, including the baud rate have to be set using the Windows Modem Setup dialog. If you are using the same modem for dialup access you may have to change the settings for the different applications.

**Advanced**

**Call-back Enabled** - Enabling call-back tells LoggerNet to watch for a call-back from the datalogger on this port. If there is a phone modem attached it will be set to monitor for incoming calls.

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

**Delay Hangup** - The amount of time, in milliseconds, that LoggerNet should delay before hanging up the link to the device. If a new command to the device is issued before the delay has expired, communication will not be terminated.

**NOTES**


---

To communicate with dataloggers using the TAPI modem you have to set the baud rate to match the communication capability of the devices in the link. If you are using COM200 modems, the baud rate must be set to 9600 on the TAPI modem. For use over cell phone modems 1200 or 4800 baud may be required.

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.

---

**4.2.4.4 Datalogger**

Dataloggers have several different tabs. Similar to the 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, whether to automatically update the datalogger's clock, and a tab to send a program. Note that not all dataloggers will have all the settings described below.

**4.2.4.4.1 Hardware Tab****Standard**

**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.

**Call-back Enabled** - Enabling call-back tells LoggerNet to watch for a call-back from this datalogger. The parent PakBus port will be set up to receive incoming calls.

**Maximum Time On-line** - A time limit can be set for the length of time LoggerNet will stay connected to the datalogger on any scheduled call. Once this time limit has been exceeded, communications with the device will be terminated. Setting the time to zero will disable this time limit. This setting can be set to zero for most communications links without serious repercussion. However, with phone modem communication you may wish to leave this setting at the default of 10 minutes, to avoid costly phone bills.

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

**PakBus Address** - Each device in a PakBus network has a unique address. Valid addresses are 1 through 4094. 4094 is a broadcast address, and is therefore reserved for the LoggerNet PC.

### **Advanced**

**Extra Response Time** - In this field, specify the additional time that LoggerNet should delay before terminating 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.

**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.

**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. The security code is set by the datalogger program, through a keyboard display, or the remote keyboard utility. If a datalogger program that sets or changes security is loaded into the datalogger, the Security Code in LoggerNet must be changed to match so that the server can access the datalogger. (Security is not available in the CR5000, CR9000, and CR200 Series dataloggers.)

**Call-back ID** - Call-back is a mode supported by some dataloggers where an instruction in the datalogger program can initiate a call to the computer. The call-back ID is sent by the datalogger to identify itself when contacting the host computer. LoggerNet uses this value to know which datalogger initiated the call-back.

**Delay Hangup** - The amount of time, in milliseconds, that LoggerNet should delay before hanging up the link to the device. If a new command to the device is issued before the delay has expired, communication will not be terminated.

**NOTES**

---

Maximum Time On-line only applies to scheduled communications. Using the Connect screen to connect to the datalogger will stay connected until terminated by the user.

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.

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

---

**4.2.4.4.2 Schedule Tab**

The Schedule tab defines when LoggerNet will automatically check the datalogger for new data.

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

**Apply to Other Stations** - This button allows the schedule setup for this datalogger to be copied to other stations in the network. Clicking the button brings up a window that lists all of the dataloggers in the network. You can select one or more dataloggers and then press OK to use the entered schedule. To select more than one datalogger, hold down the Ctrl key while clicking the dataloggers to select.

**Base Date** - The base 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 when the schedule is enabled and applied.

**Base Time** - This field is used to define the first time for scheduled data retrieval. As with the Base Date field, if the time has already passed, a data collection attempt will be made when the schedule is enabled and applied. This setting is also used with the Collection Interval to determine the time data collection will be performed.

**CAUTION**

---

Entering a zero for any of the intervals below will cause LoggerNet to try and collect as fast as possible.

---

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

Example: If the Base 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.

**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 all the collection attempts fail, then the server will commence calling on the Secondary Retry Interval if it is enabled.

**Secondary Retry Interval** - If the secondary retry interval box is checked, the specified interval is a calling interval that will be followed 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. If the box is not checked the collection schedule will return to the normal collection schedule and continue through the primary retry schedule until communications are restored.

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.

**Collect Ports and Flags** - If this box is checked, the current state of the ports and flags is collected and stored in LoggerNet's internal data cache. This allows functions such as the Numeric Display and view ports and flags to get updated data with scheduled data collection.

**When the Server's Table Definitions are Invalid** - This option determines what action should be taken in the data cache during scheduled data collection when LoggerNet determines that the table definitions it has stored for a table-based datalogger and the table definitions actually in the datalogger do not match.

- **Automatically Reset Changed Tables** - LoggerNet will reset (delete and recreate) any tables that have changed. Unchanged tables will not be reset. Scheduled data collection will continue without action from the user.
- **Stop Collection Until Manually Updated** - Scheduled data collection will be halted until the user manually updates the table definitions (Setup window, Data Files tab, Get Table Definitions button). At that time, the user will be prompted to Merge or Reset the table definitions.

#### 4.2.4.4.3 Final Storage Area 1 and 2 Tab (Edlog Dataloggers with Mixed-array Operating System)

Mixed-array dataloggers include the 21X, CR500, CR510, CR10, CR10X, 21X, CR23X, and CR7. When the datalogger program stores data in a mixed-array datalogger, the data arrays are stored in a final storage area. Some dataloggers, such as the CR10X, have two final storage areas while others, such as the 21X, have only one. This tab is used to define the output file name and location, the data file format and other output options for the data stored in the final storage area.

**Enabled for Collection** - The specified final storage area will be included in the collected data if this box is checked.

**Output File Name** - This is the name and directory path for the output file where the final storage data will be saved after being collected from the datalogger during manual data collection from the Connect window or during scheduled data collection.

**Use Default File Name** - Checking this box will set the collected data file name to the default value, which consists of the name of the station and number of the final storage area.

**File Output Option** - This option allows you to choose whether new data collected from the station is appended to the data file, overwrites the old data in the data file, or is not stored to a data file. The default option is to append to the data file so the old data is not lost. If the data file is used only for a real-time display or such that only the last data collected is needed, overwrite can be used to replace the old data with the new collected data. If the data is only going to be used within LoggerNet for display on the Connect Screen graph or numeric display, or for RTMC, you can choose no output file and a limited amount of data will be kept in LoggerNet's internal data cache.

**Output Format** - Select the format for the output file.

- **ASCII, Comma separated** writes data to the file in ASCII text format one record per line with commas between the data values. This file can be opened in LoggerNet View, a text editor, processed using Split, or brought into a spreadsheet application.
- **ASCII, Printable** writes data to the file in ASCII text format separated into columns separated by tabs. The column number precedes each data value in the record. Only 80 characters will be placed on each line, columns that don't fit the 80 characters are placed on the next line. This file format can be opened in LoggerNet View, a text editor, or processed using Split. See the example data file below.

```
01+0109. 02+2002. 03+0038. 04+1639. 05+15.00 06+13.20 07+24.79 08+073.9
09+269.0 10-1.000
01+0109. 02+2002. 03+0038. 04+1639. 05+25.00 06+13.20 07+24.79 08+073.9
09+279.0 10-.988
01+0109. 02+2002. 03+0038. 04+1639. 05+35.00 06+13.20 07+24.79 08+074.0
09+289.0 10-.946
01+0109. 02+2002. 03+0038. 04+1639. 05+45.00 06+13.20 07+24.79 08+074.0
```

```
09+299.0 10-.875
01+0109. 02+2002. 03+0038. 04+1639. 05+55.00 06+13.20 07+24.79 08+074.0
09+309.0 10-.777
01+0112. 02+2002. 03+0038. 04+1640. 05+0.000 06+13.20 07+24.79 08+074.0
09+074.1 10+1638. 11+18.00 12+13.19
01+0109. 02+2002. 03+0038. 04+1640. 05+5.000 06+13.20 07+24.78 08+074.0
09+319.0 10-.656
```

- **Binary** writes the data to a file in a binary format. The advantage of the binary format is that it is more compact so the size of the file is much smaller than for the ASCII based files. The disadvantage is that it's unreadable except using View or by post-processing with Split.

**Collect Mode** - The collect mode allows you to choose how much data to collect when getting data from the datalogger.

- **Data Logged Since Last Collection** - When LoggerNet calls the datalogger to collect data, it will try to get all of the data stored by the datalogger since the previous call. If this is the first call to a datalogger there might be a lot of historical data stored. When Collect All on First Collection is checked, LoggerNet will collect all data in the datalogger the first time data is collected. If Collect All on First Collection is not checked, the first call to the datalogger will collect the number of arrays specified in the Arrays to Collect on First Collection field. This allows you to avoid keeping communications tied up while all the historical data is collected.
- **Most Recently Logged Arrays** - This option is used when you are interested in only the most recently stored data. When this option is selected you can specify how many arrays back from the most recent array should be included when data is collected from the datalogger.

#### 4.2.4.4.4 Data Files Tab (CRBasic Dataloggers, and Edlog Dataloggers with Table Data and PakBus Operating systems)

Table-based dataloggers include the CR10T, CR510TD, CR10X-TD, CR23X-TD, CR1000, CR5000, CR9000, and CR200 Series. Data output to final storage is stored as records in tables. The Data Files tab is used to define what data tables will be collected from the datalogger, along with the output file name and format.

**Tables to be Collected** - All of the available tables in the datalogger are listed in the column on the left. If no tables are listed, click the Get Table Definitions button. The tables selected for collection are shown with a green check mark and the excluded tables are shown with a red 'X'. Data from the selected tables will be collected from the datalogger during scheduled data collection. For a description of the built-in tables see Appendix B.3.

The individual tables can be highlighted by clicking the table name. The settings on the right side of the window apply to the highlighted table. The name of the highlighted table appears at the top of the settings. Double clicking a table name will toggle collection of that table on or off.

**Included for Scheduled Collection** - If this box is checked the specified table is included in data collection. This can be changed either by clicking the check box or double clicking the name of the table in the list.

**Output File Name** - This setting defines the file name and path for the output data file that contains the data collected from the datalogger. Clicking the browse button ( ... ) at the right of the box will allow you to choose another directory or file name for the collected data. The data from each table is stored in a separate output file.

**Use Default File Name** - Checking this box will set the collected data file name to the default value, which consists of the name of the station and the name of the table.

**File Output Option** - This option allows you to choose whether new data collected from the station is appended to the data file, overwrites the old data in the data file, or is not stored to a data file. The default option is to append to the data file so the old data is not lost.

If the data file is used only for a real-time display or such that only the last data collected is needed, overwrite will replace the old data with the newly collected data.

If the data is only going to be used within LoggerNet for display on the Connect Screen graph or Numeric Display, or for RTMC, you can choose no output file and the data will only be kept in LoggerNet's internal data cache (see Appendix C.2 for more about the data cache).

**Output Format** - Select the format for the output file.

ASCII, table date, no header - The data is output with timestamp and data values separated by commas with no header.

ASCII table data, short header (TOAC11) - The data is output in Table Oriented ASCII format type 1 which has a two line header and data formatted the same as the ASCII table format.

ASCII table data, long header (TOA5) - The data is output in Table Oriented ASCII format type 5 which has a multi-line header and data values separated by commas.

Binary table data (TOB1) - The data is stored in Table Oriented Binary format type 1.

**Collect Mode** - The collect mode allows you to choose how much data to collect when getting data from the datalogger.

- **Data Logged Since Last Collection** - When LoggerNet calls the datalogger to collect data, it will try to get all of the data stored by the datalogger since the previous call. If this is the first call to a datalogger there might be a lot of historical data stored. When Collect All on First Collection is checked, LoggerNet will collect all data in the datalogger the first time data is collected. If Collect All on First Collection is not checked, the first call to the datalogger will collect the number of records

specified in the Records to Collect on First Collection field. This allows you to avoid keeping communications tied up while all the historical data is collected.

- **Most Recently Logged Records** - This option is used when you are interested in only the most recently stored data. When this option is selected you can specify how many records back from the most recent record should be included when data is collected from the datalogger.

**Get Table Definitions** – When this button is pressed, LoggerNet will query the datalogger for its table definitions. This should only be needed the first time connecting to a station or when the datalogger program has changed. New table definitions will cause the previous output data file to be saved with a different name and a new data file will be created to save the data.

#### 4.2.4.4.5 Clock Check/Set Tab

**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, any time a clock set is performed (manual or automatic). A positive value sets the datalogger's clock ahead of the PC's clock; a negative value sets the datalogger's clock behind the PC's clock. This may be useful if the server and the datalogger are in different time zones.

**Enabled** – Select this box to automatically compare the datalogger's clock to the LoggerNet 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 event. Special consideration should be given if the PC clock automatically adjusts for Daylight Savings Time.

Refer to Section 4.2.6 for additional information on setting and checking the clock.

**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 data collection.

**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 data collection.

**Interval** - The interval at which a clock check should be performed is specified in the Interval field. If this interval is set at 1 day, 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.

**Check Clocks** - Press this button to manually initiate a clock check of the LoggerNet server and datalogger clocks. The two values are displayed in the PC System Date/Time and Station Date/Time fields, respectively.

**Set Station Clock** - Press this button to manually set the clock to that of the LoggerNet server.

---

**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.

---

#### 4.2.4.4.6 Program Tab

The program tab displays the name of the program currently running in the datalogger, if it is known by the LoggerNet server. Table-based dataloggers store program information in the table definitions. If the table definitions have been retrieved from the datalogger, LoggerNet should know the name of the running program. However, mixed-array dataloggers do not store the program name internally. LoggerNet is made aware of the program name and output table information when a new program is sent, or when a DLD file that contains this information is associated with the datalogger. To send a new program to the datalogger, press the Send button. To associate a program name with the datalogger, press the Associate button (associate applies only to mixed-array dataloggers).

#### 4.2.4.5 PhoneBase

The PhoneBase is a telephone modem connected to one of the server's ComPorts to provide access to other devices in the datalogger network. The PhoneBase has only the Hardware tab. This device must be properly installed and configured in the operating system to use one of the computer's ComPorts before it can be used by LoggerNet.

##### **Standard**

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

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

**Edit 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 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. If you change the settings for one of the standard modems you will have to save it to a new name to use it. The only modems that can be deleted from the list are modems that have been added by the user.

**Modem Type** - Use the drop down list box to select the type of modem that is attached to the server computer's communications port. In most instances, the <default modem> should work.

### **Advanced**

**Extra Response Time** - In this field, specify the additional time that the LoggerNet server should delay before terminating 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.

**Delay Hangup** - The amount of time, in milliseconds, that LoggerNet should delay before hanging up the link to the device. If a new command to the device is issued before the delay has expired, communication will not be terminated.

---

### **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.

---

#### **4.2.4.6 PhoneRemote**

The Hardware tab of the remote phone modem is used to set up the dialing string for the attached remote device. It 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 is used to enter the telephone numbers or dialing strings for the remote modem. To add a number to the list simply click the <add phone number> field and type in the number. Spaces, dashes and parentheses are usually ignored by the modem. To remove or change a number, highlight the number and delete or click the number and use backspace.

Multiple phone numbers 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. As an entry is created a new line

is added for additional entries as needed. The first phone number will be dialed following the specified delay. The next number will then be dialed after the delay specified. The amount of time to delay is in milliseconds so a 5-second delay would be entered as 5000 milliseconds.

#### 4.2.4.7 RFBASE

The RF base modem acts as a gateway device that provides RF communication with the remote RF modems connected to dataloggers at the field site.

##### **Standard**

**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 base is enabled.

**Maximum Time On-line** - A time limit can be set for the length of time LoggerNet will stay connected to the device on any scheduled call. Once this time limit has been exceeded, communications with the device will be terminated. Setting the time to zero will disable this time limit. This setting can be set to zero for most communications links without serious repercussion. However, with phone modem communication you may wish to leave this setting at the default of 10 minutes, to avoid costly phone bills.

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

##### **Advanced**

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

**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.

**Delay Hangup** - The amount of time, in milliseconds, that LoggerNet should delay before hanging up the link to the device. If a new command to the device is issued before the delay has expired, communication will not be terminated.

---

#### **NOTE**

The RFBASE device is not used for RF400 radios.

---

#### 4.2.4.8 RFRemote

The RF modem has only a Hardware tab.

##### **Standard**

**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.

**Address** - The hardware for each RF modem is configured for a certain address with internal switches. 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 Address field. Each RF modem used as a repeater only site still needs its own unique Address.

##### **Advanced**

**Use F Command** - The “F” command forces the baud rate to 9600. In the modem enabled (ME) state, the serial I/O port of the end-of-link modem will communicate with the datalogger at 9600 baud with the “F” command. In the synchronous device communication (SDC) state, the baud rate from the computer to the start-of-link modem will be 9600.

**Use U Command** - The “U” command will force RF communication between radios to 2400 baud rather than 3000 baud. DC95s purchased before February 1989 can only be used at 2400 baud. For further information see Appendix F in the Radiotelemetry Network Instruction Manual.

**Use W Command** - The “W” command will force the RF modem to wait until there is no carrier detect before transmitting. This option is used when the computer is connected to more than one RF base.

**Custom Dial String** – The custom dial string is specifically used to send commands to an RF95 modem. The values entered into this field will be inserted between the S command and the string of switch identifiers sent when the modem is dialed.

#### 4.2.4.9 RF95T

The RF95T device is used to set up communication for a table-based RF95 base modem.

##### **Standard**

**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.

**Maximum Time On-line** - A time limit can be set for the length of time LoggerNet will stay connected to the device on any scheduled call. Once this time limit has been exceeded, communications with the device will be terminated. Setting the time to zero will disable this time limit. This setting can be set to zero for most communications links without serious repercussion.

However, with phone modem communication you may wish to leave this setting at the default of 10 minutes, to avoid costly phone bills.

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

**Advanced**

**Extra Response Time** - In this field, specify the additional time that LoggerNet should delay before terminating the communications link if there is no response from the RF95T. Additional time may be needed in instances where the communications link is noisy or network traffic is heavy.

**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.

**Delay Hangup** - The amount of time, in milliseconds, that LoggerNet should delay before hanging up the link to the device. If a new command to the device is issued before the delay has expired, communication will not be terminated.

#### 4.2.4.10 RF95TPass

This device is used to configure the remote modem in an RF95T network.

**Standard**

**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.

**Address** - The hardware for each RF modem is configured for a certain address with internal switches. 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 Address field. Each RF modem used as a repeater only site still needs its own unique Address.

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

**Advanced**

**Extra Response Time** - In this field, specify the additional time that LoggerNet should delay before terminating the communications link if there is no response from the RF95T remote. Additional time may be needed in instances where the communications link is noisy or network traffic is heavy.

**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.

#### 4.2.4.11 MD9 Base

The MD9 base modem has only a Hardware tab.

---

**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.

---

#### **Standard**

**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 LoggerNet will stay connected to the device on any scheduled call. Once this time limit has been exceeded, communications with the device will be terminated. Setting the time to zero will disable this time limit. This setting can be set to zero for most communications links without serious repercussion. However, with phone modem communication you may wish to leave this setting at the default of 10 minutes, to avoid costly phone bills.

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

#### **Advanced**

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

**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.

**Delay Hangup** - The amount of time, in milliseconds, that LoggerNet should delay before hanging up the link to the device. If a new command to the device is issued before the delay has expired, communication will not be terminated.

#### 4.2.4.12 MD9 Remote

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

##### **Standard**

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

**Address** - 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 Address field.

**Maximum Time On-line** - A time limit can be set for the length of time LoggerNet will stay connected to the device on any scheduled call. Once this time limit has been exceeded, communications with the device will be terminated. Setting the time to zero will disable this time limit. This setting can be set to zero for most communications links without serious repercussion. However, with phone modem communication you may wish to leave this setting at the default of 10 minutes, to avoid costly phone bills.

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

##### **Advanced**

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

**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.

#### 4.2.4.13 RF400

If the RF400 is being used in a point-to-point network (one base radio to one remote radio) or in a PakBus network, where all of the settings in the radios are identical, then the communications link can be depicted on the device map as a direct connection (COM Port with datalogger or PakBus routing device attached -- no RF400s shown in the device map). However, in a point-to-multipoint network where all remote radios have a separate address, the RF400s are depicted on the device map. Refer to the RF400 Users Manual for complete information on RF400 radio setup.

The RF400 has only one tab, the Hardware tab.

### **Standard**

**Communication Enabled** - Before communication can take place, all devices in the chain must be enabled. When this box is selected, the RF400 radio is enabled for communication.

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

**Attention Character** - Enter the character that will be used to reset the RF400 modem. By default, the radios are programmed to use the + character as the Attention Character. However, if the RF400 is being used in a communications link that includes a phone modem, you will most likely need to change this character in the RF400 radio setup and on LoggerNet's Setup window. Most phone modems use + as the reset character, and sending this character unexpectedly will reset the modem and terminate the communications link.

### **Advanced**

**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.

## **4.2.4.14 RF400 Remote**

This device is used to set up the RF remote in a point-to-multipoint RF communication network containing CSI's RF400 modems. It has only one tab, the Hardware tab.

### **Standard**

**Communication Enabled** - Before communication can take place, all devices in the chain must be enabled. When this box is selected, the RF400 radio is enabled for communication.

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

**Network Address** - Enter the network address that is set up in the RF400 radio. A unique network address is required only if there is more than one network of dataloggers within the communication range of the network you are configuring; otherwise, the default of 0 can be used. All devices in a network must have the same radio network address. Valid Radio Net Addresses are 0 through 63.

**Radio Address** - This is the unique radio address for the RF400 remote. Valid addresses are 0 through 65,535.

**Advanced**

**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.

**4.2.4.15 Generic Modem**

The Generic Modem is used to set up any device in the communications network whose behavior is controlled by scripts. No default strings (such as reset and initialization strings) are sent; therefore, all commands must be entered by the user. The generic modem has two tabs that must be configured, Hardware and Modem.

**Hardware Tab, Standard**

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

**Maximum Time On-line** - A time limit can be set for the length of time LoggerNet will stay connected to the device on any scheduled call. Once this time limit has been exceeded, communications with the device will be terminated. Setting the time to zero will disable this time limit. This setting can be set to zero for most communications links without serious repercussion. However, with phone modem communication you may wish to leave this setting at the default of 10 minutes, to avoid costly phone bills.

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

**Hardware Tab, Advanced**

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

**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.

**Delay Hangup** - The amount of time, in milliseconds, that LoggerNet should delay before hanging up the link to the device. If a new command to the device is issued before the delay has expired, communication will not be terminated.

**Modem Tab**

**Dial script** - Enter the complete ASCII text string (initialization commands, telephone number, etc.) that is required to set up the device for communication.

**End script** - Enter the reset string that should be sent at the end of communication with this device. Care should be taken to ensure that the script puts the device in an off-line state.

**Half Duplex** - Select this check box to enable half-duplex communication. This means that communication will take place in one direction at a time; therefore, it will significantly slow the communication speed.

**Raise DTR** - Select this check box to set the DTR (data terminal ready) line high. This tells the remote device that the computer is ready to receive data.

**RTS CTS use** - Determines what mode to use for Request to Send/Clear to Send functions:

- Hardware handshaking will be enabled - The computer uses the RTS and CTS hardware lines to control the flow of data between the computer and the remote device. Most newer modems support hardware flow control.
- The RTS line will be raised - Sets the RTS (ready to send) line high. This tells the remote device that the computer is ready to send data.
- The RTS line will be lowered - Sets the RTS line low.

#### 4.2.4.16 PakBusPort

A PakBusPort must be added to the network map if you want to add a datalogger capable of PakBus communication (CR510-PB, CR10X-PB, CR23X-PB, CR1000, or CR200 Series). PakBus is a packet-based communications protocol developed by CSI for its dataloggers and some communications peripherals. PakBus offers a robust, efficient means of communication within larger datalogger networks, and allows routing of data from one datalogger (or other PakBus device) to another within the network. All PakBus devices within the network are assigned a unique address. Transmissions within the network can be broadcast to all devices or to only one device using the unique address.

PakBus Graph can be used to visually monitor and retrieve settings from devices in a PakBus network.

Refer to Campbell Scientific's PakBus Networking Guide for more information about setting up PakBus networks.

#### **Hardware Tab, Standard**

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

**PakBusPort is Dialed** - The computer running the LoggerNet server is included as a PakBus device in the network. Because of the nature of broadcast messages within the Pakbus network, the computer can keep the PakBus port open, and therefore, can "listen" for transmissions from other PakBus devices. In most instances, keeping this port open is not an issue. However, if there are other hardware or software components on your computer that must have

access to the physical port to which the PakBus port is attached, you will want to check the PakBus Port is Dialed box so that LoggerNet opens the port only when communication is initiated as part of scheduled data collection or manually by the user. This way, the port remains available for other uses, except when it is in use by LoggerNet.

**Maximum Time on Line** - A time limit can be set for the length of time LoggerNet will stay connected to the datalogger on any scheduled call. Once this time limit has been exceeded, communications with the device will be terminated. Setting the time to zero will disable this time limit. This setting should be set to zero for most communications links other than phone modems.

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

**Beacon Interval** - Routing devices in a PakBus network are capable of "learning" about other devices in the network. To learn about other devices in the network the router can be configured to send out a beacon, and any devices in the PakBus network that can "hear" the broadcast will respond with an identification message. The beacon interval is how often the computer will send out a beacon to the PakBus network. If the PakBus Port is Dialed check box is enabled, the beacon will not be transmitted at the specified interval unless LoggerNet is actively communicating over the PakBus Port.

#### ***Hardware Tab, Advanced***

**Extra Response Time** - In this field, specify the additional time that LoggerNet 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.

**PakBus Address** - This read-only field displays the address that has been assigned to the PakBus port. In most instances, the LoggerNet server has only one PakBus address that is used for all PakBus ports; however, if you want to have multiple independent PakBus networks, you can set the PakBus address for each port separately. Multiple PakBus ports can be assigned individual addresses using the Options | LoggerNet PakBus Settings menu option.

**Delay Hangup** - The amount of time, in milliseconds, that LoggerNet should delay before hanging up the link to the device. If a new command to the device is issued before the delay has expired, communication will not be terminated.

#### ***PakBus Nodes Tab***

The PakBus Nodes tab displays the address and device type of PakBus devices that LoggerNet has discovered (from direct communication or from the routing table of another device), but are not yet added to the network map.

Select a PakBus port in the device map and press Start Search to have LoggerNet query the network for devices. When a device is found, its PakBus address will be added to the Node PakBus ID column. When Get Device Type is pressed, the type of device (e.g., CR1000, CR10X-PB) will be displayed in

the Device Type column. To automatically add a device to the network map, highlight it and press Add To Network Map. The datalogger will be added to the network as the correct device type and PakBus ID.

## 4.2.5 Setting Up Scheduled Data Collection

### 4.2.5.1 Data Collection Scheduling Considerations

One of the goals in datalogging applications is to retrieve the data from the datalogger's memory to a computer so that it can be analyzed further. LoggerNet can be used to retrieve the data from the datalogger manually, on demand, or you can set up an automatic data collection schedule. When the schedule is enabled and LoggerNet is up and running, the LoggerNet server will initiate calls to the datalogger on the defined schedule and collect its data. When the data is collected, it is stored to a file (unless the No File option is chosen when setting up what data to collect), and it is also stored in LoggerNet's data cache. LoggerNet client applications, such as the Numeric and Graphical Displays or RTMC, retrieve their information from this data cache.

Remember that what data is collected, and how it is stored, is configured on the datalogger's Data Files or Final Storage Area tab. If a table or final storage area is not selected for collection, no data will be stored or displayed on the display screens. The update rate of display screens in LoggerNet will depend up two factors -- the rate at which data is being stored to an output table in the datalogger and the rate at which data collection from the datalogger is taking place.

Input locations and Public variables do not have to be scheduled for collection to be displayed on the Numeric Display or the Graph. These values are updated based on the update interval of the display (but limited by how fast measurements are actually being made in the datalogger). However, final storage data will be updated only when that data is collected using either a scheduled data collection or a manual data collection.

### 4.2.5.2 Intervals

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

#### 4.2.5.2.1 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 or Output Data Interval** – Most data tables or final storage arrays are set up to store data records at regular intervals. The data record consists of a record number and time stamp, followed by the output processing (i.e., sample, average, min, max, etc.) of the input location or variable values. This interval 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 that is stored to the datalogger's final storage memory.

#### **4.2.5.2.2 Data Collection Setting Intervals**

The collection interval at which the LoggerNet server requests new data from the datalogger is set up on the Schedule tab for that datalogger in the Setup Screen. If the collection interval 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) on CR10X-TD/PB family dataloggers or Public tables on CR1000, CR5000 or CR9000 dataloggers, the current values will be collected every time a scheduled collection occurs, whether or not the values have been updated. Therefore if your scan interval were one minute in the datalogger program, and the data collection interval were 30 seconds, you would get two identical records in the Public table for each one minute scan.

#### **4.2.5.2.3 Communications Path Considerations**

When setting up data collection intervals for the dataloggers you should consider the communications path to the datalogger, and its affect on how often you can retrieve data.

Phone modems require anywhere from 10 – 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 networks with repeaters also add time delays since each modem must be contacted and then pass the message on to the next RF modem and so on. Each of these operations takes time so the time schedule for RF networks with repeaters should allow enough time for the link to be established with the datalogger and collect the data.

Consideration should also be given to other operations such as clock sets, program downloads, or manual data requests 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 associated with RF data collection.

### 4.2.5.3 Setting Up Scheduled Data Collection

The data to be collected and the output file locations and format are specified on the datalogger's Data File or Final Storage tab. The Schedule tab is used to define the interval on which the LoggerNet server will check the datalogger for new data. If new data exists, it will be stored in the data files and the LoggerNet data cache.

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 or final storage areas should be collected from the datalogger each time a data collection attempt is made. If no tables are selected on the Data Files tab or the Final Storage Area is not enabled for collection, no data will be collected from the datalogger.

You should check the directory path and the data file options to make sure the files are where you want them and in the right format. Note that for table-based dataloggers, each table must be configured separately (i.e., selected for collection, file name provided, file format specified, etc.).

---

**NOTE**

For table-based dataloggers, if no table names appear on the Data Files Tab, click the Get Table Definitions button.

---

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 immediately.

The Status Monitor screen (Section 6) can be used to ensure that data collection is occurring on the defined schedule. Some issues will also be identified by the Troubleshooter (Section 6). If data is not being collected, check the following:

- The Scheduled Collection Enabled box on the Schedule 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 or final storage areas from which you desire data should be enabled for collection in the Data Files or Final Storage Area tab of the Setup Screen.

- All devices in the communications path to the datalogger must have the Communications Enabled check box on the Hardware tab selected.
- 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 Status Monitor.
- Look at the collection state data for the datalogger in the Status Monitor. This is displayed as one of four states.
  - Normal collection
  - Primary retry
  - Secondary retry
  - Collection disabled
- Check the Status window and ensure the Pause Schedule check box is cleared.
- For table-based dataloggers, ensure the table definitions have been retrieved from the datalogger and are current.

### 4.2.6 Setting the Clock

A datalogger's Clock tab can be used to define a schedule at which an automatic clock check will be performed. The datalogger's clock will be set 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.

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 PC's clock.

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 computer 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 computer's clock, the result would be duplicate time stamps that contained different data.

**NOTES**

---

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. 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.

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

Changing the computer system clock while the display screens are running will terminate the connection for most of the screens. This can also affect LoggerNet operations or even crash the program.

---

## 4.2.7 Sending a Program to the Datalogger from Setup

In most instances, you will send a new program to the datalogger from the Connect window. Once the program is sent from Connect, you can launch the Numeric Monitor from that window to view measurements and ensure the program is working correctly. However, for convenience when configuring a datalogger for communication and data collection, a program can also be sent from the Setup window .

Select the Program tab for the highlighted datalogger. If a program exists in the datalogger that LoggerNet has knowledge of, it will be displayed in the Current Program fields. To send a new program, press the Send button.

For mixed-array dataloggers, if the datalogger currently has a program running in it but LoggerNet is not aware of that program, you can Associate a program with the datalogger. When a program is associated with a datalogger, input location labels and final storage values will be available for viewing on LoggerNet's Numeric and Graphical Displays.

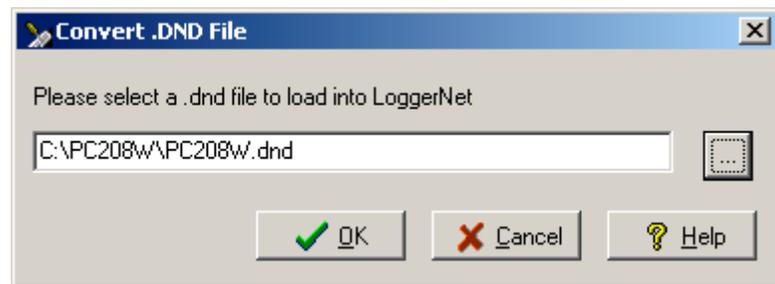
## 4.2.8 Setup's Options Menu

### 4.2.8.1 Convert from PC208W Network

The convert utility (Options | Convert from PC208W Network) converts a datalogger network that was created under PC208W version 3.0 or higher to a network that will work with LoggerNet. This allows you to keep the network map and configurations as they were with the PC208W software and avoid starting over.

It is a good idea to make sure that data collection with PC208W is up to date and all the data from the dataloggers has been collected and saved into files. You should also make a backup copy of the PC208W files.

To run Convert, shut down PC208W and bring up LoggerNet. Make sure that the network map in LoggerNet's Setup Screen is empty. If you try to convert a network map onto an existing network an error message will remind you that the current network map is not blank. (Make sure to apply the changes after you delete all the devices or the conversion will fail.)



Select “Convert from PC208W Network” from the Options menu on the Setup screen. A pop-up message will remind you that data collection should be up to date. Then the dialog box shown above will come up allowing you to choose the PC208W network description file (pc208w.dnd) you want to convert. You can either type in the path or click the button to the right with the ellipsis (...) that will allow you to browse for the file.

PC208W network descriptions are files ending with .dnd and have the name and path as shown in the dialog box above. Once you have found the network description file to convert, click OK and LoggerNet will run the scripts to create the network with settings in LoggerNet format. (The converter requires that wmodem.ini be in the same directory as the pc208w.dnd.)

The new network should have all of the same devices and communication paths as under PC208W. Note that LoggerNet shows almost all physical devices in the network map, where PC208W "assumed" some devices were present so the device map may look slightly different. For instance, LoggerNet will show both a PhoneBase and Phone Remote (the modem connected to your computer and the modem connected to the datalogger, respectively). PC208W showed only the PhoneBase. The remote modem connection was assumed.

The data files created by PC208W will remain in the PC208W directory. They are not copied over to the LoggerNet directory. However, the Output Filename field on LoggerNet's FS Areas 1/2 tabs will reflect the PC208W data file directory. Any newly collected data will be stored to the files in this directory unless you make changes. The new network is saved by pressing Apply. When collecting data, LoggerNet will begin collection based on where PC208W last collected data from the datalogger(s). If scheduled data collection was set up in PC208W, LoggerNet will resume that scheduled collection at the next time to call.

We strongly recommend that you review all settings once a conversion has been performed. Following are some areas to check:

- Review the directory where LoggerNet will save data files. You may want to change this to a different directory, since LoggerNet will collect data to the PC208W directory. If you want contiguous data files but want to store data in the LoggerNet directory (or some other directory), simply copy the existing data files to the desired directory and ensure it is specified in the FS Areas 1/2 tabs.
- Double-check all settings for all devices to ensure they were converted correctly. This includes modem settings, schedules, COM port selections, etc.
- Any Tasks that existed in PC208W will also be converted. Review these to make sure file names, directories, and schedules are correct. The Task Master can be opened from LoggerNet's main Toolbar from File | Open.

#### 4.2.8.2 LoggerNet PakBus Settings

The LoggerNet PakBus Settings are used to set up the PakBus ID for one or more PakBus ports in LoggerNet. LoggerNet can be configured so that each PakBus port in the network map will be an independent PakBus network (where communication between PakBus devices exists only in that network), or so that all PakBus ports in the network map are part of one PakBus network. To link -- or "bridge" -- all of the PakBus networks, select the Bridge PakBus Ports check box. The PakBus ID for LoggerNet is then entered into the PakBus ID for Global PakBus Router field.

If the PakBus ports are set up as separate networks, the PakBus ID for each PakBusPort is entered in the PakBus ID/PakBus Port table.

Valid PakBus IDs are 1 through 4094, though typically numbers greater than 3999 are used for PakBus ports. This is because, when a neighbor filter is set up, a PakBus datalogger will answer a Hello message from any device with an ID greater than 3999, but will ignore devices with IDs less than 4000 that are not in their neighbor list.

### 4.2.8.3 LoggerNet Clock Settings

LoggerNet allows you to select the time option that will be used for the LoggerNet server. There are three options:

**Use local time without correction for daylight saving time** - The LoggerNet server will use the clock from the computer on which LoggerNet is running. When the time changes to daylight saving time, the server's time is not affected.

**Use local time with correction for daylight saving time** - The LoggerNet server will use the clock from the computer on which LoggerNet is running. When the time changes to daylight saving time, the server's time will be adjusted.

**Use Greenwich Mean Time (GMT)** - The LoggerNet server will use Greenwich Mean Time, regardless of whether daylight saving time is applicable.

### 4.2.8.4 LoggerNet IP Manager Settings

These settings are used to configure LoggerNet for receiving incoming transmissions from an AirLink digital cell phone (Raven or PinPoint) with a dynamic IP address.

LoggerNet will "listen" for update notification messages from the modem, and change the IP address for a device based on the information received. A setting in the modem defines the interval at which transmissions to LoggerNet will be made.

The IP Manager Port is the UDP port on which LoggerNet will "listen" for an incoming transmission. If this setting is left at the default of 0, LoggerNet will not listen for incoming notifications. The default value for AirLink modems is 17338. However, if a firewall is in use, the port value may be changed when passed through the firewall.

The IP Manager Key is the 128-bit encryption key used for decoding transmissions from all AirLink modems used in the datalogger network. This setting must match the setting in the modems for the transmission to be successfully interpreted. LoggerNet's default value in the IP Manager Key field is the default for the AirLink modems.

Note that in addition to these settings, an ID for each modem is set on the IPPort's Hardware tab of the Setup screen (AirLink Device ID).

#### 4.2.8.4.1 Troubleshooting Tips

If the computer on which the LoggerNet server is running is set up behind a firewall, the administrator of that firewall must open a hole in the firewall to allow the incoming notifications to reach the server.

LoggerNet will be unable to open the UDP port if another process is using it. To determine what ports are being used by processes on the system, the following command can be issued at a command prompt:

**netstat -a -p UDP -n**

where:

- a** Displays all connections and listening ports
- p UDP** Specifies to show connections for UDP protocol
- n** Returns addresses and port numbers in numerical form

## 4.3 Device Configuration Utility

The Device Configuration, or DevConfig, is run separately from LoggerNet. It is used to set up PakBus information in PakBus-capable dataloggers and to configure peripheral communication devices. Refer to Section 12, Miscellaneous Clients and Utilities, for information on the Device Configuration Utility.

# Section 5. Real-Time Tools

LoggerNet's real-time tools are used to manage your stations in the datalogger network. Tools are provided for sending new programs, setting the clock, toggling ports and flags, collecting data, and displaying data numerically and graphically.

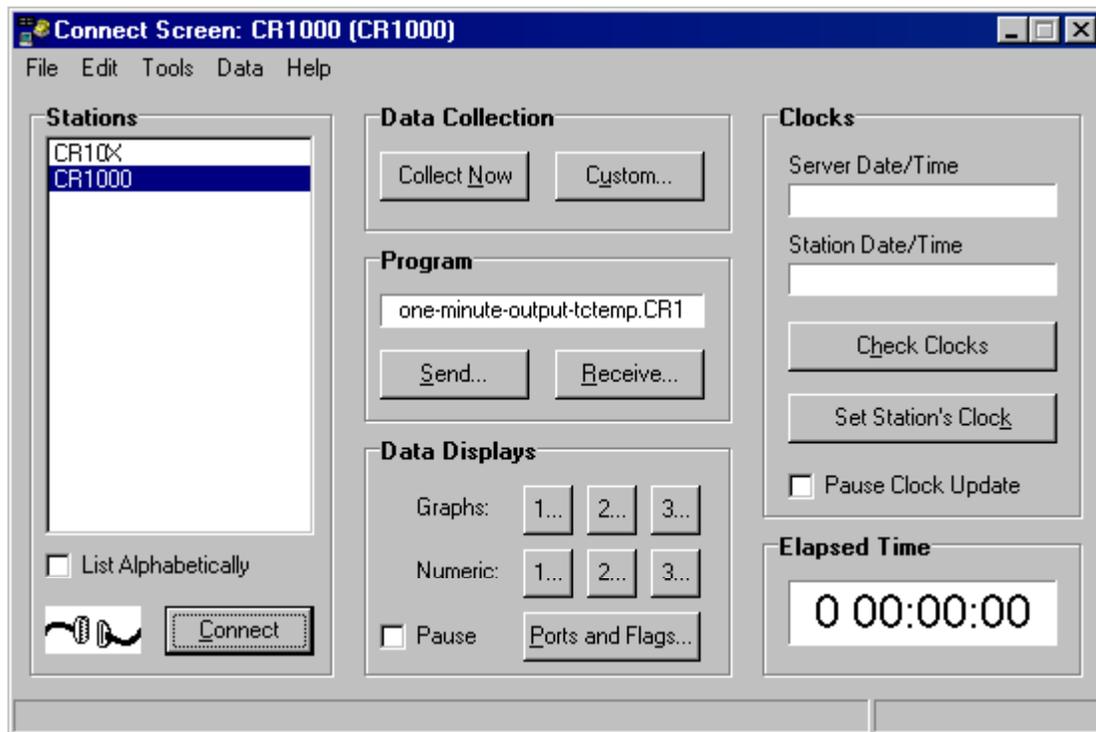
## 5.1 The Connect Screen Toolbox

The Connect screen provides a real-time connection to a datalogger in the datalogger network. Tools are provided for transferring programs to the datalogger, manually setting the datalogger's clock, viewing and collecting data, and communicating with the datalogger in terminal mode.

These tasks can be accomplished while actively connected to a station in the datalogger network, or, if you are not actively connected, LoggerNet will make the connection, perform the desired action, and then end communication with the station.

### 5.1.1 Connecting to the Datalogger -- or Not

The Connect screen works with and displays data from only one datalogger at a time. The name of the selected datalogger and the datalogger type appear in the title bar at the top of the window. For ease of use in large networks of many dataloggers you can list the dataloggers in alphabetical order by selecting the List Alphabetically check box. All the station names will be listed alphabetically to make it easier to find a specific station.



As noted above, you can work with a datalogger station while actively connected to it or when you are in a disconnected state. Even when not actively connected, you can choose to collect data, check or set the clock, etc. When a button is pushed, LoggerNet will attempt to contact the datalogger, performed the desired action, and then terminate communication.

If you want to perform only one task, such as collecting new data, it may be more efficient not to actively connect to the datalogger. LoggerNet will merely contact the datalogger, collect the data, and end communication. If you were actively connected, LoggerNet would also update the clock displays during this process, which, when collecting large amounts of data over a slow communication link, could affect the speed of data collection. If you want to perform multiple tasks -- e.g., send a new program and view measurements on the numeric monitor to ensure the program is running correctly -- then it is usually more efficient to establish an active connection, perform the tasks, and then terminate the connection yourself. Otherwise, LoggerNet must establish communication with the datalogger twice. Over remote communication links, this connect/disconnect/connect sequence will increase the time to complete the tasks.

When you select the Connect button the animated graphic will indicate an active connection state. It will show that LoggerNet is trying to establish the connection; the two connectors join together when the connection is made. You can also connect to the datalogger by double clicking the datalogger name or selecting Connect from the File menu.

Once the datalogger connection is established the Elapsed Time counter starts. This counter will continue as long as the datalogger connection is maintained. The user should be aware of the how long the Connect screen is connected to the datalogger. A manually initiated connection to a datalogger takes priority over other communication in the network map. Other devices that share any part of the communication path will not be contacted, even for scheduled data collection. Max Time Online is disabled when a manual connection is established so this communications link will be active until the user disconnects or closes the Connect screen.

---

**NOTE**

Once a connection is made to a datalogger with the Connect screen, this connection takes precedence and will prevent communication with other devices sharing the same serial port or base modem.

---

To disconnect from the datalogger, click the button that now reads Disconnect. To work with another datalogger you must disconnect from the first one. Double clicking another datalogger will disconnect from the first datalogger and connect to the new one without prompting.

If LoggerNet fails to make a connection to the datalogger, it will time out and display an error message that it could not connect. It will immediately attempt the connection again and will continue trying until the user clicks Cancel.

## 5.1.2 Data Collection

The Data Collection section of the Connect screen allows you initiate a manual data collection from the datalogger. LoggerNet keeps track of two separate data collection pointers for each datalogger: (1) the pointer for scheduled data collection and manual data collection from the Connect screen's Collect Now button, and (2) the pointer for manual data collection from the Custom Collection window.

The data for each of these two pointers is, by default, stored in two separate data files on the computer. The default directory for scheduled data collection/manual data collection is C:\Campbellsci\LoggerNet. The default directory for data collection via the Custom Collection window is C:\Campbellsci\LoggerNet\Data. Because different pointers are kept in LoggerNet for each of these collection options, if you select Collect Now from the main Connect window, and your Setup option is set to collect only new data, and then you do a Custom Collection and also choose to collect only new data, the new data collected using the Custom Collection window is the new data since the last time you collected *using this window*. Similarly, new data collected using Collect Now from the main Connect screen is the new data since the last time you chose Collect Now, or since the last scheduled data collection.

### 5.1.2.1 Collect Now

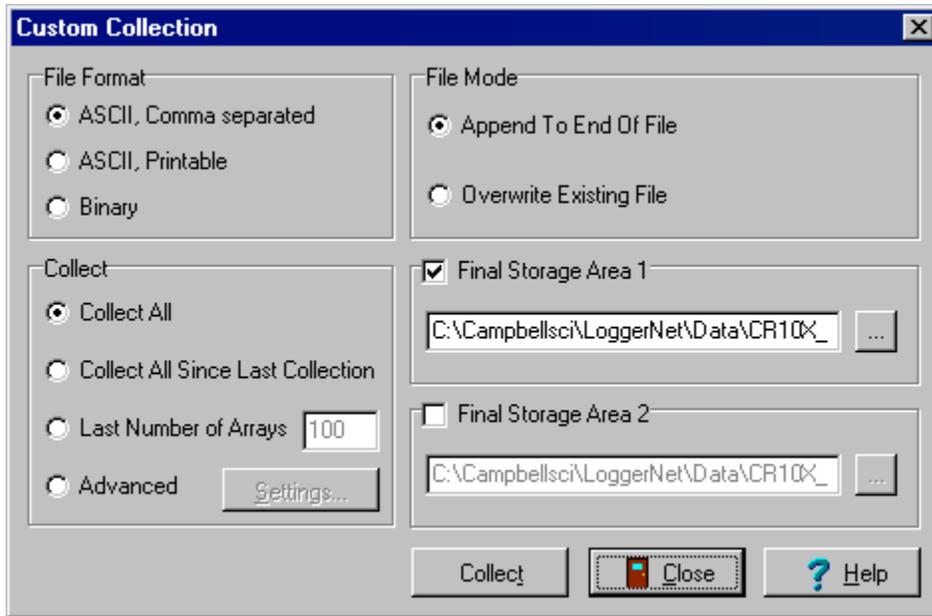
The Collect Now function is the equivalent of doing a scheduled data collection for the datalogger without waiting for the scheduled time. Clicking the Collect Now button will initiate a call to the datalogger and any available data will be collected and stored as specified on the Setup screen. This function is often used to manually update data collection to see the latest data in the Numeric Monitor or a Graph; or before data files are copied for processing.

Once you have started data collection with Collect Now, you can stop it by clicking the Cancel button on the animated screen. This might be necessary if you started a data collection that is bringing in more data than you really wanted, especially over a slow communications link.

### 5.1.2.2 Custom Collection

#### 5.1.2.2.1 Mixed-array Dataloggers

Clicking the Custom button for a mixed-array datalogger brings up the dialog box shown below. The options you can specify include the file format and whether to overwrite or append to the existing file, how much data to collect and which final storage areas you want to collect.

**NOTE**

While retrieving data from the datalogger using Custom Collection, scheduled data collection will be suspended. The directory where the files are stored for custom collection is separate from the files for scheduled collection data and by default is a Data directory under the LoggerNet directory (e.g., C:\CampbellSci\LoggerNet\Data).

**File Format**

This option is used to select the file format in which to store the collected data. Appendix XX provides information on File Formats.

**Collect**

- **Collect All** will get all of the data available in the selected final storage areas. For a datalogger that has a lot of data stored, this could result in a large file and take a long time.
- **Collect All Since Last Collection** - When this option is selected, LoggerNet will attempt to collect all the data since the last Custom Collection from the datalogger. Note that separate data collection pointers are kept for the Custom Collection option; therefore, collecting from this window will not affect data files created by scheduled data collection or a manually initiated collection from the main Connect window.
- **Last Number of Arrays** specifies how many of the last stored records will be retrieved from the datalogger.
- **Advanced** allows the user to specify the memory pointer address for each of the final storage areas. Data collection will begin at the specified memory pointer and go through the last record collected. Use of this option depends on knowing the memory pointer values for data collection.

Pointers can be manually reset but are updated and stored with each data collection.

### **File Mode**

The File Mode determines if the data to be collected will be added to the end of the data file, if it exists, or if it will overwrite an existing data file. This option only applies if a file with the same name exists in the directory specified.

- **Append to End of File** - When this option is selected, the data collected using the Custom Collection option will be appended to the end of the file from previous Custom Collection.
- **Overwrite Existing File** - When this option is selected, the newly collected data file will overwrite any existing file with the same name.

### **Select Final Storage Areas**

The Final Storage Area section on the dialog selects which final storage areas to collect data from. The file name and directory show where the collected data will be saved. Clicking the browse button ( ... ) next to the file name will bring up the file select dialog.

---

**NOTE** CR500, CR7, and 21X dataloggers have only one final storage area.

---

#### **5.1.2.2.2 Table-based Dataloggers**

Clicking the Custom button for a table-based datalogger brings up the dialog box shown below. The options you can specify include how much data to collect; the file format; whether to overwrite, append or create a new file; and the tables you want to collect. With table-based dataloggers you also have the option of choosing specific records or a time interval to collect.

Each table is saved in a separate file so there will be one file created for each table that is selected.

---

**NOTE** While retrieving data from the datalogger using Custom Collection, scheduled data collection will be suspended. The default data file names for custom collection are separate from the files for scheduled collection data and by default are placed in a Data directory under the LoggerNet directory.

---

**Custom Collection**

**Collection Options**

Collect Mode: Newest Number of Records

File Mode: Append to End of File

File Format: ASCII Table Data, Short header (TOACI1)

Starting Record Information

Starting Record #: 0

Number of Records: 100

Starting Date/Time

Date: 11/02/04 Time: 12:00:00 AM

Ending Date/Time

Date: 11/02/04 Time: 4:31:38 PM

**Table Collection**

Table	File Name
<input type="checkbox"/> Public	C:\Campbellsci\LoggerNet\Data\CR1000_Public.dat
<input type="checkbox"/> Status	C:\Campbellsci\LoggerNet\Data\CR1000_Status.dat
<input checked="" type="checkbox"/> TCTemp	C:\Campbellsci\LoggerNet\Data\CR1000_TCTemp.dat

Select All

Change File Name... Start Collection

Reset Tables... Close Help

### Collection Options

- **Newest Number of Records** will retrieve the number of records specified in the Starting Record Information area going back from the last record collected.
- **Specific Records** will retrieve the specified number of records beginning with the Starting Record # as specified in the Starting Record Information area.
- **Data Since Last Collection** will retrieve the data stored since the last time a custom collection was performed. LoggerNet keeps track of the records collected from each table every time a custom collection is executed. This option will work even if the last custom collection used a different option.
- **All the Data** will get from the datalogger all the data available from all of the selected tables. If the datalogger is full this could take a long time, especially with large memory dataloggers or over slow communication links.
- **Data From Selected Date and Time** uses the starting and ending time and dates to get the data from the datalogger stored between those times. If the datalogger does not have data for the specified time range a blank file will

be created. (CR200 Series dataloggers do not support this collection option.)

### ***Starting/Ending Date and Time***

The Starting and Ending Date and Time are used to specify the timestamp range for the selected date and time option. The date can be chosen from the drop down calendar and the time can be entered or edited using the arrows to the right of the control.

### ***Starting Record Information***

- **Starting Record #** is used to specify a range of records to collect. The data collection will begin at this record number and get the number of records specified in the Number of Records.
- **Number of Records** is used to specify a range of records or the number of records to go back from the last record stored in the datalogger.

### ***File Mode***

File Mode is used to choose whether the collected data should be appended to the file if it exists, overwrite the existing file, or create a new file. If Create New File is selected and the named file exists, a new file will be created with the specified file name and a sequence number added to it.

### ***File Format***

This option is used to select the file format in which to store the collected data. Appendix XX provides information on File Formats.

### ***Reset Tables***

(CR5000, CR1000, CR9000, and CR200 Series dataloggers only) lets you clear out the data and reinitialize the data tables in the datalogger and the data cache. Clicking Reset Tables will bring up a list of the tables defined in the datalogger. Select the tables you want to reset (or click the Select All check box to get all of the tables) and click OK. This will delete all the data stored in the datalogger for the selected tables and will also reset LoggerNet's data cache collect areas.

---

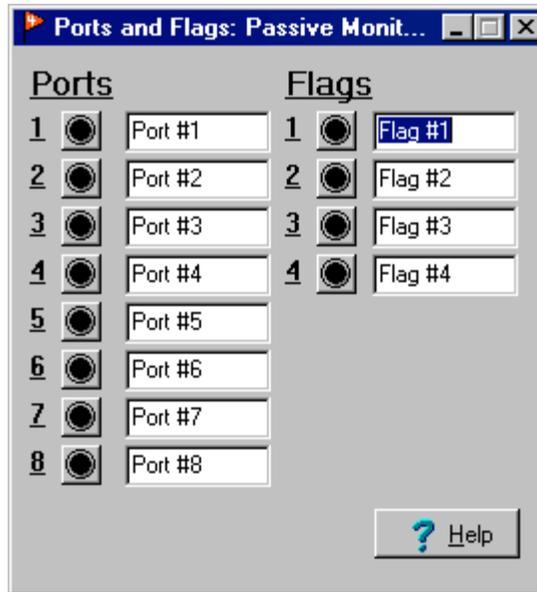
**WARNING**

**The Reset Tables function will delete the data stored in the datalogger and the data cache (Appendix D) for the selected tables. Make sure you do not need the stored data or the data has been saved to a file.**

---

### 5.1.3 Ports and Flags

The Ports and Flags window shows the current state of the ports and flags for the datalogger. If a flag is enabled or a port is high, the button next to the name appears green; if the flag or port is disabled or low, the button appears black.



Click the button next to the flag or port to turn it on or off. You do not have to be actively connected to the datalogger to toggle a port or flag. If you are not connected, when the port or flag is toggled, LoggerNet will connect to the datalogger, make the change, and disconnect.

Not all dataloggers have the same number of ports and flags. The Ports and Flags dialog box will display only those ports and flags available for the datalogger type. For CR1000, CR5000, CR9000, and CR200 dataloggers, LoggerNet will display up to 8 flags if a Public variable array by the name of Flag() has been dimensioned in the program. Otherwise, no flags will be available.

You can customize the labels for the Ports and Flags by clicking within the label field and typing the desired text.

---

#### NOTE

Remember that ports can only be set if they are configured as outputs in the datalogger program.

---

### 5.1.4 Datalogger Clock

If a connection to a datalogger is established, the datalogger's clock is checked continuously and displayed along with the computer's clock as updates are received. The clock update can be paused by selecting the check box next to Pause Clock Update. In some situations it is desirable to pause the clock update to minimize data traffic over the communications link.

You can manually set the clock by clicking the Set Station Clock button. LoggerNet attempts to set the datalogger clock as closely as possible to the computer clock. A slight difference in the clocks might exist after the clock is set because of the communications time delay. Over some communication links it is impossible to match the computer clock exactly. LoggerNet uses advanced compensation to get the best possible synchronization between the computer and the datalogger clocks.

If you are not connected to the datalogger, or if you are connected but the clock update is paused, you can press the Check Clocks button and LoggerNet will check both the datalogger's clock and the computer's clock and display the results on the screen.

An automatic scheduled clock check can be set up in the Setup screen (see Section 4.2.4.4.5).

Setting the clock may affect the time stamps assigned to your data. Refer to Section 4.2.6 for further discussion.

## 5.1.5 Program Management

The Program section on the Connect screen is used to send programs to or retrieve programs from dataloggers in the network. Edlog is used to create programs for the CR7, 21X, and the CR10(X), CR510, and CR23X Series dataloggers (CRXX, CRXX-TD, and CRXX-PB). The CRBasic Editor is used to create programs for the CR5000, CR9000, CR1000, and CR200 Series dataloggers. Short Cut can be used to create programs for any datalogger type. After a program is created in one of the program editors, the Connect screen is used to transfer it to the datalogger. (Program editors are discussed in Section 7.)

---

**NOTE**

Programs for the CR7, 21X, and the CR10(X), CR510, and CR23X Series dataloggers must be compiled in the editor to create the \*.dld file that is downloaded to the datalogger. The CR200 Series datalogger also requires a precompiled file (\*.bin), which can be done in the editor or when the program is sent using LoggerNet. CRX000 dataloggers compile their program on-board.

---

### 5.1.5.1 Sending a Datalogger Program

To transfer a program, press the Send button. A standard file select dialog box will come up so you can choose the file to send. The Files of Type selector at the bottom of the dialog can be used to filter the files that are displayed. This filter is set to the appropriate file type for each datalogger automatically (\*.dld for CR7, 21X, CR10(X), CR510, and CR23X dataloggers, and \*.CR# for the CR5000, CR1000, CR9000, and CR200 Series dataloggers).

After selecting a datalogger program file a warning will appear to remind you that data may be lost when the new program is sent. (For mixed-array dataloggers data is not lost if the memory configuration does not change; sending a new program to table-based dataloggers always clears all data

memory.) If there is any data in the datalogger that has not been collected, click Cancel to stop the program send, and collect the needed data.

If OK is selected at the warning, the progress bar will come up with the program transfer progress. Once the program has been sent, the text changes to Compiling Program. When the datalogger finishes compiling the program the progress box will close.

---

**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 Setup screen. Otherwise, access to the datalogger may be limited or completely denied.

---

### 5.1.5.2 CR200 Series Programs

Programs for the CR200 Series dataloggers must be precompiled before being sent to the datalogger. The compiled file is a binary image file with a \*.bin extension. Unlike the other dataloggers, CR200 Series dataloggers do not have an on-board compiler. Consequently, the \*.bin file must be generated with a version of the precompiler that matches the operating system in the datalogger, either during program creation in the CRBasic Editor, or when the file is downloaded to the datalogger by LoggerNet. If a \*.bin file is downloaded to a CR200, and the version of that binary file does not match the datalogger's OS, the download will fail and an error will be returned.

The LoggerNet installation includes all of the compilers for the CR200 that were available at the date of release. When sending a program, if you choose to send the \*.CR2 file, LoggerNet will first check the OS version of the datalogger and then attempt to compile the \*.CR2 file with the matching compiler. If you choose to send the \*.bin file, LoggerNet will not check the CR200's OS or precompile the file, it will just send the \*.bin file. In this instance, it is up to you to ensure that the \*.bin file was created with the correct precompiler.

CR200 dataloggers are shipped with a support software CD. This CD contains the precompiler that matches the OS of your datalogger. It may be necessary to copy this into the CR200 compiler directory so that it is available for LoggerNet's use. By default, this directory is C:\Campbellsci\Lib\CR200Compilers.

### 5.1.5.3 Retrieving Datalogger Programs

The program running in the datalogger can be retrieved and saved to a file by pressing the Retrieve button. You will be prompted for a name and directory in which to store the retrieved file. Files for the CRBasic dataloggers (\*.CR1, \*.CR5, \*.CR9) can be opened directly in the CRBasic Editor. A retrieved \*.dld file can be imported into Edlog for editing by using Edlog's Document DLD feature (Section 7). The binary image files (\*.bin) for the CR200 Series dataloggers cannot be retrieved.

This feature is useful if the original file has been corrupted, lost, or erased. Note that programs may not be reliably retrieved over noisy or slow communications links.

## 5.1.6 Program Association

A table-based datalogger maintains final storage table information internally -- this is referred to as the datalogger's table definitions. The LoggerNet server uses this information for the Add dialog box when you select values to view on a Numeric or Graphical display and for the LoggerNetData applications.

Mixed-array dataloggers do not store final storage information internally. However, this information is contained in the \*.dld file as commented text. When you download a program to a datalogger, LoggerNet uses the input location and final storage information from this file. If you communicate with a datalogger that already has a program in it, you can use the Tools | Associate Program option to select a \*.dld file from which LoggerNet should get this information. A program file can also be associated with a datalogger from the Setup window's Program tab.

Programs created with Edlog version 2.0 and greater include both the input location information and the final storage information in the \*.dld file. Previous versions of Edlog stored only the input location information in the \*.dld. If final storage information is not available for viewing in LoggerNet after associating the file, you may need to recompile the program file with a version of Edlog that stores this information in the \*.dld file.

---

### NOTE

If you are using Edlog Version 2.0 or greater and labels are still not available for use, check Edlog's Options | DLD File Labels menu item and ensure that labels are being stored in the file when the program is compiled.

---

## 5.1.7 Data Displays

The Connect screen's Data Displays provide display screens to monitor numeric values collected from the datalogger or to view the collected values graphically. Both types of displays get their data from the LoggerNet data cache, so the data must be collected from the datalogger before it can be displayed.

Scheduled data collection will supply updated final storage data for these displays automatically as it is received from the datalogger. Additionally, for table-based dataloggers only, an active connection will supply values for final storage data. For mixed-array dataloggers, final storage data must be collected first (either by scheduled data collection or manually from the Connect screen). Input locations and variables, along with port and flag information, are automatically retrieved from the datalogger when selected for display, regardless of the datalogger type.

Updates to the displays can be suspended by selecting the Pause check box. This will stop the updates to all displays even though new data may be coming in to LoggerNet.

---

**NOTE** If clock updates and display updates are paused while connected to a datalogger, the connection may time out and terminate the connection.

---

There are three Numeric Display screens and three Graphical Display screens. Each screen is launched as a separate window that can be moved or resized as needed. The display screens are not minimized if the Connect screen is minimized, but they can be minimized independently.

### 5.1.7.1 Data Display Limitations

There are limits to the number of Input Locations that can be retrieved for display from some of the dataloggers. Older mixed-array dataloggers can transfer 62 input location values at a time. The newer mixed-array dataloggers can transfer up to 500 input locations. The CR10X-TD family of dataloggers is limited to a maximum of 500 input locations by packet size and record format. CR5000, CR1000, and CR9000 dataloggers don't have practical limits for the number of Public Variables that can be requested or displayed.

Both mixed-array and CRBasic dataloggers will transfer only the requested input locations or public variables. However, CR10X-TD type dataloggers must transfer the entire input locations record in one packet. Because of packet size, if more than 500 input locations are used in the datalogger program the record cannot be collected.

### 5.1.7.2 Numeric Display Screens

Data values collected from the datalogger can be displayed in numerical format on the Numeric Displays. Selecting one of the three numbered boxes next to the text "Numeric" will bring up a Numeric Display screen. Each of the three Numeric Displays is configured separately, and all three can be active at the same time if desired. The settings and selected data values for a display are saved when the display, Connect screen, or LoggerNet is closed. The display settings for each datalogger are also saved independently, so a different datalogger will have different settings.

An example of a Numeric Display is shown below. If a Numeric Display is already active but hidden behind other windows, clicking its number will bring it to the front.

---

**NOTE** A mixed-array datalogger's final storage data must be collected by LoggerNet before it can be displayed.

---

RecNum	154146.00	RecNum	2569.00	TCTemp_TMx(4)	11:57:56 AM		
TimeStamp	1:58:23 AM	TimeStamp	11:58:00 AM	TCTemp_TMx(5)	11:57:49 AM		
PTemp	25.04	TCTemp_Avg(1)	24.42	TCTemp_Min(1)	24.34		
TCTemp(1)	24.47	TCTemp_Avg(2)	24.41	TCTemp_Min(2)	24.34		
TCTemp(2)	24.47	TCTemp_Avg(3)	24.39	TCTemp_Min(3)	24.32		
TCTemp(3)	24.47	TCTemp_Avg(4)	24.33	TCTemp_Min(4)	24.23		
TCTemp(4)	24.41	TCTemp_Avg(5)	25.44	TCTemp_Min(5)	25.39		
TCTemp(5)	25.46	TCTemp_Max(1)	24.51	TCTemp_TMn(1)	11:57:04 AM		
Flag(1)	false	TCTemp_Max(2)	24.49	TCTemp_TMn(2)	11:57:03 AM		
Flag(2)	false	TCTemp_Max(3)	24.46	TCTemp_TMn(3)	11:57:02 AM		
Flag(3)	false	TCTemp_Max(4)	24.41	TCTemp_TMn(4)	11:57:02 AM		
Flag(4)	false	TCTemp_Max(5)	25.48	TCTemp_TMn(5)	11:57:02 AM		
PortHigh	false	TCTemp_TMx(1)	11:58:00 AM				
		TCTemp_TMx(2)	11:58:00 AM				
		TCTemp_TMx(3)	11:57:58 AM				

Update Interval: 00 m 01 s 000 ms

Add... Delete Options... ? Help

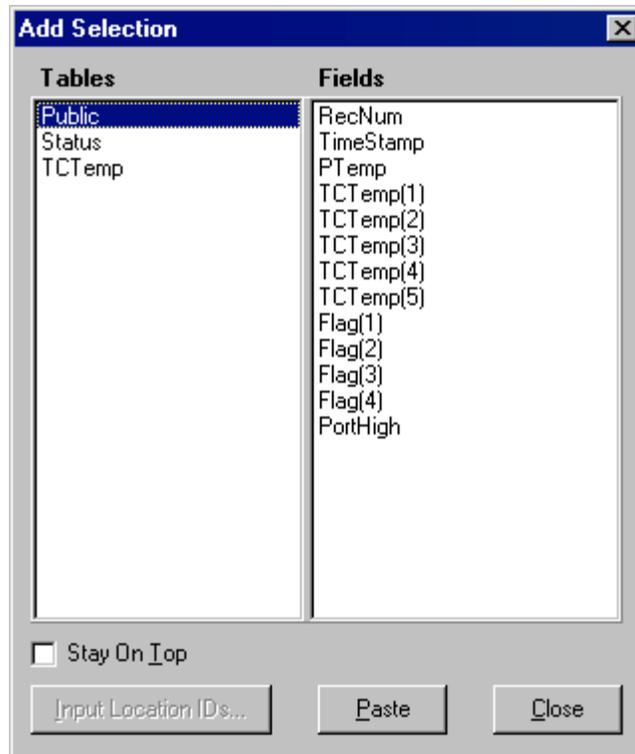
#### 5.1.7.2.1 Adding and Removing Values

The Numeric Display is initially blank when opened; the fields to be displayed must be selected. Press the Add button to bring up the Add Selection dialog box that lists the data fields for the available datalogger tables or final storage arrays.

#### NOTES

Table-based dataloggers - If the Add Selection dialog box is empty, return to the Setup window's Data Files table and select Get Table Definitions.

Mixed-array dataloggers - If no array IDs are listed and the input locations are listed as InputLocation\_# instead of with labels, return to the Setup window's Program tab and associate a DLD file with the datalogger.



Selecting a table name or final storage array ID will bring up a list of data fields in the right hand window. Select the fields to add by clicking the data field names. Multiple data fields can be selected by holding down the Shift or Ctrl key while clicking additional names. An entire table or array can be selected by clicking the table name.

Up to 60 data fields can be placed on each Numeric Display. The selected data fields can be added to the display either by clicking the Paste button to enter them on the display starting at the selected cell, or dragging the selected fields to the display cells. The Add Selection dialog can be kept in front of other windows by clicking the Stay on Top check box.

For mixed-array dataloggers that have an associated datalogger program, the final storage labels and input locations names are available to select and add to the display. If the datalogger doesn't have an associated file, generic labels for up to 28 input locations are still displayed in the Fields pane of the window. Additional input locations can be displayed by clicking the Select Input Locations button at the bottom of the screen. The numbers of the input locations to display can then be entered using dashes to indicate ranges and commas to separate numbers. For example 2-4,6,14-17 would add eight input locations to the display: input locations 2,3,4,6,14,15,16, and 17.

Once the fields have been added to the Numeric Display, they will update automatically as new data is collected from the datalogger. Input Locations and fields from a Public table that are not included as part of normal scheduled data collection will be collected from the datalogger based on the Update Interval field at the bottom right on the display (see below for more information on the Update Interval).

To delete data fields from the Numeric Display, select the data fields on the display and press the Delete button. Adding new data fields on top of existing fields in the display will overwrite the existing fields.

#### **5.1.7.2.2 Update Interval**

The Update Interval controls how often LoggerNet will get updated input locations from mixed-array dataloggers, and table values (public, status, or data table values) from table-data dataloggers. Setting this interval applies to all fields on the Numeric Display.

Final storage values for mixed-array dataloggers are updated only when data collection occurs from the datalogger. The frequency of update for all values in all dataloggers is limited by the frequency in which these values are updated in the datalogger program itself.

#### **5.1.7.2.3 Display Options**

The appearance of a Numeric Display can be customized by selecting the Options button. The Cell Options feature allows you to set up how the data should appear, set visual alarms, and set the number of rows and columns.

##### **Display Tab**

**Decimal Places to Show** - Sets the number of digits to display to the right of the decimal point. If one or more cells are selected before choosing this option, the change will affect only those cells. If no cells are selected, the change will affect the entire numeric display. This option can be used to show greater precision or present a cleaner appearance, depending on the data need.

**Boolean True/False String** - Allows you to choose the text that will be displayed for Boolean values (such as ports and flags, or variables declared as Boolean). By default the strings are True/False, though they could be set to High/Low, On/Off, etc.

**Show the Date When Displaying Timestamps** - When this check box is enabled, the date will be included in timestamps that are displayed on the numeric monitor. Otherwise, only the time will be displayed.

**Enable Alarms** - Alarms can be set to turn the background of a field a different color depending on the value of a data point. Select the Enable Alarms check box to turn on the alarm feature. The alarm levels can be set for one or more selected cells by entering a value in the High Alarm Value field and the Low Alarm Value field. When the data value displayed in a cell exceeds the limits set, the cell color will change based on the colors selected for each of the alarm values.

##### **Data Cell Configuration**

**Data Justification** - Determines whether the data value will be flush with the right or left side of the cell. If one or more cells are selected, the change will affect only the selected cells. If no cells are selected, the change will affect the entire display.

**Label Justification** - Determines whether the data labels will be flush with the right or left side of the cell. If one or more cells are selected, the change will affect only the selected cells. If no cells are selected, the change will affect the entire display.

**Number of Rows/Columns** - Allows you to configure the number of rows or columns for a numeric display. The total number of rows X number of columns cannot exceed 60. After changing the number of rows or columns, you may need to resize the numeric display window (by dragging one of the window's corners) for the font size and cell size to properly accommodate the values.

**Restore Configuration Defaults** - Select this button to restore all of the settings for the numeric display to their original defaults.

### 5.1.7.3 Graphical Display Screens

Data values collected from the datalogger can be plotted on a line graph in the Graphical Display. Like the Numeric Display, there are three graphs and all of them can be active at the same time if desired. The settings and selected data values for a graph are saved when the graph, Connect screen, or LoggerNet is closed. The graph settings for each datalogger are also saved independently, so a different datalogger will have different settings.

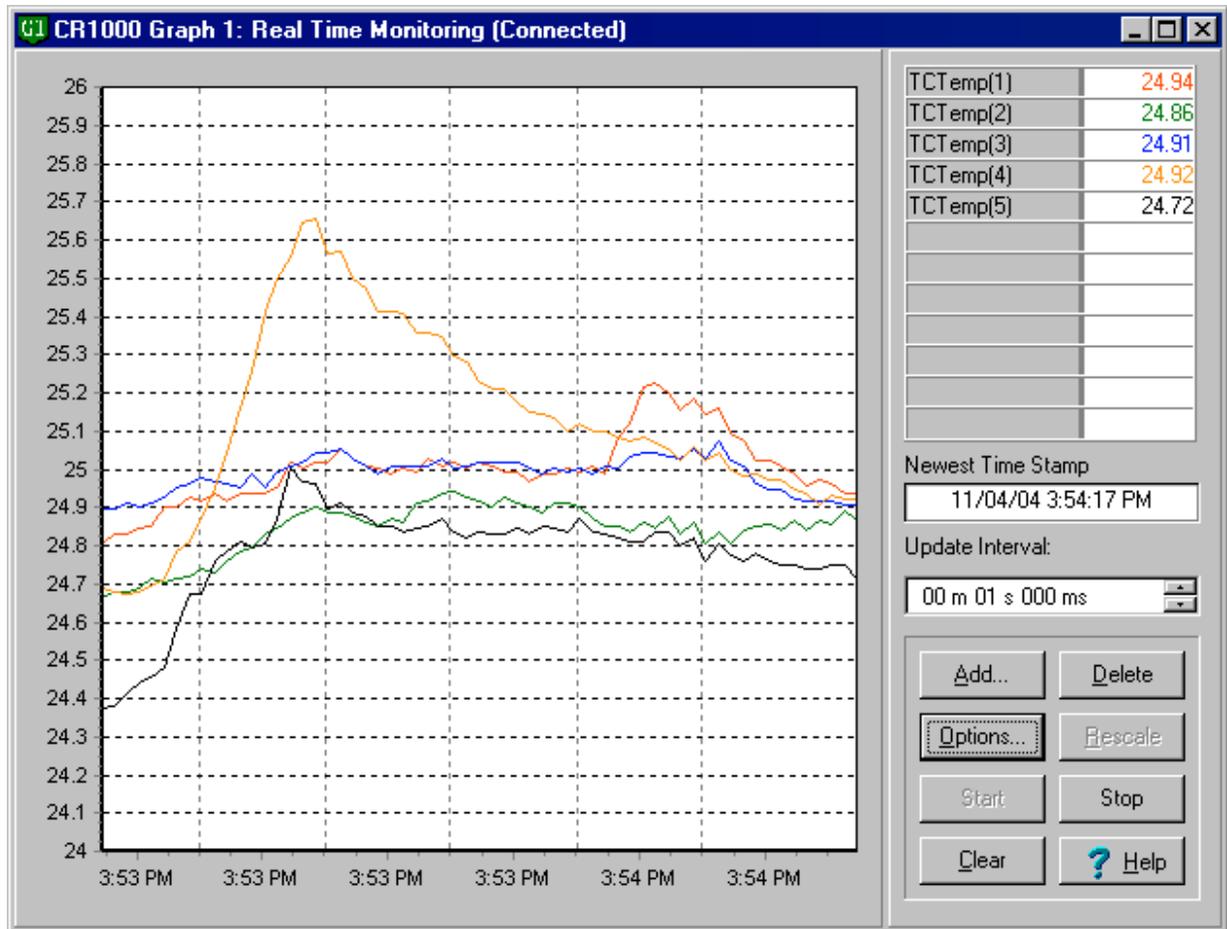
Selecting one of the three numbered buttons next to the text "Graphs" will bring up a graphical display. An example is shown below. If a Graphical Display screen is already active but hidden behind other windows, clicking its number will bring it to the front.

---

**NOTE**

A mixed-array datalogger's final storage data must be collected by LoggerNet before it can be displayed.

---



The timestamp for the right side of the graph is displayed above the configuration buttons on the right side of the window. This indicates the timestamp for the most recent data. The Update Interval is displayed below this field. The Update Interval controls how often the display will be updated with new data. Refer to the discussion on the update interval above, for additional information.

#### 5.1.7.3.1 Displaying Values on a Graph

A graph is initially blank when opened; the fields to be plotted must be selected. Press the Add button to bring up the Add Selection dialog box that lists the data fields for the available datalogger tables or final storage arrays. Up to twelve data fields can be graphed simultaneously. Data values are added to a graph in the same way they are added to the numeric display. Refer to the discussion above about adding values to a display using the Add Selection dialog box.

**NOTES**

---

Table-based dataloggers - If the Add Selection dialog box is empty, return to the Setup window's Data Files table and select Get Table Definitions.

Mixed-array dataloggers - If no array IDs are listed and the input locations are listed as InputLocation\_# instead of with labels, return to the Setup window's Program tab and associate a DLD file with the datalogger.

---

Once data fields have been added, the graph will start plotting data automatically. If there is historical data for the selected fields in the data cache, it will be displayed on the graph. Note that Input Location or Public table data does not have history stored in the data cache; therefore, no historical data will be displayed.

To delete data fields from the Graphical Display, select the data fields on the Graphical Display and press the Delete button. Existing data fields can be replaced by adding new data fields to the same cells.

Plotting of data can be temporarily stopped by pressing the Stop button. To resume plotting, press Start. The contents of the graph can be cleared by pressing the Clear button.

The Rescale button is used to bring outlying data values back within the vertical axis of the graph when using one of the Powers of 10 scaling options (see below for information on Powers of 10 scaling).

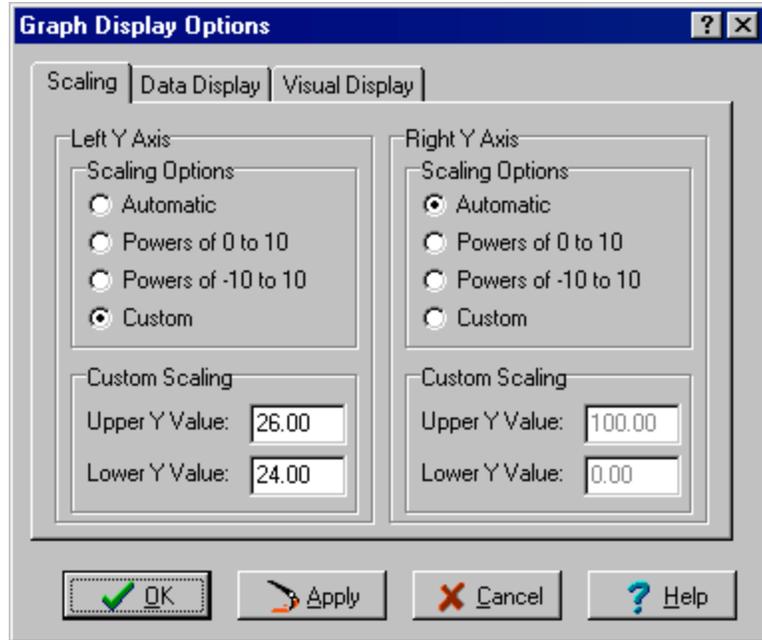
**5.1.7.3.2 Graph Options**

To customize the Graphical Display, press the Options button. The Graph Display Options dialog box has three tabs: Scaling, Time Display, and Visual Display.

**Scaling Tab**

The Scaling options are used to set up the scale for the right and left axes. The axes can be scaled automatically, fixed to a specific range or set to Power of 10 Scaling.

The left and right y-axes can be set independently. Data values can be set to graph relative to the right or left axis by customizing the Trace Options (see below). The axes scales are only shown on the graph if there are data fields linked with them.



**Automatic Scaling** - Adjusts the axis according to the values currently being displayed on the graph. The maximum and minimum will be set to display the largest and smallest values of all the fields being graphed. If there are values that are much larger or much smaller than the others, they can dominate the scaling and make variations in the other fields harder to see.

**Power of 10 Scaling** - Applies a Power of 10 multiplier to each of the data fields so the graph will have a scale of either 0 to 10, or -10 to 10 so that all values can be displayed on the same graph. The data values are scaled so they will fit between the option chosen. When this option is in effect, pressing the Rescale Button on the Graph tab will rescale any value that has fallen outside of the scale.

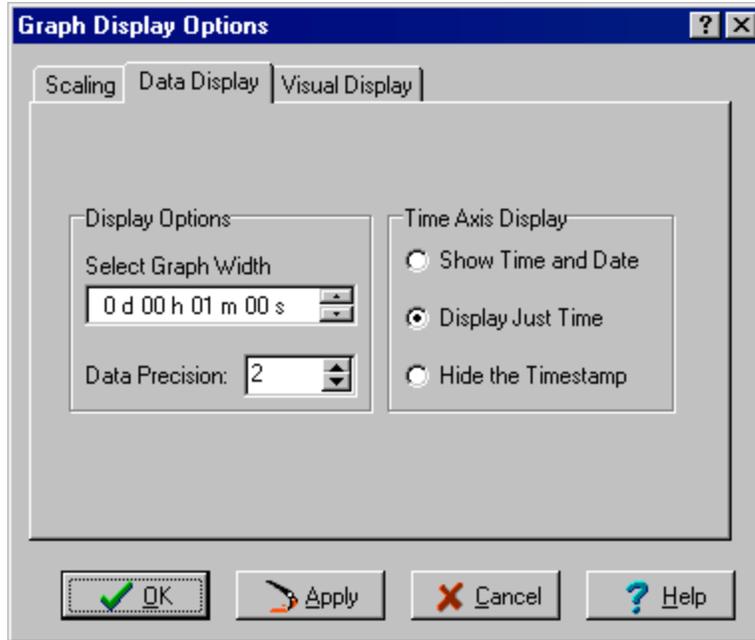
**Custom Scaling** - The Upper Y Value and Lower Y Value fields are used to specify a fixed range for the Y axis. When this option is in effect, the Rescale button on the Graph will be disabled. Any data values that do not fall within the specified range will not appear on the graph.

### ***Data Display Tab***

**Select Graph Width** - Determines the amount of time displayed across the width of the graph. The default setting of 1 minute will display 1 minute of data. If larger graph widths are specified the graph will backfill to plot whatever data is available in the data cache.

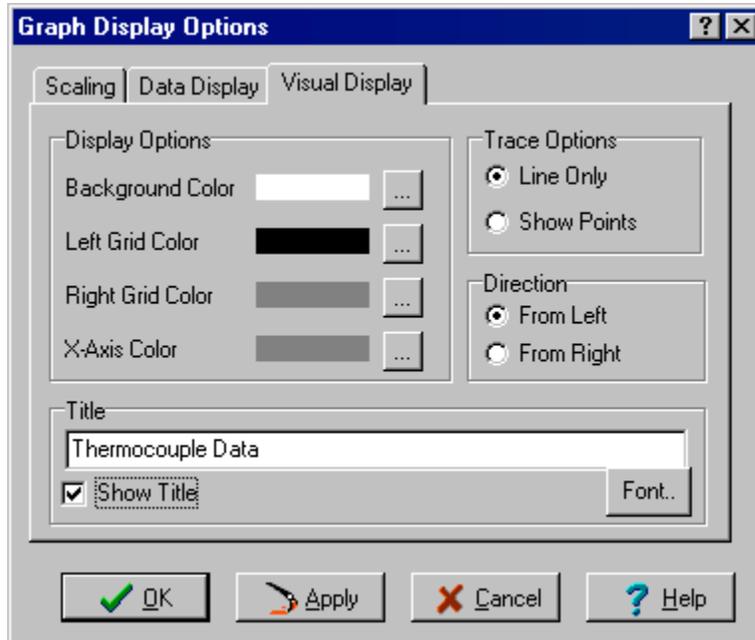
**Date Precision** - Determines how many decimal points will be displayed for the plotted value.

**Time Axis Display** -These options affect the horizontal time axis. You can choose to Show Time and Date, Display Just Time, or Hide the Timestamp.



**Visual Display Tab**

The Visual Display options allow the user set the appearance of the graph. The colors for the grids and background as well as how the trace should appear can be set here. A title can also be added to the graph.



**Display Options** - These options are used to set the look of the graph itself. The Background Color selects the color of the graph background. This is white by default. Be careful to select a background color that does not make any of the data traces disappear.

The Left / Right Grid Colors select the color of the grid lines that go with the left or right axis scale.

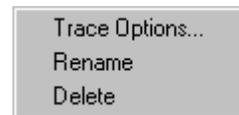
**Trace Options** - Determines whether the trace will be displayed with only a line or if points for the data will be shown also. Note that the line color, width and style for the trace are set in Cell Options (see below).

**Direction** - This option determines how the data values will populate the graph -- from the left (oldest data on the left and newest data on the right) or from the right (oldest data on the right and newest data on the left).

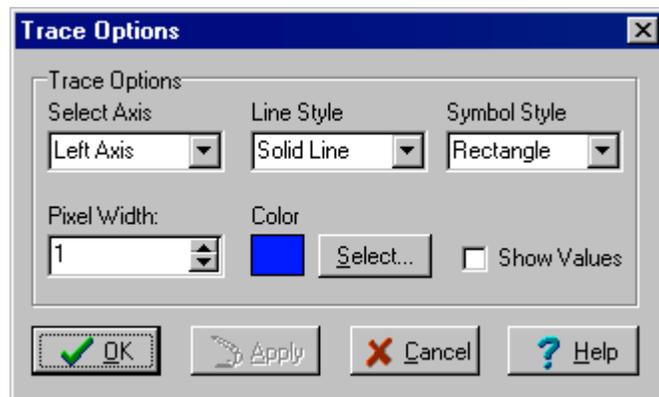
**Title** - Allows you to place a descriptive title on the graph. Enter the text of the title in the text box, then click the Show Title check box to make the title visible. Click the Font button to choose the font style, size and color for the title.

### 5.1.7.3.3 Cell Options

The appearance of the data traces can only be changed by selecting a cell and pressing the right mouse button. A floating menu will be displayed:



Select Trace Options to display the Cell Options dialog box:



**Select Axis** - Sets whether the data trace is displayed on the left or right axis.

**Style** - Selects the style for the line.

**Symbol Style** - Selects the appearance of the individual data points. Data points are displayed on the line in the Graph Options, Visual Display tab (see Graph Options above).

**Pixel Width** - Sets the width of the trace. Wider traces are easier to see but may obscure other traces or small variations of the data.

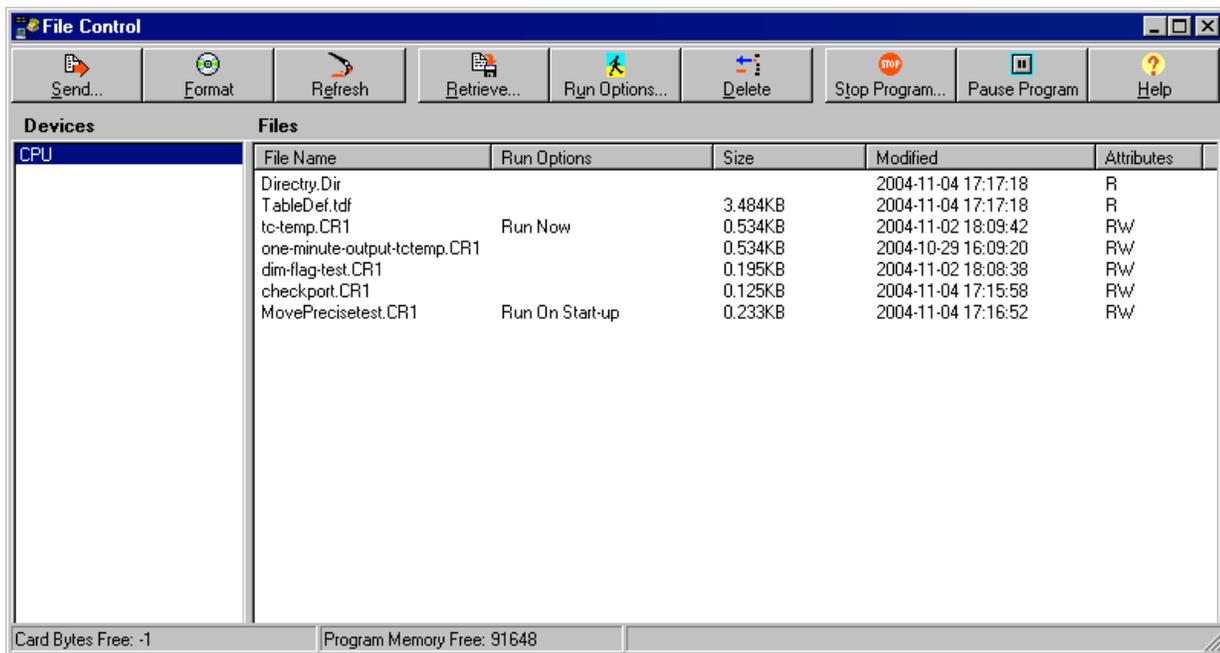
**New Color** - Sets the color of the trace and the data points. The user can choose from the Windows color palette for the color. The color for this trace is shown in the color window.

**Show Values** - Displays the numeric value for a data point on the trace in a small gray window. This option is impractical for data that is being updated at a very fast rate, as the values will be too numerous to read and may obscure other traces.

### 5.1.8 File Control for CR5000, CR1000, and CR9000 Dataloggers

CR5000, CR1000, and CR9000 dataloggers have a built in file system much like a computer hard disk. Multiple files can be stored in the datalogger's memory or on a PC card, including data files and datalogger programs. Note that unlike other dataloggers, these dataloggers retain in memory programs that have been downloaded to them unless the programs are specifically deleted or the datalogger memory is completely reset.

File Control is used to manage all the files on these dataloggers. File Control is opened from the Connect window's Tools | File Control menu.



The File Control window displays a list of files stored on the datalogger's CPU or PC card. The window on the left lists all of the data storage devices available for the selected datalogger (CPU or CRD). Selecting a device shows a list of the files stored there.

The run options for a file indicate whether it is set to Run Always, Run Now, or Run On Start-up. The currently executing program is indicated by the Run Now or Run Always attribute. The file size is displayed, as well as the last time the file was modified and whether or not the file is Read Only or Read/Write.

Note that the Size, Modified date, and Attributes may not be available for all dataloggers.

There are several options to work with the files and directories on the datalogger.

**Send** is used to transfer files from the computer to the datalogger. Clicking the Send button brings up a standard file selection dialog box. A new file can be chosen to send to the highlighted device.

Datalogger programs, data files, and other ASCII files can be sent to the datalogger.

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

**Refresh** will update the list of files for the selected device.

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

**Run Options** brings up a dialog box that is used to control what program will be run in the datalogger. Highlight a file, and then select the Run Options button. *Run Now* will stop the current program in the datalogger and load and run the selected program. All data tables in the datalogger will be reset. *Run On Start Up* will set a program to be run when the power is cycled on the datalogger. *Run Now and On Start Up* sets the selected program to run immediately, and if the datalogger loses power, the program will run when the power is restored. Note that a file marked with Run Now and On Start-Up option is shown as Run Always in the File Control display.



**Delete** - Highlight a file and press the delete button to remove the file from the datalogger's memory.

**Stop Program** halts execution of the currently running datalogger program. This option will delete all data tables.

**Pause Program** will temporarily pause execution of the program. The button text changes to **Resume Program** until the program is started again. Unlike

the Stop Program option, when a program is paused, all data tables in the datalogger are maintained.

### **Right Click Menu Options**

When a device is selected (CPU or CRD), pressing the right mouse button displays a menu with the Send File and Format Device functions.

When a file name is selected, pressing the right mouse button displays a menu with the Retrieve File, Delete File, Refresh List, Run Options, Stop Program, and Pause Program options.

## **5.1.9 Terminal Emulator**

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 remote keyboard select Tools | Terminal Emulator from the menu.

---

### **CAUTION**

The remote terminal mode 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.

---

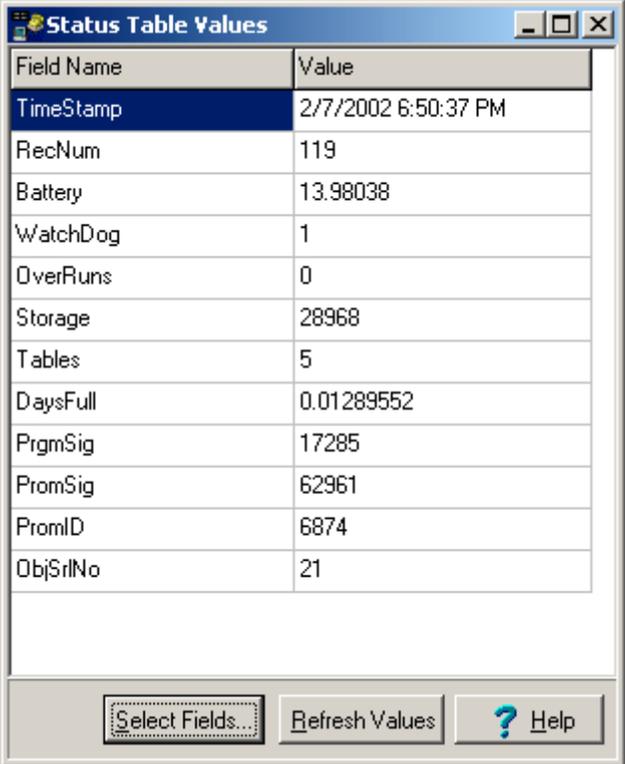
## **5.1.10 Status Table Information (Table Dataloggers Only)**

Table-based dataloggers have a default system table called a Status table. The Status table contains information about the datalogger type; OS version, date and signature; program statistical information; etc. (Refer to each datalogger's manual for complete information about the Status table).

To view the status table for a table-based datalogger select Tools | Status Table from the menu on the Connect screen. A window similar to the one below will be displayed. Click Select Fields to choose the status information you are interested in seeing.

Once the fields are selected, LoggerNet will contact the datalogger to get the latest status data for the selected fields.

Click Refresh Values to have LoggerNet query the datalogger again to get updated values. Each datalogger has a slightly different set of values in the status table. Information about the meaning of the values in the status table can be found in Appendix XX.



Field Name	Value
TimeStamp	2/7/2002 6:50:37 PM
RecNum	119
Battery	13.98038
WatchDog	1
OverRuns	0
Storage	28968
Tables	5
DaysFull	0.01289552
PrgmSig	17285
PromSig	62961
PromID	6874
ObjSrlNo	21

Buttons at the bottom: Select Fields..., Refresh Values, ? Help

## 5.2 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, as well as input locations or variables. 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. One copy of RTMC Run-Time is provided with LoggerNet; additional copies to run on remote machines can be purchased separately.

---

**NOTE** Scheduled data collection must be enabled in LoggerNet, or RTMC's display will never update.

---

### 5.2.1 Development Mode

RTMC Development is a graphic display editor that allows the user to easily place graphical components on the display screen and associate them with data values.

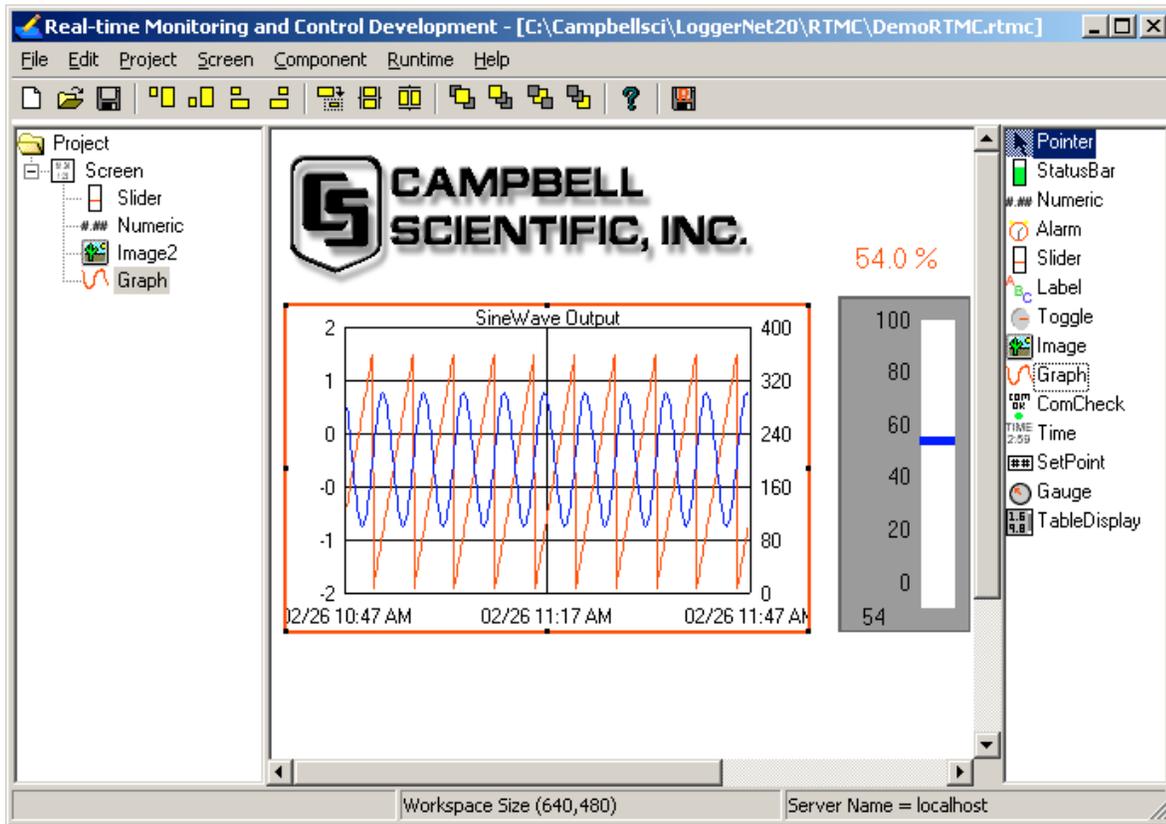
The RTMC Development window, as shown below, has three sections.

**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 graphical 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. The directory in which these files are stored is C:\Campbellsci\Lib\RTMCMediaLib. Custom images can be

used as well; these should be placed in the media library directory to make them available for RTMC's use.

### 5.2.1.1 The RTMC Workspace

The RTMC workspace is a fixed size container for holding one or more display screens. As new display screens are added (Screen | Add New Screen) they appear as tabs in the project. The size of the workspace (and the resulting run-time window) can be changed by selecting Project | Change WorkSpace Size.

### 5.2.1.2 Display Components

Display components are the objects that are used to display data. To add a component to the workspace, click 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 on a schedule. If scheduled data collection is not set up in LoggerNet or the selected data value is excluded from scheduled collection, the values will not update. Input locations, ports and flags for mixed-array dataloggers are collected at the scheduled collection interval or any time a manual collection is done.

---

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 opened 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	<p><b>Pointer</b> 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.</p>
 StatusBar	<p><b>Status Bar</b> 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.</p>
 Numeric	<p><b>Numeric</b> 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.</p>
 Alarm	<p><b>Alarm</b> is used to set up a display component that changes appearance when an alarm condition occurs. You can choose the images to be displayed for normal and alarm conditions, 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.</p>
 Slider	<p><b>Slider</b> 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 of the Slider component.</p>
 Label	<p><b>Label</b> is text that can be used to put titles or labels on the graphical components.</p>
 Toggle	<p><b>Toggle</b> 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.</p>
 Image	<p><b>Image</b> 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.</p>



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 mixed-array dataloggers, the data for one array ID is displayed. 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.



ValueForwarder

**ValueForwarder** allows you to set a value in one datalogger, based on the value in another datalogger. The value itself can be sent to the other datalogger, or you can set a value to a specified constant. This component can be hidden from the run-time display if desired.

**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.

### 5.2.1.3 Functions Available from the RTMC Menus

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 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.

**Select All** selects all of the components in the workspace. The components can then be cut, copied, deleted, grouped, etc.

---

#### **NOTE**

Cut/Copy does not go to the Windows clipboard so these objects are not available to paste into other applications.

---

### **Project Menu**

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. By default, LoggerNet's port number is 6789. Note that the remote LoggerNet server must have Remote Connections enabled (Options | Allow Remote Connections) for RTMC to be able to display remote data. See below for additional information.

**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.

**Configure Auto Tabbing** lets you enable or disable the automatic switching between project tabs when an RTMC form is run, and set the rate at which a new tab will be displayed. When RTMC is in AutoTab mode, it will display a tab for a set amount of time and then display the next tab.

### **Screen Menu**

Screen Menu options work with the tabbed screens in the project. The Screen Menu is also available by right clicking 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 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.

### **Component Menu**

The Component Menu is used to set the component properties, placement and alignment. The Component Menu is also available by right clicking 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.



**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.

---

**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.

---

**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** - The only option is 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.

### 5.2.1.4 Expressions

RTMC has a built-in expression interpreter that allows the user to condition the data or create displays based on calculations of 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 the Setup screen, followed by the table name or array ID, followed by the data label. An example for a mixed-array datalogger would be CR10X\_2.109.BattV (datalogger.arrayID.input location).

---

#### NOTE

Spaces must be used before and after the predefined constants and functions. Operators do not require spaces.

---

#### 5.2.1.4.1 Operators

<u>Operator</u>	<u>Description</u>
( )	Prioritizes parts of an expression within the larger expression.
*	Multiply by
/	Divide by

<u>Operator</u>	<u>Description</u>
^	Raised to the power of
+	Add
-	Subtract
=	Equal
<>	Not equal
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to

#### 5.2.1.4.2 Predefined Constants

<u>Constant</u>	<u>Description</u>
e	2.718282
PI	3.141593
True	-1
False	0

#### 5.2.1.4.3 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))

<u>Function</u>	<u>Description</u>
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.

<b><u>Function</u></b>	<b><u>Description</u></b>
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.

<u>Function</u>	<u>Description</u>
(x)XOR(y)	Performs a logical exclusion on two numbers.

#### 5.2.1.4.4 Order of Precedence

When processing mathematical expressions, the order of precedence is:

- 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.

#### 5.2.1.5 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, or the "localhost".



To set a connection to LoggerNet on another computer, open 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 that supports security and the network administrator has implemented security. The default port number for LoggerNet is 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 or when using RTMC Run-Time

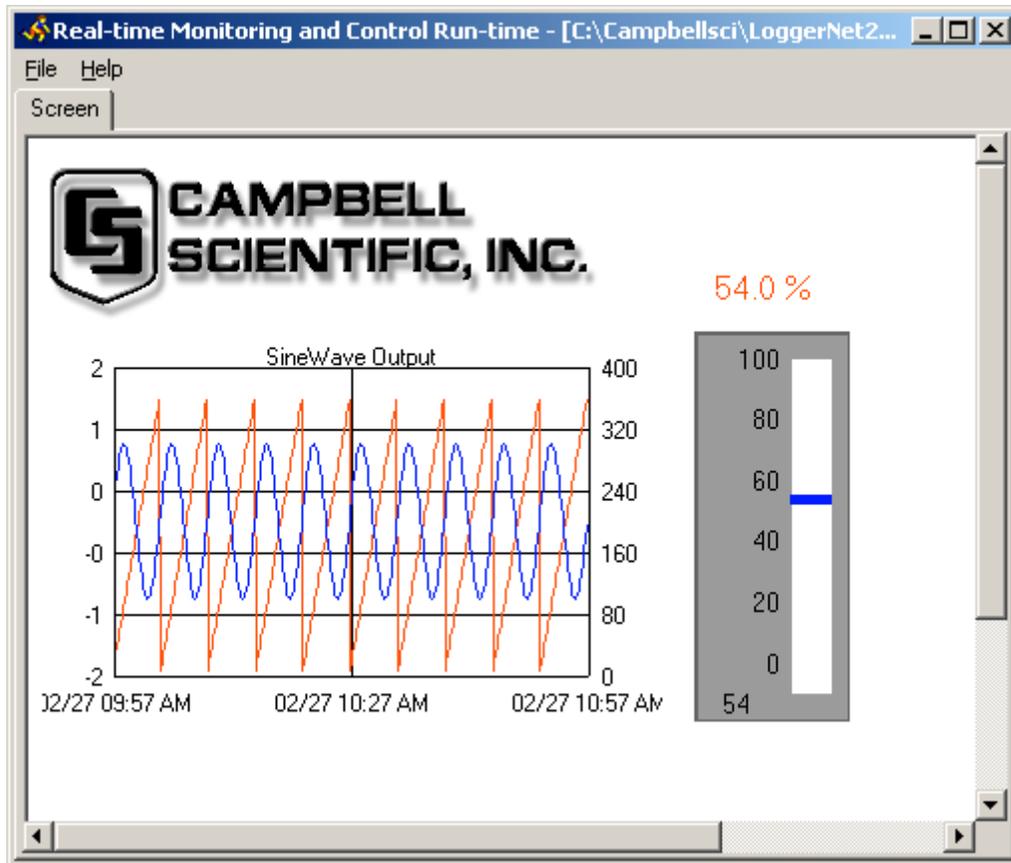
---

**NOTE** LoggerNet must be installed with the “Allow Remote Connections” option enabled for RTMC to be able to connect remotely.

---

## 5.2.2 RTMC 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 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 on a schedule 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 RTMC Run-Time. In the Run-Time window select File | Open Project to select the RTMC project screen to run.

In Run-time mode, you can print an image of the RTMC display screen by selecting File | Print Screen. A new form to be run is selected under File | Open Project.

A copy of RTMC Run-time comes with LoggerNet. If you want to run RTMC projects on remote computers, additional copies of RTMC Run-time can be purchased separately. One copy is required for each computer on which RTMC Run-time will be used. As noted above, when running RTMC Run-time on a remote computer, the host computer must have Remote Connections enabled (LoggerNet Toolbar, Options | Allow Remote Connections).

# Section 6. Network Status and Resolving Communication Problems

---

*LoggerNet provides several tools for monitoring the status of a datalogger network and troubleshooting communication problems within that network.*

*The Status Monitor screen provides a way to monitor communications statistics. Statistics are displayed for data collection attempts and communication failures. PakBus Graph provides a visual representation of the devices in a PakBus network and lets you edit PakBus device setting. The Log Tools utility provides a way to read communication logs more easily. The Troubleshooter highlights potential problems in a communication network and provides access to a Communications Test and other troubleshooting tools.*

*Note that a Troubleshooting section is also provided in Section 15.*

## 6.1 Status Monitor

The Status Monitor is opened by pressing the Status button from the LoggerNet Toolbar. The leftmost column of the Status Monitor displays the network map showing the devices in the network. Statistics for each device are displayed in columns to the right. The display defaults with the most commonly monitored statistics for each device and can be customized by the user. See below for a description of the statistics and their meaning.

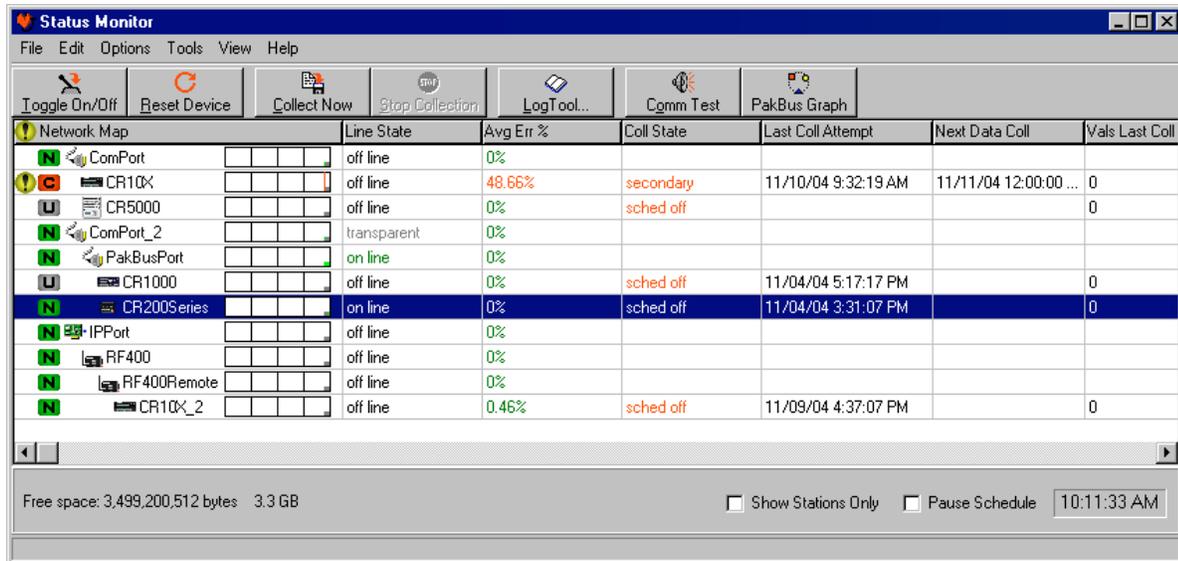
Information in the Status window can be sorted by column. Click on a column heading to sort the devices. For instance, if the Network Map column is clicked once, the devices will be sorted alphabetically by name. Clicking the column a second time will sort the devices in reverse alphabetical order, and a third time will restore the Network map to its original order.

Above the display area is a row of buttons providing quick access to many of the functions available in the Status Monitor (these buttons are discussed in subsequent sections). Below the display area is an indication of the disk space available on the computer and check boxes for Show Stations Only and Pause Schedule.

Checking Show Stations Only will hide all of the devices in the network map except the dataloggers and storage modules. This can make the display easier to view, especially when there are many dataloggers in the network or the communication paths are complex. This option can also be accessed by selecting the Options | Show Stations Only menu item.

Checking Pause Schedule suspends scheduled data collection for all of the dataloggers in the network. This can be useful when trying to isolate specific problems. This option can also be accessed by selecting Options | Pause Schedule.

The current LoggerNet server time is displayed in the lower right hand corner.



## 6.1.1 Visual Status Indicators

There are three visual status indicators in the first column of the Status Monitor.

A Communication Status Image **N** to the left of each device indicates the current communication state for that device. The color and letter for the image are a way to quickly verify the LoggerNet communication server's connection with the device. A device has one of four states: Normal (green N), Marginal (blue M), Critical (red C), or Unknown (gray U). The status image changes to reflect the device's current status.

If a critical problem is found when communicating with a device, a Trouble Indicator icon **!** will be displayed to the left of that device. The column heading for the network map will also display the Trouble Indicator icon.

The Communication History  is a chart that is placed to the right of the device name. This graph shows the state of the communication link with the device over the previous 12 hour period. The colors of bar for the chart indicate whether communication was Normal, Marginal, Critical, or unknown.

One or all of these visual indicators can be toggled on/off from the View menu.

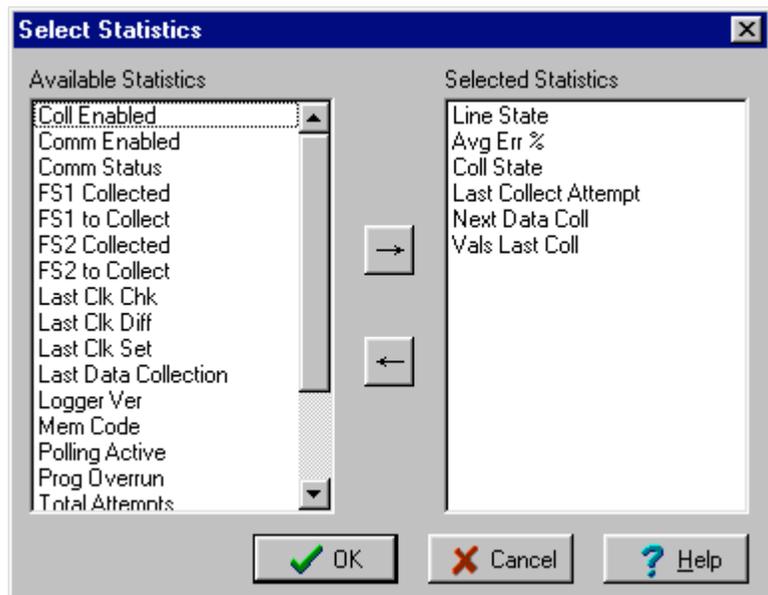
## 6.1.2 Status Monitor Functions

The Status Monitor includes a number of functions that provide control over the LoggerNet data collection process, configure the display appearance, or bring up other applications such as Log Tool or PakBus graph. The Status Monitor functions can also be accessed through the menus at the top of the screen.

### 6.1.2.1 Selecting Columns

The Status Monitor can be customized to display only those columns containing statistics of interest. To add columns to the Status Monitor window,

click Edit | Select Columns. The Select Statistics window appears. Right clicking in the middle of the Status Monitor window will also bring up the Select Statistics window.



Entries in the Available Statistics field will not be displayed on the main screen. Entries in the Selected Statistics 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. These statistics values are updated continuously as LoggerNet communicates with the devices in the network. The actual text as displayed on the screen is shown below in parentheses.

- **Average Error Rate (Avg Err %)** - 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 (Coll Enabled)** - Indicates whether or not scheduled data collection is enabled for the device. (Data collection is set up and enabled in the Setup Screen.)
- **Collection Retries (Coll Retries)** - The total number of data collection attempts that have occurred under the primary and secondary retry collection states (Coll State).
- **Collection State (Coll State)** - Indicates the current state of scheduled data collection.

Normal - The normal data collection schedule is active.

Primary Retry - A data collection failure has led to the primary retry collection schedule.

Secondary Retry - The number of primary retries set in the Setup Screen have run out and the secondary retry schedule is now active.

Schedule Off - Scheduled data collection is not enabled for this device.

Comm Disabled - Communications for this device or one of the devices in the communication link from the computer has been disabled.

Invalid Table Definitions - A change in the table definitions for this device has been detected. This has caused scheduled collection to be suspended until LoggerNet has updated table definitions to match the datalogger.

Network Paused - Scheduled data collection has been suspended for the whole network.

Unreachable - The device cannot be reached through the network.

- **Communication Enabled (Comm Enabled)** - Indicates whether or not communication is enabled for the device. (Communication is set up and enabled in the Setup Screen.)
- **Communication Status (Comm Status)**- The current communication state of the device: Normal, Marginal, Critical, or Unknown. This value matches the state of the indicator next to the device name. Marginal is indicated when warning messages for the device are logged in communications. Critical is indicated when communication failures are logged in communications.
- **Final Storage Area 1, 2 Collected (FS1 Collected, FS2Collected)** - (Array-based dataloggers only) This value will show the number of data values that have been collected from final storage area 1 or 2. In combination with the FS to Collect statistic this can show progress toward the completion of data collection.
- **Final Storage Area 1, 2 to Collect (FS1 to Collect, FS2 to Collect)** - (Array-based dataloggers only) When LoggerNet first contacts the datalogger for data collection, the datalogger indicates how much data there is to collect from each of the storage areas. This statistic shows the number of values for the current data collection activity. When the number of values collected matches the number to collect, all of the data for this call has been collected.
- **Holes** - A "hole" is a discontinuity in record numbers for dataloggers that support data advise. This statistic is the number of values that are in holes.
- **Last Clock Check (Last Clk Chk)** - The computer date and time of the last time the clock was checked for this device.
- **Last Clock Difference (Last Clk Diff)** - The amount of time the datalogger clock deviated from the LoggerNet computer's clock when the

last clock check was performed. If the datalogger clock is slower than the computer clock, this will be a positive value.

- **Last Clock Set (Last Clk Set)** - The computer date and time that the datalogger's clock was last set to match the LoggerNet computer's clock.
- **Last Collect Attempt** - The computer date and time when data collection was last attempted for this device.
- **Last Data Collection** - The date and time that data was last collected from the device by LoggerNet.
- **Line State** - This is an indication of the current communications activity for the device. In normal communication these will be changing often and will provide a feel for how the network is working. The possible states are listed below.

Off-Line - There is no active communication with this device.

On-Line - LoggerNet is actively communicating with this device.

Transparent - The device is being used as a bridge to communicate with another device.

Hanging Up - LoggerNet has finished communication with this device or the other devices on the link and is closing down communication.

Comm-Disabled - Communications for this device have been disabled for this device, a parent device, or the network.

- **Lithium Battery Voltage (Lith Batt Volt)** - The voltage level of the datalogger's lithium SRAM back-up battery.
- **Logger Version (Logger Ver)** – (Array-based dataloggers only) This number indicates the version level of the datalogger. This number is used by LoggerNet to adjust communication for older dataloggers.
- **Low 5v Battery Detect (Low 5V)** - A counter that indicates the number of times the datalogger's 5V supply has dropped below 5V. The counter's maximum limit is 99; it can be reset in the datalogger's \*B mode.
- **Low Voltage Stopped (Low Volt Stopped)** - The number of times the datalogger program has been halted because the datalogger's 12 V power source has dropped below the minimum power requirement. The counter's maximum limit is 99; it can be reset in the datalogger's \*B mode.
- **Memory Code (Mem Code)** – (Mixed array dataloggers only) This number is an indication of the amount of memory available in the datalogger. The amount of memory represented depends on the datalogger. See the datalogger operator's manual for more information.
- **Next Data Collection (Next Data Coll)** - The date and time of the next scheduled data collection for the device.

- **Polling Active** – True indicates that LoggerNet is currently receiving data for this device.
- **Program Overrun (Prog Overrun)** – (Mixed array dataloggers only)  
This is the number of table overruns recorded for the running datalogger program. This is the same number available from \*B mode with keyboard displays or through the terminal mode.
- **Table Defs State** - Indicates the status of the datalogger's table definitions, as known by the LoggerNet server.

None - No table definitions have been retrieved from the datalogger.

Current - The table definitions from the datalogger match what LoggerNet has stored as the table definitions for the datalogger.

Suspect - A collection attempt has returned an invalid table definitions code. LoggerNet will attempt to verify the table definitions for the datalogger.

Getting Table Defs - LoggerNet is in the process of retrieving the table definitions from the datalogger.

Invalid Table Defs - The table definitions from the datalogger do not match what LoggerNet has stored as the table definitions for the datalogger. Table definitions will need to be updated before data collection can occur.

- **Total Attempts** - The total number of communication attempts with the device since LoggerNet was started or retries were reset.
- **Total Failures** - The total number of communication attempts with the device that have failed since LoggerNet was started or retries were reset.
- **Total Retries** - The total number of communication attempts with a device after the original attempt failed since LoggerNet was started or retries were reset.
- **Uncoll Holes** - The number of values in holes that cannot be collected from the datalogger (most often, because the data has been overwritten by newer data).
- **Values In Last Collect (Vals Last Collect)** - The number of values that were collected during the last data collection attempt. Used in combination with the Values to Collect the user can get an idea of how much data is left to collect.
- **Values to Collect (Vals to Collect)** – The total number of values to be collected. When LoggerNet first contacts the datalogger it finds out how many data values are waiting to be collected.
- **Watchdog Error (Watchdog Err)** – (Mixed array dataloggers only) This is the number of Watchdog Errors being reported by the datalogger. This is the same number that is available from \*B mode with keyboard displays

or using the remote keyboard. See the datalogger operator's manual for more information.

### 6.1.2.2 Toggle Collection On/Off

Pressing the Toggle On/Off button toggles scheduled collection on or off for the selected datalogger. The change to scheduled collection is reflected on the Setup screen for this datalogger. The function is only enabled when a datalogger is selected in the network map. Collection can be toggled either by clicking the button or selecting Options | Toggle Collection On/Off.

### 6.1.2.3 Reset Device

Resets all of the statistics for the selected device. Reset Device is done either from the button or selecting Options | Reset Retries.

### 6.1.2.4 Collect Now/Stop Collection

Collect Now starts data collection for the selected datalogger. This is the equivalent of the Collect Now button on the Connect screen. Data collection will use the file settings from the Data Files or Final Storage Area 1 or 2 tabs on the Setup screen. While data collection is in progress, pressing Stop Collection will stop the collection. This might be needed if the datalogger has a lot of uncollected data and it would take too long to get it all. These functions are available by clicking the buttons or from the Options menu.

## 6.2 LogTool

There are four logs kept by LoggerNet that track the operation of the server, communications with the dataloggers and data collection. These logs can be used for troubleshooting communication problems. The LogTool utility allows you to view the communication packets transferred between the computer running the datalogger support software and other devices in the network. LogTool can be launched from the Status Monitor, or from the LoggerNet Toolbar's Tools menu.

Each of the logs is explained here briefly. Operation of the LogTool utility is also explained. For additional information on interpreting logs, see Appendix C.

### 6.2.1 Log Types

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

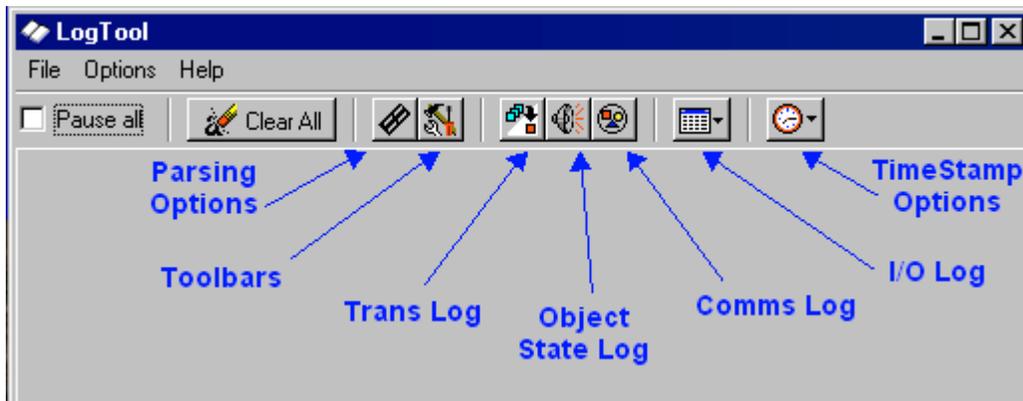
**Communication Log** (comms\$.log) - This log records information on the quality of communications in the datalogger network. Three types of messages are recorded: status messages, warning messages, and fault messages.

**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.

**Low Level I/O Log** (io\$SerialPort\_1) - This log displays low level incoming and outgoing communications for a root device (i.e., serial port).

## 6.2.2 Using LogTool

LogTool can be opened from the Status Monitor, the LoggerNet Toolbar's Tools menu, or the Troubleshooter. When the LogTool is first opened, two logs are displayed: the transaction log and the communication log. The Object State log and the Low Level I/O log can be displayed by toggling their associated button on the toolbar.



**Parsing Options** - Toggles the display of the message parsing window.

**Toolbars** - Toggles the display of an individual tool bar for each of the logs. You can pause the display of messages for a tool bar by selecting the Pause check box. You can clear all messages for a log by pressing the Clear Messages button.

**Trans Log, Object State Log, Comms Log** - Toggles the display of the associated Log.

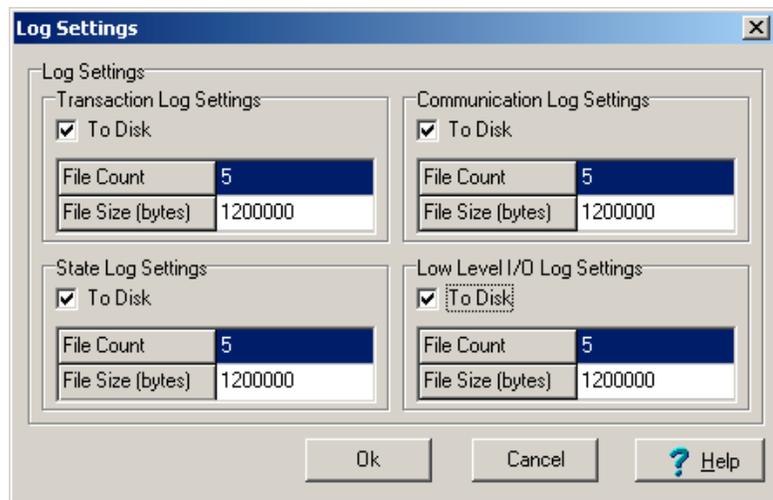
**I/O Log** - Opens the Low Level I/O log for a specific COM port in a new window.

**TimeStamp Options** - Allows you to select the format for the time stamp in the logs. If none of the options are enabled (an option is enabled if a check mark appears to the left of the option name), only the time is displayed (hh:mm:ss AM/PM). If Date is selected, a date (MM/DD/YY) will be added to the time stamp. If Military is selected, the time stamp will be displayed in 24 hour format instead of 12 hour format. If ms Resolution is chosen, the time stamp will also include milliseconds.

### 6.2.3 Saving Logs to File

The log messages currently being displayed on the LogTool screen can be manually saved to a text file by selecting File | Save Log Windows to File. In addition, these logs can be printed by selecting File | Print Log Windows.

Log files can be saved to individual files automatically by selecting Options | Log File Settings from the LogTool menu and selecting To Disk for the appropriate log. In addition to controlling whether the logs are saved to files, Log File Settings also lets you set the size and number of log files that are created. One set of log files each is created for the Transaction, Communication, and Object State logs. For the Low Level I/O logs, one set of logs will be created for each of the root level devices in the network. Once the maximum number of logs have been created (set by the File Count) the next log file created will overwrite the oldest file.



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

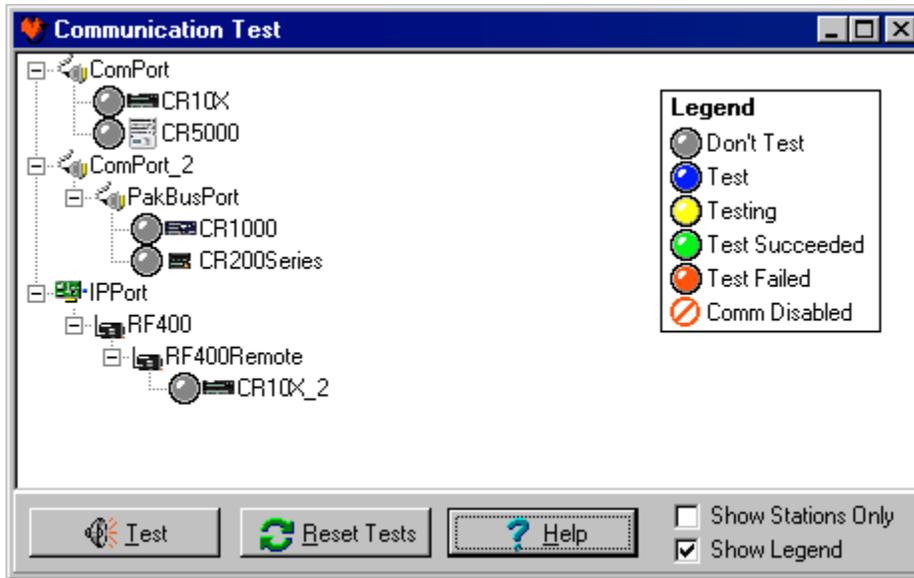
**To Disk** – Selecting this check box enables saving the associated logs to files on the server computer hard disk.

**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.

**File Size** – This setting determines how big the log file is allowed to grow before being saved to an archived file. 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 (e.g. tran0.log, tran1.log, tran2.log...).

## 6.3 Comm Test

The Comm Test tool allows you to test the communication links to the datalogger stations. When Comm Test is launched, you will see a window similar to the one below with all of your devices in the network map listed.



The color of the circle to the left of each datalogger station indicates the quality of communication. The legend displayed on the right side of the window provides a key to the color codes. The legend can be removed from the window by clearing the Show Legend check box.

When you first open Comm Test, the state of the devices is unknown, so the circles for each device will appear grey. To initiate the test, click on one or more of the datalogger stations to select them (the circles will appear blue), and press the Test button. The LoggerNet server will attempt to contact the selected device(s) and perform a simple clock check. While a test is in progress, the circle for a device will appear yellow. Once the test is performed, the resulting circle will be green (clock check successful) or red (clock check failed).

Press Reset Test to clear the test results before running the test again.

By default, the Communication Test window shows all devices in the network. You can display on the stations by selecting the Show Stations Only check box.

## 6.4 PakBus Graph

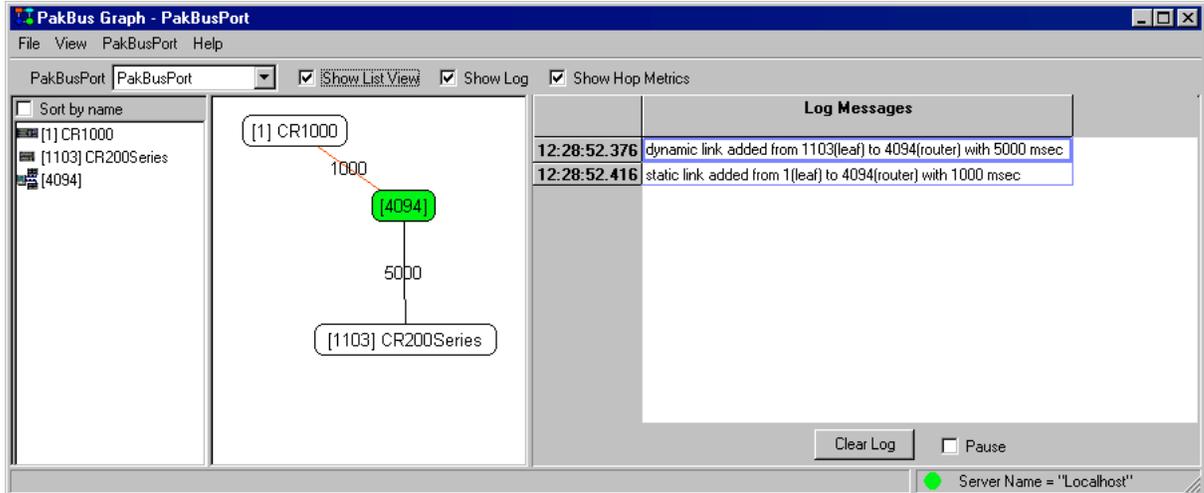
PakBus Graph is a utility that graphically depicts the connections in a LoggerNet PakBus datalogger network. It provides a look at LoggerNet's PakBus routing table. In addition, the utility can be used to change the settings of a PakBus device.

The window for PakBus graph is divided into three sections: the list of PakBus devices, a graphical depiction of the PakBus network, and the log messages for PakBus communication. The list of devices and the log can be toggled off by clearing the Show List View and Show Log options, respectively.

Software servers are identified in PakBus Graph by the color green. Other devices remain colorless unless they have been selected with the mouse cursor. When selected, they are colored cyan.

The default PakBus address for LoggerNet is 4094. Other PakBus devices will be shown by name and address, if known.

PakBus Graph can be opened from the Status Monitor, the Troubleshooter, or from the LoggerNet Toolbar's Tools menu.



### 6.4.1 Selecting the PakBus Network to View

When PakBus Graph is opened, it is set to view the first PakBus network on the computer on which the datalogger support software is running. If more than one PakBus network is set up on the computer, the different networks can be viewed individually by selecting a port name from the **PakBusPort** drop-down list. If the PakBus ports set up in the software have been bridged, the resulting single port will be named "global".

PakBus Graph also can be opened independently from the software toolbar, by double-clicking the PakBusGraph.exe in the software's program files directory (e.g., C:\Program Files\CampbellSci\LoggerNet). If opened independently, the host computer to which PakBus graph should connect can be selected from File | Change Server Connection on the PakBus Graph menu.

The Change Server Connection option has the following settings:

**Server IP** - The name of the computer with which to connect. This must be the valid name of an existing computer or a TCP/IP address (in the form ###.###.###.### consisting of the IP network number, ###.###.###, and the host number, ###).

**User Name** - Your user name on the software server.

**Password** - Your password for the software server.

The **User Name** and **Password** fields are required only if your server administrator has set up security on your system.

## 6.4.2 Dynamic and Static Links

There are two types of links to PakBus dataloggers that the server recognizes: static links and dynamic links. Static links (depicted using red lines) are the communication links to dataloggers that have been set up in the software, but which have not been confirmed by communicating with the datalogger(s). You will see these dataloggers listed in the software's network map. Dynamic links (black lines) are communication links to dataloggers that have been confirmed. You may also see links to leaf node dataloggers that have not been set up in the software, but which the server has "learned about" by querying the PakBus network.

## 6.4.3 Viewing/Changing Settings in a PakBus Datalogger

If you right-click a device in PakBus graph, you will be presented with a floating menu. From this menu, select **Show Settings** to display a list of the PakBus settings for the datalogger. Some of these settings are read-only, but other settings can be changed. Click within the cell for a setting, enter a new value, and then press return to make a change. If the change is accepted, the cell will appear green. If the change was denied (which likely means the setting is read-only), the cell will appear red.

## 6.4.4 Right-Click Functionality

There are several options available from the floating menu that is displayed when you right-click a device (not all devices will have all settings):

**Show Settings** - This option shows the PakBus settings of a device (see above).

**Ping Node** - This option will send a packet to the selected device to determine if it is reachable in the PakBus network. The results of the ping will be displayed in the Log Messages. Each ping message will include the size of the packet sent, and the time of response from the pinged device. The last message recorded will include summary information from the ping.

**Reset Node** - This option will reset the routing table in a PakBus device.

**Change Server ID (server only)**- By default, the PakBus address of the software server is 4093 (PC400) or 4094 (LoggerNet). This option lets you change this default.

**Search for Neighbors (server only)** - When this option is selected, the software server will broadcast a Hello Request every 5 seconds to search for PakBus neighbors with which it can communicate. During this time, the PakBus port is kept on-line.

**Broadcast Reset (server only)** - This option will reset the routing table in the selected PakBus device, as well as any neighbors of the selected device that are acting as routers.

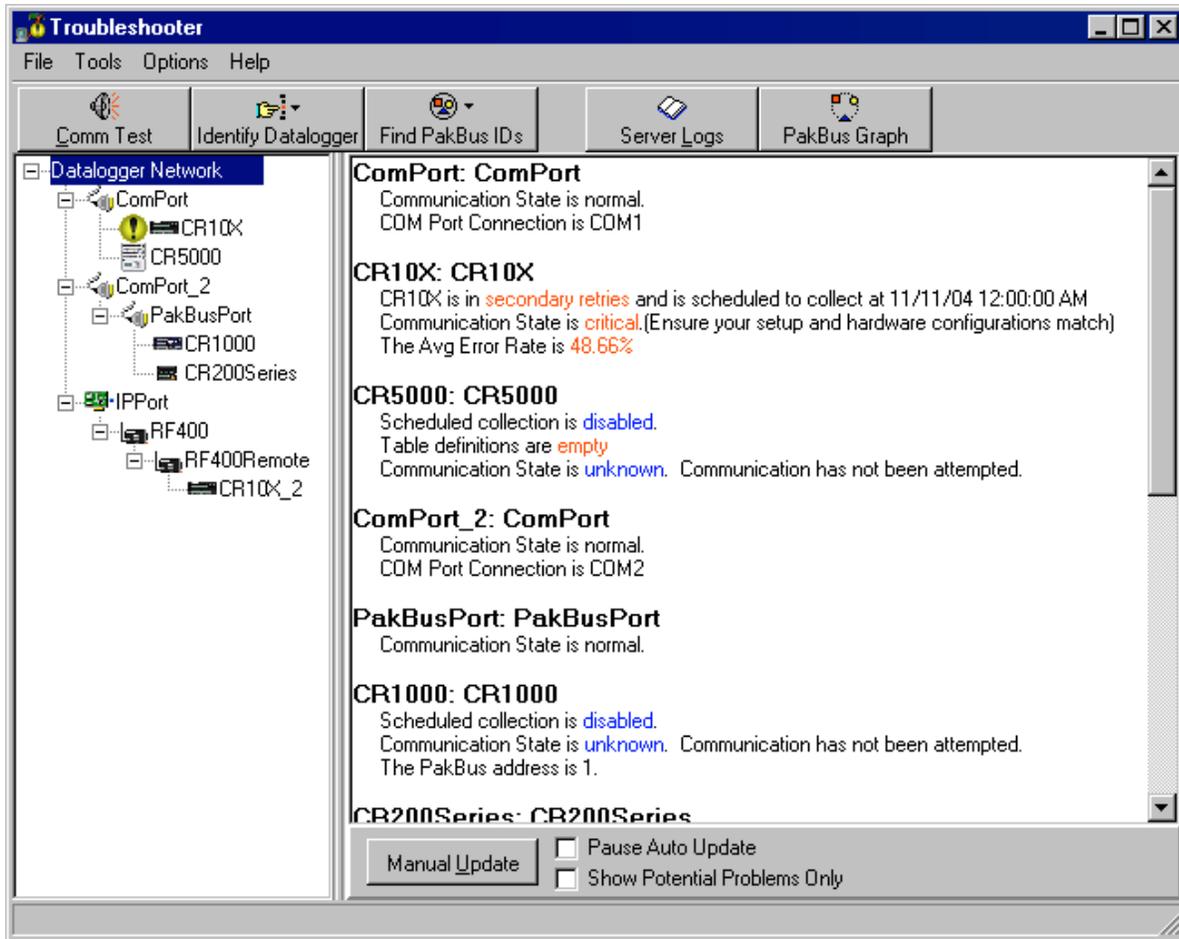
### 6.4.5 Discovering Probable Routes Between Devices

You can view the probable route that communication will take between two PakBus devices by sequentially clicking on the two devices in Pakbus Graph. The probable communication route will be highlighted in cyan. If the Show Hop Metrics check box is selected, the graph will include the time, in milliseconds, that communication takes between the two devices. The results are also displayed in the Log Messages portion of the window.

## 6.5 Troubleshooter

The Troubleshooter is a tool that can be used to help assess communication problems in a datalogger network. The Troubleshooter can be opened from the LoggerNet Toolbar's Tools menu.

When the Troubleshooter is opened, all communication ports, peripheral communication devices, and dataloggers that have been set up in LoggerNet will be displayed on the left side of the window. If a device is highlighted, the right side of the window will display status information for it and for other devices in its communications link. If Datalogger Network is highlighted, the status of all devices will be displayed. Note that you can suppress all status information but potential problems by selecting the Show Potential Problems Only check box. You can also limit the display to show only dataloggers by selecting Options | Show Stations Only.



The information will be different for each device, but may include the type of device, the device name in the network map, the state of communication with the device, whether or not scheduled data collection is enabled, whether or not table definitions are valid, modem type, phone number, device address, and the error rate.

Note that the information in the Troubleshooter is not updated until communication is attempted with the datalogger (either manually from the Connect window or automatically based on scheduled data collection). If the Troubleshooter is opened and you suspect a change in the status of one or more devices, select the Manual Update button to have LoggerNet retrieve the latest information from the network. You can also suppress the update, regardless of whether or not data is collected, by selecting the Pause Auto Update check box.

### 6.5.1 Status Information

Much of the status information is self-explanatory. Some information which may require further definition follows.

**Communication State** - Communication State will be listed as Unknown until communication is attempted with a device. Once a communication attempt has been initiated, it will be listed as Normal (communication is successful), Marginal (there are some errors in communication, but none that cannot be recovered from), or Critical (communication failed).

**Scheduled Collection** - This is the automatic data collection schedule. The state is either enabled, disabled, or paused. Scheduled data collection is paused from the Status window (Pause Schedule check box).

**Table Definitions** - Table-based dataloggers return information on all the data tables in the datalogger to the LoggerNet server. The state can be Valid or Out of Date. If table definitions are out of date, they must be updated before data can be collected from the datalogger (LoggerNet attempts to do this automatically. Table definitions can be updated manually from the Setup window's Data File tab for the datalogger.)

**Avg Error Rate** - A running average, expressed in percent, of the number of communication failures and retries over a period of time.

## 6.5.2 Retrieving Information for a Datalogger

This Identify Datalogger option is used to determine the type of datalogger attached to a device such as a COM port or PakBus port. When the Identify Datalogger button is pressed a list of all devices is displayed. Select a device (i.e., a COM port) and wait a few moments for the datalogger information to be returned. The message will vary, depending upon the datalogger type:

Message	Datalogger Type
A classic datalogger was found.	Edlog dataloggers, default OS (CR10X, CR510, CR23X, CR7, 21X)
A BMP1 datalogger was found.	Edlog dataloggers, table-based OS (CR10X-TD, CR510-TD, CR23X-TD)
A PakBus datalogger with a PakBus address of XXXX was found.	Edlog datalogger, PakBus OS
A CRX000 was found.	CR1000, CR5000, CR9000
A PakBus datalogger with a PakBus address of XXXX was found.	CR200

The Find PakBus IDs option is used to determine the PakBus IDs for one or more Pakbus dataloggers. The option works similarly to the Identify Device option. When the Find PakBus IDs button is pressed, a drop down list of all PakBus ports is displayed. Select a port and wait a few moments for the addresses of all PakBus devices found to be returned.

### **6.5.3 Archiving Troubleshooter Results**

The information displayed in the Troubleshooter can be saved to a text file or printed using the Save Analysis to File or Print Analysis option from the File menu. If Datalogger Network is selected, information for all devices will be printed or saved. If only one device is selected, only information for that device is printed or saved.

### **6.5.4 Other Tools in Troubleshooter**

Other network status tools can be launched from the Troubleshooter utility. These are the Comm Test, LogTool (Server Logs button), and PakBus Graph. Each of these utilities has been explained previously in this section. A Terminal Emulator can also be opened. Terminal Emulation is explained in the Troubleshooting section of this manual.

# Section 7. Creating and Editing Datalogger Programs

---

*Datalogger must be programmed before they can make measurements. LoggerNet offers three options for programming dataloggers. Short Cut, Edlog, and CRBasic Editor.*

*Short Cut (also referred to as SCWIN) is an application for generating programs for all of Campbell Scientific's dataloggers and preconfigured weather stations. Users do not have to know individual program instructions for each datalogger. Short Cut not only generates a program for the datalogger, but also a wiring diagram that can be left with the datalogger for field servicing.*

*The CR7, CR10, 21X, CR500, CR510, CR10X, and CR23X dataloggers are programmed using the Edlog editor. Edlog supports all operating systems for these dataloggers, including the table-data or "TD" and PakBus or "PB" versions. It is a full-featured editor that requires the user to understand the program instructions for the datalogger, but it can be used to develop more complex programs than what can be created using SCWIN.*

*The CRBasic Editor is the full-featured program editor for the CR5000, CR9000, CR1000, and CR200 Series dataloggers. Like Edlog, it requires that the user have more knowledge of datalogger program instructions than SCWIN.*

*In addition to the above programming tools, the Transformer utility is offered in LoggerNet for those users of CR10X dataloggers who need to develop programs for the CR1000 dataloggers.*

## 7.1 Review of CSI Datalogger Models

Campbell Scientific dataloggers can be broken down into two categories: Edlog dataloggers and CRBasic dataloggers.

Edlog dataloggers, the 21X, CR7, CR10, CR10X, CR500, CR510, and CR23X, come by default with operating systems that store data in one or two areas of final storage, with all intervals typically stored "end-to-end" in the same area of memory as individual arrays, hence the name "mixed-array" operating systems. Each array (e.g., 15 minute, hourly, daily) will have its own identifier that appears as an integer in the first position of the array. This is referred to as the Array ID. The other "elements" of the array store year, Julian day, hour-minute, seconds, and any of a variety of processing of measurements, such as average air temperature, total rainfall, minimum battery voltage, etc. To analyze the data, the user may find it useful to post-process the mixed-array data to extract the interval array of interest. Split (see Section 8) is ideally suited to do this.

Some of these Edlog dataloggers, specifically the CR510, CR10X, and CR23X, can alternatively be configured with table-data or PakBus operating systems. In these table-based configurations (CR510-TD, CR510-PB, CR10X-TD, CR10X-PB, CR23X-TD, and CR23X-PB), they measure the sensors the same

way, but store the processed data in individual tables instead of arrays. Each final storage table will contain only data for that interval – e.g., fifteen minute, hourly, and daily data records will be in different tables. The user can more closely control the size of these tables (for example to store a “buffer” of twelve hours of one minute data without taking up all of the available memory). In addition, the collected data file will have the date/time value in a single string – e.g.; “2004-05-15 13:50:00” – that is more readable in third party post-processing software. While there are some differences in communications between the table-data (TD) and PakBus (PB) operating systems, their measurement and final storage instructions are the same, so Short Cut treats them identically.

The alternatives to Edlog dataloggers are CRBasic dataloggers. These include the CR200/CR205, the CR1000, the CR5000, and the CR9000 (Short Cut does not support the CR9000). These dataloggers are also table-based, but programming syntax follows a format more like the Basic programming language, complete with declarations, scan sequences, and Basic-style logical statements.

---

**NOTE**

Those users who are moving from Edlog to the CRBasic dataloggers and who also need more control over datalogger programs, may find Short Cut to be an excellent way to learn CRBasic. You can follow the same steps in Short Cut for a CRBasic datalogger as you would for an Edlog datalogger, but then open the program in the CRBasic Editor to see how Short Cut created the program.

---

## 7.2 Short Cut

### 7.2.1 Overview

Short Cut program generator creates programs for Campbell Scientific dataloggers in four easy-to-follow steps. You select the datalogger, choose how often and which sensors you wish to measure, and select intervals or trigger conditions on which to store data, and what processing to perform on the raw measurements for final storage. Short Cut also generates a wiring diagram for connecting your sensors to the datalogger.

Short Cut was designed to help the beginning datalogger programmer create datalogger programs quickly and easily. Short Cut effectively insulates the user from having to know the nuances of datalogger programming and the Edlog versus CRBasic programming languages. It supports the most commonly sold sensors from Campbell Scientific, as well as generic measurements (such as differential voltage, bridge, and pulse), commonly used functions (such as heat index calculation, alarm conditions, and simple controls), and multiplexer analog channel expansion devices.

Short Cut cannot be used to edit existing Edlog, CRBasic, or Short Cut for DOS programs. Program editing and more complex datalogger programming functions should be accomplished using our Edlog or CRBasic Editor programming tools.

Short Cut was designed with extensive built-in help. Help can be accessed at any time by pressing the F1 key. There are also Help buttons on most screens. You can also select the Help Contents page or the Help Index from Short Cut's Help menu. Help for each sensor can be accessed by searching the Help Index, following the links from the Contents page, or pressing the Help button from the sensor template.

Short Cut begins and ends with the Home screen. The four buttons on the home screen guide you through the four basic steps to generate a program:

Step 1 - Create a new file or open an existing Short Cut file

Step 2 - Choose sensors to monitor

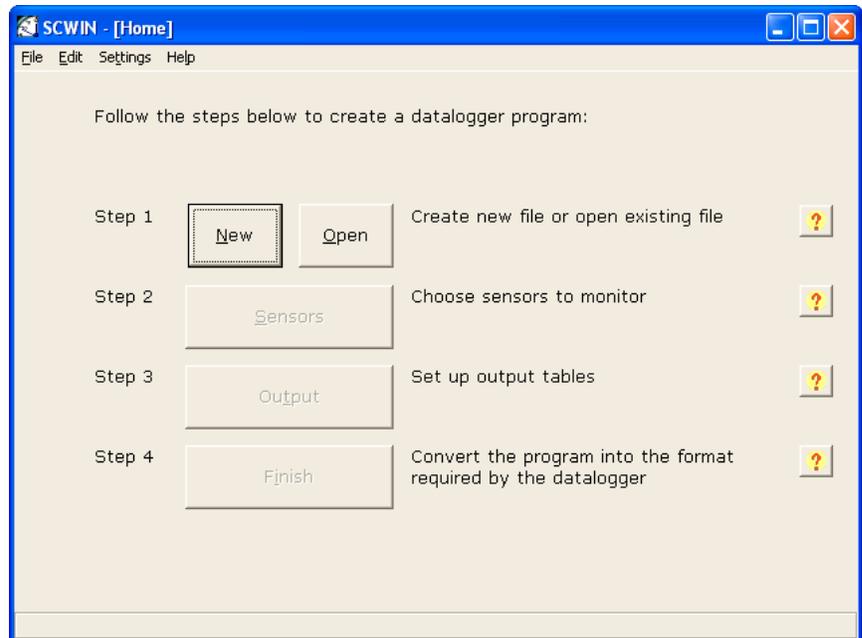
Step 3 - Set up output tables

Step 4 - Convert the program into a format required by the datalogger

After generating the program, you can send it to the datalogger from LoggerNet's Connect window or from the Setup window's Program tab.

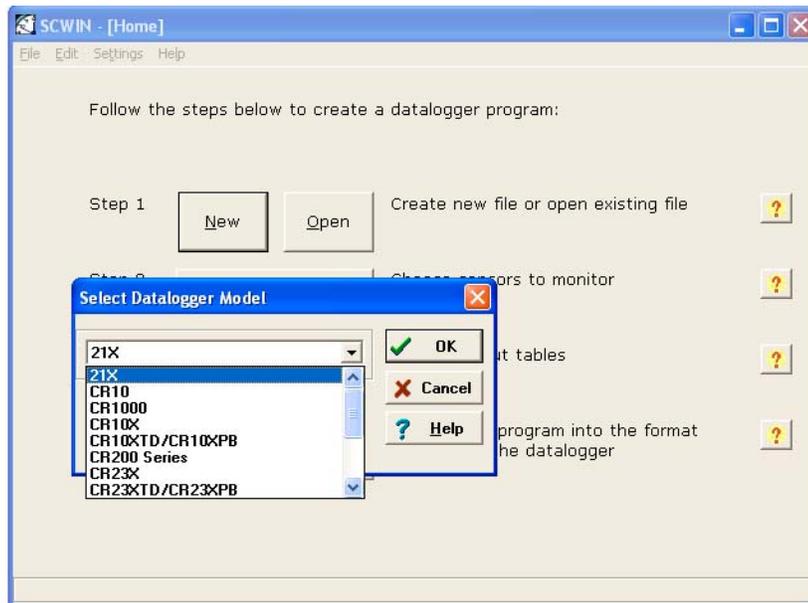
## 7.2.2 Creating a Program Using Short Cut

On opening, Short Cut presents a “home screen”. The four buttons lead the user to the primary tasks necessary to generate a datalogger program, but you can always return to the home screen.



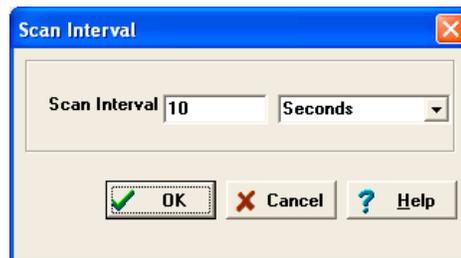
The “?” symbols at the right of each step open help screens that guide you through the most important aspects of each step.

### 7.2.2.1 Step 1 – Create a New File or Open Existing File



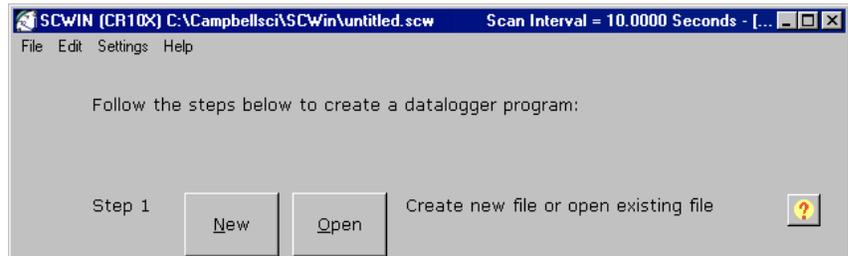
Step 1 offers to generate programs for any of Campbell Scientific’s dataloggers, including Edlog dataloggers with mixed-array, table-data, or PakBus operating systems or CRBasic dataloggers.

After selecting the datalogger type, Short Cut will ask for a Scan Interval, or period for each measurement cycle. Edlog dataloggers offer only units of seconds; CRBasic dataloggers offer milliseconds, seconds, or minutes.



When choosing a scan interval, remember that faster scan intervals will use more power. For most applications, a 10 to 60 second scan interval is sufficient. If faster scan intervals are required for your application, make sure there is sufficient time for the execution of all instructions in the program (refer to the section in the datalogger manual on Execution Intervals for additional information).

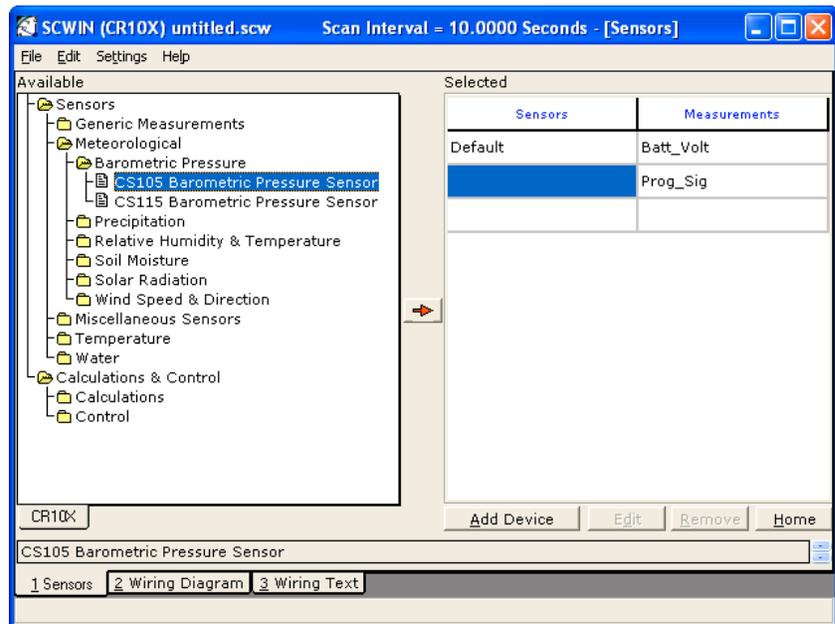
After making your selections, note that the title bar shows the datalogger type and scan interval:



Once you have saved the file, the filename will replace “untitled.scw”.

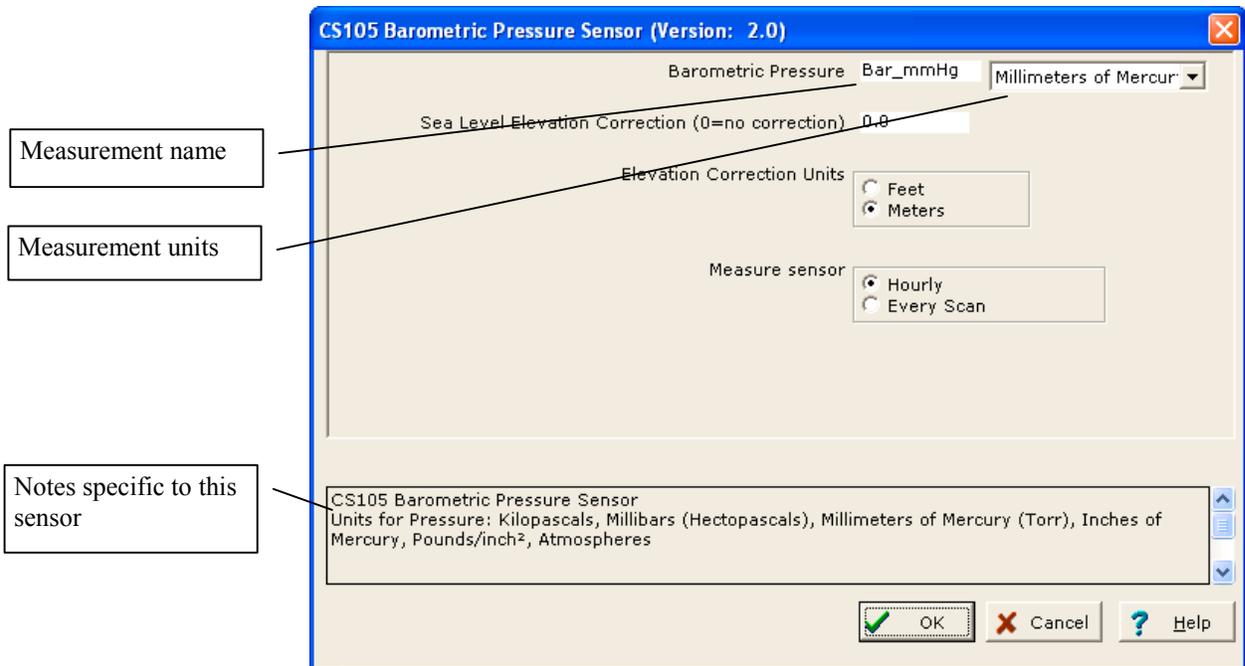
### 7.2.2.2 Step 2 – Choose Sensors to Monitor

In step 2, you tell Short Cut which sensors you’ll be measuring. Short Cut organizes sensors into application groups:



Some major groups have subgroups. Double-clicking the Meteorological group folder shows several subgroups of meteorological sensors, with the Barometric Pressure subgroup expanded to show two possible sensors. Refer to the documentation for your sensors for the name of the sensors you have. If your sensor is not shown, you may be able to measure it with a generic measurement. Contact your Campbell Scientific application engineer for more assistance, if needed.

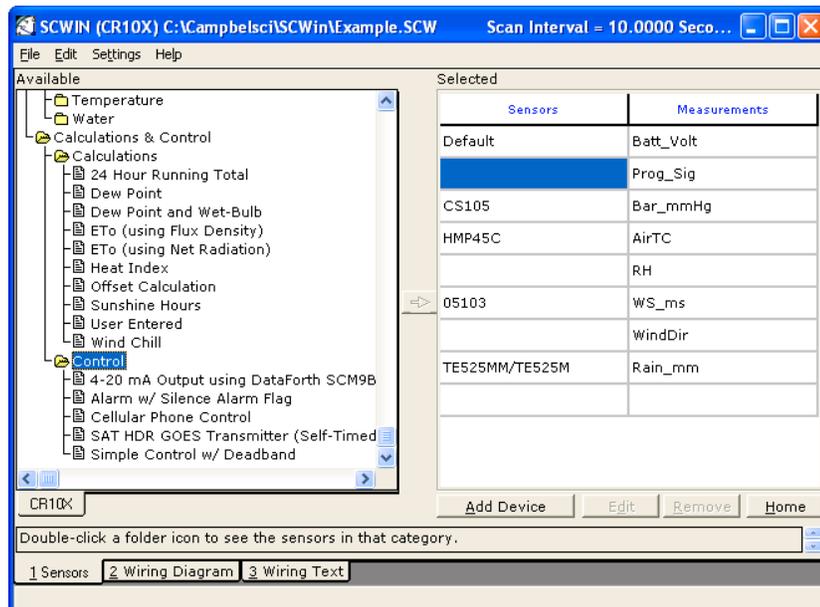
You “add” sensors to your program by double-clicking them or selecting them and clicking the red arrow in the middle of the screen. Most sensors will require you to at least review the default settings for that measurement, including the measurement name, units, etc. An example of choosing the CS105 Barometric Pressure Sensor is below.



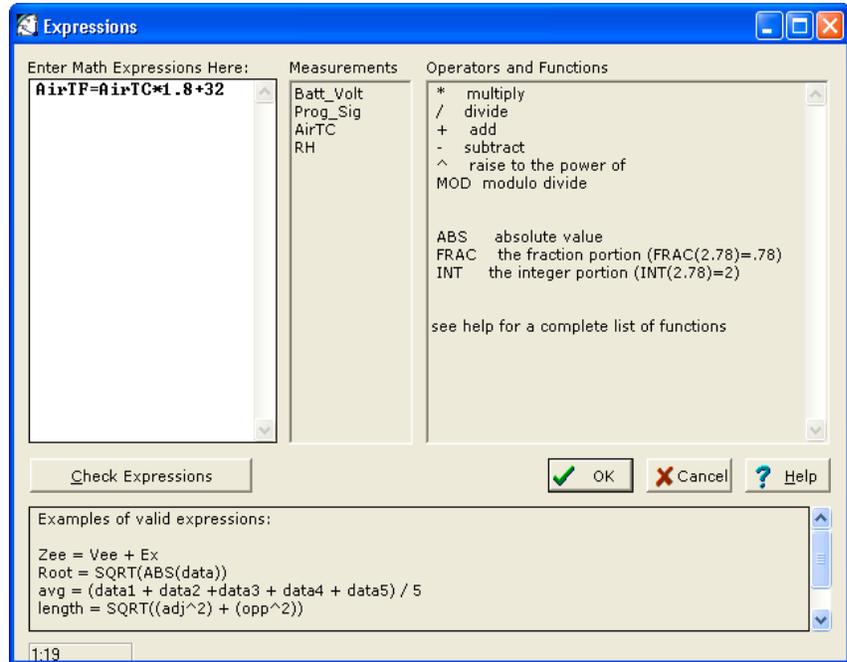
Note that this sensor offers not only a custom name field and units, but also allows you to correct for sea level, a common practice in measuring atmospheric pressure. In the middle of the screen, look over the notes (or refer to the Help for this sensor), for this sensor may require other sensors or have limitations. When you choose OK, Short Cut adds the necessary instructions with appropriate multipliers and offsets.

At any time, you may choose a measurement label on the right side of the screen and edit it or remove it.

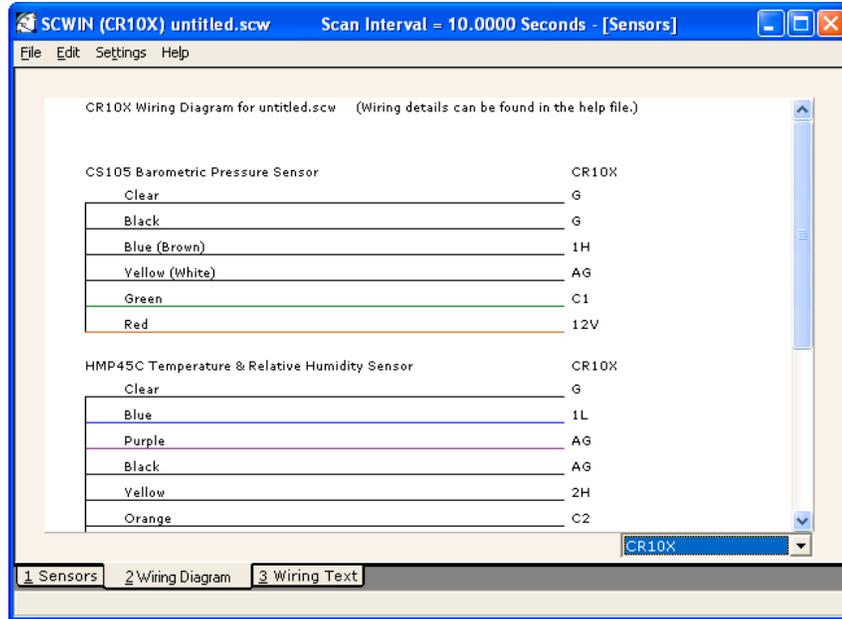
In addition to actual sensors, Short Cut provides “sensor files” to perform various calculations and effect some simple control:



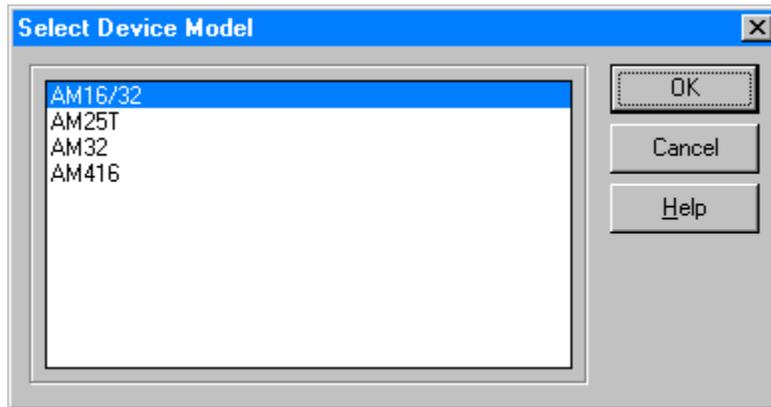
Some of these calculations may require additional sensors, or sensor measurements stored in particular units. See the help for each calculation to determine the necessary inputs. Note that there is also a User Entered calculation available in the Calculations folder. With it you can enter your own custom calculation:



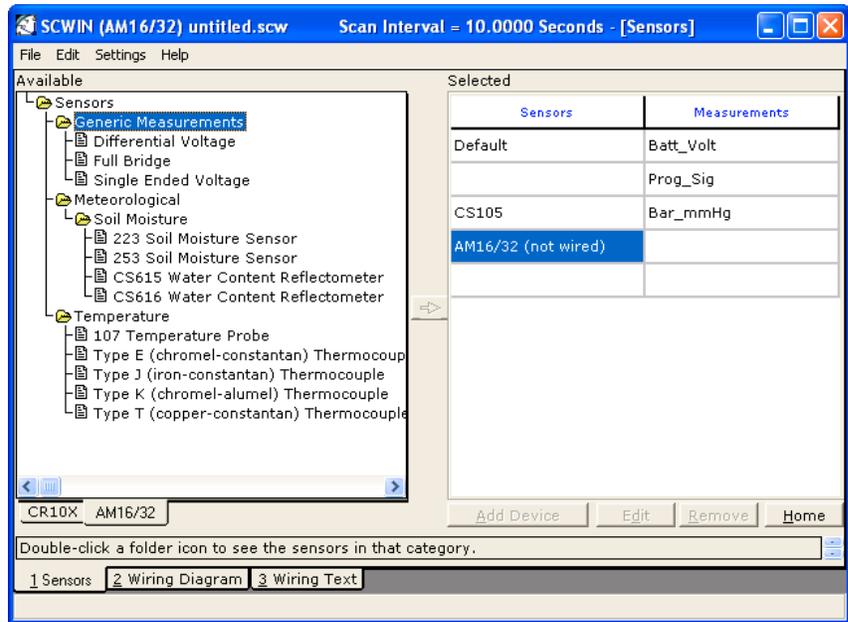
Short Cut provides you with a wiring diagram by clicking on the **Wiring Diagram** tab at the bottom of the Sensors window. In the example below, Short Cut was told to measure a CS105 Barometric Pressure sensor, an HMP45C Air Temperature and Relative Humidity sensor and a 05103 Wind Speed and Direction sensor. Each sensor was allocated the necessary terminals. Short Cut will not let you add more sensors than there are terminals on that datalogger or device. You can print this diagram (or the textual equivalent) by choosing the File | Print option from the main menu. Many users find it handy to leave a printed wiring diagram in the enclosure with the datalogger in case a sensor has to be replaced.



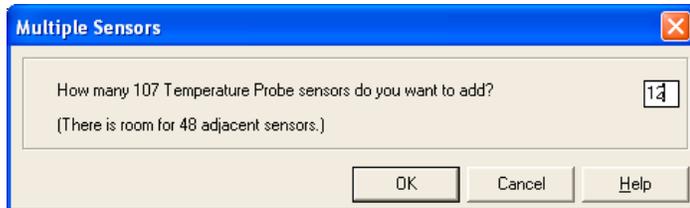
Short Cut can also create programs for dataloggers using a variety of interface devices, including multiplexers and special interfaces for vibrating wire sensors and shaft encoders. Add these devices with the **Add Device** button and resulting pop-up as below:



Once you've added a device, such as the AM16/32 multiplexer, a tab is added to the bottom of the screen for that device, and the sensors available for that device are shown:



You can then add sensors to that device just as you would to the main datalogger. When doing so, Short Cut may ask how many sensors to measure:

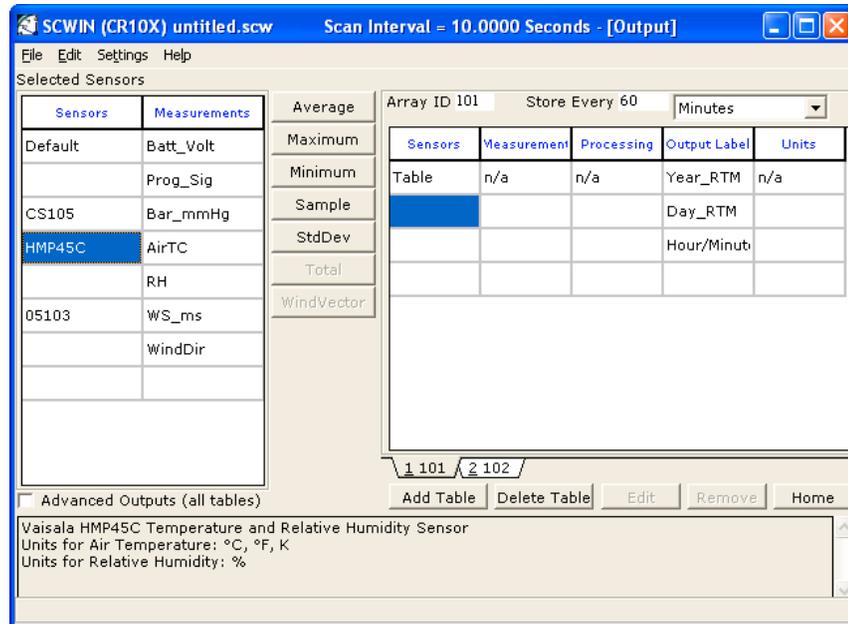


It will limit the number you can add to the available terminals on the device, and it will add the number of measurements you request, with unique measurement names to the list on the right side of the screen. Note that, once you add a sensor to a multiplexer, it may limit what kind of sensors can be added thereafter, as each sensor on the multiplexer must share the same wiring between the multiplexer and the datalogger.

When you're finished adding interface devices and sensors, click "Home".

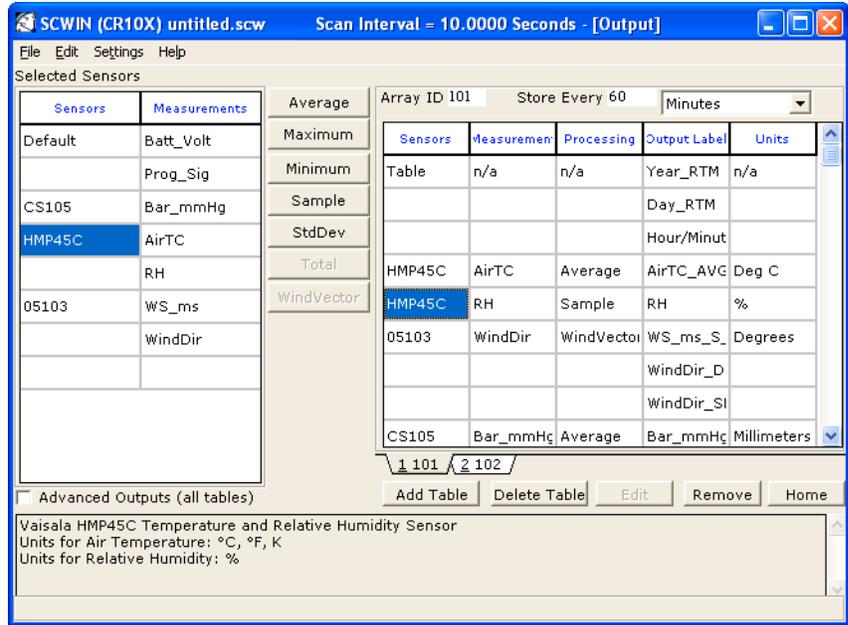
### 7.2.2.3 Step 3 – Setup Output Tables

After selecting the sensors to be measured, you can choose what data to store by clicking the Step 3 Output button:



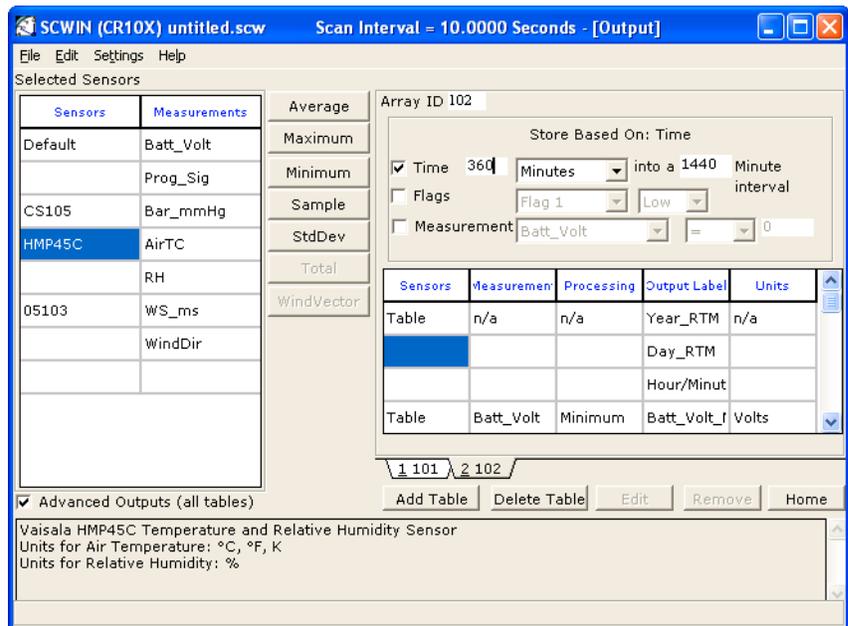
On the left, Short Cut will show the sensors you've added to be measured, with the measurement labels you've used. On the right is a multi-tabbed grid that shows the output tables. Most users store data as summaries over time – hourly and daily are common intervals for meteorological stations – but you can choose the table names (Array ID for Edlog mixed-array dataloggers) and intervals needed. You can even add additional tables as necessary.

To store a measurement to final storage, simply click on a measurement label at left, choose the data processing you want for that measurement by clicking one of the enabled buttons in the middle, and Short Cut adds the necessary instructions to save that data. In the example below, average air temperature, a sample of relative humidity, wind vector outputs, and average barometric pressure were selected for a 60-minute interval. Short Cut enables the most logical outputs for each measurement. If you require an output that is not enabled you can right-click on the measurement to get a pop-up menu containing all output options. You can also select a block of measurements (left-click+drag or left-click and shift+left-click) to do the same output on all of them. Note however that only output options common to all of the selected measurements will be enabled.



Note that outputs for a sensor don't have to be added in the same sequence as the measurement. You can even drag and drop the outputs to rearrange their order. Note also that multiple outputs can be added for any one sensor. For example, you may want to store the maximum and minimum air temperature as well as the average.

The Advanced Outputs check box in the lower left corner provides more output options:



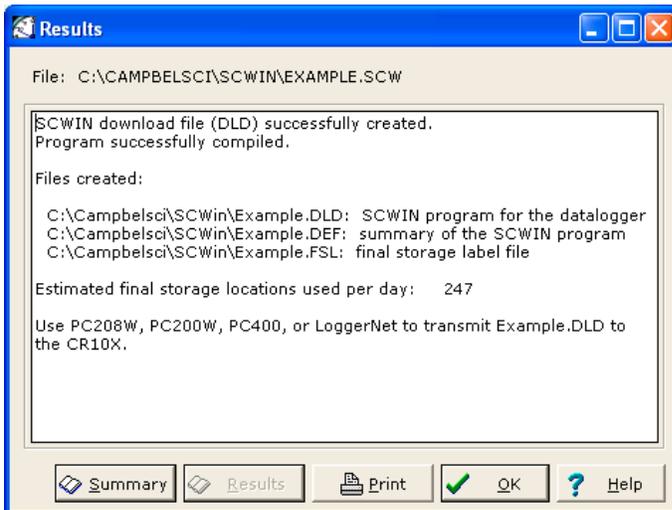
“Advanced mode” allows you to choose an offset into an interval – in the example above, the daily data is being output at 6:00 a.m. in the morning (or “360 minutes into a 1440 minute interval”). You could also choose to output based on an internal flag or some measurement value. For example, you may create a rain table that is only stored to when there is rain.

Be cautious in using more than one checkbox, for the logic for the check boxes in the advanced mode are inclusive – that is, they must all be true in order for any output to be stored. If you were to choose to store rain only if both the time and the measurement logic were chosen, then the very last measurement of rain – in the last 10 seconds – would have to have been greater than zero RIGHT AT 6:00 a.m. Otherwise, both conditions wouldn’t be true and the value wouldn’t be stored.

When you’ve configured all of your outputs, click Home again.

#### 7.2.2.4 Step 4 – Convert the Program into the Format Required by the Datalogger

The Finish button completes the process. If you haven’t yet saved the program, Short Cut asks for a program name and offers the default directory within its program working directory (default is C:\Campbellsci\SCWin). Short Cut also displays a Results window:



Several files are created when a program is saved:

- *ProgramName*.SCW (“Example.SCW” in this example) at the top of the screen is the file in which Short Cut keeps all of your selections for datalogger, sensors, outputs, etc.
- For mixed-array dataloggers, *ProgramName*.DLD is the ASCII text file that must be sent to the datalogger for it to make the measurements and store the data you want. For CR5000 or CR1000 dataloggers, this file will be the .CR5 or .CR1 file. For CR200 dataloggers, this file will be a .BIN (binary image) file.

- *ProgramName*.DEF is the text file that describes the wiring for the sensors and devices to the datalogger, measurement labels, flag usage, and the output expected. You can view the contents of the DEF file by clicking the Summary button on the Results screen.
- *ProgramName*.FSL is a text file containing output labels (created for mixed-array dataloggers only). This file can be used by Split or View or other software to provide column headers for the data file (not necessary for table-data and CRBasic dataloggers as their data files typically include header information).

Also note that, while Short Cut can generate a program file for the datalogger, you must use LoggerNet or some other datalogger communication software to transmit that program to the datalogger.

## 7.2.3 Short Cut Settings

The Settings menu on the Short Cut menu offers several settings that may prove useful.

### 7.2.3.1 Scan Interval

If you wish to change the scan interval after you've created the program, this option provides that opportunity. It presents the same pop-up window as when you first choose a datalogger.

### 7.2.3.2 Program Security

Some dataloggers allow you to set security by entering one or more numbers into their security fields. You can allow different levels of access (e.g.; only allow data retrieval, or also allow monitoring of values, or also allow sending a new program or setting the clock) by entering multiple levels.

Datalogger security is not meant to be extremely tight. Rather, it is designed to prevent honest people from making mistakes.

Notwithstanding its intention, one mistake you can make is to set security and then forget the values. If you send a program with security set, you will then need to add that security setting to LoggerNet's Setup window for that datalogger. If you don't, you may find that you can no longer communicate with the datalogger. Should this happen and you forget the security code and have lost the Short Cut program file, you may have to visit the datalogger site and cycle power on the datalogger to be able to communicate with it. Most dataloggers that offer security will communicate over their CS I/O port directly with a keyboard/display or PC in the first few seconds of powering up. See the datalogger manual for a full description of the security features.

### 7.2.3.3 Datalogger ID

Mixed-array dataloggers keep a memory location available for a datalogger ID value. This is typically an integer that you can read from within the program and store into final storage to keep track of the identity of the datalogger that created the data. Valid Datalogger IDs are 1 through 12 and 14 through 254. Use the Datalogger ID instruction in Short Cut (found under Miscellaneous Sensors) to use the ID in the datalogger program.

### 7.2.3.4 Power-up Settings

Some dataloggers offer the option to retain interim measurements or calculations or the states of flags or ports when they power-up from a low battery or loss of power condition. This may be useful when calculations are used to control devices. You may, for example, want to ensure that pumps or controls are off when a datalogger powers up so as to make the control decision based on a fresh measurement. See the datalogger manual for a full description of this feature.

### 7.2.3.5 Select CR200 Compiler

Use this setting to select the directory and executable name that will be used to pre-compile the CR200/205 program to check for errors.

Most Campbell Scientific dataloggers are sent an ASCII program file, which they then compile into machine code. The CR200/205 does not have enough memory and processing capability to do this compilation, so it's necessary to compile the program file into the binary version used by the datalogger itself. This compilation is done by Short Cut to check for errors in the program before sending it. It's done again by LoggerNet when sending the program to the datalogger. Compilation is performed using a special executable that mimics the functions and capability in the datalogger's operating system. Therefore, the compiler executable must match the datalogger's operating system or the datalogger may fail to run the compiled binary (\*.BIN) program. LoggerNet and Short Cut are installed with precompilers for all of the released versions of the CR200/205 operating systems. If, at some time in the future, you acquire a newer CR200/205, or choose to install a later operating system, you must make sure you also have the compiler executable that matches. These compiler executables are typically installed in library directory. By default, this directory would be installed as:

```
C:\Campbellsci\Lib\CR200Compilers
```

If you receive an operating system update, you should copy the compiler associated with it to this directory. If, for some reason, you put the compiler in a different directory, this menu item provides a way to choose that compiler executable.

### 7.2.3.6 AC Noise Rejection

Some dataloggers have parameters available in their measurement instructions to provide some rejection of noise due to AC electrical signals. These parameters will be used by Short Cut if possible, but the frequency of this noise varies. In most of North America, the AC frequency is 60 Hz. However, in

many countries the frequency is 50 Hz. If you know the frequency of this AC noise, you can select one or the other frequency. This setting remains in effect for other programs generated by Short Cut until you change it.

### 7.2.3.7 Font

Use this setting to change the appearance of the font used by Short Cut. Most windows other than the wiring descriptions (which require a non-proportional font to make sure wiring diagrams are aligned) will use this font.

### 7.2.3.8 Set Working Directory

This setting changes the directory that Short Cut offers as a default for your programs. Upon installation, the default is set to C:\CampbellSci\SCWIN.

## 7.2.4 Editing Programs Created by Short Cut

Short Cut is very flexible and has many features. It does not, however, support all of the functionality in Campbell Scientific dataloggers. Some users will need to develop programs with capabilities beyond that offered by Short Cut, but will want to take advantage of the library of instructions and settings known to a program generator in order to get a head start.

For Edlog dataloggers, the easiest method is to Document the DLD file from within Edlog (discussed later in this Section). Short Cut creates a .DLD file to send to the datalogger that includes input location and final storage labels. Documenting a .DLD file causes Edlog to use the same labels and to show you the individual instructions being used to carry out the program. You can then add and delete instructions from within Edlog to add functionality to the program. Short Cut cannot import the files created by Edlog, however. Short Cut reads only its own SCW-formatted files.

For CRBasic dataloggers, you can use the CRBasic Editor to open the .CR# files directly. Again, Short Cut will not be able to open the files you've edited with the CRBasic Editor, since they are not an SCW file.

## 7.2.5 New Sensor Files

Short Cut was designed with future flexibility in mind. Datalogger and sensor support is provided as individual files and not part of the SCWIN executable. As new dataloggers and sensors become available, new definition files will be created to add and modify the necessary features known to Short Cut. To update these files, you can download the latest version of Short Cut from the Campbell Scientific website:

<http://www.campbellsci.com/resource.html>

It is also possible to have custom sensor files created for sensors your organization uses that are not included with Short Cut. Contact your Campbell Scientific applications engineer for details.

## 7.3 Edlog

### 7.3.1 Overview

Edlog is a tool for creating, editing, and documenting programs for Campbell Scientific's mixed-array dataloggers: CR7, CR500, CR510, CR10, CR10X, 21X, CR23X. Edlog also supports these same dataloggers configured with table-based operating systems, including the table-data or "TD" and PakBus or "PB" versions. It provides a dialog box 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.

#### 7.3.1.1 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 errors. Errors that the precompiler misses should 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), that provides an estimate of program execution time, is also created. For mixed-array 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.

#### 7.3.1.2 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 menu bar or toolbar.

#### 7.3.1.3 Programming Efficiency

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**

We recommend that you do not copy instructions from a program written for one datalogger to a program for a different type of datalogger. Instructions may differ between dataloggers and cause unexpected results when downloaded.

---

#### 7.3.1.4 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 that 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 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.

### 7.3.1.5 Final Storage Label Editor

The Final Storage Label Editor allows you to change the default labels assigned by Edlog. The labels are stored in the Final Storage Label file for mixed-array dataloggers and as part of the datalogger program for mixed-array and table-based (both table-data and PakBus) dataloggers.

### 7.3.1.6 Expression Compiler

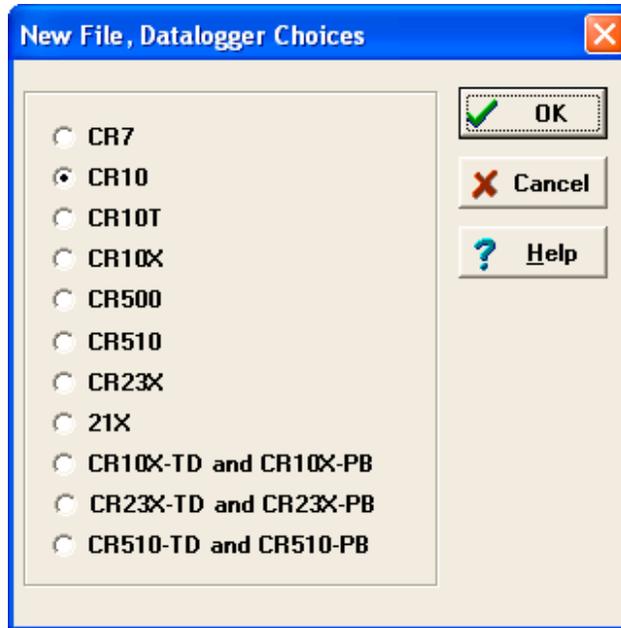
Mathematical calculations can be written algebraically using Input Location labels as variables. When the program is compiled, Edlog will convert the expressions to datalogger instructions.

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.

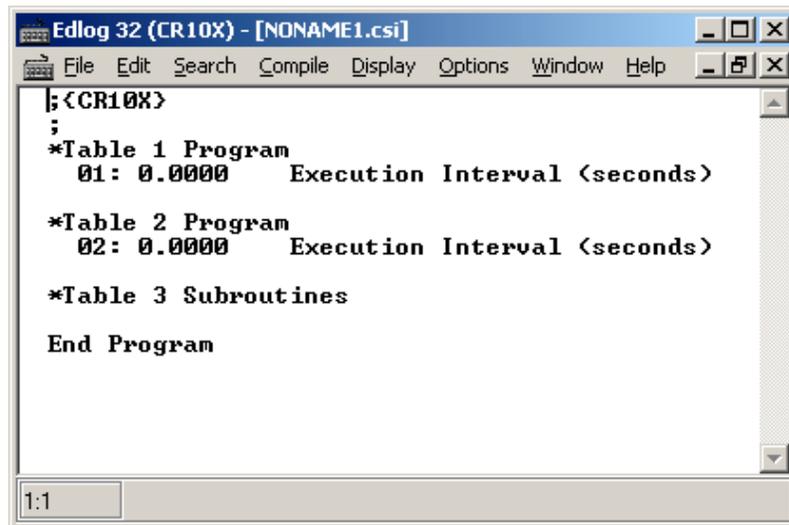
$$\text{TempF}=\text{TempC}*1.8+32$$

## 7.3.2 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 window 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. (The asterisk is used to identify the beginning of a program table in the datalogger.) 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 compiled the code will be checked for errors. After compiling, the datalogger program can be sent to the datalogger using the Connect screen.

### 7.3.2.1 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.

### 7.3.2.2 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. Although CSI files are ASCII files they require a particular format, so editing the \*.CSI files with some other text editor can corrupt the Edlog programs so that they no longer load or 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 – Program Trace Information 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 – Final Storage Label files contain the final storage labels for the data values in the output data records. This file is used by Split to show labels for data values in reports, and by View for column headings. FSL files are not created for table-based dataloggers. Table-based datalogger program files contain the final storage labels.

Other files that are used in Edlog but are generated by other means than compiling the program include:

- \*.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.

---

**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 will result in errors.

---

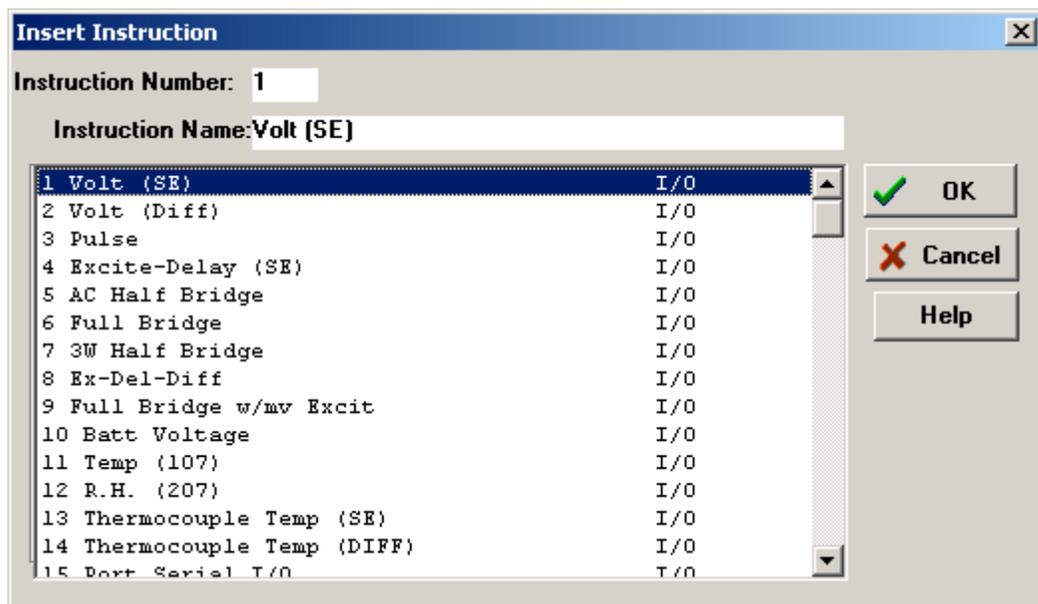
- \*.TXT - Printer output files created by Edlog are saved with a TXT extension. These files can be sent to a printer or viewed with a text editor. A TXT file is created by selecting File | Print to File.

### 7.3.2.3 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 Edit | Insert Instruction from the Edlog menu.
- Press <Shift>+<Insert> on the keyboard.
- Right click a blank line and select Insert Instruction from the pop-up menu.
- Type the instruction number onto a blank line and press enter.

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



To insert an instruction into the program, select it 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 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.

### 7.3.2.4 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 mouse 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 or potentially 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.

### 7.3.2.5 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 pop-up menu.
- Select Edit | Comment or Edit | Uncomment from the Edlog menu.
- 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.

### 7.3.2.6 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 7.3-1 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 innermost parentheses are evaluated first.
- To continue an expression to the next line, end the first line with an underscore ( \_ ).

TABLE 7.3-1. 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 <sup>2</sup>
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<sup>(-2)</sup>  
 Root = SQRT(ABS(data))  
 avg = (data1+data2+data3+data4+data5)/5  
 length = SQRT((adj<sup>2</sup>)+(opp<sup>2</sup>))  
 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)
  1: 1      REPS
  2: 2      Channel
  3: 1      Excitation Channel
  4: 2      Loc [TempC]
  5: 1      Mult
  6: 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)

1: 1  
2: 2  
3: 1  
4: 2  
5: 1.0  
6: 0.0

2: Z=X (P31)

1: 2  
2: 5

3: Z=F (P30)

1: 1.8  
2: 0  
3: 3

4: Z=X\*Y (P36)

1: 3  
2: 5  
3: 5

5: Z=F (P30)

1: 32  
2: 0  
3: 3

6: Z=X+Y (P33)

1: 3  
2: 5  
3: 6

### Errors That Can Occur With Expressions

Some of the error messages that occur when using expressions need no further explanation:

- Missing left parenthesis
- Missing right parenthesis
- Variable name expected
- Number expected
- Floating point numbers limited to 5 digits
- Function expected
- New line expected
- Equal sign expected

Other errors are explained below.

### 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:

```
5=e1*(Vee+en)
(lambda) = COS(theta)
10-(zee/2)=bee
```

These are correct ways of entering the above expressions:

```
five=e1*(Vee+en)
lambda = COS(theta)
bee=10-(zee/2)
```

### 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, 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=7.52(ef/2)
sigma=-17(RCP(alpha))
```

These are correct ways of entering the above expressions:

```
tee=5*(2)
mu=(nu)*103
bee=7.52*(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
```

### 7.3.2.7 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 7.3-2 provides a list of keystrokes that can be used in editing programs and moving around in Edlog.

TABLE 7.3-2. 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

### 7.3.2.8 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.

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

Edit | Delete, Edit | Cut, Edit | Copy, and Edit | Paste allow sections of the program to be deleted, 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.

## 7.3.3 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 library 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.

---

## 7.3.4 Documenting a DLD File

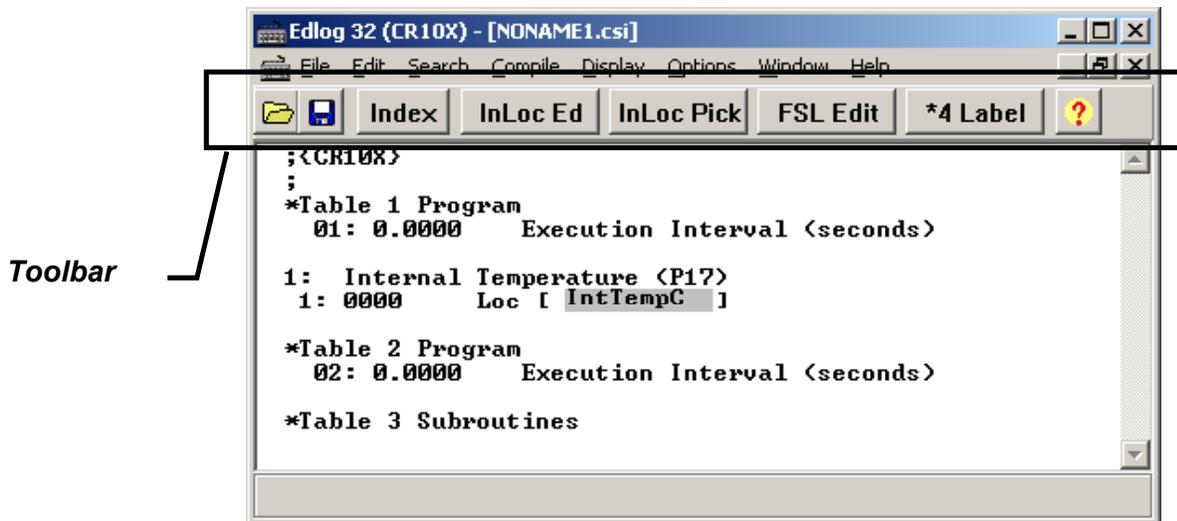
The CSI file is the file created by Edlog that is used to generate the DLD code and other files. If for some reason your CSI file is missing, you can import the DLD file into Edlog to create another editable CSI file. From the Edlog menu select File | Document DLD. Select the DLD file to be imported 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. The DLD version of these programs can be imported into current versions of Edlog by using this Document DLD feature, though any comments will be lost.

## 7.3.5 Display Options

### 7.3.5.1 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).



Open the input location editor.



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



Open the final storage label editor



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.

### 7.3.5.2 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. If automatic renumbering is disabled, you can manually renumber the instructions by selecting Display | Renumber.

### 7.3.5.3 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.

### 7.3.5.4 Indentation

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

The settings for indentation are found under Options | Editor. Turn on Automatic Indentation by checking the box next to it. The distance for each indentation (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 Indentation 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.

## 7.3.6 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.

### 7.3.7 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.

Labels can have up to 9 characters for mixed-array 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.

### 7.3.8 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 input location of an instruction is linked to the instruction. 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)
   1: 1      REPS
   2: 2      Channel
   3: 1      Excitation Channel
   4: 1      Loc [TempC]
   5: 1      Mult
   6: 0      Offset

11: Battery, Volt (P10)
   1: 2      Loc [BatteryV]
    
```

```

12: If time is (P92)
  1:  0      minutes into interval
  2:  60     minute interval
  3:  10     Set high Flag 0(output)

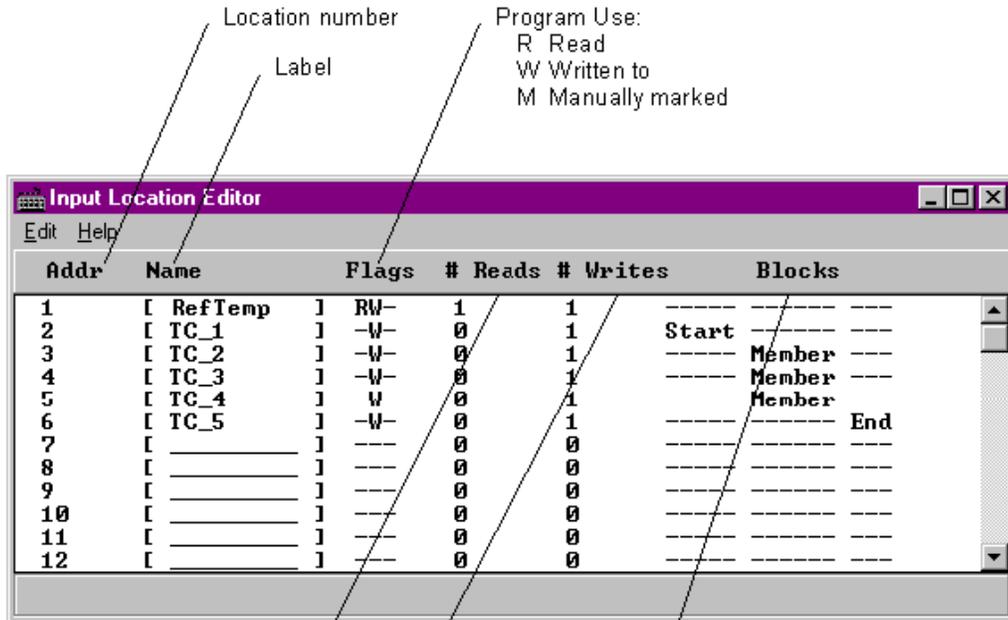
13: Sample (P70)
  1:  2      Reps
  2:  1      Loc [TempC]
    
```

When the program is executed, the datalogger will perform the Sample (P70) 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 BatteryV location.

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

### 7.3.9 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 or press <F5>.



Number of Instructions in the program that read the location

Number of Instructions in the program that write to the location

Reps or Manual Block  
Start First of Rep.  
End Last of Rep.

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 existing 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.

### 7.3.10 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 139, Detailed Program Signatures
- Instruction 188, SDI-IO16
- Instruction 189, SDM-LI7500
- 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.

### 7.3.11 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 processing instruction that created it. In the example below BattVolt\_AVG is the average battery voltage that is stored as part of array 112.

For mixed-array dataloggers the final storage labels are stored in an \*.FSL file when the program is compiled, as well as in the DLD files. For table-based dataloggers the final storage labels are included as part of the datalogger program in the \*.DLD file; no FSL file is created. LoggerNet gets the final storage labels as part of the table definitions from the datalogger. Split, the Graphical and Numeric Displays, View, and the LoggerNet Data applications use the final storage labels.

The user can create a custom label to reflect the meaning of the value that is being stored. Click 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 a mixed-array 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.

- Array ID or Table Name identifies the set of output data instructions the data is associated with. For mixed-array 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 store 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 from the Final Storage Label Editor menu.

### 7.3.12 Datalogger Settings Stored in the DLD File

Certain settings for the datalogger, which are normally accessed through the datalogger's \* modes, can be included in the DLD file. These settings include options such as program security, final storage allocation, the type of labels saved in the DLD file, power up and compilation settings, and PakBus address and router settings. When the new program is downloaded to the datalogger and compiled, the settings will take affect. These settings are accessed using Edlog's Options menu.

### 7.3.13 Program Security

Setting security in the datalogger allows you to restrict access to certain functions, which helps ensure the program or data are not altered. Security is unlocked in the datalogger when, upon attempting to connect, LoggerNet sends the code entered in the Setup window.

#### 7.3.13.1 Setting Passwords in the DLD

In the Program Security Dialog Box, there is a field for three levels of security. Enter a non-zero number in the appropriate field to enable security at the desired security level. This number is used as a password to unlock the associated security level when needed. If you choose to set level 02, level 01 must also be set. Likewise, if you set level 03, levels 01 and 02 must be set.

**NOTE** CR7 and 21X dataloggers have only 1 level of security.

---

#### 7.3.13.2 Disabling Passwords

Passwords of 0000 disable the program security. When you disable level 01, you also disable program security for levels 02 and 03. Similarly, disabling level 02 disables level 03. All passwords are set to 0000 upon power-up of the datalogger. When the program is run, security is enabled.

Refer to the datalogger manual or Edlog's help file for additional information on Security in the datalogger.

### 7.3.14 Final Storage Area 2

The ring memory for CR10, CR10X, CR510, and CR23X dataloggers can be divided into two final storage areas. By default, all memory is allocated to final storage area 1. However, the datalogger's memory can be partitioned into two final storage areas using this option. To allocate memory to Final Storage Area 2, enter the number of locations into the Final Storage Area 2 Locations field.

Final storage area 1 is reduced by the amount of memory allocated to final storage area 2. Data stored in final storage area 2 is protected when Input and/or Intermediate Storage is reallocated. Data stored in final storage area 1 is erased when any of the memory areas are reallocated.

### 7.3.15 DLD File Labels

This option allows you to determine the labels that are saved in the DLD file for the datalogger. While labels are useful, they can increase the size of the datalogger program considerably. If program size is a concern, you can limit or completely eliminate the labels that are saved in the DLD file.

#### 7.3.15.1 Mixed-array Dataloggers

Mixed-array dataloggers can store the labels for input locations and final storage output in the DLD file. LoggerNet uses this information on the Monitor Values displays. If you do not include these labels in the DLD file, you will see generic names for input locations, and will not be able to display final storage locations at all.

Options for mixed-array dataloggers are:

**Minimize DLD Size** - No input location labels or final storage labels are saved in the DLD file.

**Default** - Up to 255 input location labels and all final storage labels are saved in the DLD file.

**All** - All input location labels and all final storage labels are saved in the DLD file.

#### 7.3.15.2 Table-Based Dataloggers

Table-based (both TD and PB) dataloggers store all final storage labels in the DLD file and there is no option to remove or reduce them. If you do not include input location labels in the DLD file, you will not be able to display input locations on the displays in the software.

The label options for table-based dataloggers are:

**Include All Input Location Labels** - All input location labels are saved in the DLD file.

**Include First X Input Location Labels** - Allows you to specify a certain number of input location labels to be saved in the DLD file.

If you are trying to minimize the size of your DLD file but still want to be able to monitor input locations in the software, you can put all of the labels that you

want to view at the beginning of your list of input locations, and put the labels for scratch and less important values at the end. Then, use the second option above to display only those values of interest.

### 7.3.16 Power Up Settings/Compile Settings

These two options allow you to clear or retain settings for ports, flags, storage locations, and timers when the datalogger is powered-up or when a program is compiled. Whether it is advantageous to clear or retain these settings depends on your application. For most applications, it is best to keep the default option of Do not change current datalogger Power-up settings. The affected settings are:

**Port Status** - The state of the ports (high/low) the last time the datalogger was on.

**Flag Status** - The state of the flags (high/low) the last time the datalogger was on.

**User Timer** - Allows you to continue timing events that occurred when the datalogger was on last.

**Input Storage** - Allows the values that were stored in the input locations before you turned the datalogger off to be included in the sample, average, and total when you turn the datalogger back on.

**Intermediate Storage** - Allows data processing to continue from when the datalogger was on last.

---

**NOTE**

Not all dataloggers have a Compile Settings option. This option refers only to the CR510, CR10X, and CR23X.

---

### 7.3.17 Datalogger Serial Port Settings

The serial port settings are used to set the baud rate to which the datalogger's port(s) should be set when the datalogger is powered-up or when a program is compiled. If the "Do not change current CS I/O Port settings" option is selected, the baud rate option used will be that at which the datalogger is currently using.

When the "Fixed Baud Rate" check box has been selected, the datalogger is forced to communicate at the baud rate selected. When it is not selected, the datalogger will first try to use the initial baud rate, but will try the other baud rates if it cannot connect.

The CR23X has an "RS232 Power Always On" check box. This keeps the power to the RS232 port on at all times. In some instances, this may be desirable but it consumes much more power than when the datalogger turns on the port as needed.

---

**NOTE**

Not all dataloggers have a Serial Port Settings option. This option refers only to the CR510, CR10X, and CR23X.

---

## 7.3.18 PakBus Settings

PakBus dataloggers have various settings that allow them to function properly in a PakBus network. In Edlog dataloggers with PB operating systems, these options can be set in the datalogger's \*D mode with a keyboard/display, but they can also be set in the DLD program file

For any of the options, if the check box Do Not Change Current Settings is enabled, then those settings will not be changed when the program is downloaded to the datalogger.

### 7.3.18.1 Network

The Network option is used to set the PakBus address in the datalogger and to configure the datalogger as a router if required. This option is the same as the datalogger's \*D15 mode.

**Address** - Enter the PakBus address that should be assigned to the datalogger. Each node (PakBus addressable device) in the PakBus network should have a unique PakBus address.

**Maximum number of nodes** - Enter the total number of nodes (including leaf node and router dataloggers, non-datalogger routers such as NL100s, and PCs) in the PakBus network.

**Maximum number of neighbors** - Enter the number of dataloggers in the PakBus network that the datalogger can communicate with directly (i.e., without going through another router).

**Maximum number of routers** - Enter the number in the PakBus network, including the PC.

While it is possible to calculate the exact number of nodes, neighbors, and routers in a PakBus network, it is often advisable to build in some "room to grow". For example, you might want to add 3-4 nodes, neighbors and routers. Be aware that each device you add means the datalogger must allocate memory for its routing table, so if you add too many, the datalogger won't have enough memory left to run its program.

### 7.3.18.2 Beacon Intervals

This option is used to set the interval on which the datalogger will transmit a beacon out a particular port to the PakBus network. Use the drop-down list box to select the port over which the beacon will be transmitted, and enter the desired interval in the Communications Interval field. This option is the same as the datalogger's \*D18 mode.

---

#### NOTE

In some networks, a beacon interval might interfere with regular communication in the PakBus network (such as in an RF network), since the beacon is broadcast to all devices within range. In such cases, it may be more appropriate to use the Neighbor Filter instead, which broadcasts a beacon only to those dataloggers which it has not received communication from within a specified interval.

---

### 7.3.18.3 Neighbor Filter

This option allows you to list expected neighbors that are available to the datalogger in the PakBus network. The datalogger will attempt to issue a "hello" command to all the dataloggers listed in the neighbors filter list, and will transmit an expected communication interval. The communication interval is the interval on which the datalogger expects to receive communication from the neighbors. If communication is not received from a neighbor within 2.25 times this interval, then the datalogger will attempt to issue another "hello" command to that datalogger only (thus, creating less network traffic than the Beacon Interval).

The expected interval is entered into the Communication Interval field in seconds. The neighbors are defined by entering their addresses into the table. A range of addresses can be entered by using the Swath field. For example, entering 1 for the address and 5 for the swath will set up dataloggers with PakBus addresses 1, 2, 3, 4, and 5 as neighbors to the current datalogger. This option is the same as the datalogger's \*D19 mode.

### 7.3.18.4 Allocate General Purpose File Memory

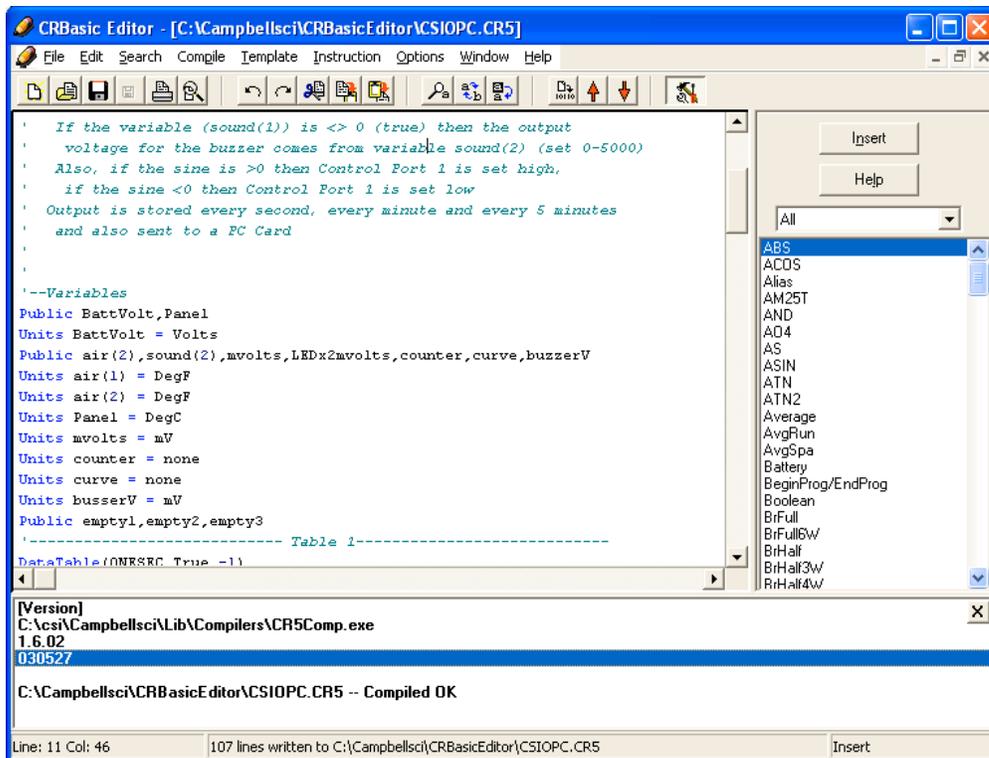
PakBus dataloggers have the ability to store files transmitted from an NL100 in a general purpose memory area. This memory area is configured as ring memory. A value can be entered to specify the number of 64K blocks of memory that should be used for this purpose. Final storage memory will be reduced by the amount of memory specified in this option. This option is the same as the datalogger's \*D16 mode.

## 7.4 CRBasic Editor

### 7.4.1 Overview

The CRBasic Editor is a programming tool for the CR5000, CR1000, CR9000, and CR200 dataloggers. It is intended for use by experienced datalogger programmers who need more flexibility and control over the datalogger operation than what can be achieved using SCWIN. 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 becomes visible after a program is compiled and shows results of the compile and any errors detected.



## 7.4.2 Inserting Instructions

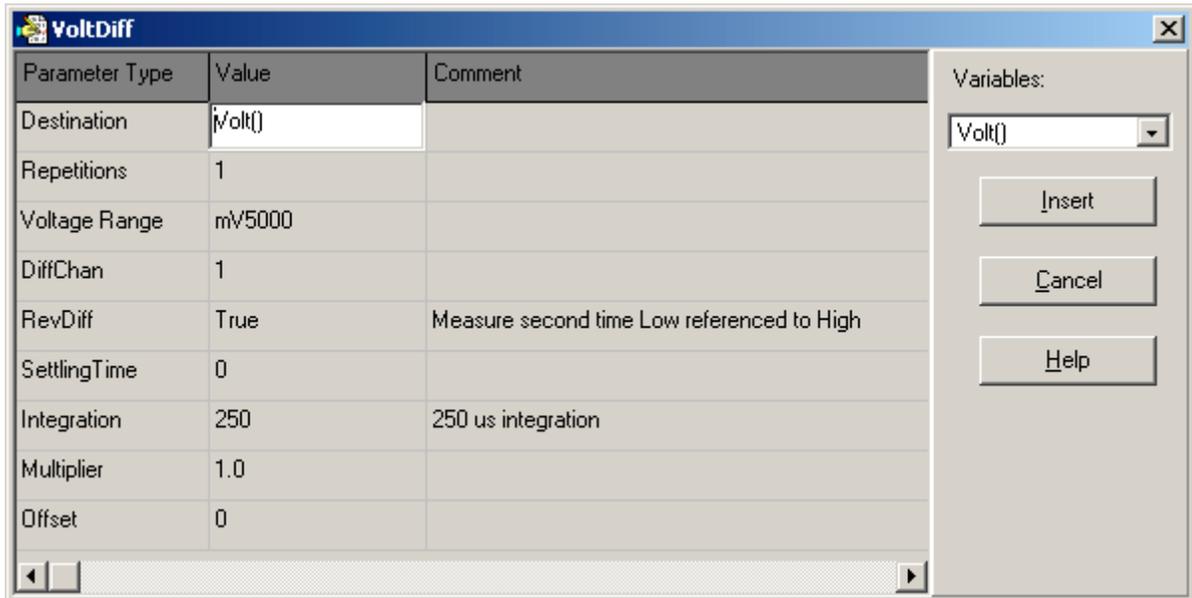
An instruction can be easily inserted into the program by highlighting it in the Instruction Panel list and pressing the Insert button or by double clicking the instruction name. If an instruction has one or more parameters, an instruction dialog box will be displayed to facilitate editing the parameters. Complete the information in the parameter fields and press Insert to paste the instruction into the program. (Some users will disable this instruction dialog box by clearing the option in the Options | Instruction Panel Preferences | Show Instruction Dialog check box.)

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 later in this section.

### 7.4.2.1 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).



### Short Cuts for Editing the Parameters

Right clicking or pressing F2 on a parameter that uses a Variable as an input type will display 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 above, 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.

## **7.4.2.2 Right Click Functionality**

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

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

Right click 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 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 the desired option.

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

Right click 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 indentation 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.

## **7.4.3 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 (enabled only for the CR9000) – 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.



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 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.

### 7.4.3.1 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 opened, 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.

When a CR200 program is being edited, the Pick CR200 Compiler menu item is available. This menu selection opens a dialog box from which a compiler can be selected for the CR200 datalogger. It is important that the compiler used for creating programs (and the associated binary image file) match the OS version of the datalogger or errors will be returned when the program is sent.

The error window can be closed by selecting the Close Message Window menu item or by clicking the X in the upper right corner of the message window.

---

**NOTE**

For CR5000, CR1000, and CR9000 dataloggers, this function only verifies the integrity of the program. Actual compilation of the program takes place in the datalogger. For CR200 dataloggers, a compiled binary image is created.

---

### 7.4.3.2 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 open 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.

---

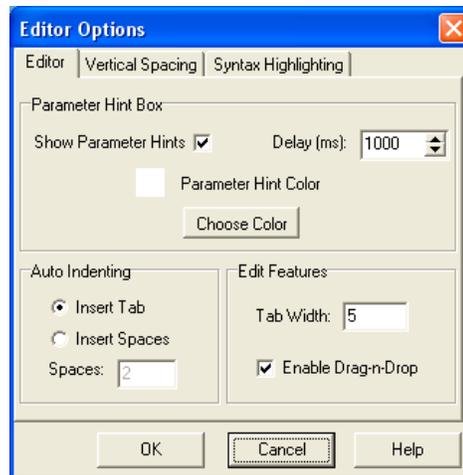
### 7.4.3.3 CRBasic Editor Options

This menu item allows you to specify the files used in the CRBasic Editor and customize its look and syntax highlighting.

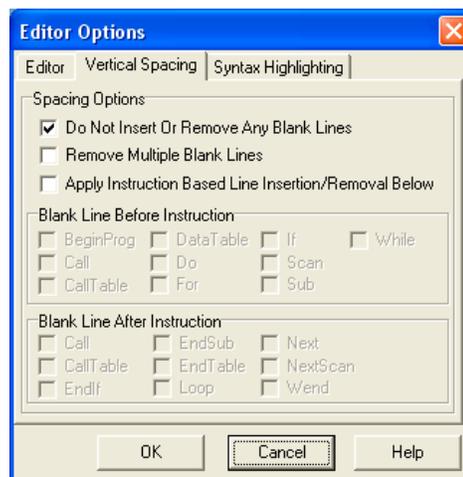
#### 7.4.3.3.1 Editor Preferences

This option sets up the appearance options for the text instructions and the behavior of popup hints.

The **Editor** tab allows the user to toggle on or off the popup hints for parameters in instructions, set the amount of time the cursor must hover over the instruction before the popup hint appears, and the background color of the popup hint. This is also used to choose whether CRBasic automatic instruction indenting indents using tabs or spaces, and set the number of spaces if that option is chosen.



The **Vertical Spacing** tab is used to set up the rules for the CRBasic Editor's Rebuild Indentation function (Edit | Rebuild Indentation). You can control whether blank lines are inserted before or after certain instructions, and how the CRBasic Editor will process multiple blank lines in the program. If "Do Not Insert or Remove Any Blank Lines" is selected, all other fields on this tab will be disabled. If either of the other two line options is chosen, the remaining fields will be available for the user to customize as desired.



The **Syntax Highlighting** tab sets up the appearance of different text elements in the program using different font styles and colors. You can customize the appearance of the text by giving normal text, keywords, comments, operators, numbers, strings, and parentheses 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.



Note that if special formatting (font style, color) is assigned to Matched Parentheses, when your cursor is on an opening or closing parenthesis it will be highlighted with the formatting, and the "other half" of that parenthesis will also be highlighted. When your cursor moves off the parenthesis, the formatting will return to normal text.

#### 7.4.3.3.2 Instruction Panel Preferences

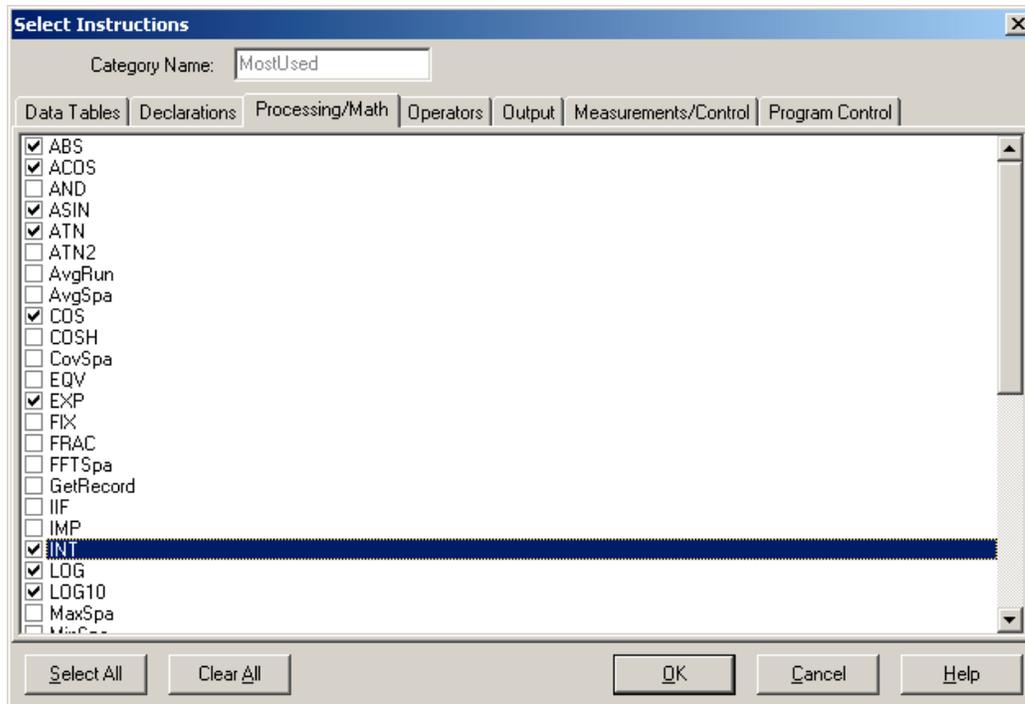
This option determines whether or not the instruction dialog box will be displayed when the user inserts an instruction.



#### 7.4.3.3.3 Edit Instruction Categories

Edit Instruction Categories allows the user to create one or more custom list of instructions. If a category of instructions is selected from the Instructions Panel, the entire list of instructions in the Editor will be filtered to show only those instructions in the selected category. (Note: The default categories cannot be edited or deleted.)

To create a new list, first select the Add New Category button and provide a name for the user-created category. Next, ensure the category name is selected and click the Edit Category button to bring up the Select Instructions dialog (shown below). Instructions that should be included in the new list are indicated by a check in the box to the left of the instruction name. This feature allows the user to display a filtered instruction list containing only those instructions most often used. Press OK to save the list.



#### 7.4.3.3.4 Other Options

**Font** - Displays 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** - Displays a 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.

#### 7.4.3.4 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.

## 7.4.4 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 provides a summary of a program, its syntax, structure, and sequence. Refer to the datalogger users manual or the on-line help for detailed information on program instructions.

### 7.4.4.1 Programming Sequence

The structure of a datalogger program requires that variables, data tables, and subroutines be declared 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>	

#### 7.4.4.2 Program Declarations

Variables must be declared before they can be used in the program. Variables declared as Public can be viewed in display software. Variables declared using Dim cannot be viewed. Variables assigned to a fixed value are used as constants.

For example, in a CRBasic program there may be multiple temperature (or other) sensors that are wired to sequential channels. Rather than insert multiple instructions and several variables, 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. To continue the example above, TCTemp(3) could be renamed using the following syntax:

```
Alias TCTemp(3) = AirTemp
```

In the display software, the more descriptive alias, AirTemp, would be used for the cell name.

### 7.4.4.3 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:

$$\text{TCTempF} = \text{TCTemp}(1) * 1.8 + 32$$

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  $\text{TCTemp}(1) > 100$ .

$$\text{DataTable}(\text{TempTable}, \text{TCTemp}(1) > 100, 5000)$$

### 7.4.4.4 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:

$$\text{PanelTemp } (\text{Dest}, \text{Integ})$$

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:

$$\text{PanelTemp}(\text{RefTemp}, 250)$$

### 7.4.4.5 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

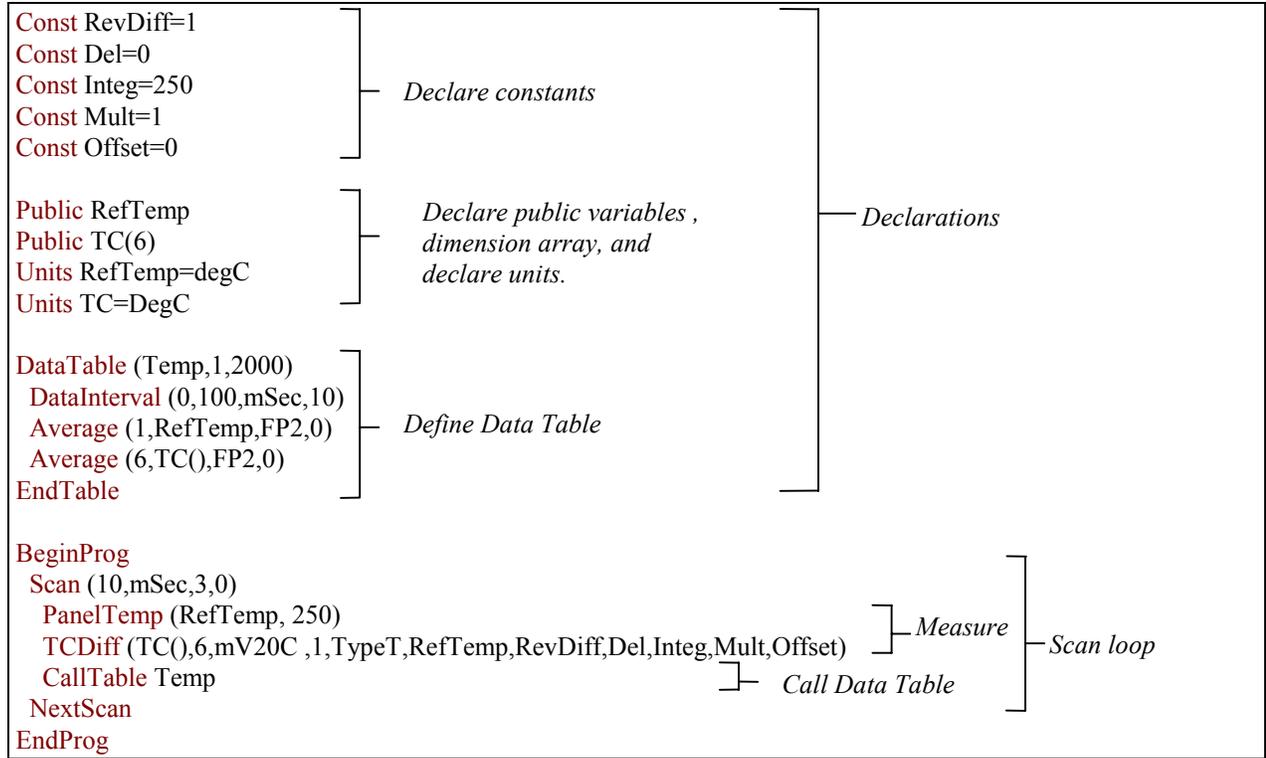
'The following program is used to measure
'4 thermocouples

'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 datalogger compiler will ignore the commented text.

### 7.4.4.6 Example Program

The following program will serve as a programming example in this section to illustrate the concepts and program structure. This is a program for a CR5000 datalogger. Note that other dataloggers may have slightly different parameters for some instructions.



### 7.4.4.7 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 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 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 size is the number of records to store in the table. You can specify a fixed number, or enter -1 to have the datalogger auto allocate the number of records. 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. (Note: The `DataInterval` instruction for the CR200 does not have 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 using the `CallTable` (*Tablename*) instruction in order for the output processing to take. 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 7.3-1), 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.

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)

#### 7.4.4.8 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 CR200 datalogger's Scan instruction has only two). The Interval is the time between scans. Units are the time units for the interval. 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 CR5000 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.

### 7.4.4.9 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 7.4-1).

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.

### 7.4.4.10 Logical Expression Evaluation

#### 7.4.4.10.1 What is True?

Several different words are 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 7.5-1. The datalogger evaluates the test or parameter as a number; 0 is false, not equal to 0 is true.

TABLE 7.5-1. 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

#### 7.4.4.10.2 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.

#### 7.4.4.10.3 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.

### 7.4.4.11 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 selected from the Monitor tab. If a Flag array is found, up to 8 flags will be displayed in the Port and Flags dialog box.

### 7.4.4.12 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 7.7-1 lists the maximum length and allowed characters for the names for Variables, Arrays, Constants, etc.

TABLE 7.7-1. 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 0-12. The name must start with a letter. CRBasic is not case sensitive.
Constant	16	
Alias	16	
Data Table Name	8	
Field name	16	

#### 7.4.4.12.1 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. 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.

#### 7.4.4.12.2 Arrays of Multipliers and 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())

Note that one exception to this is when the Multiplier or Offset points to an index into the array, then the instruction will not advance to the next Multiplier or Offset but use the same for each repetition. For instance in the above example, if Mult(2) and Offset(2) were used, the instruction would use 0.115 and 0.234 for the Multiplier and Offset, respectively, for each repetition. To force the instruction to advance through the Multiplier and Offset arrays while still specifying an index into the array, use the syntax **Mult(2)()** and **Offset(2)()**.

#### 7.4.4.13 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 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.

*TableName.EventCount* = the number of data storage events that have occurred in an event driven DataTable.

*TableName.Tablefull* = 1 to indicate a fill and stop table is full or a ring-mode table has begun overwriting its oldest data, = 0 if the data table is not full/begun overwriting oldest data.

*TableName.TableSize* = the size allocation, in number of records, of the selected DataTable.

---

**NOTE**

The values of *Tablename.output(1,1)* and *Tablename.eventend(1,1)* are only updated when the tables are called.

---

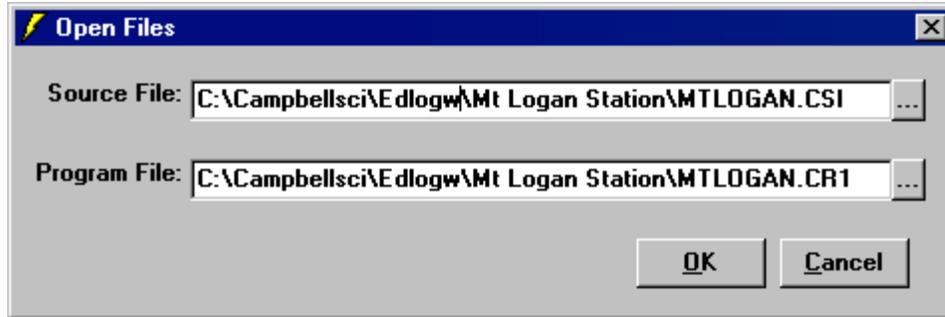
## 7.5 Transformer Utility

The Transformer application converts a datalogger program created in Edlog to a CRBasic program. With CSI's introduction of the CR1000, this program was specifically designed to help ease the transition from programming in Edlog to programming in CRBasic. Currently, conversion is supported for CR10X to CR1000 only. Additional dataloggers may be supported in the future.

### 7.5.1 Transforming a File

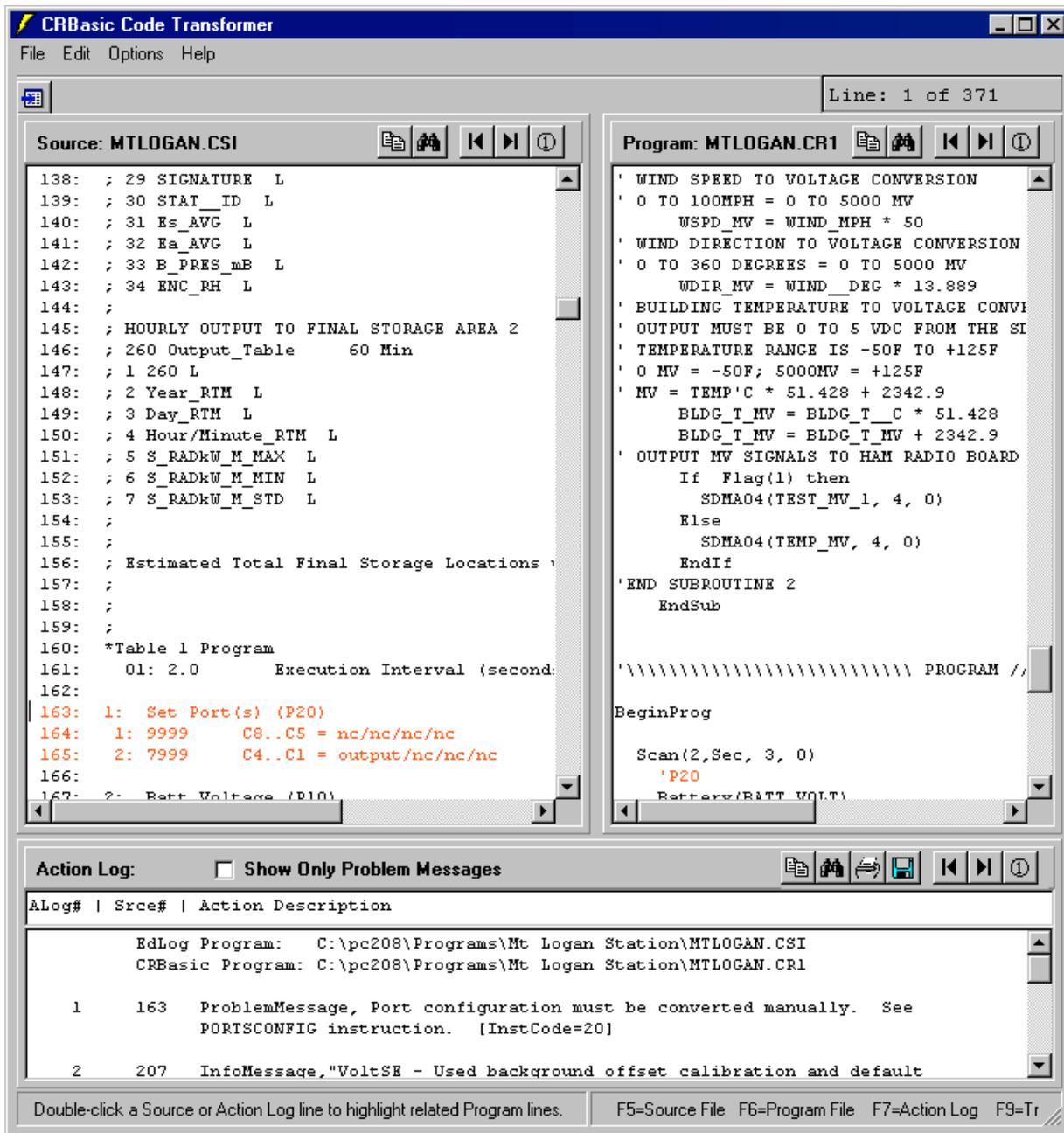
The Transformer utility is not available from the LoggerNet Toolbar. It is installed as one of the utilities with LoggerNet, and can be opened from the Window's Start Menu, Programs | LoggerNet | Utilities.

When the Transformer is first opened, a dialog box is displayed in which to enter the Source File and the Program File.



The Source File is the CSI or DLD file to be converted. The Program File is the new CR1 file that will be created. By default, the resulting file name for the CR1000 program that will be created is the name of the original program with a CR1 extension. This can be changed if desired by typing in a new path and/or file name directly, or by pressing the Browse button to the right of the Program File field.

When OK is chosen, the Edlog program file -- or Source file -- is opened up in the left window of the Transformer. The CR1000 program file is created in the right window.



Comments about the conversion are shown in the Action Log (bottom portion of the window). The Action Log should be reviewed carefully; it provides useful comments and alerts you to any problems that may exist in the converted file. To view only the messages related to problems in the field, enable the Show Only Problem Messages check box.

If a comment in the Action Log is double-clicked, the associated instructions in both the Edlog program and the CRBasic program will be highlighted. If an instruction is double-clicked in the Edlog file, the associated instruction in the CRBasic program will be highlighted. In the window above, the first comment was selected, resulting in the Edlog instruction and a comment in the CRBasic program both being highlighted in red.

The transformed file cannot be edited in the Transformer. Once transformed, it can be opened in the CRBasic Editor or saved under a new file name. To open the program in the CRBasic Editor, press the CRBasic program icon  at the top left of the window. To save the file under a different name, choose File | Program File | Save As.

If an Edlog file previously has been opened in the Transformer, when the file is opened a second time you will receive a message "This file, <filename>, already exists. If you overwrite it, the information it contains will be lost. Do you want to overwrite it?" If you choose Yes, the existing CR1 file will be overwritten. If you choose No, you will be given the opportunity to provide a new name for the file. This message can be suppressed by selecting Options | Suppress "Overwrite File" Warning from the Transformer menu. However, note that you should strongly consider keeping this message intact to avoid the possibility of overwriting a file that you transformed and then subsequently edited in the CRBasic Editor.

## 7.5.2 Controls

The following buttons are used within the Transformer to move to a different location in the file, or save or print the file.



Copies the highlighted text to the Windows clipboard. The information can then be pasted into another application.



Searches for specific text in the file.



Moves the mouse cursor to the beginning of the file.



Moves the mouse cursor to the end of the file.



Jumps to a specific line number in the file.



Prints the contents of the window (Action Log only).



Saves the contents of the window (Action Log only).



# Section 8. Working with Data Files on the PC

---

*After data has been collected from the datalogger, you need a way to analyze that data. LoggerNet provides two tools to do this.*

*View is a simple file viewer that provides a way to look at the collected data. It will open data files (\*.DAT, \*.PRN, \*.CSV) saved in a variety of formats including files from mixed-array and table-based dataloggers. View can also open other CSI file types (\*.DLD, \*.CSI, \*.PTI, \*.FSL, \*.LOG, \*.CR2, \*.CR5, \*.CR1, \*.CR9). Graphing capabilities are provided for one or two traces.*

*Split is a tool that is used to post-process collected data from either mixed-array or table-based dataloggers. Split can create reports by filtering data based on time or conditions. It can generate statistics, perform calculations, reformat files, check for data quality (limit testing), and generate tables with report and column headings. It can also handle the time synchronization necessary to merge up to eight data files.*

## 8.1 View

The View button on the LoggerNet main screen brings up the file view program. This program can be used to look at any data file collected from dataloggers, regardless of the format (comma separated, table oriented ASCII, binary, or printable).

Once a file is opened, data can be printed, graphed, or displayed in comma separated, tabular, or hexadecimal format. However, since View is primarily a file viewing utility, a file cannot be edited or saved using this program.

### 8.1.1 Opening a File

View provides two ways to open a file. The one you use depends on the type of file. Use File Open  to open a data file and File Open as Text  to open other types of files.

Files with a particular extension can be configured in Windows to be opened by View automatically when double clicked in Windows Explorer. Refer to Section 8.1.6.1 Assigning Data Files to View for more information.

#### 8.1.1.1 Opening a Data File

To open a data file, click the File Open  icon or select File | Open from the menu. Any data file type stored by LoggerNet can be viewed. The default file extension is DAT, but PRN and CSV files can also be opened in this mode. When a file is first opened, it is displayed in the Data Panel in a comma-separated format. The file can be viewed in a tabular (field formatted) format or graphed. The graphically displayed data is referred to as the Graph Panel.

---

**NOTE** View will try to assign a time/date stamp automatically when opening a data file. Therefore, it may take some time to completely open a large file.

---

When a mixed-array data file is opened in this mode View will automatically associate an FSL file with a matching name if one exists in the active directory (see Final Storage Label below). If a matching FSL file does not exist, a box will appear asking whether or not you want to select one. When an FSL file is associated with a data file, View uses information from the file to provide label names for the Data Panel column headings and Chart Panel traces.

---

**NOTE** View does not need FSL files for Table Oriented ASCII (TOA) files from table-based (CRBasic or Edlog table-data) dataloggers because column labels are included in the headers in these data files.

---

View can be run as a stand-alone program by double clicking the View.exe file in Windows Explorer.

### 8.1.1.2 Opening Other Types of Files

To open a file that is not a data file (e.g., \*.DLD, \*.CSI, \*.PTI, \*.FSL, \*.LOG, \*.CR2, \*.CR5, or \*.CR9) press the  icon or select File | Open As Text from the menu. Files opened in this mode are not processed for time stamps, and thus cannot be graphed or displayed in the tabular format.

A file opened in this mode can be viewed only in its original format or in hex mode. The Array Selection, Array Definitions, Tab options, and Graph View options do not work. This mode is most often used to open files other than data files (or to quickly open data files, but without any of View's graphing or time stamping capabilities). In Text Mode, data can be copied to the Windows clipboard and pasted into other applications.

### 8.1.2 Final Storage Label (FSL) Files

A final storage label (FSL) file is created by Edlog when a mixed-array datalogger program is compiled. It provides information about the output tables in the program and the labels associated with each output instruction. The information in this file is used by View to provide column headers in the Data Panel view and trace labels in the Graph Panel, as well as to assign date and time values. To add or change an FSL file association, press the Array Definitions icon on the right side of the Toolbar and select the desired FSL file.

After choosing a data file from File | Open, View will handle the FSL association in one of the following ways:

- 1) If an FSL file exists that has the same name as the data file, View automatically uses that FSL file.
- 2) If no FSL file exists with the same name, View will prompt you to select an FSL file. If you choose to select an FSL file, another file dialog box appears so you can select the FSL file. If you choose not to select an FSL

file, View will still open the file and attempt to calculate date/time stamps based on likely positions of date and time data elements in the file. Selecting an FSL file allows column headers and time stamps to be displayed.

- 3) If you have opened a file before, View keeps track of what option you chose for the FSL the first time it was opened, and opens it with that FSL option. It keeps this information for the last 50 files.
- 4) A different FSL file can be associated with a data file by selecting the Array Definitions button (rightmost icon on the Data Panel) and choosing a new FSL file.

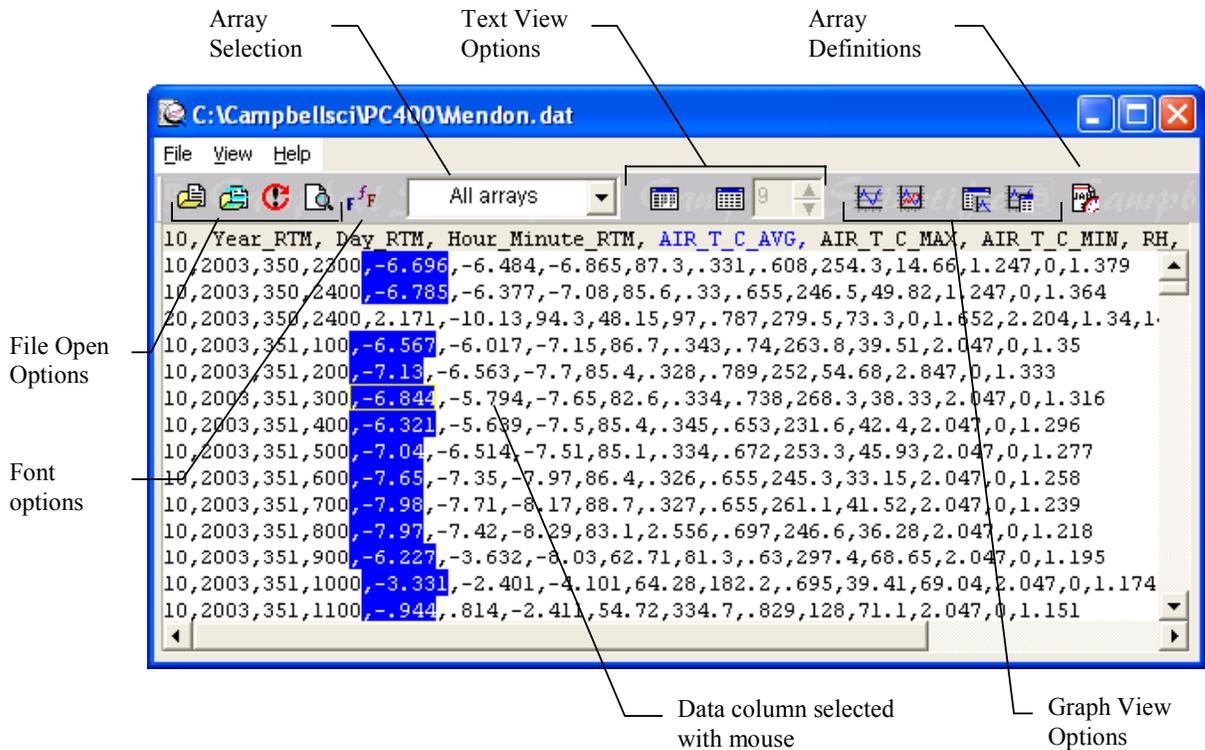
**NOTE**

When viewing numeric data in the Data Panel, if more than one array ID exists in the data file, View will display the header for the last array that has been selected with the mouse pointer.

Table Oriented ASCII (TOA) files do not use FSL files. Data label information is included directly in the data file.

### 8.1.3 Data View

The initial display for data files or other files opened by View is as normal text, typically comma-separated for data files. Data can be alternatively be viewed as tab separated values in columns or in hexadecimal format.



### 8.1.3.1 Array Selection

When a data file is first opened, all the arrays in that file will be displayed in the Data Panel. You can display, print, or graph elements from only one array by picking the array from the Array Selection list. This essentially acts as a filter so that all other arrays are hidden from view.

In the Array Selection list, there are two symbols that can precede the arrays displayed in the list. The check mark indicates the array is in the FSL file and is found in the data file being viewed. The X indicates the array is in the FSL file but is not found in the data file. If no symbol precedes the array, the array is in the data file but is not listed in the FSL file.

---

**NOTE**

Array Selection only applies to mixed-array dataloggers (CR510, CR10X, CR23X). Table-based dataloggers only store data for one table in each data file. CR5000, CR1000, CR9000, CR200, and CRxxx-TD/PB dataloggers store data in a table-based format.

---

### 8.1.3.2 Text View Options

By default, the data is displayed in ASCII format (comma separated, printable ASCII, and TOA files are displayed in the format they were saved to disk; binary files are interpreted into ASCII format). Press the Toggle Hex Mode  on the toolbar to display the text as hexadecimal characters. This button is a toggle, so if you press it again the display will return to the default view. You can also display the text as hexadecimal characters by selecting View | Hex from the menu.

Data displayed on the Data Panel can be changed to an adjustable-tabular format by pressing the Expand Tabs button,  or selecting View | Expand Tabs. The tab button is a toggle that turns the tabular display on or off. The column widths of the adjustable-tabular display can be changed by typing a number directly into the combo box, or selecting the up or down arrow to change the value. The range for the tab width is 9 to 27 characters.

---

**NOTE**

The tabular format is unavailable for files opened using Open As Text.

---

### 8.1.3.3 Changing the Font

The font used for the text display can be changed with the font selection dialog. Click the Font button or select View | Font from the menu to change the font used for the printer and data display.

Only fixed width fonts can be selected to allow the data columns to line up. Typical fonts include Courier, Courier New, Fixedsys, Lucida Console, MS LineDraw, and Terminal. Normal font options such as color, bold, underline and italic are also available.

## 8.1.4 Graph Panel

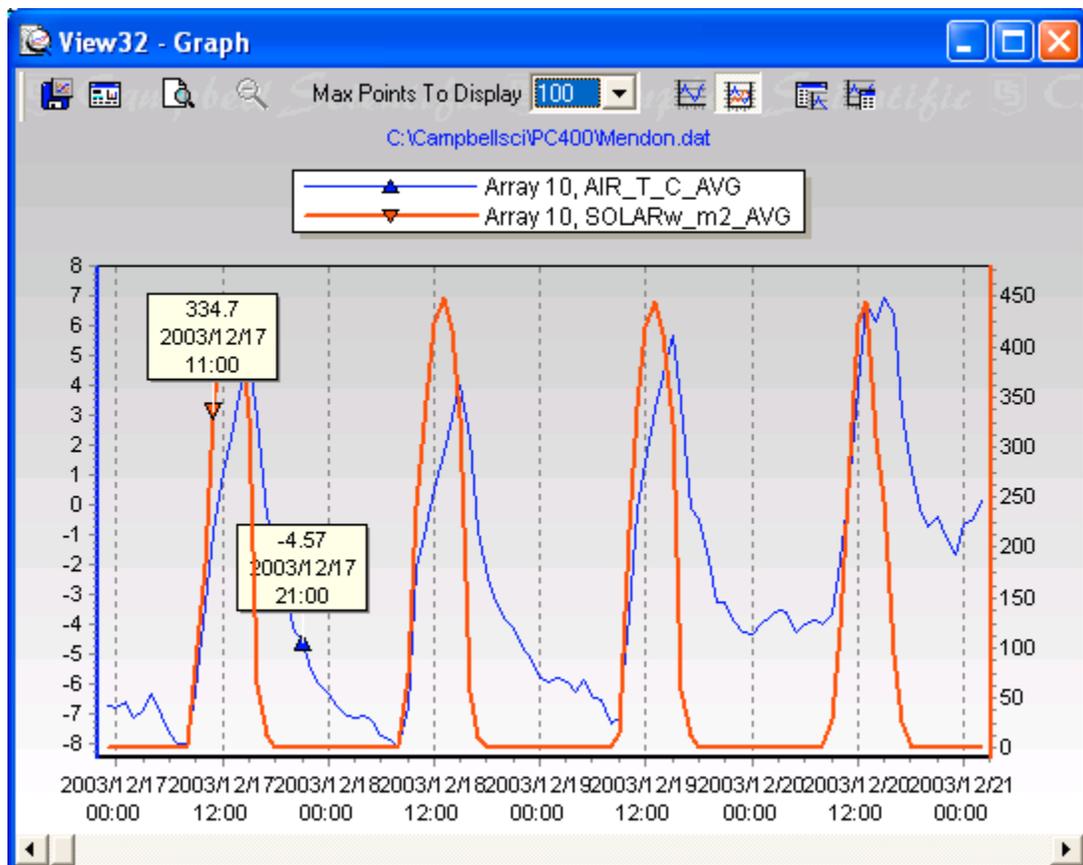
The Graph Panel is used to display a line graph of one or two data values. The values to graph are selected in the data view by clicking one column and then, if desired, holding the <ctrl> key down while clicking the second. The selected columns will be highlighted in blue and red.

### NOTE

The Graph Panel is unavailable for files opened using Open As Text.

You can choose whether to display two traces on the same or different vertical axes with buttons on the Toolbar. Click the single axis graph button and the graph will be displayed with only one vertical axis. Click the two axis graph button and the graph will be displayed with two axes, one on each side.

The scale for the graph axes defaults to automatic. The upper and lower limits will depend on the range of values in the selected data. You can set the graph's Y scale to a fixed value using the Customize Chart option (second from left-most icon).



### 8.1.4.1 Graph Options

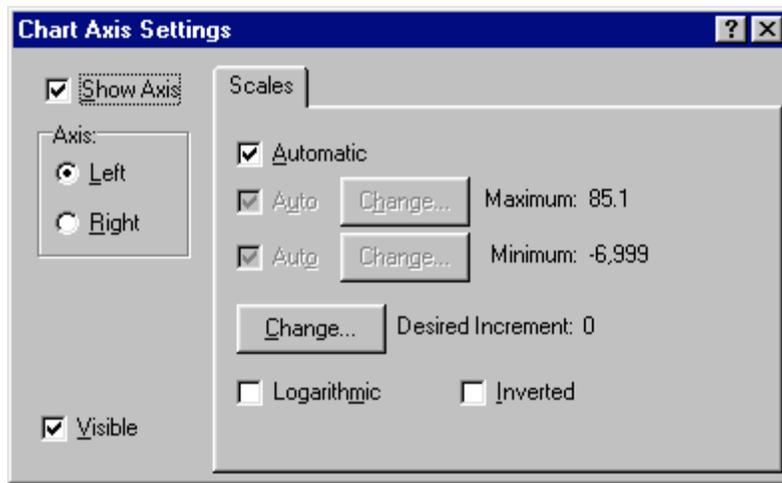
The following options are available to set up how the Graph displays the line graph. These options are available by clicking the buttons at the top of the graph. Let the mouse cursor hover over a button for a few seconds to see a pop-up hint describing the button's function.



Saves the graph to a file in BMP, WMF, or EMF format. The file type is selected using the Save as Type option at the bottom of the Save As dialog.



Brings up the Customize Chart dialog that allows you to set the properties of the Y axes. Choose whether to set the Left or Right axis on the left side of the window and then clear the Automatic checkbox.



Displays the Print Preview screen. Paper orientation and margins can be adjusted and the graph can be printed.



Returns a zoomed graph to normal view. The Zoom feature allows you to zoom in on a particular area of a graph by holding the left mouse button and dragging the mouse cursor from left to right over the area to be enlarged. To return to normal view, click the Zoom Cancel button or hold the mouse button and drag the cursor from right to left.



Allows you to set the number of points that are displayed across the width of the graph. Select the arrow to the right of this field to display a drop-down list box to select the number of points.



Graphs two traces on the same Y-axis.



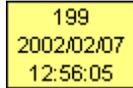
Graphs one trace on the left axis and the other on the right.



Forces the graph window to be on top. This button is toggled on or off to keep the graph in front of other windows.



Forces the data view to be on top. This button is toggled on or off to keep the data in front of other windows.



Data Labels Feature. Data Labels allow you to get the exact numeric value and timestamp for a data point on the graph. Click a data point to display a data label for that point. One data label is displayed for each trace on the graph.

You can use the scroll bar at the bottom of the graph to scroll forward or backward in time.

## 8.1.5 Printing Options

All printing is done through the Print Preview screen. Bring up the screen by clicking the Print Preview button. Print Preview for the Data Panel will display the pages of text as they will be printed. You can view any of the pages and select the pages you want to print.

### 8.1.5.1 Printing Text

To print numerical data, press the Print Preview button or select File | Print Preview from the menu. From Print Preview you can browse among the pages that will be printed and change the paper orientation if desired. You can zoom in on a particular area of the previewed page by left clicking the page. To return to normal view, right click the page, or choose the Page Width or the Full Page icon. Simply press the print button  on the toolbar to print one or more pages. See the online help for details of the Print Preview options.

### 8.1.5.2 Printing Graphs

With the Graph Panel window opened, click the Print Preview to preview the printed page, and then select the Print button to print the graph. You can also right click the graph to bring up a menu from which you can select Print Preview. The preview screen allows you to set options for the margin on the page and whether to use portrait or landscape paper orientation.

## 8.1.6 Advanced Topics

### 8.1.6.1 Assigning Data Files to View

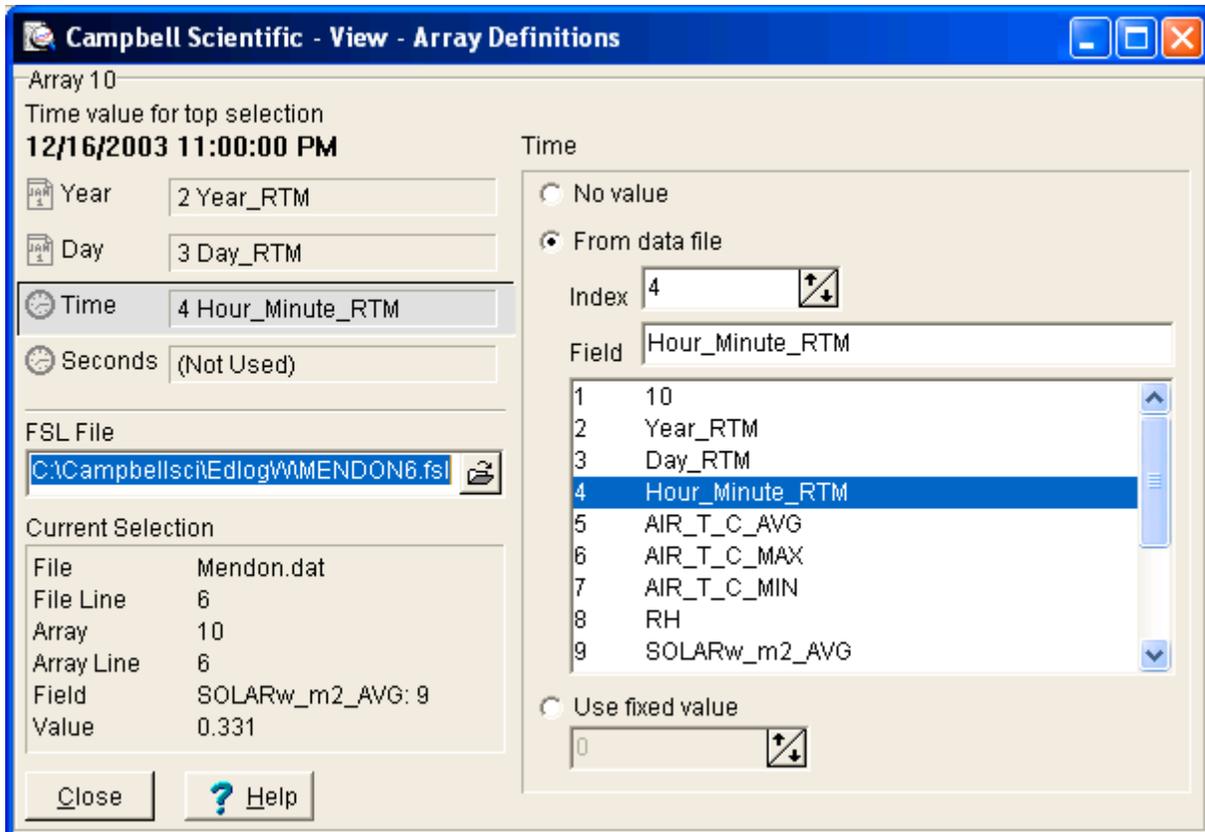
Windows will let you assign the program with which a particular file type will be opened based on that file's extension. When a file with an assigned extension is double clicked, it will be opened with the associated program. You may want to associate \*.DAT files with the View program for quick opening of data files. This association can be made by selecting Tools | Folder Options | File Types from the Windows Explorer menu.

**NOTE**

Only one association can be made for each file type. If the \*.DAT file is assigned to another program, you must remove this association before the new association can be made.

### 8.1.6.2 Array Definitions (Mixed-array dataloggers only)

The Array Definitions button  on the toolbar brings up the Array Definitions window. You can use this window to manually assign correct time and date stamps to a mixed-array data file, or to change or add an FSL file association.



When you bring up the Array Definition screen, View will attempt to automatically assign elements from the selected array to the date and time values if they exist in the data file.

If these elements are not assigned automatically, you can assign values manually. Click anywhere in the Year, Day, Time, or Seconds box, and in the second column, you can choose to assign no element, an element from the data file, or a fixed value to any of the date and time values.

When the Array Definitions window is opened, if one of the top lines of this window reads "No array selected", click a text element in your data file for the array you want to assign values. View will automatically assign elements from the array to the year, date, and time values if your file's date and time values are at the beginning of the array and are in descending order (i.e., day is before time) and they are not separated by any non-date or non-time elements. The

assigned date and time values will most likely be correct for files of more than a few hundred lines.

In this window, you can assign a \*.FSL file (final storage label file) to the data file if one was not assigned when the data file was first opened. Enter a file name into the FSL file name field to make this assignment. View will then use information from the \*.FSL file to provide label names for the data window column headings and graph traces.

---

**NOTE**

This option is unavailable for files opened using Open As Text and for TOA Files. (TOA files do not use FSL files. Column header information is included directly in the data file.)

---

## 8.2 Split

### 8.2.1 Functional Overview

Split is a tool to analyze data collected from Campbell Scientific dataloggers. Its name comes from its function of splitting out specific data from a larger data file. Originally, Split could only process mixed-array files, and it was used to “split” the different arrays – typically different time intervals – of a file into separate files (e.g., for hourly versus daily data).

In addition to splitting out mixed-array data, Split can filter output data based on time or conditions, calculate statistics and new values, reformat files, or check data quality (limit testing). Split can generate tables with report and column headings, as well as time synchronize and merge up to eight data files.

Input Files (maximum of eight) are read by Split, specific operations are performed on the data, and the results are output to a new Output File or a printer. Split creates a parameter file (*filename.PAR*) that saves all of your settings such as which data files are read, what operations are performed on the data set, and where the final results will be saved. The parameter file may be saved and used again.

Input Files must be formatted in Printable ASCII, Comma Separated ASCII, Field Formatted ASCII, Final Storage (Binary) Format, Table Oriented ASCII (TOACII or TOA5), Table Oriented Binary (TOB), or Raw A/D data (such as the results of a burst measurement).

Split can be used to convert a file of one format to a different format. For example, a Table Oriented ASCII file can be converted to the Comma Separated ASCII format used in mixed-array datalogger data files. This is useful to convert table-based data files to work with applications that were written to work with mixed-array files.

Output files generated by Split can be Field Formatted (default), Comma Separated ASCII, or Printable ASCII. Split can also create reports in ASCII as well as html formats, or send them directly to a printer.

Split lends itself to experimentation. The processed data are displayed on the screen, giving immediate feedback as to the effect of changes or new entries to the parameter file. Split does not modify the original Input File.

## 8.2.2 Getting Started

The most common use of Split is to separate array data collected on a particular interval from a data file containing data output at several different intervals.

In the following example, hourly data are split from a data set that contains 15 minute, hourly and daily data. The data was collected from BirchCreek, a CR10X datalogger. The CR10X was loaded with a program created by Edlog named Birch.dld.

The 15 minute data, array 99, the hourly data, array 60, and the daily data, array 24, are intermixed in the data file (Figure 8.2-1).

99	Year_RT	Day_RTM	Hour_Mi	BattVol	BattVol	BattVol	Encl_RH	Encl_RH	LogrTem	LogrTe
99	2002	35	2130	14.23	14.00	2118	22.07	2128	-15.45	
99	2002	35	2145	14.24	14.03	2133	22.31	2143	-15.92	
60	2002	35	2200	14.19	13.96	2113	22.44	2155	-15.43	
99	2002	35	2200	14.19	14.01	2158	22.44	2155	-16.69	
99	2002	35	2215	14.23	14.01	2200	22.38	2214	-16.93	
99	2002	35	2230	14.25	14.04	2218	22.48	2229	-16.96	
99	2002	35	2245	14.26	14.05	2233	22.58	2242	-17.36	
60	2002	35	2300	14.26	14.01	2200	22.62	2252	-16.93	
99	2002	35	2300	14.26	14.05	2256	22.62	2252	-17.69	
99	2002	35	2315	14.24	14.05	2300	22.55	2300	-17.42	
99	2002	35	2330	14.18	14.01	2329	22.51	2329	-17.31	
99	2002	35	2345	14.24	14.00	2336	22.51	2331	-17.14	
60	2002	35	2400	14.26	14.00	2336	22.55	2300	-17.14	
24	2002	35	2400	14.26	13.70	1244	23.67	730	-4.479	
99	2002	35	2400	14.26	14.05	2348	22.51	2355	-17.20	
99	2002	36	15	14.24	14.05	13	22.55	12	-17.50	
99	2002	36	30	14.26	14.05	18	22.65	29	-17.56	

FIGURE 8.2-1. Data File

When Edlog compiled Birch.dld, it also created the Final Storage Label file, Birch.fsl that lists the final storage locations for each data element (Figure 8.2-2).

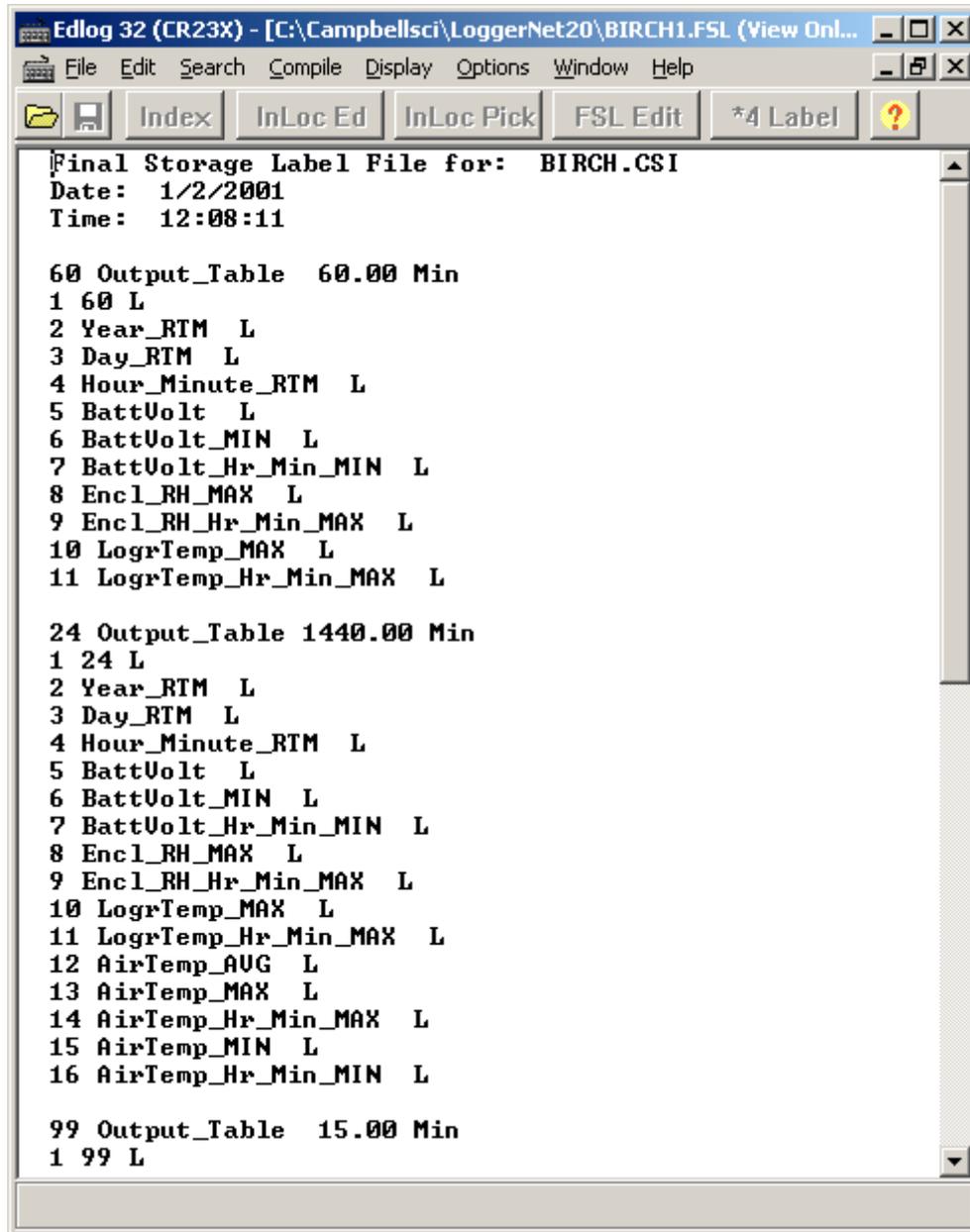


FIGURE 8.2-2. FSL file

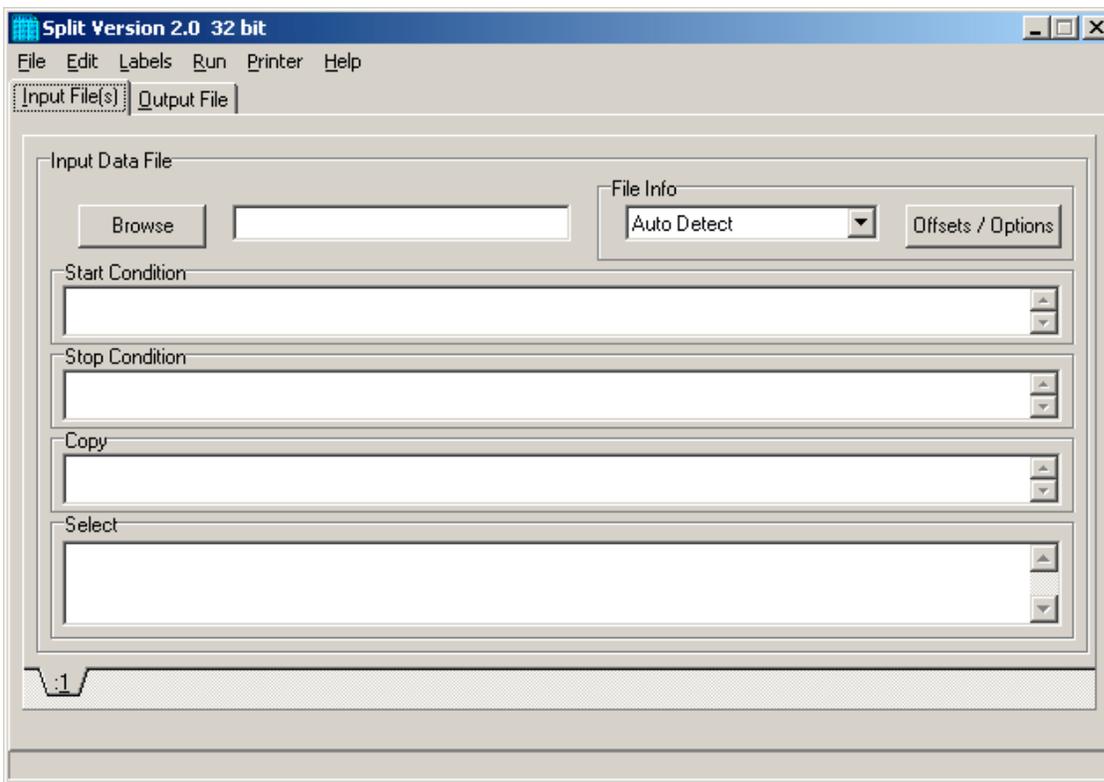


FIGURE 8.2-3. Input File Template

When you start Split a blank template similar to the one above is shown. This template is used to enter the parameters that will define what data from the input file to include in the output file. The parameters entered on this template can be saved as a parameter file (\*.PAR) and reused for other data.

On the **INPUT FILE** tab you only need to specify the input file name, copy condition, and the data to select. Split allows start and stop conditions to be specified but if they are left blank, the entire file will be read.

The name of the Input Data File can be typed in or the **Browse** button can be used to select from available files. In this example BirchCreek.dat will be selected as the input data file.

Selecting the data to copy is simplified by the use of the Birch.fsl file. From the toolbar menu, click Labels | Use Data Labels. From the Data File Labels pop-up, Select File is used to find Birch.fsl. When one of the Output Arrays is highlighted, the Field Names of the data in that array are displayed (Figure 8.2-4).

#### NOTE

In this example, a mixed array data file is processed and the Use Data Labels feature uses an FSL file. When processing a table-based datalogger file, change the file type to “Table-based data file to use for labels” and select the table-based DAT file. Split will use the header information from this file for its labels.

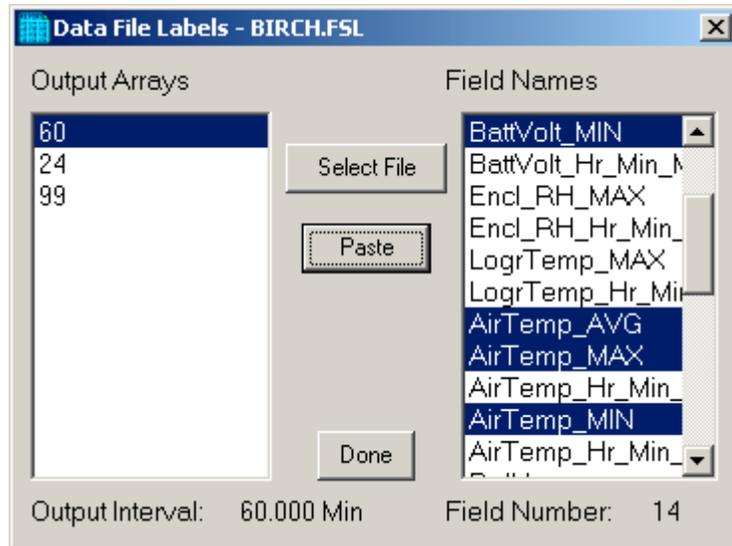


FIGURE 8.2-4. DATA FILE LABELS Screen

In this example we want the hourly data (note the Output Interval at the bottom of the Data File Label window), so click array 60. To paste the desired values from this array into the Select box, select the field names while holding down the <ctrl> key. All of the values could be selected by clicking the first one and holding the mouse button down, and dragging to the end. Once the values you want have been selected click Paste (Figure 8.2-5).

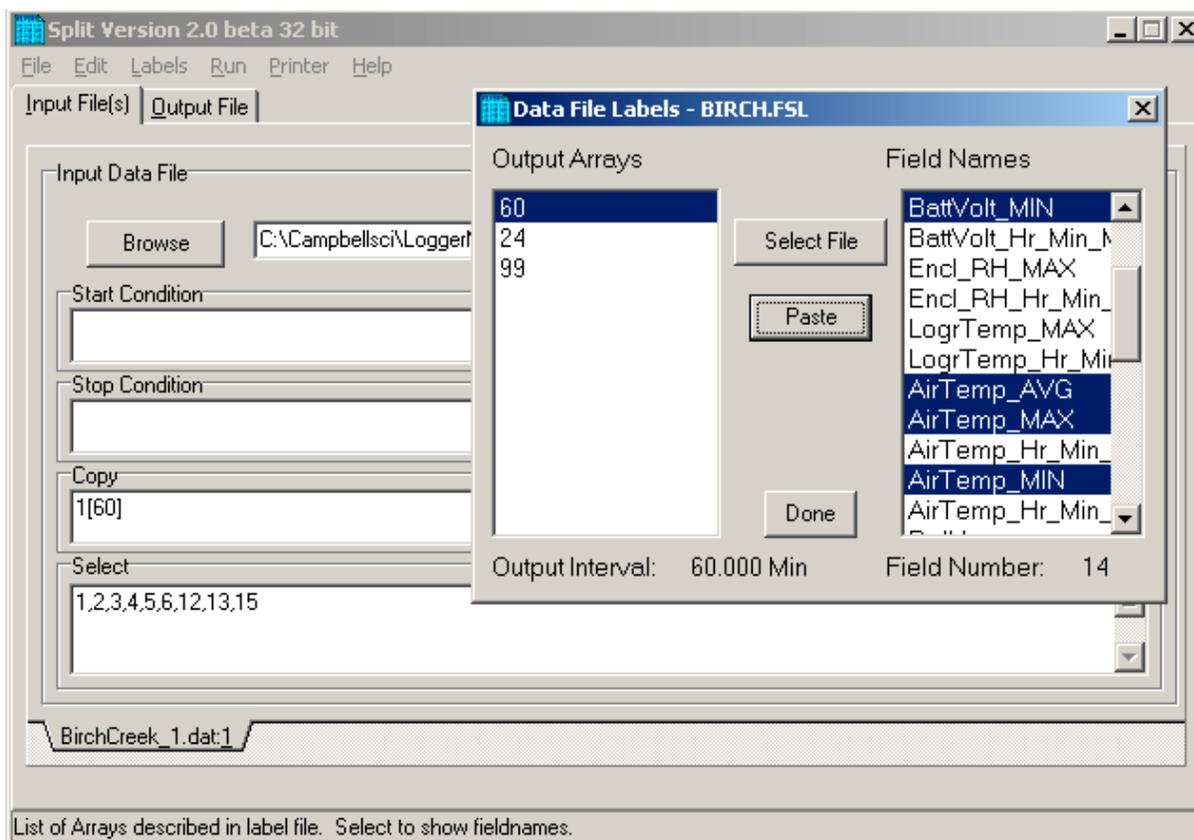


FIGURE 8.2-5. Pasting Values into Split

Note that the cursor in the **INPUT FILE(S)** screen must be in valid paste area (Copy or Select). If the cursor is in the File name box or in Start/Stop condition, you will get the error message “Cannot Paste There”.

The Paste operation copied the numbers of each of the fields into the Select box. Notice also that it pasted the Array ID into the copy condition: 1[60] tells Split that in order to copy a line of data, the first value in that line must be 60. Split uses the Array ID to discriminate between the hourly and daily data.

Now specify the Output File name. (Without one specified, Split will run and display results but no output file will be created.) Click the **OUTPUT FILE** tab. Type in “hourly” for the name of the output file. By default, Split will use the file extension “PRN”, creating the output file: hourly.prn. Depending upon the option chosen in the "If File Exists then" list box, an existing PRN file may be overwritten, appended to, or saved under a new name.

The Labels option from the toolbar can also assist in labeling the output values. Once again, choose **LABELS | USE FINAL STORAGE LABELS** and select array 60 and all the field names. This time move the cursor to Line 1 of the first column of labels on the **OUTPUT FILE** tab and press Paste (Figure 8.2-6). The labels from the final storage file will be pasted into each of the columns. Split will automatically break a label name into multiple rows at the “\_” in a label name.

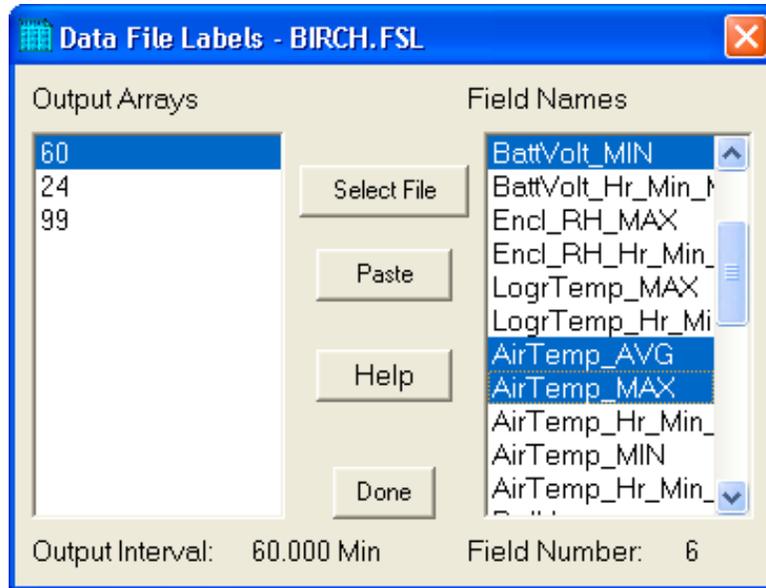


FIGURE 8.2-6. Pasting Labels Into Split

Maximum column heading width is one less than the number entered in the Default Column Width field. However, entering a number in the Width row for the column will set the column width for an individual column. Any FSL labels that are too long for Split column headings will be shown in red. They should be edited before running Split. To edit one of the labels, press the <Enter> key or use a mouse to copy, cut, and paste. A Report Heading can also be entered (Figure 8.2-7) using the same editing technique.

Report and Column Headings						
Report Heading	Hourly Data					
Column#	4	5	6	7	8	9
Element/Field#	4	5	6	12	13	
Filename	BirchCreek_1.d	BirchCreek_1.d	BirchCreek_1.d	BirchCreek_1.d	BirchCreek_1.d	
Line 1	Hour_Minute	Battery	BattVolt	AirTemp	AirTemp	AirTemp_MIN
Line 2		Volts	Min	Average	Max	
Line 3				C	C	
Decimal						
Width	12		12			12

FIGURE 8.2-7. Edited Headings

For table based data files the timestamp is normally the first column and is a quoted text string ("2002-02-26 10:30:00"). To display these timestamps in the output you will need to change the column width for the first column to at least 24. If the column width is too small to accommodate the value output, the

string will be highlighted in red and preceded by an asterisk, with the words “Bad Data” in the lower right corner when the file is processed.

To run Split, select RUN | GO. The hourly data will be split out and stored in hourly.prn. The results are displayed on the screen as shown in Figure 8.2-8.

Close the Run window. If you wish to save this parameter file for future reports, choose FILE | SAVE. The file will be saved with a .PAR extension.

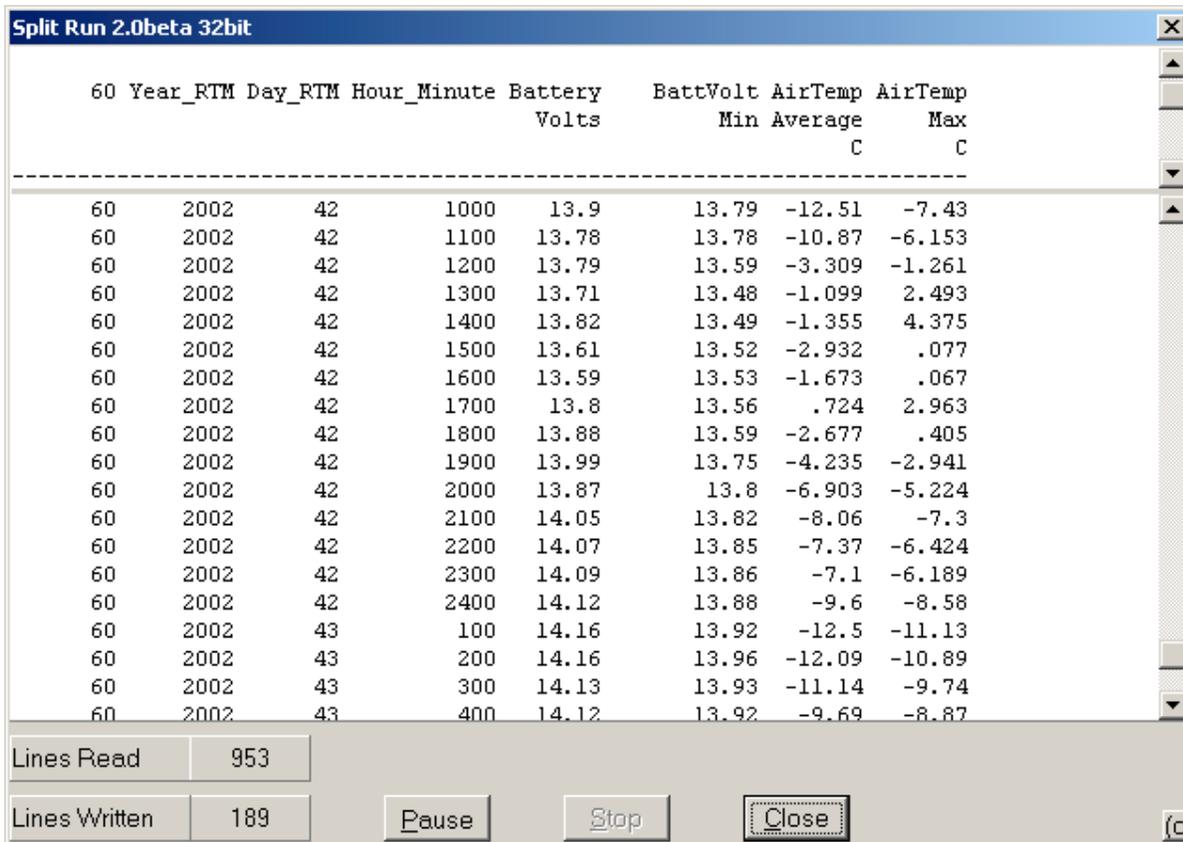


FIGURE 8.2-8. Split Run Showing Hourly Data

## 8.2.3 Split Parameter File Entries

### 8.2.3.1 Input Files

The name of the Input File is entered in the space to the right of the **Browse** button. The default directory is the working directory for Split (if the default installation directories were chosen, this will be c:\campbellsci\splitw). If the input file is not in the default directory, use the **Browse** button to find the input file.

In LoggerNet, mixed array datalogger files are stored in a simple comma separated ASCII format; table-based datalogger files are stored in TOA5 (a comma separated format with headers). Split can process Input files from other

software, but they must be formatted in Comma Separated ASCII, Final Storage (Binary) Format, Field Formatted ASCII (Split default output format), Printable ASCII, Table Oriented ASCII (TOAC11 or TOA5) or Raw A/D data (refer to special Burst Mode instruction in your Campbell Scientific datalogger manual).

Files stored in Table Oriented Binary (TOB) format are converted to Table Oriented ASCII files when Split uses them. The converter comes up automatically when you run Split to create the output file. You cannot use the Data Label browser to select the columns of data from a binary file. If you want to use the Data Label browser you can open the file first using View, which converts the binary file to ASCII and saves it under a new name, prior to processing it with Split.

Split's default output file, a field-separated ASCII format with a \*.PRN file extension, can be processed a second time if desired.

Table 8.2-1 provides an example of Comma Separated, Field Formatted, Printable ASCII, and Table Oriented ASCII input file types. The data in the various formats are identical. Each line of data represents an "Output Array", starting with an Output Array ID (in this case 115). Each data point in the Output Array is referred to as an "element". The element number is given in the Printable ASCII format, and implied in the other formats. Data presented in Table 8.2-1 is used for example purposes in the following sections.

<b>TABLE 8.2-1. Comma Separated, Field Formatted, Printable ASCII, and Table Oriented ASCII Input File Format Types</b>						
<b>COMMA SEPARATED</b>						
115,189,1200,89.6,55.3,25.36,270						
115,189,1300,91.3,61.5,27.25,255.4						
115,189,1400,92.7,67.7,15.15,220.1						
115,189,1500,94.1,69,20.35,260.6						
<b>FIELD FORMATTED</b>						
115	189	1200	89.6	55.3	25.36	270
115	189	1300	91.3	61.5	27.25	255.4
115	189	1400	92.7	67.7	15.15	220.1
115	189	1500	94.1	69	20.35	260.6
<b>PRINTABLE ASCII</b>						
01+0115	02+0189	03+1200	04+089.6	05+055.3	06+25.36	07+270.0
01+0115	02+0189	03+1300	04+091.3	05+061.5	06+27.25	07+255.4
01+0115	02+0189	03+1400	04+092.7	05+067.7	06+15.15	07+220.1
01+0115	02+0189	03+1500	04+094.1	05+069.0	06+20.35	07+260.6

Element 1	=	Output Array ID# (115)
Element 2	=	Julian day (189)
Element 3	=	hour, minute
Element 4	=	average temperature in deg. F
Element 5	=	average soil temperature in deg. F
Element 6	=	average wind speed in mph
Element 7	=	wind direction in degrees
TABLE ORIENTED ASCII		
"TOASCII","CR10T","15Minute"		
"TMSTAMP","RECNBR","TCTempF_MAX","BattVolt_MIN"		
"2002-02-26 10:30:00",0,73.97,13.99		
"2002-02-26 10:45:00",1,74.03,13.98		
"2002-02-26 11:00:00",2,74.53,13.98		
"2002-02-26 11:15:00",3,74.82,13.98		
"2002-02-26 11:30:00",4,75.23,13.98		
Element 1	=	Timestamp
Element 2	=	Record Number
Element 3	=	temperature in degrees F
Element 4	=	minimum battery voltage

A maximum of eight input files may be processed by Split at one time. Additional input files are added using the EDIT | ADD DATA FILE menu option. Split looks for a file extension of .DAT if no extension is specified. If the Input File does not exist, an error message is displayed when RUN | GO is selected from the menu options.

For instance, to process two files named TEST.DAT and TEST\_1.DAT the user would select TEST.DAT and TEST\_1.DAT as Input Files. Two blank input file templates will be generated. To change from one template to the other, click the appropriate tab on the bottom of the screen. Both templates must be completed before Split will process the data. To merge different output arrays from the same input file into one array, open the data file once for each different array.

### 8.2.3.1.1 File Info

In most instances, Split automatically recognizes the type of data file it is reading when using Auto Select in the File Info field. However, there are two exceptions for which you should choose the appropriate option manually:

- **Reading Raw A/D Data from Burst Measurements**

To read this type of data and convert it to ASCII, select Burst in the File Info box. Once Burst is selected, the Number of Values in Each Burst window in the Offset Menu will become accessible. Enter the number of elements in each Burst. This number does not include the array ID number or calibration data.

- **Reading Data in Final Storage (Binary) Format**

If the data is in binary format and Start and Stop Offsets are used, Final Storage (Binary) Format must be selected in the File Info field. This tells

Split that the file must be decoded as Final Storage before counting the bytes. If Offsets are not used, Auto Detect may be chosen and the file will be processed correctly.

### 8.2.3.1.2 File Offset/Options

#### **Last Count**

Each time Split runs a parameter file, it keeps track of the number of bytes it read from the input file and saves this information in the parameter file. Split can then start where it last left off. This is done by clicking the **Offsets** button and selecting the Last Count option (see Figure 8.2-9). This feature may be used to process only the new data from a file in which new data are being appended periodically to the data file.

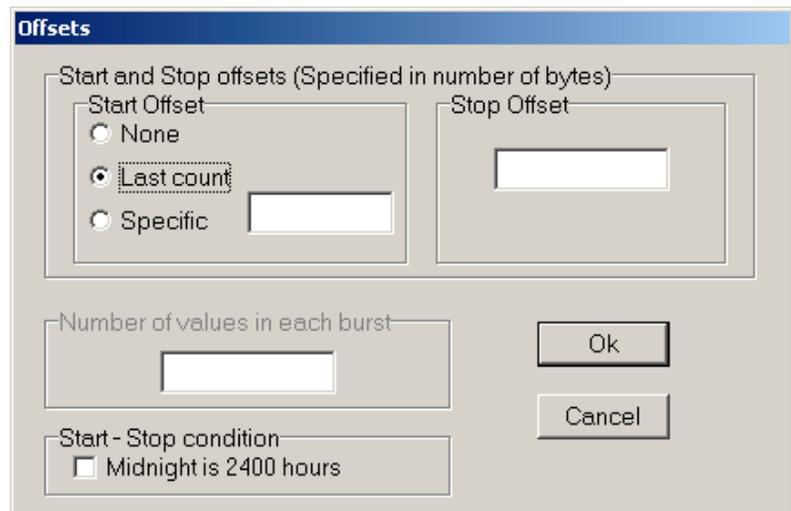


FIGURE 8.2-9. Offsets options

#### **CAUTION**

When using the Last Count option, if the Start and Stop Conditions are specified, they must exist in the newly appended data or Split will never begin execution.

Because Last Count keeps track of the number of bytes in the file, if you delete data from the beginning of a file, Last Count will not work properly.

#### **Specific**

By selecting the Specific option and entering a number, Split will "seek" that position in the file. This option saves time by starting (or stopping) part way through a large data file. The number specifies the number of bytes into the file to seek before processing data. A positive or negative number can be entered. If the number is positive, Split will start reading from the beginning of a file; if the number is negative, Split will start reading from the end of a file. All characters, including spaces, carriage returns, and line feeds, are counted. A stop offset can also be specified.

In Figure 8.2-10, Split will skip the first 256 bytes of data before it begins processing the data in Input File. In Figure 8.2-11, Split will skip the first 256 bytes of data before beginning and stop execution on byte 1024.

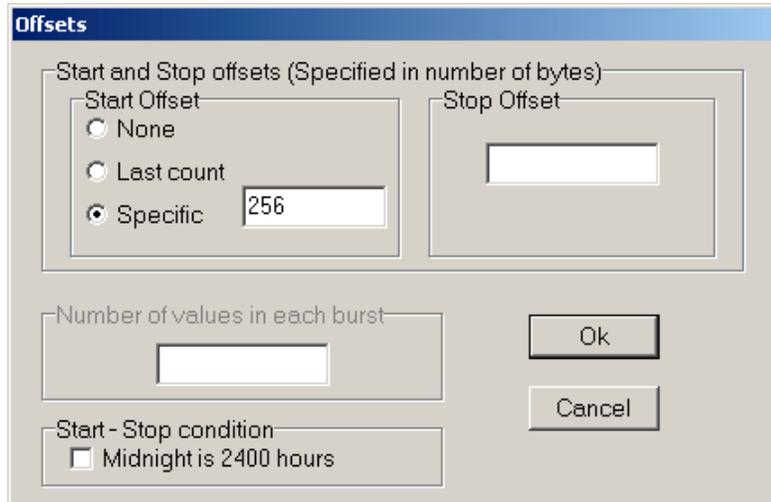


FIGURE 8.2-10. Example of a specific byte count

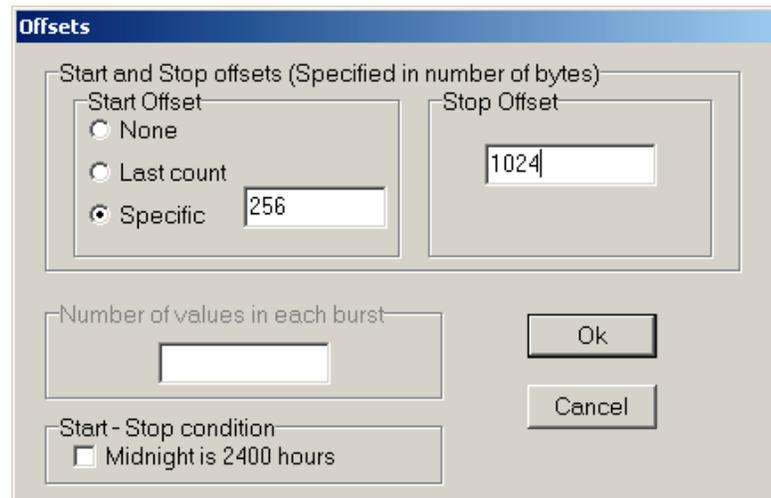


FIGURE 8.2-11. Example of a stop offset

### **Number of Values in Each Burst**

When processing a burst data file, enter the total number of values recorded for each Burst (this is the number of burst scans multiplied by the number of channels per scan). This number does not include the array ID or calibration data.

To break the results into a column for each channel, enter the number of channels for the Break Arrays value (Output File Tab, Other button).

**Midnight is 2400 hours**

When programming mixed-array dataloggers, the Real Time instruction (P77) has two different options for the midnight time stamp: midnight = 2400 of the day just ending or midnight = 0000 of the day just beginning.

When processing mixed-array data files using time synchronization, select this check box if the time stamp is midnight at 2400 of the day just ending. This will ensure that Split processes the data file correctly.

**8.2.3.1.3 Start Condition**

A starting point may be specified to begin processing data. If the Start Condition field is left blank, Split will start processing data at the beginning of the data file. The starting point can be any element within the array or a combination of elements within an array.

The syntax can be expressed as:

$$e_i[val_i]$$

where  $e_i$  = the position number of the element within the array

$val_i$  = the value of that element.

For example, the data in Table 8.3-1 contains seven elements per Output Array, representing hourly data. Assume that this data file contains one month of hourly data. To start processing data at 1500 hours on the first day, the Start Condition is expressed as 3[1500], where 3 means the third element within the array and 1500 is the value of that third element.

The element must match this start value exactly to trigger the start condition. However, when starting based on time, you can enable the "Start-Stop On/After Time" function to trigger the start of processing when the exact time is found or at the first instance of data after that time has occurred. This option is found on the Output tab, Other button.

**NOTE**

Table data files contain the time and date as a single quoted string at the beginning of each data record. Split handles the dates as long as you include a colon separator as a placeholder for each of the fields in the timestamp. 1[Year]:1[Day of Year]:1[Time of Day]:1[Seconds]

See the examples below:

:1[60]:: Day of Year 60

1[2002]:1[60]:1[1250]: Year 2002, Day of Year 60, Time of Day 12:50

::1[1445]:1[30] Time of Day 14:45, Seconds 30

Logical “and” and “or” statements can be used when specifying the Start Condition. A logical “and” statement means that all conditions must be true for the statement to be true. Up to three conditions can be connected with “and” statements. If too many “and” statements are used, an error message will be displayed when you run Split.

The logical “or” statement means that if *any* of the conditions are true, then the statement is true. Split allows up to six conditions to be connected with “or” statements. Additionally, each “or” statement can contain up to three “and” conditions. As with the “and” statements, if the maximum number of valid statements is exceeded, an error message will be displayed.

These rules for logical statements also apply to the Stop and Copy Conditions.

An example of a simple logical “and” statement follows:

2[189]and3[1200]

Element two (the Julian day) must equal 189, and element three (the time in hours/minutes) must equal 1200.

If the following “and” statement was used:

2[189]and3[1200]and4[92]and5[67]

an error would be returned because the maximum number of allowable “and” statements has been exceeded.

A range can be specified for val<sub>i</sub> by putting “..” between the lower and upper limit. For example:

2[189]and7[200..275]

In this example two conditions must be satisfied to start processing data. First, the day of year must be 189, and second, element 7 must be between 200 to 275 degrees, inclusive.

#### 8.2.3.1.3.1 Starting Relative to PC Time

Split has the ability to start relative to the current PC TIME (computer time). This feature allows a .PAR file to be run on new data files without changing the Start Conditions, provided the Input Data File is collected at a fixed interval and Split is run at a fixed interval. For example, the same PAR file could be run every day to display the last 48 hours of data without changing the start conditions. For example, using a table based data file:

Start Condition = 1:1[-1]:1[1200]:1:

In this instance, Split will begin processing data when the date for both files is one less than the current date (1:1[-1]:1[1200]:1:)and the time is 1200 (1:1[-1]:1[1200]:1:).

As an expanded example, assume that LoggerNet is used to append data to an archive file. SplitR is executed using a desktop shortcut. In this case the frequency of data collection and data reduction is the same. Time values in the data file (day, hr:mn, sec.) are different each time the data are collected, but by telling Split where to Start reading relative to the PC clock, the Start Conditions do not need to be changed. To accommodate variations in the data collection and reduction frequencies, an interval in minutes or seconds may be specified as shown in the examples below.

**2[-0]:3[-60,5]** tells Split to start at a timestamp in the data that is between 55 and 65 minutes prior to the current PC time (the closest 5 minute interval of the current day that is less than the PC time minus 60 minutes). If you are processing data stored at the top of the hour and the PC time is 1404, Split calculates 1304 and looks for hour 1300 to start reading.

**2[-3]:3[-120,60]** tells Split to find the closest 60 minute interval that is less than the PC time minus 3 days and 2 hours. If the PC time is the day of year 159, hour 0017, Split will start reading on data output at 2200 hours on day 155.

**2[-3]:3[-120]:4[20,5]** tells Split to find the closest 5 second interval that is less than the PC time minus 3 days, 2 hours and 20 seconds. If the PC time is 27 seconds after noon on day 30, Split will begin reading on data output at 1000 hours and 05 seconds on day 27.

Split can also begin processing a file on a particular month and day. Use the syntax :E[Month%Day]::, where E is the element that contains the Julian Day, and Month and Day are either constants or a value related to PC time. For example:

**:2[-1%1]::** tells Split to begin processing on the first day of the previous month.

**:2[-0%15]::** tells Split to begin processing on the fifteenth day of the current month.

**:2[5%1]::** tells Split to begin processing on May 1.

This function can be used in both the Start and Stop conditions. It provides a simple way to create a monthly report. For additional information, refer also to Section 8.2.3.1.15.2 -- Using Time Synchronization While Starting Relative to PC Time

---

### CAUTION

Split will not start reading if the exact specified starting time cannot be found, unless you enable the "Start-Stop On/After Time" feature. The interval (5 minutes, 60 minutes, and 5 seconds in the examples above) must be evenly divisible into 60 minutes.

---

**NOTE**

- If the start time is a certain number of days prior to the PC time, the file will be processed beginning at midnight of the day specified.
- To specify a start time in minutes from the current PC time, you must also specify a day parameter of [-0]. Otherwise, processing will begin at the first instance in the data file that the minutes parameter equals the current minutes.

#### **8.2.3.1.4 Stop Condition**

The Stop Condition specifies when to stop processing data. This feature allows segments of data to be removed from large data files. For instance, if a data file contains one month of data and just one day is desired, the start and stop values allow the user to get just that day's data.

The Stop Condition is expressed with the same syntax as the Start Condition. If the Stop Condition parameter is left blank, Split will execute until the end of the file. As with the Start Condition, logical "and" and "or" statements can be used when specifying the Stop Condition (Section 8.2.3.1.3), as well as stopping based on PC time.

The array or record containing the Stop Condition is not included in the output file. If the stop value is not found, Split will display a dialog box that gives the option to select a new file and continue processing the data. This feature is useful when data are contained in more than one data file.

The "Start-Stop On/After Time" function can be used with a Stop Condition. This will stop processing of the file when the exact time is found or at the first instance of data after that time has occurred. This option is found on the Output tab, Other button.

The C and F commands alter the meaning of the Stop Condition.

##### **8.2.3.1.4.1 "C" Option: Formatting Event Tests Containing Conditional Output Arrays**

The C option is used to combine data from two or more conditional arrays onto one Split output line. A conditional array is one that is only output when a defined event occurs.

Assume that two or more conditional Output Arrays with unique Output Array IDs compose a test period, followed by an unconditional Output Array that defines the end of a test. The unconditional "end of test" Output Array is at the end of each test, but the conditional Output Arrays may or may not be present. The data file is comprised of several of these tests.

As an example, let's look at a vehicle test application. The start of the test is when the vehicle is turned on, and the end of the test is when the vehicle is turned off. The conditional output arrays could be:

- monitoring the engine temperature and outputting data to a unique array when the temperature exceeds a limit

- outputting data to a unique array when the brakes are applied
- outputting data when engine RPM exceeds a limit

The unconditional array data (the stop condition) would be output to a unique array when the engine is turned off. By processing the data with Split using the C option, the data collected during each test could be merged on to one line, with blanks inserted if a set of data didn't exist (e.g., if the engine temperature never exceeded the defined limit).

- An Input File must be set up for each array ID in the test. The first Input File is configured on the Input File tab that appears when you open Split. Additional Input Files are added by choosing Edit | Add Data File from the Split menu. The same data file will be used as the Input File for each array.
- Type in the array ID in the Copy field of the Input File tab for each array. The array ID is the first element of a data file, so the line should read 1[123], where 123 is the actual array ID you want to process.
- In the Select field, type in the number for each element (data value) you want to be output in the report.
- In the Stop Condition field, type in a "C," followed by the ID of your stop condition array. If your "end of test" array was array ID 200, the Stop Condition field would read: C,1[200]. This should be typed into the Stop Condition fields of each array, including the "end of test" array.

Set up the Output File as you would for any Split process. If you are including column headings, the arrays and elements will appear in the order they are listed on the Input File tabs. That is, the first column will be Input File number 1, element number 1; the next column is Input File number 1, element number 2... Input File number 2, element number 1 follows in the column immediately after the last element of Input File number 1.

Consider Table 8.2-2 below:

TABLE 8.2-2. Example of Event Driven Test Data Set	
100,12.1,10.,32.6	Data from arrays output during the first test.
101,92.7,67.7	
102,56.1,48.7,98.,220.1	
200	Second test.
100,12.5,9.89,30.1	
102,56.2,50.,100.5,210.6	
200	Third test.
100,13.1,10.1,33.1	
101,94.1,69	
200	

This table contains four different output arrays: 100, 101, 102, and 200. During the first test, data was output from all three conditional arrays (100, 101, and 102), with 200 signaling the end of the test. During the second test, data was output from arrays 100 and 102. During the third test, data was output from arrays 100 and 101.

To process these files using the C option, the parameter file would be set up as follows (assuming the name of our data file is Data\_1.DAT):

First Input File = Data\_1.DAT:1  
 Stop condition = C,1[200]  
 Copy = 1[100]  
 Select = 1,2,3,4

Second Input File = Data\_1.DAT:2  
 Stop condition = C,1[200]  
 Copy = 1[101]  
 Select = 1,2,3

Third Input File = Data\_1.DAT:3  
 Stop condition = C,1[200]  
 Copy = 1[102]  
 Select = 1,2,3,4,5

Fourth ("end of test") Input File = Data\_1.DAT:4  
 Stop condition = C,1[200]  
 Copy = 1[200]  
 Select = (leave blank)

**NOTE**

The *:(number)* after the data file name is inserted automatically by Split.

**TABLE 8.2-3. Processed Data File Using Option C**

100	12.1	10	32.6	101	92.7	67.7	102	56.1	48.7	98	220.1
100	12.5	9.89	30.1				102	56.2	50	100.5	210.6
100	13.1	10.1	33.1	101	94.1	69					

When Split is run, the resulting data file will look similar to Table 8.2-3. Each line of data represents one test. Notice that blanks were inserted if the data set (conditional array) did not exist.

**8.2.3.1.4.2 Trigger on Stop Condition (F Option) Output of Time Series**

The Trigger on Stop Condition, or F option, changes the function of the Stop Condition when one or more Time Series functions (Section 8.2.3.1.11) are contained in the Select field. When a Stop Condition is met, the time series data is calculated and written to the output file. However, instead of stopping at this point, processing resumes and time series data is output the next time the Stop Condition is met. This continues until the end of file or until the user stops Split manually.

The Trigger on Stop Condition is enabled by clicking **Other...** on the Output Tab and checking the box next to the Trigger on Stop Condition field. When the Trigger on Stop Condition is enabled, the function affects all files being processed that have a Stop Condition specified. If multiple files are being processed but it is desired that the function affect one or more—but not all—of the files, the F option is used in the Stop Condition field of the files that you want processed using the function. The syntax for the F option is: F,e<sub>i</sub>[val<sub>i</sub>].

A typical application for the Trigger on Stop Condition is to reduce days of hourly data into daily summaries. A logical element to use for the Stop Condition is time (hrmn). Assuming the third element of the hourly Output Array is hrmn, and midnight is output as 0, the Stop Condition is entered as 3[0] (or F,3[0] if the F option is used). The Time Series processing is performed over a day defined by midnight to midnight.

If only hourly Output Arrays were contained in the Input File, the Copy line could be left blank. If other Output Arrays are present which need not be included in the Time Series processing, a logical Copy condition would be the Output Array ID of the hourly output.

The Trigger on Stop Condition functions the same for multiple Input files as it does for a single Input File. If the option is enabled on several Input Files, and the Stop Conditions do not occur at the same point in each file, when a file's Stop Condition is met, its time series data are output and blanks are output for data selected from the other Input Files.

Say, for example, that you were interested in the average value of the first data point (element 2) for each test, in the data set listed in Table 8.2-2. The Input File template would look like that shown in Table 8.2-4.

**TABLE 8.2-4. Input File Entries to Process the First Data Point for each Test**

First Input File =	DATA_1.DAT:1
Stop Condition =	F,1[200]
Select =	AVG(2)

### 8.2.3.1.5 Copy

The Copy Condition tells Split which arrays should be used for the output data. After the Start Condition is satisfied, and before the Stop Condition is met, the Copy condition must be satisfied before any data will be processed according to Select line instructions. If the Copy condition is left blank, all arrays are processed between the Start and Stop values. Syntax for the Copy condition is similar to the Start and Stop values mentioned above. Logical "and" and "or" statements (see Section 8.2.3.1.3) can be used when specifying the Copy condition.

For example, referring to Table 8.3-1, if only those hours during day 189 when the temperature was above 90 and the soil temperature was below 62 is desired, or, during day 189 when the average wind speed was below 21 while the wind direction was between 255 to 265 is desired, the Copy condition would be:

1[189]and4[90..150]and5[0..61.99]or1[189]and6[0..20.99]and7[255..265]

Only Output Arrays with hours 1300 and 1500, Table 8.3-1, conform to the above Copy conditions.

### 8.2.3.1.6 Select

The Select line specifies which elements of an Output Array are selected for processing and/or output to the specified Output File. The Select line becomes operable only after the Start Condition and Copy condition are met, and before the Stop Condition is satisfied. If the Select line is left blank, all elements in output arrays meeting the Start Condition and Copy conditions are output to the Output File.

Processing is accomplished through arithmetic operators, math functions, spatial functions, and time series functions.

### 8.2.3.1.7 Ranges

Element numbers may be entered individually (e.g., 2,3,4,5,6,7), or, in groups (e.g., 2..7) if sequential. Range limits (lower to upper boundary conditions) may be placed on elements or groups of elements specified in the Select or Copy lines. For example, 3[3.7..5],4..7[5..10] implies that element 3 is selected only if it is between 3.7 and 5, inclusive, and elements 4,5,6, and 7 must be between 5 and 10, inclusive.

If range limits are used in the Select condition, when Split is run, any data which are outside of the specified range will be highlighted according to the options chosen for the output file. Table 8.2-5 summarizes what each option produces on the screen and in the output file if out of range data are encountered. This type of range testing is a quick way to identify data problems.

**TABLE 8.2-5. Effects of Out of Range Values for Given Output Options**

Output Option	Screen Display*	PRN File	RPT File or Printer Output
Report = None; No other options defined (default)	bad values displayed in red and preceded by asterisk; the text "bad data" highlighted in a red box at bottom right of screen	blanks inserted for bad values	N/A
Report = File or Printer; no other options defined	bad values displayed in red and preceded by asterisk; the text "bad data" highlighted in a red box at bottom right of screen	blanks inserted for bad values	bad values preceded by asterisk
Report = None; replacement text (abc) in "Replace bad data with" field	bad values displayed in red and preceded by asterisk; the text "bad data" highlighted in a red box at bottom right of screen	abc inserted in place of bad values	N/A

Report = File or Printer; comment in "Replace bad data with" field	bad values displayed in red and preceded by asterisk; the text "bad data" highlighted in a red box at bottom right of screen	comment inserted in place of bad values	bad values preceded by asterisk
Report = None; "Display only bad data" option enabled	only lines with bad data are displayed; bad values displayed in red and preceded by asterisk; the text "bad data" highlighted in a red box at bottom right of screen	only lines with bad data output; blanks inserted for bad values	N/A
Report = File or Printer; "Display only bad data" option enabled	only lines with bad data are displayed; bad values displayed in red and preceded by asterisk; the text "bad data" highlighted in a red box at bottom right of screen	only lines with bad data output; blanks inserted for bad values	only lines with bad data output; bad values preceded by asterisk
*The Screen Display box must be checked; if not, no data will be displayed on the Split Run screen.			

---

**NOTE** In this instance, out of range data refers to data outside of the specified output range. It is not to be confused with out of range data generated by the logger.

---

**8.2.3.1.8 Variables**

Variables can be assigned names in the Select line. For example,  $x = 4 - 5 * (6 * 3.0)$  means that x is equal to element 6, times the number 3, times element 5, subtracted from element 4. A numeric value is distinguished from an array element by the inclusion of a decimal point. Variables must be declared before they can be used in the Select line. A variable name must start with an alpha character, can include numbers and must not exceed eight characters. Variable names can start with the same character but they must not start with another complete variable name (e.g., the variable XY is not valid if there is also the variable X). A comma must follow each variable statement, as with all parameters in the Select line. Once the variables have been declared they can be used later in the Select line (i.e.,  $x = 4 - 5 * (6 * 3.0)$ ,  $y = 6 / 3, 2, 3, 6, 7, 7 * x, 6 + y$ ).

---

**NOTE** Variables can be defined in the **first four Input File's Select lines** only, but may be used in subsequent Input File's Select lines.

---

Illegal operations (e.g., logarithm of a negative number) will cause Split to store blanks for the Output. It is possible to get a run time error (error 0/1) if the floating point math exceeds the limits of the PC.

**8.2.3.1.9 Numerical Limitations**

The greatest number that can be output is determined by the field width (Table 8.2-3, "Column Widths" option). If the width is eleven or greater, the maximum number is 99,999,999; for widths from eight through ten the maximum is 99,999; for widths less than eight the maximum is 9999. If a column is not large enough for a value, it will be stored as a 9,999, 99,999 or 99,999,999 based on the column width. In some instances, such as when a column is not large enough for the date function, you will see the text "bad data" on the Split Runtime window.

**8.2.3.1.10 Mathematical Functions, Details, and Examples**

**TABLE 8.2-6. Split Operators and Math Functions**

OPERATORS	OPERATOR PRECEDENCE ORDER (3 = high, 1 = low)
^ = raise to the power	3
x Mod y = Modulo divide of x by y	2
* / = multiplication, division	2
+ - = addition, subtraction	1

**EXAMPLES OF SYNTAX FOR MATHEMATICAL OPERATORS**

3*5	multiply element 3 by element 5
3/5	divide element 3 by element 5
(3..5)/(8..10)	same as 3/8, 4/9, 5/10
3+5	add element 3 to element 5
3-5	subtract element 5 from element 3
(3,9,5)-(8,7,10)	same as 3-8, 9-7, 5-10
3*2.0	multiply element 3 by a fixed number 2
2^3.0	raise element 2 to the third power

**MATH FUNCTIONS**

Abs(x)	= Absolute value of x
Arctan(x)	= Arc tangent of x (in degrees)
Cos(x)	= Cosine of x (in degrees)
Exp(x)	= Natural Exponent function (e <sup>x</sup> )
Frac(x)	= Fractional portion of x
Int(x)	= Integer portion of x
Ln(x)	= Natural logarithm of x
Sin(x)	= Sine of x (in degrees)
SpaAvg(x..y)	= Spatial average of elements x through y
SpaMax(x..y)	= Spatial maximum of elements x through y
SpaMin(x..y)	= Spatial minimum of elements x through y
SpaSd(x..y)	= Spatial standard deviation of elements x through y
Sqrt(x)	= Square root of x

The following array of ASCII data will be used for all Mathematical function examples.

0105 0176 1200 -07.89 55.10 12.45 270.5

**Abs(x)** returns the absolute, or positive value of element x.  
 Examples:  
 Abs(4) = 7.89  
 Abs(4\*5) = 434.74

<b>Arctan(x)</b>	returns the arc tangent of element x in degrees. Examples: Arctan(7) = 89.788 Arctan(7/6) = 87.365
<b>Cos(x)</b>	returns the cosine of element x in degrees. Examples: Cos(5) = .57215 Cos(5-6) = .73551
<b>Exp(x)</b>	returns the exponential base e to the power of element x. Example: Exp(4) = .00037
<b>Frac(x)</b>	returns the fractional value of the element x. Examples: Frac(4) = -.89 Frac(6+7) = .95
<b>Int(x)</b>	returns the integer portion of the element x. Examples: Int(7) = 270 Int(5*6) = 685
<b>Ln(x)</b>	returns the natural log of element x. Examples: Ln(6) = 2.5217 Ln(7/6*5/1) = 2.4337
<b>Sin(x)</b>	returns the sine of element x in degrees. Examples: Sin(7) = -.99996 Sin(7-2+5) = .50603

Spatial functions, included under Mathematical functions, operate on a per Output Array basis. The average, maximum, minimum, and standard deviation of a specified group of elements within an array are calculated.

<b>SpaAvg(x..y)</b>	returns the spatial average of elements x through y. Examples: SpaAvg(1..7) = 258.74 SpaAvg(1,4,7) = 122.54
<b>SpaMax(x..y)</b>	returns the maximum value of elements x through y. Examples: SpaMax(1..7) = 1200 SpaMax(1,2,5) = 176
<b>SpaMin(x..y)</b>	returns the minimum value of elements x through y. Examples: SpaMin(1..7) = -7.89 SpaMin(1,2,5) = 55.1

**SpaSd(x..y)** returns the standard deviation of elements x through y.  
 Examples:  
 SpaSd(1..7) = 394.57  
 SpaSd(5,2,1) = 49.607

**Sqrt(x)** returns the square root of element x.  
 Examples:  
 Sqrt(3) = 34.641  
 Sqrt(3^ 2.0) = 1200

**8.2.3.1.11 Time Series Functions, Details, and Examples**

**TABLE 8.2-7. Time Series Functions**

**TIME SERIES FUNCTIONS**

Avg(x;n)	= Average
Blanks(x;n)	= Number of blanks in element
Count(x;n)	= Number of data points in element
Max(x;n)	= Maximum
Min(x;n)	= Minimum
RunTotal(x;n)	= Running total
Sd(x;n)	= Standard deviation
Smpl(x;n)	= Sample raw value
SmplMax(x;y;n)	= Sample (y) on a maximum (x)
SmplMin(x;y;n)	= Sample (y) on a minimum (x)
Total(x;n)	= Totalize
WAvg(x;n)	= Unit vector mean wind direction (in degrees)

**NOTE:** x can be an element or a valid expression. n is optional and is the number of arrays to include in the function. Date and Edate can be used for the “n” in the Time Series functions to produce monthly output (see Table 8.2-8 Special Functions, this Section).

Time Series functions are used to perform vertical processing on selected elements, such as calculating the average of an element over a specified range of data. Time Series results are output in three instances:

1. when a Trigger on Stop Condition (F option) is met
2. at the end of a data file (or within a range specified by Start and Stop Conditions)
3. when an interval count is met

When the Trigger on Stop Condition (or F option) is used, any time series data defined in the Select line is output each time the Stop Condition is met. Refer to Section 8.2.3.1.4.2 for more information on the Trigger on Stop Condition.

Results which are output at the end of a file or a range of data are referred to as Final Summaries. A typical select line that would produce a Final Summary is:

1,2,3,4,Avg(4)

This line would output values for elements 1 through 4 each time an array was output. Additionally, an average value for element 4 would be calculated for the entire file and output as the last line of data in the output file.

1,2,3,4,Avg(4;24)

This line would output values for elements 1 through 4 each time an array was output, and an average value for element 4 would be calculated every 24<sup>th</sup> array and output as an additional column in the file. An additional summary would occur for an Interval Count if the count was not evenly divisible into the number of output arrays present in the Input File. The summary, in this case, is calculated from an incomplete interval count.

The date( ) function can be used for the interval in a time series function to produce monthly output. Refer to the Monthly summary example in Section 8.2.3.1.12.

**NOTE**

When Date and Edate are used within other functions they must be used with the older format Date(doy;y) and Edate(doy;y) instead of using the extended date functions. For example AVG(1;Date(2;2002.0)). The decimal is needed to indicate a fixed number. Numbers without the decimal are interpreted as element IDs.

The interval count in a Time Series Function is optional and does not require a decimal point. To determine the interval, Split counts the number of arrays which meet the specified conditions (Stop, Start, and Copy). If the time synchronize function is enabled, the Time Series functions remain synchronized to the starting time even if a complete array is missing from the input data. When elements are missing, the Time Series calculations are based on the actual number of elements found.

Semicolons are used in Time Series functions to separate the elements or expressions from the count which determines the interval. SmplMax and SmplMin require two elements separated by a semicolon. The first is checked for a maximum or minimum, while the second is sampled on the maximum or minimum.

The following set of weather data from Mt. Logan in northern Utah gives a total of seven elements each hour. This Field Formatted output, with title and column headers, was generated by Split. These data are used in the following examples of Time Series functions.

**Mt. Logan Weather Data**

Day	Time	Airtemp deg F	RH	Mean Wind Speed mph	Mean Wind Direction	Std Dev of Direction
178	100	58.56	17.42	5.855	338.3	6.562
178	200	57.48	17.65	8.27	344.8	7.51
178	300	56.85	17.76	7.75	330.8	5.065
178	400	56.55	18.89	7.6	319.7	10.93
178	500	56.57	19.6	10.41	307.3	4.23
178	600	55.33	23.32	8.99	317.7	6.258
178	700	55.95	24.79	9.52	322.3	4.609
178	800	58.12	23.98	6.588	315.6	9.43

178	900	59.79	23.46	5.458	312	15.32
178	1000	61.09	24.12	4.622	299.3	18.3
178	1100	61.34	25.03	5.926	303	17.26
178	1200	60.61	27.46	6.815	309.7	18.71
178	1300	61.01	25.44	8.35	310.2	18.37
178	1400	60.93	25.48	10.92	317.5	12.68
178	1500	62.3	23.79	8.43	310.6	19.21
178	1600	63.75	24.31	8.88	321.4	15.22
178	1700	66.15	22.45	7.97	341	17.77
178	1800	67.33	23.06	6.758	344.1	20.74
178	1900	66.59	24.75	7.08	341.8	16.09
178	2000	64.52	26.03	8.76	337.2	14.91
178	2100	59.84	27.45	11.81	305.4	12.36
178	2200	56.19	35.46	15.62	316.7	19.01
178	2300	55.48	38.8	17.12	338.7	11.41
179	0	55.22	37.13	11.86	351.6	8.22

**Avg(x;n)** returns the average of element x over a full data set or every n<sup>th</sup> value.

Examples:

Avg(3) = 59.898 (average daily temp)

Avg(3;4) = 57.36 (average 4 hour temp)

56.493 (average 4 hour temp)

60.708 (average 4 hour temp)

61.998 (average 4 hour temp)

66.148 (average 4 hour temp)

56.683 (average 4 hour temp)

**Blanks(x;n)** returns the number of blanks or bad data in element x over a full data set or every nth value. Refer to Table 8.3-9 for definition of blank or bad data. Example:

Blanks(3) = 0 (no holes in data set).

**Count(x;n)** returns the number of data points (non blanks) in element x over a full data set or every n<sup>th</sup> value.

Example:

Count(1) = 24 (24 data points in data set).

**NOTE**

Blanks and Count are functions designed for checking the integrity of the data file. A common use for these two functions is “100.\*BLANKS(x;n)/BLANKS(x;n)+COUNT(x;n)” which gives the percentage of holes (bad data) in the file.

**Max(x;n)** returns the maximum value of element x over a full data set or every n<sup>th</sup> value.

Examples:

Max(5) = 17.12 (max WS for day)

Max(5;12) = 10.41 (max WS for 12 hours)

17.12 (max WS for 12 hours)

**Min(x;n)** returns the minimum value of element x over a full data set or every n<sup>th</sup> value.

Examples:

Min(7) = 4.23 (min std. dev. of WS for day)  
 Min(3;8) = 55.33 (min temp for 8 hours)  
           59.79 (min temp for 8 hours)  
           55.22 (min temp for 8 hours)

**RunTotal(x;n)** returns a running total of element x for every line in the data set. If an n<sup>th</sup> value is specified, a running total will be output every n<sup>th</sup> value.

Example: RunTotal(5) =  
           5.85  
           14.12  
           21.87  
           29.47  
           39.88  
           48.87  
           :  
           :  
           :  
           166.76  
           182.38  
           199.50  
           211.36  
           211.36

Running total of hourly average wind speed provides up-to-the-hour wind run for that day. Because an n<sup>th</sup> value was not specified, the Final Summary output, which is daily wind, is the same as the "total" output.

**Sd(x;n)** returns the standard deviation of element x over a full data set or every n<sup>th</sup> value.

Examples:  
 Sd(3) = 3.6593 (std. dev. temp for day)  
 Sd(3;8) = 1.011 (Sd temp for 8 hours)  
           1.1182 (Sd temp for 8 hours)  
           4.965 (Sd temp for 8 hours)

**Smpl(x;n)** returns a sample of element x every n<sup>th</sup> value.

Examples:  
 Smpl(4;8) = 23.98 (RH every 8 hours)  
           24.31 (RH every 8 hours)  
           37.13 (RH every 8 hours)

**SmplMax(x;y;n)**

looks for a maximum value in element x and samples element y when the maximum is found. If an n<sup>th</sup> value is specified then it outputs the sample on a maximum every n<sup>th</sup> value, otherwise it outputs the sample on a maximum at the end of file.

Examples:  
 SmplMax(5;(3)) = 55.48 (on max wind speed sample temperature)  
 SmplMax(5;(3,6);8) = 56.57 307.3  
 60.93 317.5

55.48 338.7  
 (on max wind speed sample temperature and wind direction every 8 hours)

**SmplMin(x;y;n)** looks for a minimum value in element x and samples element y when the minimum is found. If an n<sup>th</sup> value is specified then it outputs the sample on a minimum every n<sup>th</sup> value, otherwise it outputs the sample on a minimum at the end of file. Examples:

SmplMin(3;5) = 11.86 (on min temp sample wind speed)

SmplMin(3; (5,6);8) = 8.99 317.7

5.458 312

11.86 351.6

(on min temperature sample wind speed and wind direction every 8 hours)

**Total(x;n)** returns the total of element x over a data set or every n<sup>th</sup> value.

Examples: Total(5) = 211.36 (daily wind run)

**WAvg(x;n)** Returns the unit vector mean wind direction in degrees of element x (wind direction in degrees) over a full data set or every nth value.

Example:

WAvg(6) =

323.14 (mean wind direction for the day)

WAvg(6;4) =

333.41 (mean wind direction for 4 hours)

315.73 (mean wind direction for 4 hours)

306 (mean wind direction for 4 hours)

314.92 (mean wind direction for 4 hours)

341.03 (mean wind direction for 4 hours)

328.09 (mean wind direction for 4 hours)

### 8.2.3.1.12 Special Functions, Details, and Examples

**TABLE 8.2-8. Split SPECIAL FUNCTIONS**

<b>CrLf</b>	= Insert carriage return line feed in Output File.
<b>Date("format"S;H;D;Y)</b>	= Convert day of year and time to a timestamp with calendar date and time, where format uses Windows conventions to specify output format. S=seconds, H = HoursMinutes, D = Day, Y = year. The output timestamp is quoted text. Date can be used to create monthly time series summaries.
<b>Edate("format"S;H;D;Y)</b>	= The same as the Date function except that the output text is not quoted. EDate can be used to create monthly time series summaries.
<b>"Label"</b>	= Insert Comment in Output file. (Label is anything within the quote marks.)

<b>Line</b>	= Number of lines written to Output file.
<b>smp1(.pa;n)</b>	= Page break such that n is the number of lines per page for the printer or the .RPT file.
<b>PCdate or PCEdate</b>	= Used in a report header to print the current date.
<b>WDQ(n)</b>	= Outputs the wind direction using an alphabetical abbreviation, based on 8 quadrants .
<b>WDQS(n)</b>	= Outputs the wind direction using an alphabetical abbreviation, based on 16 quadrants .

The Mt. Logan data set is used for the Special Function examples. These functions are helpful in converting time fields to formatted timestamps and formatting the output. Since one of the main differences between mixed-array data files and table based data files is the time format, these functions can be used to convert between file types.

**NOTE**

If you are processing the data file in multiple passes including formatting of the date and time fields, you should put the date processing in the final pass. Split cannot read all of the timestamp formats that it can produce. For example, the quoted timestamp in table based data files has a specific structure. Any changes to the structure will make the timestamp unreadable for Split.

**Crlf** returns a carriage return and line feed where the Crlf is placed in the parameter file.  
 Examples:  
 Smp1("Max Temp";24),Max(3;24),  
 Smp1(Crlf;24),Smp1("Max RH";24),Max(4;24)  
 = Max Temp 67.33  
 Max RH 38.8  
 The Crlf is placed after the maximum temperature 67.33 so that the maximum RH is on the next line.

**NOTE**

A carriage return/line feed is recognized by Split as an element, and may throw the column headers off in the output file.

**"Label"** returns a comment in the output file. This is a useful formatting function when labels are desired on the same line as the data. The label includes anything within the quote marks, the quote marks are not output but must be in the parameter file. The label cannot exceed the width of the output column (default is eight characters). A maximum of thirty (30) labels are allowed per Select line.

Make sure that the column widths are big enough for the label to fit. Otherwise the output will indicate Bad Data.

Examples:

```
"Max Temp" =
  Max Temp (outputs Max Temp
  Max Temp 24 times)
.
.
.
  Max Temp
```

```
Smpl("8 hour ";8),Smpl("Max Temp";8), Max(3;8) = 8 hour
  Max Temp 58.56
8 hour Max Temp 63.75
8 hour Max Temp 67.33
```

This example samples the labels called "8 hour" and "Max Temp" and looks for a Maximum temp for every 8 hour interval.

**Line**

numbers each line written to the report file or printer. This differs from the Count function in that Count looks at how many lines were read.

Examples:

```
Line, 4, 5 =
  1  17.42  5.855
  2  17.65  8.27
  3  17.76  7.75
  4  18.89  7.6
  5  19.6   10.41
  6  23.32  8.99
  7  24.79  9.52
```

```
.
.
.
```

```
19  24.75  7.08
20  26.03  8.76
21  27.45  11.81
22  35.46  15.62
23  38.8   17.12
24  37.13  11.86
```

```
Smpl (Line;8), Smpl (4;8), Smpl (5;8)
```

```
1  23.98  6.588
2  24.31  8.88
3  37.13  11.86
```

**smpl(.PA,n)**

Outputs the data to the printer or .RPT file with **n** lines per page.

Examples:

```
2, 3, Smpl (.PA;12) =
  100      58.56
  200      57.48
.
.
.
1100     61.34
```

1200	60.61
1300	61.01
1400	60.93
.	.
.	.
.	.
2300	55.48
0	55.22

**WDQ(n)** Outputs the wind direction using an alphabetical abbreviation, based on 8 quadrants (N, S, E, W, NE, NW, SE, SW). n is an element containing wind direction. For example, if n = 182, S would be returned in the output file.

**WDQS(n)** Outputs the wind direction using an alphabetical abbreviation, based on 16 quadrants (N, S, E, W, NE, NW, SE, SW, NNE, ENE, ESE, SSE, SSW, WSW, WNW, NNW). n is an element containing wind direction. For example, if n = 111, ESE would be returned in the output file.

**Date("format"; S; H; D; Y)** Converts a datalogger's time stamp to a different format and encloses it in double-quotes (edate will produce a date without quotes). "Format" is a string which identifies how the date should be output. The "format" string is similar to the date format used by Windows. See the online help in Split to get a complete list of the format parameters.

S is the element number that contains seconds; H is the element number that contains hours/minutes; D is the element number that contains day; and Y is the element number that contains the year. A constant can be used in place of any of the element numbers (the constant must be a valid value for the type of date field and include a decimal point; e.g., 2000.0 for the year). If only three elements are specified, these will be assumed to be hour/minute, day, and year.

When using the Date function for a table-based datalogger (e.g., a time stamp in the format "2002-02-03 21:16:00"), if the time stamp is the first element in the array, a 1 is used for all of the time stamp elements (S; H; D; Y).

If "serial" is entered for the "format" string, a serial date will be output. Other special functions are "hourarray" and "dayofyear". Both of these are used when processing data from table-based dataloggers so that the timestamps are similar to that of mixed array dataloggers. Hourarray changes a 0000 hourly timestamp to 2400, and dayofyear produces a Julian Day.

In older versions of Split, the `date()` and `edate()` functions were limited to converting the Julian day to a MM-DD format, with a syntax of `date(doy;y)` where `doy` = the element number for the day of the year; `y` = the element number for the year. This older format is still supported.

**NOTE**

Split will mark the date as Bad Data if the time and date resulting from the conversion will not fit in the specified column width. The on-screen display and the report file will precede the date with asterisks. In the .PRN output file, Split uses the Bad Data string.

When Date and Edate are used within other functions they must be used with the older format `Date(doy;y)` and `Edate(doy;y)` instead of using the extended date functions as shown in the table. For example `AVG(1;Date(2;2002.0))`. The decimal is needed to indicate a fixed number. Numbers without the decimal are interpreted as element IDs.

**Date Format Examples**

Assume that in a mixed array data file, element 2 is Year, element 3 is Day of Year, element 4 is Hour/Minute, and element 5 is Seconds.

<u>String Entered</u>	<u>Output</u>
<code>date("mm/dd/yy, h:nn";5;4;3;2)</code>	"02/25/02, 4:10"
<code>edate("mm/dd/yy, hh:nn";5;4;3;2)</code>	02/25/02, 04:10
<code>edate("dddd, mmmm d, yyyy";5;4;3;2)</code>	Monday, February 25, 2002
<code>edate("'Date:' mmm d, yyyy";5;4;3;2)</code>	Date: Feb 25 02

If a time element is missing from a mixed array data file, use a valid constant instead.

If processing a table-based data file, use a 1 for all time elements (assuming the time stamp is the first element in the data file). For the examples above:

<code>date("mm/dd/yy, h:nn";1;1;1;1)</code>	"02/25/02, 4:10"
<code>edate("mm/dd/yy, hh:nn";1;1;1;1)</code>	02/25/02, 04:10
<code>edate("yyyy", "dayofyear", "hhmm";1;1;1;1)</code>	2002, 56, 0410

Notice that this last example essentially creates an array-type of timestamp.

**NOTE**


---

When processing a data file from a mixed array datalogger, if the time stamp uses midnight as 2400 with "today's" date, the date function will convert that time stamp to 0000 hours with "tomorrow's" date. The "No Date Advance" function can be used to stop the date from rolling forward (Other button, No Date Advance check box).

---

**edate("format"; S; H; D; Y)**      **edate( )** functions identically to **date( )** above, except that the time stamp is not surrounded by quotes.

**Monthly Summary Example**

The Date function can be used to produce a monthly summary of daily time series data by using Date( ) for the interval in the time series function. This will trigger time series output for the first day of each month. The syntax is `avg(7,date(3;2))`, where you want to take a monthly average of element 7, and the day of year is contained in element 3 and the year in element 2. If you have data recorded on a once per minute or once per hour basis, it must first be processed into a 24 hour summary for this function to produce the output expected.

**NOTE**


---

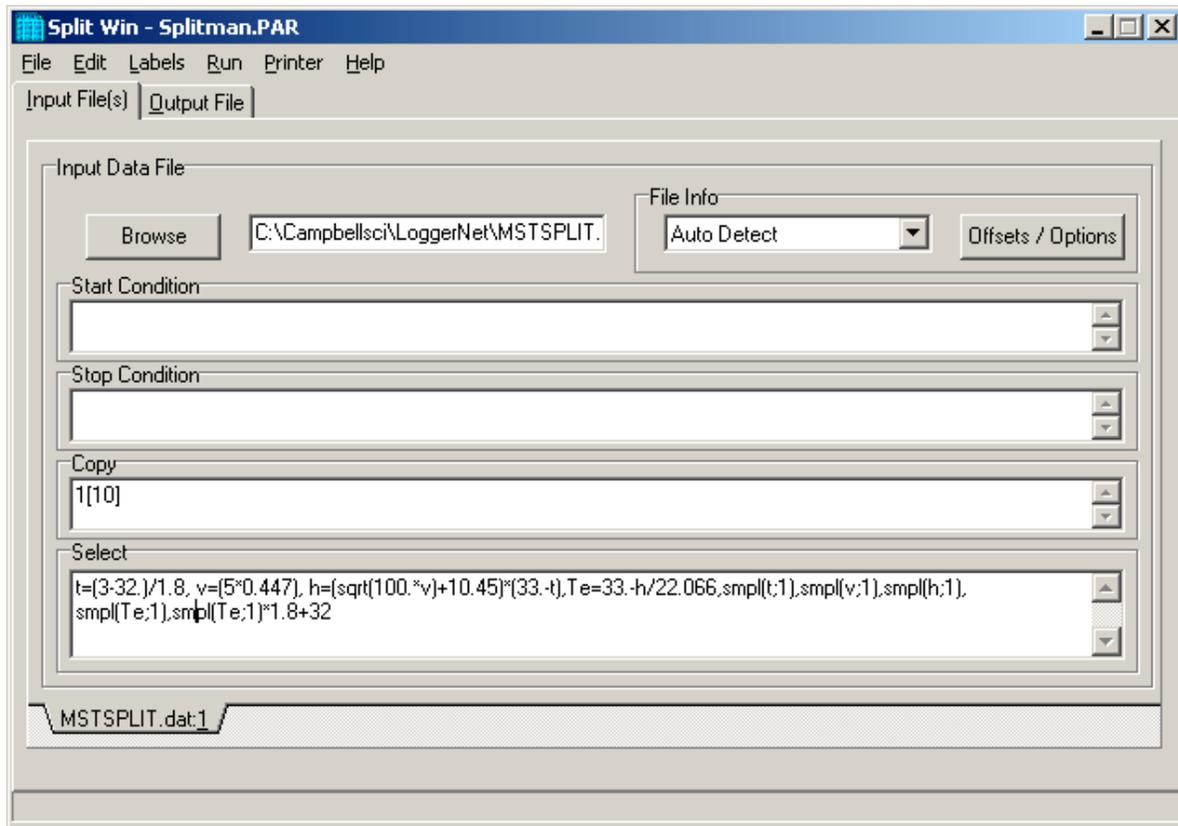
When Date and Edate are used within other functions they must be used with the older format `Date(doy;y)` and `Edate(doy;y)` instead of using the extended date functions. For example `AVG(1;Date(3;2))`. When used with table based data files the format would be `AVG(1;Date(1;1))`.

---

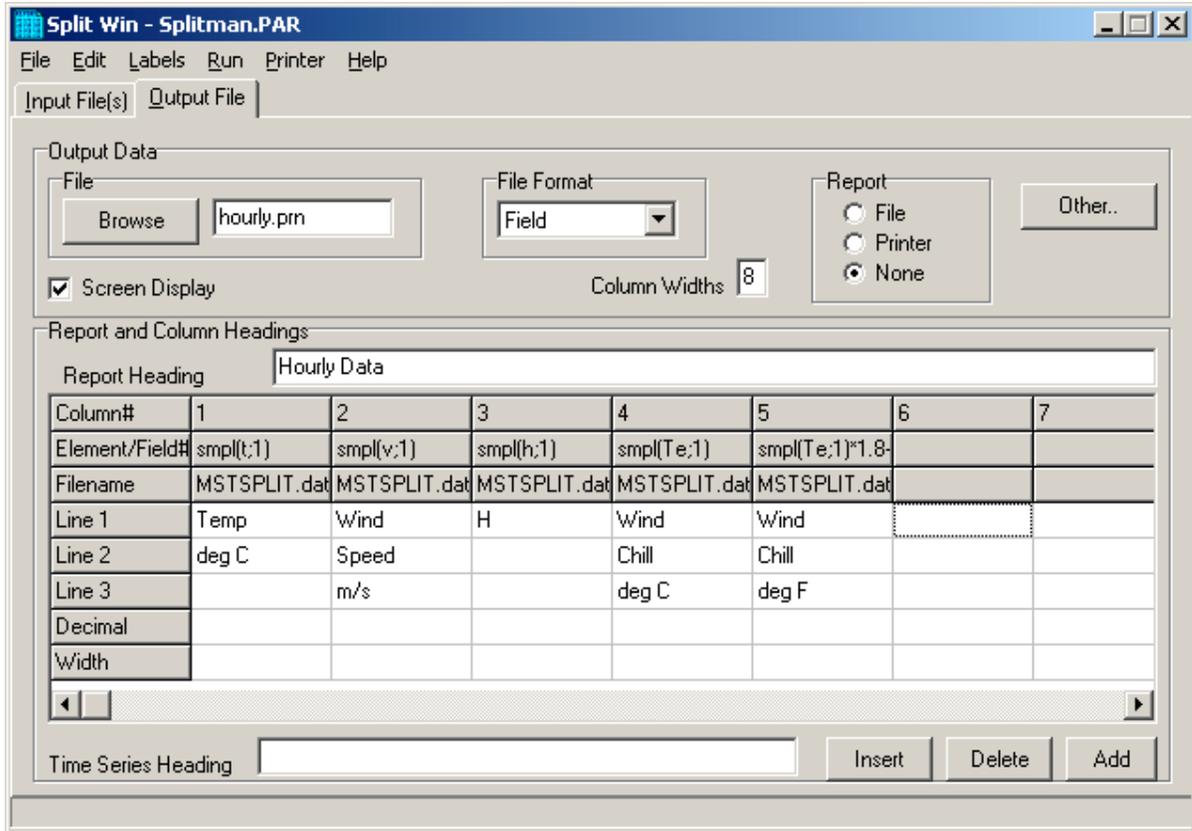
When producing a monthly summary and outputting the month along with the data, you might want to set up the value for the month as "month -1", to correctly reflect the month that the data actually represents.

**8.2.3.1.13 Split Functions Example**

The following is a parameter file that operates on the Mt. Logan data with several of the Split features being utilized. This first screen shows the input file and the select criteria that were programmed. This example does calculations based on temperature and wind speed to determine the wind chill.



The following screen shows the output file setup including the column headings and the units.



This .PAR file produces a wind chill summary of the Mt. Logan Peak data set. The formula for calculating wind chill is given as follows:

$$T_e = 33 - (h/22.066)$$

where

$$T_e = \text{Wind Chill equivalent temperature, degrees C}$$

$$h = ((100V)^{0.5} + 10.45 - V)(33 - T)$$

where

$$h = \text{Kcal m}^{-2} \text{ hr}^{-1} \text{ wind chill index}$$

$$v = \text{wind speed in meters/second}$$

$$T = \text{temperature in degrees C}$$

Note that at wind speeds between 0 to 4 mph (0 to 1.8 m/s), the wind chill should be ignored because this formula results in wind chill temperatures that are greater than the ambient temperature. The National Weather Service includes wind chill in reports only when temperatures drop below 35°F (1.7°C).<sup>1</sup> The formula is for example purposes and is not endorsed by Campbell Scientific as a standard.

When this .PAR file is executed, the following output is displayed on the screen.

Wind Chill Report from Mt. Logan				
Temp deg C	Wind Speed m/s	H	Wind Chill deg C	Wind Chill deg F
14.756	2.6172	438.06	13.148	55.666
14.156	3.6967	489.58	10.813	51.463
13.806	3.4643	491.34	10.733	51.319
13.639	3.3972	493.4	10.64	51.151
13.65	4.6533	529.57	9.0005	48.201
12.961	4.0185	530.58	8.9547	48.118
13.306	4.2554	528.27	9.0596	48.307
14.511	2.9448	456.04	12.333	54.199
15.439	2.4397	414.97	14.194	57.55
16.161	2.066	383.21	15.633	60.14
16.3	2.6489	402.08	14.778	58.601
15.894	3.0463	425.2	13.731	56.715
16.117	3.7325	439.59	13.078	55.541
16.072	4.8812	468.26	11.779	53.202
16.833	3.7682	421.85	13.882	56.988
17.639	3.9694	405.59	14.619	58.314
18.972	3.5626	361.39	16.622	61.92
19.628	3.0208	331.76	17.965	64.337
19.217	3.1648	345.62	17.337	63.207
18.067	3.9157	393.08	15.186	59.335
15.467	5.2791	493.51	10.635	51.142
13.439	6.9821	584.71	6.5016	43.703
13.044	7.6526	607.86	5.4526	41.815
12.9	5.3014	566.29	7.3368	45.206

Reference

<sup>1</sup>“Wind Chill Errors”, Edwin Kessler, Bulletin of the American Meteorology Society, Vol. 74, No. 9, September 1993, pp 1743-1744.

**8.2.3.1.14 Summary of Select Line Syntax Rules**

- A fixed numeric value must include a decimal point "." or be in scientific notation. There are some exceptions to this as noted below.
- Scientific notation has the format "mantissa E power of ten" (e.g., 3E5 = 3 x 10<sup>5</sup>).
- Element numbers are entered without a decimal point.
- Commas separate Select line parameters (e.g., 2,3,(3+4)/3.2,6).
- Two decimal points are used to select consecutive elements between starting and ending elements (e.g., 3..6, refers to the elements 3,4,5, and 6).
- A set is a group of two or more elements and/or expressions separated by commas and enclosed by parentheses. No member of a set can include parentheses. Therefore, a set cannot include a set or a function as one of its members. For example:

**VALID EXPRESSION**

Arctan (2/3)  
 Arctan (2/3, 3/4, 4/5)  
 Arctan (COS(2))

**INVALID EXPRESSION**

Arctan ((2/3))  
 Arctan ((2/3, 3/4), 4/5)  
 Arctan (COS(2), COS(3))

- A single expression can operate on a set of elements. For example, the expression (3..6,8)/2.0 is the same as 3/2.0, 4/2.0, 5/2.0, 6/2.0, 8/2.0; (3..6)/(2..5) is the same as 3/2, 4/3, 5/4, 6/5.
- The element or expression that is the argument of a math or Time Series function, must be enclosed in parentheses. A range of elements can be specified, resulting in as many outputs as elements (e.g., Avg(3..5,7) will output 4 averages).
- Square brackets are used to enclose an allowable range for a value (e.g., 3[3.6..12] ) to indicate that the allowable range for element 3 is from 3.6 to 12. Whole numbers within brackets do not require a decimal point. Table 8.3-6 explains how values outside the specified range are treated.
- The interval in a Time Series function is optional and does not require a decimal point.
- Semicolons are used in Time Series functions to separate the elements or expressions from the number that determines the interval. Sample on maximum and sample on minimum require two elements or expressions also separated by a semicolon.

**8.2.3.1.15 Time Synchronization**

The time synchronize function is useful when data is missing from a file or if several files of data need to be merged together. The files are synchronized according to time; any missing data in the file (or files) will be replaced with blank data.

This function synchronizes according to day, hrnm (hour-minute), and/or seconds. The syntax used to identify the time elements for array data is:

$$e_i[\text{day}]:e_i[\text{hrnm}]:e_i[\text{seconds}]$$

Referring to Table 8.3-1, to identify the day of year for a mixed-array data file, type:

$$2[189]::$$

for hrnm type:

$$:3[1200]:$$

and seconds are expressed as:

$$::4[5]$$

A single colon is assumed to be between day and hrnm (e.g., 2[189]: means day, :3[1200] means hours, and 2[189]:3[1200] means day and hour-minute). When the time synchronize function is used, a time interval must be specified in the Copy line of the first data file. For example, 4[60] in the Copy line will create a synchronized file containing the data from the input files that occurred every 60 minutes. If no time interval is specified in the Copy line then the time

specified in the Start Condition becomes simply a starting time with no time synchronization.

Typically, the starting time specified must actually be found in the input file before the Start Condition is satisfied (e.g., if the input file starts at 1100 hrs and 1000 hrs is entered for the starting time, with no day specified, Split will skip over arrays until it reaches 1000 hrs the next day). However, the Start-Stop On/After Time function can be enabled (Output tab, Other button) to trigger the start of processing when the exact time is found or at the first instance of data after that time has occurred.

#### *Table-based dataloggers*

Because the time stamp for a table-based datalogger is all one string, and therefore read by Split as one element, the syntax is somewhat different. All elements in the time stamp are specified by a 1 (if the time stamp is the first item in each row of data).

The 1s in the string identify the position of the time stamp in the line of data. Each colon represents a portion of the time stamp. The format is 1[year]:1[day]:1[hour/minute]:1[seconds]. The colons in the time stamp must be present or the function will not work correctly.

---

**NOTE**

Time synchronization can only be done for data from a single year. It will not work over a year boundary.

---

Time elements can be identified without specifying a starting time (e.g., 2:3). If you are working with only one file, Split will begin processing that file at the first record in the file. If any gap in the data is found, blank data (or the "Replace Bad Data With" text) and a carriage return line feed will be inserted for each line of missing data. Note that Split will also detect a gap in data if, for instance, you specify a start time of 2[92]:3 (start at Julian day 92) and your hour/minute for day 92 starts at 9:30 a.m. The time between the start of the day (0000) and 9:30 a.m. will be considered missing data. Blanks (or the "Replace bad data with" text) and a carriage return line feed will be inserted at the beginning of the PRN file for each "missed" output interval.

If you are working with two or more files, once Split starts processing the files (based on the time of the first record of the first file), if no data exists for the other file(s), blank data will be inserted.

If multiple input files are given specific starting times, Split starts the output at the earliest specified starting time. In a PRN file, Blanks or the comment entered in the "Replace bad data with" field are inserted for values from other input files until their starting times are reached. In a RPT file only blanks are used.

---

**NOTE**

When using time synchronization with a mixed array data file, with a midnight time stamp of 2400, you will need to select the Other button, "Midnight at 2400 hours" checkbox.

---

**8.2.3.1.15.1 Time Synchronization and the Copy Condition**

To use the time synchronize function, time element(s) must be specified in the Start Condition. The user must also specify a time interval in the Copy condition. For instance, if the original data had 15 minute outputs and you only want hourly outputs, then an interval of 60 minutes must be specified following the element number. This is entered as (assuming hrnm is element number 3) “3[60]”. If time synchronization is specified in the Start Condition, Split looks for the interval in a time element in the Copy condition. Only one time interval is specified. This interval is the unit of time to synchronize each file.

The interval can be given tolerance limits by following the interval with a comma and the tolerance. For example, if 3 is the hrnm element, and the time interval is 60 minutes +/-2 minutes, the syntax is 3[60,2].

Table based data files need to use the same time format as described in Section 8.2.3.1.3. You can specify the interval for time synchronization on table files as ::1[60]: which will give you an output interval of 60 minutes.

If the time synchronize function is enabled and data are missing at one or more of the time intervals specified, then a blank (or the comment entered in the “Replace bad data with” field) is output to the Output File. See Table 8.3-5.

**8.2.3.1.15.2 Using Time Synchronization While Starting Relative to PC Time**

Split tries to time-sync files to the top of the hour when starting relative to PC time. If you are synchronizing files where the data output interval is not at the top of the hour, you will need to specify an interval in the Copy Condition that represents a window of time in which Split should look for the hour/minute. For instance, if your data is output 50 minutes into a 60 minute interval (and therefore, your time stamps are 50, 150, 250, 350...2350) your Start Condition and Copy Conditions for the first file might look like the following:

Start Condition

2[-1]:3[50]:

Copy Condition

1[106]and3[60,10]

Where:

- element 1 is the array ID
- element 2 is the Julian day
- element 3 is the hour/minute

The Start Condition directs Split to begin processing data when the time is one day prior to the current PC time and when the hour/minute value is equal to 50. The 1[106] in the Copy Condition specifies the array from which the data should be copied. The 3[60,10] indicates that the interval for the time stamp is 60 minutes and designates a 10 minute time window on each side of the top of the hour in which Split should look for the hour/minute data (10 minutes before the hour, 10 minutes after the hour).

The second file's Copy Condition should include only the array from which to copy the data. No interval is necessary.

### 8.2.3.2 Output Files

To create an Output File, click the **OUTPUT FILE** tab. The file is created on the default drive or directory unless the file name is preceded with an alternative drive or directory. Use the **Browse** button to change directories.

Split will assign this file an extension of .PRN if an extension is not specified by the user. Whenever an Output file name is entered, regardless of extension, an Output file is created only when the **RUN | GO** menu option is selected.

If the file name you have selected already exists, you can use the "If File Exists Then" drop-down list box to determine what action Split will take. By default, each time a PAR file is run the existing output files (PRN, RPT, and HTM) are overwritten (**Overwrite** option). When **Append** is selected, the PRN file will not be overwritten -- the new data will be added to the end of the existing file. However, the RPT and HTM files will be overwritten. If **Create New** is selected, Split will create all new files using the original file name and appending an **\_0**, **\_1**, and so on to each subsequent run.

In **Append** mode, if an HTM or RPT file is needed with all the data, you will need to run the PRN created by Split through the program a second time. If the Output File name is left blank, Split does not write data to an Output File on disk; rather, it will display the processed values on the screen if the **Screen Display** box is checked. If **Screen Display** is not enabled, no data will be displayed on the **Split RUN** screen.

---

#### **CAUTION**

The Output file name cannot be the same as the Input file name. Split will display an error message if this condition occurs.

---

Several output options may be specified to alter the default output to the file. Some are located on the main **OUTPUT FILE** screen and some are made available by pressing the **Other** button.

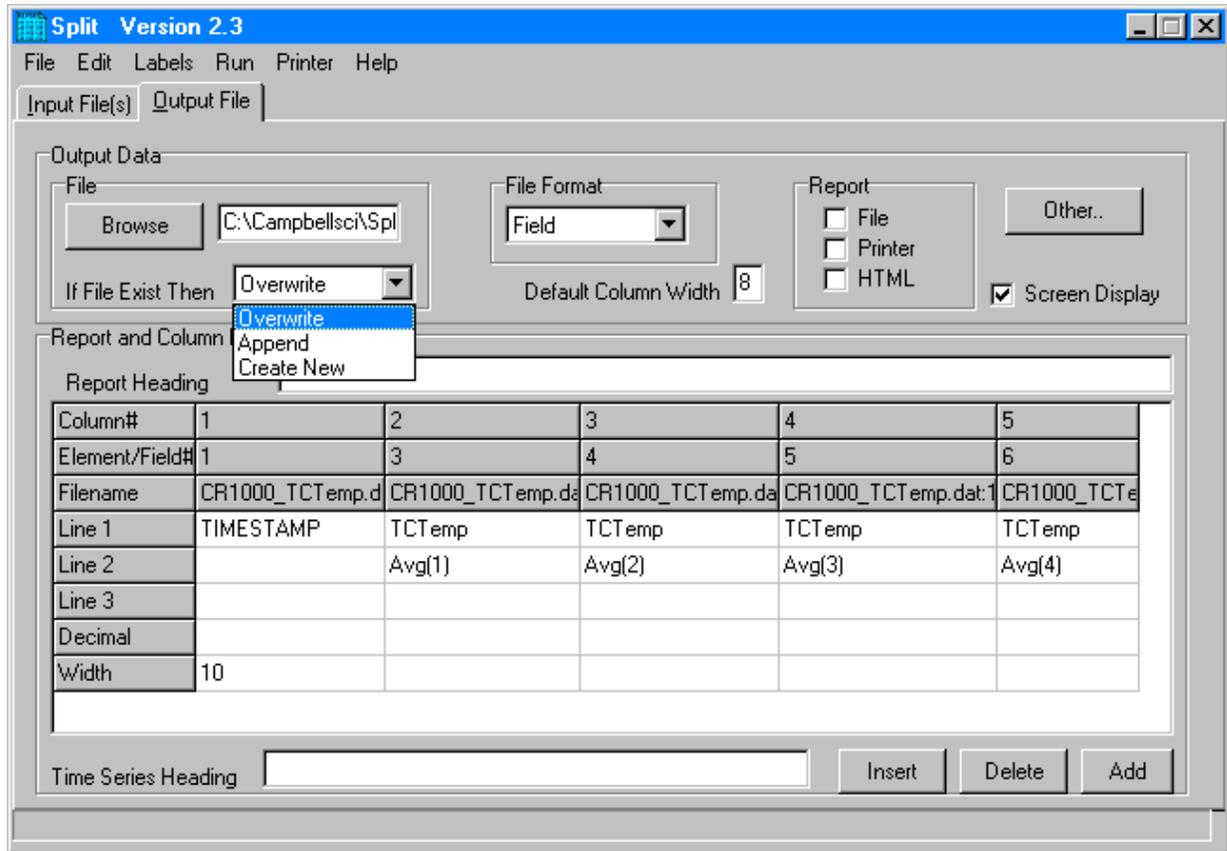


FIGURE 8.2-12. Output File Template

### 8.2.3.2.1 Description of Output Option Commands

#### File Format

There are five File Format options to choose from: No File, Field, Comma, Printable, and Custom. If No File is chosen, then only the .PRN file is saved to disk. The Field, Comma, and Printable options produce files formatted as Field Formatted, Comma Separated, and Printable ASCII, respectively. An example of each of these file types is given in Table 8.2-1 in the Input Files section.

The Custom file format uses the regional settings in the Windows operating system to determine the decimal symbol and the separator used with data values. In the Regional Settings for Numbers, the decimal symbol uses the character specified in the Decimal Symbol field; the separator uses the character specified in the List Separator field. These settings are typically found in Control Panel | Regional Settings (or Options), Numbers tab. This allows users who are used to the comma “,” as the decimal and the period “.” as a data separator to see the output data in that format.

#### Default Column Widths

The Default Column Widths field is used to set the default width of the columns. Valid entries are 6,7,8, and 9. The initial width is 8. High Resolution Final Storage data requires a minimum column width of 8. Entering a number in the Width row for each column overrides the default settings and

sets the width of individual columns. If this field is left blank, the Default Column Widths field is used.

### Screen Display

The Screen Display field controls writing the processed data to the screen. To write to the screen, check the box. For faster execution, clear the box to omit writing to screen. The data will then be written to the file only.

### Report

A report, with page and column headings, can be sent to a file or printer. There are three report options: File, Printer, HTML. One or more can be selected. A report sent to a file has the extension of .RPT. If the report is sent to a printer, the printer must be on-line. In all cases a .PRN output file is created. A basic HTML file can be created containing the formatted report data. The HTML file can be used as a display of the formatted data output in a web browser.

---

### NOTE

To remove page breaks in the HTML file, enable the “No FF” option.

---

### Other

The **Other** button provides access to the dialog box shown below.

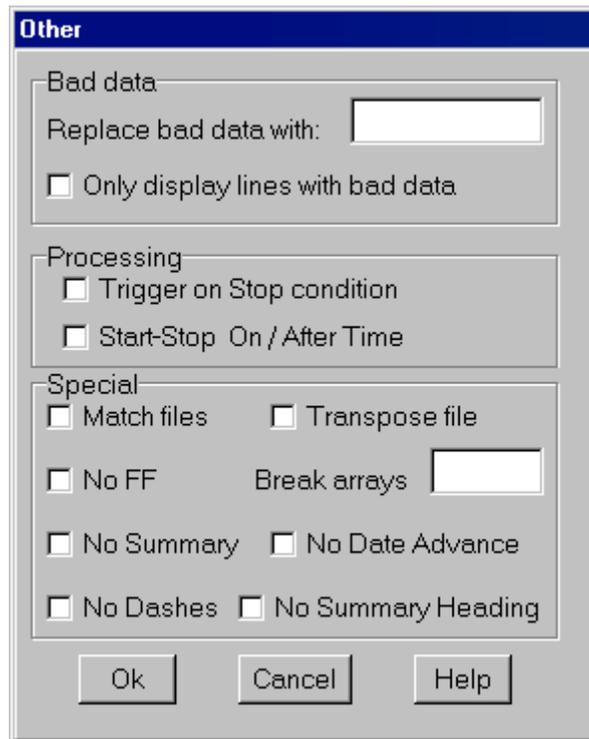


FIGURE 8.2-13. Other Output File Options

It allows the following settings to be modified:

**Replace bad data with** - The text in the field, to the right of this option, is entered into the .PRN output file data set if data are blank, bad, or out of range. See Table 8.2-9 for definition of blank or bad data. Whatever text string the user enters in the field will be entered if a blank or question mark is in the data or if data are out of range. This option is useful when the Output file is imported into a spreadsheet program, such as Excel.

**TABLE 8.2-9. Definition of Blank or Bad Data for each Data File Format**

<b>File Format</b>	<b>Definition of Blank or Bad Data</b>
Printable ASCII	????
Comma Separated ASCII	blank or any character except numeral or space
Field Formatted	blank or "" (double quotation marks)

**Only display lines with bad data** - Outputs only those arrays containing one or more Out of Range elements. If a report is generated, an asterisk precedes the Out of Range value in the .RPT file.

**Trigger on Stop condition** - Changes the meaning of Stop Condition to trigger Time Series processing output. The Stop Condition is included in the Time Series processing if it satisfies the Copy line.

If the Trigger on Stop Condition is selected, a Time Series output will occur each time the Stop Condition is met.

**Start-Stop On/After Time** - In most instances, Split will not start or stop processing a file unless the exact start condition is found. However, when starting or stopping based on time, you can enable Split's **Start-Stop On/After Time** option. This will trigger the start (or end) of processing when the exact time is found or at the first instance of data after that time has occurred (which meets other defined criteria in the PAR file).

**Match files** - This option compares two files of the same data. If good data exists in one and not the other (question marks), then Split will fill the OUTPUT file with the good data. This is used to get a more complete record from an error ridden file (e.g., one recorded at freezing temperatures by reading a tape twice and running both files through Split).

---

**CAUTION**

For the Match files option to produce a correct Output File, the differences between the two Input Files can only be question marks. Both files must have the same Start Condition or the beginning of both files must be the same.

---

**Transpose file** - Transposes the rows and columns of the input file. Only one Input File can be transposed at a time and no Select options can be specified. A maximum of 26 arrays are transposed per pass of Split.

To transpose a file containing more than 26 arrays, several passes are required. Change the Output file name and Start Condition for each pass. Split may then be used to merge the multiple files.

**No FF** – Suppresses form feeds and page breaks in RPT and HTML files. When this option is selected, a header appears on the first page only. This option is used for printing reports on continuous feed paper or for displaying HTML files in a browser.

**Break arrays** - This option breaks up the Output Array into new arrays that are #+1 elements in each new array. Split automatically assigns an array ID number equal to the first element in the first array. Only one Input File may be specified. Start, Stop, and Copy Conditions may be specified, but the Select line must be left blank.

**NOTE**

The Break Arrays function works only for mixed array data. It is typically used when processing data from burst measurements.

**No Summary** - When producing reports that include time series processing based on an interval, sometimes that interval will not divide evenly into the number of lines in the data file that is being processed. For example, you may be processing one-minute data on a five-minute interval, and the data file has 103 lines; thus, there are 3 lines of data "left over" at the end of the report. By default, the summary (average, total, maximum, etc., depending upon which time series function is being used) of the left over values is printed at the bottom of the report following the Time Series Heading. Enable the No Summary check box to omit the summary of the left over values and the Time Series Heading from the report.

**No Date Advance** - When processing a data file from a mixed array datalogger, if the time stamp uses midnight as 2400 with "today's" date, the date function will convert that time stamp to 0000 hours with "tomorrow's" date. (This is because the algorithm used by the date function is based on Windows' time format, and it does not support a 2400 time stamp.) For example:

Array ID	Year	Julian Day	Hour/Minute	Date Function	Data	Data
10	2002	151	2200	05/31/02 22:00	1.701	193.6
10	2002	151	2300	05/31/02 23:00	1.476	31.99
10	2002	151	2400	06/01/02 00:00	1.123	106.2

At Julian Day 151 (May 31) 2400 hours, the date function produces an output of June 1 00:00 hours. The date can be stopped from rolling forward by using the No Date Advance check box. The output will then be similar to:

Array ID	Year	Julian Day	Hour/Minute	Date Function	Data	Data
10	2002	151	2200	05/31/02 22:00	1.701	193.6
10	2002	151	2300	05/31/02 23:00	1.476	31.99
10	2002	151	2400	05/31/02 00:00	1.123	106.2

Caution should be used when applying the date function and enabling or disabling No Date Advance, since it is possible to produce an incorrect date. For instance, using the above example if you were to enter the following into your select line:

```
3,edate("hh:mm";4;3;2)
```

with the No Date Advance enabled, you would get the output:

151	22:00	1.701	193.6
151	23:00	1.476	31.99
151	00:00	1.123	106.2

If you were to enter:

```
edate("mm/dd/yy";4;3;2),4,6,7
```

with the No Date Advance disabled, you would get the output:

05/31/02	2200	1.701	193.6
05/31/02	2300	1.476	31.99
06/01/02	2400	1.123	106.2

**No Dashes** - When the **No Dashes** check box is selected, the dashed line that typically appears under the column headings will not be displayed. This option affects all output types (PRN, RPT, HTM, and printed page).

**No Summary Heading** - When processing data using time series functions (see No Summary, above), select this option to prevent the Time Series Heading and Column Headings from being printed at the bottom of the report. The "left over" summary data will still be printed.

### 8.2.3.2.2 Report Headings

A report is output to a printer or file with the extension .RPT. Headings are not included in the standard output to disk (.PRN or user named extension output file). However, a report can be labeled with a header by entering text into the Report Heading field. A report heading can have several lines, but it is limited to a total of 253 characters including backslashes and carriage returns. “\” characters break the report heading into multiple lines.

When Time Series functions are used in the Select field without an interval, they appear as a final summary at the end of the report. They can be labeled by entering a title into the Time Series Heading field at the bottom of the Output File page. Time Series interval summaries cannot be assigned individual titles directly, but you can use special functions such as "Label" and "CrLf" to create column headings and special formatting.

“PCDATE” within the Report Heading inserts the computer's current date (Month-Day-Year). For the European format (Day-Month-Year), enter “PCEDATE”.

### 8.2.3.2.3 Column Headings

Up to three lines per column can be entered as column headings. These headings are limited to a length of one less than the Output field width.

Column headings associated with Time Series outputs are repeated for Final Summaries if a title for the Final Summary is requested on the headings for report line.

The number of digits to report to the right of the decimal point is entered in the Decimal field and can be set independently for each column. The value output will be rounded to the specified number of digits. Leave this field blank if you do not want to round the data to a specific number of digits.

Column headings can be entered using Split's Data Labels Function (Labels | Use Data Labels).

## 8.2.4 Help Option

On-line Help is available from any location in Split. Simply select the area of Split in question and press <F1>. Split also offers a brief on-the-fly Help. Place the cursor on the area of Split in question; after a moment a brief description is displayed in the hint line of the Split window (bottom left).

## 8.2.5 Editing Commands

Split supports the Windows Cut, Copy, and Paste commands. Text from any field in Split or other Windows applications can be Cut, Copied, or Pasted.

## 8.2.6 Running Split From a Command Line

Existing parameter files can be executed using Splitr.exe which is a "run-time" version of the Split Report Generator. When Splitr.exe is run, the file is processed as if the user chose Run | Go from the Split menu. Splitr.exe can be executed from the Windows scheduler, from a batch file, or from a Windows command line prompt or shortcut.

### 8.2.6.1 Processing Alternate Files

Splitr allows the user to select different input and/or output files for an existing parameter file by entering them on the command line after the parameter file name. For example:

```
"Splitr LOGAN.PAR/R TEST.DAT TEST.PRN"
```

Replaces the Input and Output file names in LOGAN.PAR, with TEST.DAT and TEST.PRN, respectively.

A space must be used to separate command line parameters. Splitr uses as many entries as exist on the command line. However, the command line has a limit to the number of characters it can accommodate—this limit is operating system dependent. The parameters must be in the following sequence: Input file name, Output file name, Start Condition, Stop Condition, Copy Condition,

and Select. A comma space comma should be inserted for any parameter not specified.

If a parameter is to be left as it is in the parameter file, then space comma space ( , ) may be entered in the command line. For instance, if the parameter file LOGAN.PAR contained TEST1.DAT as an input file name, the following command line would leave the input file TEST1.DAT and change the output file to TEST.PRN.

```
“SPLITR LOGAN/R , TEST.PRN”
```

### 8.2.6.2 Processing Multiple Parameter Files with One Command Line

More than one .PAR file can be executed with a single Splitr command line. Each .PAR file and its associated parameters are separated from the next .PAR file by a semicolon with one space on each side ( ; ). For example:

```
“SPLITR LOGAN/R TEST.DAT TEST.PRN ; SINKS/R TEST1.DAT  
TEST2.DAT 1[189]”
```

executes the LOGAN.PAR file on TEST.DAT and outputs the results to TEST.PRN, then executes the SINKS.PAR file on TEST1.DAT and outputs the results to TEST2.DAT. Execution of SINKS.PAR starts when the first element in TEST1.DAT is 189.

### 8.2.6.3 Using Splitr.exe in Batch Files

Batch files containing one or more Splitr command lines can be useful for automating data processing. Batch files can be executed manually or by setting them up in the Windows scheduler.

Batch files process each command in succession, without waiting for execution of a command to be completed before proceeding to the next unless they are configured to do so. If multiple parameter files are being processed using Splitr in a batch file, there are no conflicts because only one copy of Splitr can be active at any one time (unless the /M switch is used. Refer to the information below on this option.). However, if other commands are used along with Splitr (such as opening the file in a spreadsheet, copying it to an archive directory, or appending it to an existing file) these commands might be executed before Splitr finishes processing data.

The Windows 95/NT Start /w (wait) parameter is added to a batch file command line to delay execution of the next command until the first command has finished. The syntax for this type of command line is:

```
Start /w Splitr LOGAN/R
```

By adding the Start /w parameter to the command line that initiates Splitr, no further commands will be processed until the LOGAN.PAR file has finished execution.

### 8.2.6.4 Command Line Switches

Splitr has four switches that can be used to control how files are processed.

#### **8.2.6.4.1 Closing the Splitr.exe Program After Execution (/R or /Q Switch)**

Typically when Split is run, after the file is processed the user must close the Screen Display window. When Splitr.exe is run from a command line, the user must also close the Screen Display window unless the /R switch is used.

The syntax for this switch is:

```
SPLITR LOGAN/R
```

where LOGAN is the parameter file name.

The /R switch should follow immediately after the parameter file name with no space between the two. If a space is used, the following message will be displayed "There was a problem opening the input file. File could not be found or may be in use."

The /Q switch is similar in function and syntax to /R. However, if Split encounters an error when processing the file, no message box is displayed that requires user response (the exceptions are a disk space error or an internal error with the Split executable). This option should be used with caution, since there will be no indication of a problem if a file cannot be processed.

#### **8.2.6.4.2 Running Splitr in a Hidden or Minimized State (/H Switch)**

Splitr can be run in a minimized state, so that the Screen Display window does not interrupt other processes on the computer. The syntax for running Splitr minimized is:

```
SPLITR /H LOGAN
```

where LOGAN is the parameter file name.

The /H switch must be positioned after SPLITR but before the parameter file name, and a space is required between the executable name and the switch.

#### **8.2.6.4.3 Running Multiple Copies of Splitr (/M Switch)**

Multiple copies of Splitr can be run at one time by using the /M switch. This switch must appear immediately after Splitr. For instance, a batch file containing the lines:

```
SPLITR /M Logan/R  
SPLITR /M Sinks/R
```

will open two copies of Splitr and process the two files simultaneously.

---

**NOTE**

When using the /M switch in a batch file under Windows NT, 2000, or XP, you must begin each line of the batch file that runs an instance of Splitr with the "start" command. Otherwise, NT will wait until the first Splitr command has finished before proceeding to the next.

---

# Section 9. Automating Tasks with Task Master

---

*The Task Master is an application that is used to set up a Task that can be run on a defined schedule or based upon a data collection event from a datalogger. A Task can be data collection from another datalogger, or anything that can be executed in a computing environment, such as a command line operation, a program executable, a batch file, or a script.*

*The Task Master is often used to post-process data files using Split after data has been collected from a station. It can also be used to launch third party software utilities such as a command line FTP client to send data to the Internet or a phone dialer to call a pager or phone upon an alarm condition.*

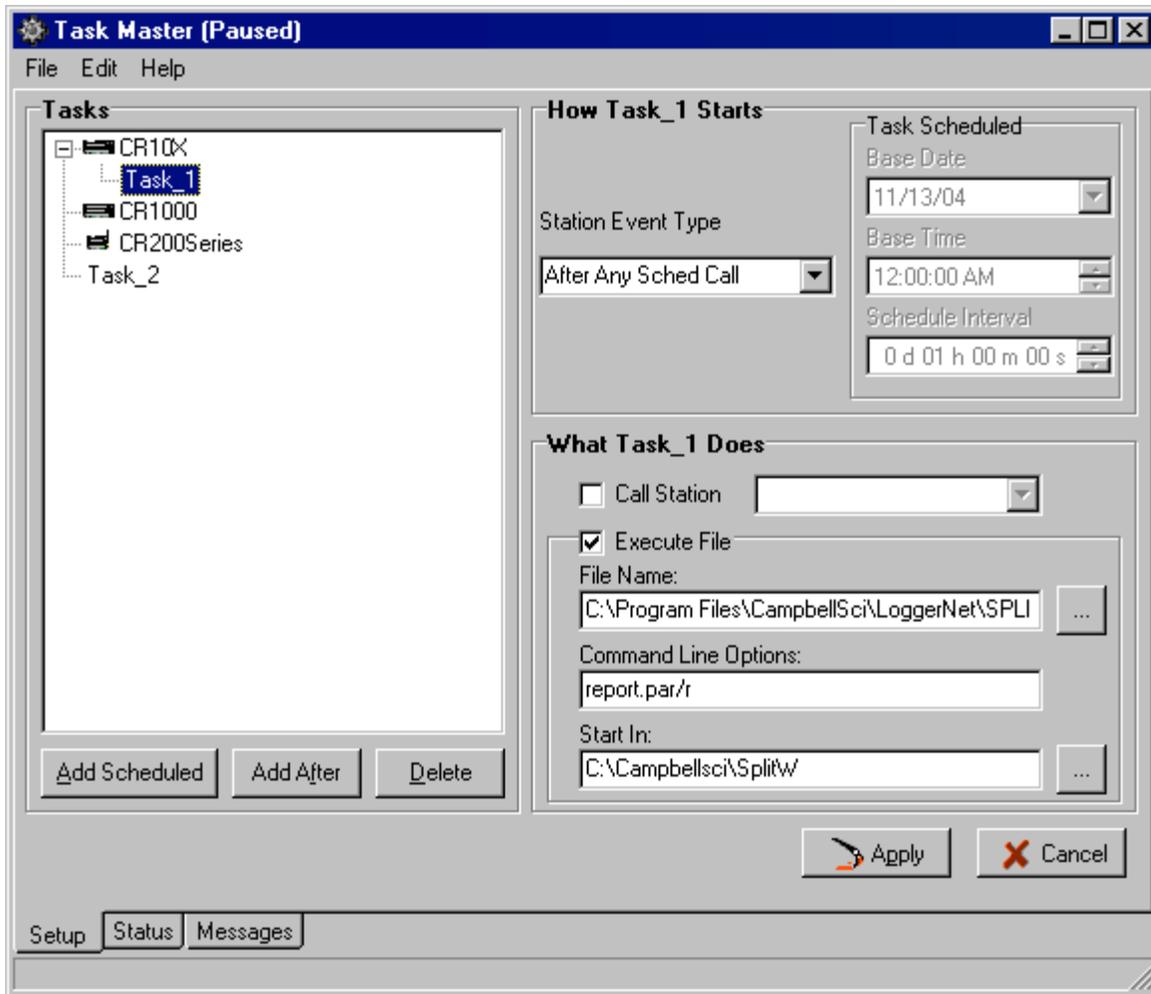
## 9.1 Task Master

The Task Master sets up and manages the optional user-defined tasks associated with data collection. Tasks can be set up to trigger data collection or execute batch file scripts or programs based on a variety of data collection events. Tasks may also be scheduled based on time intervals.

The Task Master can be opened from the LoggerNet Toolbar's File | Open menu. By default, a button for the Task Master is not placed on the Toolbar; however, one can be added by selecting Options | Configure Buttons and selecting the check box for Tasks.

When the Task Master is opened, all of the dataloggers in the network, and any tasks that may have already been defined, are displayed on the left side of the window.

There are two types of Tasks that can be created: a Scheduled task and an Add After task. If a task is shown attached to a datalogger as Task\_1 in the example below, the task execution will be based on a datalogger collection event. Task\_2 is not linked with a datalogger and will run as a scheduled event.



## 9.1.1 Setup Tab

### 9.1.1.1 Adding Tasks

To add a task that will run based on a data collection event for a datalogger, select the datalogger by clicking it. Then click the Add After button or select Add After from the Edit menu. A new task will appear attached to the selected datalogger. You can then set up the conditions for the task with the options on the right side of the window.

You can create complex combinations of tasks by linking tasks to other tasks or multiple tasks to one datalogger. A task linked to another task has no start options but will execute the specified action following the completion of the parent task. Multiple tasks linked to a datalogger will execute based on the conditions specified for the start of the task. This allows one task to be run after successful data collection and another if data collection fails.

To add a scheduled task, click the Add button or select Add from the Edit menu. A new task will be added to the list of tasks below the list of dataloggers. You can then set up the conditions for the task with the options on the right side of the window.

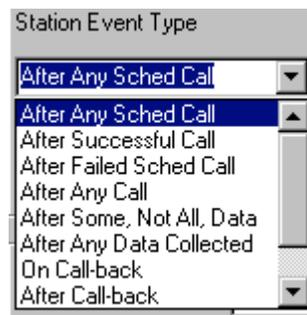
To delete a task click to highlight it and click Delete or select Delete from the Edit menu. The selected task will be deleted. If there are any tasks linked to the task, they will move up and take the place of the deleted task.

Tasks can be renamed by selecting the task and then clicking again on the task name. The name will turn into a text edit box and you can create your own task name.

There is also a right click menu that will allow the same Add, Add After, and Delete functions as described above.

### 9.1.1.2 Logger Event Tasks

There are 10 data collection events that can be selected to trigger a task linked to a datalogger. Clicking the drop down list button to the right of the event brings up a list of these events as shown in the screen shot below. These events allow flexibility in deciding when a linked task should be run.

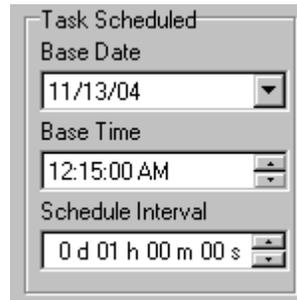


- **After Any Scheduled Call** – The task will run each time scheduled data collection is attempted. This will run whether or not the attempt succeeded or data was collected.
- **After Successful Call** – The task will run each time the datalogger is successfully contacted. This could be for data collection or any other communication with the datalogger.
- **After Failed Scheduled Call** – The task will run only when a scheduled data collection fails to communicate with the datalogger. This is useful for building in an alert or notification utility to warn the user of the failure.
- **After Any Call** – The task is run any time LoggerNet attempts to contact the datalogger. This will run even if the datalogger does not respond or no data is collected.
- **After Some, Not All, Data Collected** – The task will run after data collection from the datalogger has started but for some reason did not finish. This can be used for detecting problems where data starts to come in and communication fails before all the data is collected.
- **After Any Data Collected** - The task will run after data is collected from the datalogger, whether the data collection finished successfully or not.

- **On Call-back** – The task will run when a datalogger initiated call is received by LoggerNet.
- **After Call-back** – The task will run after LoggerNet has processed the datalogger call-back.
- **After Failed Retry** - The task is run whenever a retry fails. This can be the failure of a primary retry or a secondary retry.
- **After Primary Retries Failed** - The task is run after the specified number of primary retries has been exhausted.

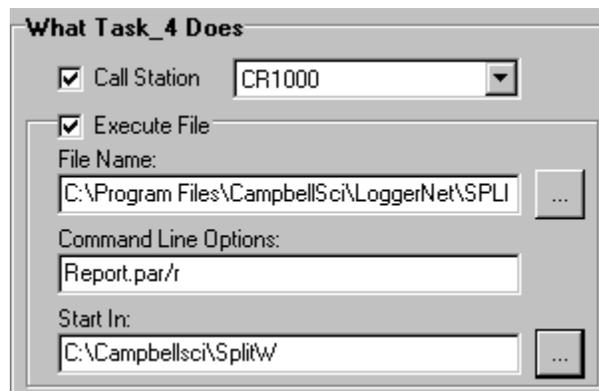
### 9.1.1.3 Scheduled Event Tasks

An alternative to event driven tasks, scheduled tasks are repeated at a specified interval. The Base Date and Time are used to set the initial date and time for the task execution. The interval specifies the time between task executions. In the example shown the task will start on November 13, 2004 and run once an hour at 15 minutes past the hour.



### 9.1.1.4 Define What the Task Does

The action to be started when the task is triggered is specified in the lower right hand panel of the task screen as shown below. The two options are to initiate a call to a datalogger and execute a file. You can choose either or both options by clicking the check box next to the option name.



**Call Station** will attempt to collect data from the selected datalogger when the task is triggered. Data will be collected according to settings in the Setup



**Task Name** - The name that was given to the task when it was set up.

**Event Type** - This column indicates what type of event will trigger the task. The event types are listed above (Station Event Types).

**Event Trigger** - If the Event Type is a Scheduled call, the schedule for the task will be listed in the format DD HH:MM:SS followed by the word "Interval". For any other type of call, the device which the task is dependent upon will be listed.

**Last Time Run** - The last time that the task was run by LoggerNet.

**Next Time to Run** - The next time that the task is scheduled to be run. If the Event Type is anything other than a Scheduled call, this field is not applicable and will be left blank. If Pause Tasks is selected, this field will read Paused.

In the example above, Task\_1 will be run after a successful with the CR10X station. Task\_2 will be run if a calls fails with the CR10X station. Task\_3 is a scheduled task, run on a 1 hour interval. The next time the task will run is 11:15 a.m. on November 13, 2004.

To temporarily stop all tasks from running, select the Pause All Tasks checkbox. This will stop all tasks until the check box is cleared. Tasks can also be Paused/Resumed, from the LoggerNet Toolbar's Options menu. When Tasks Enabled has a check mark to the side of it, the tasks will be run. When it does not, the tasks are paused. This menu option is toggled on/off by selecting it.

A task can be triggered to run even if its trigger condition has not occurred. Highlight a task from the list, and select the Run Selected Task button. This is a good way to ensure that your task will run correctly before enabling the Task Master.

### 9.1.3 Messages Tab

This Messages window shows messages from the LoggerNet server about the activity of tasks. It will show information when a task is set up or when a task is executed. To clear the messages window, select Edit | Clear Messages.

Task messages can also be saved to LoggerNet's log files by selecting Edit | Log Messages. The Task messages will be interspersed with other server messages in the Trans.log, Comms.log, and State.log files.

# Section 10. Storage Module Software (SMS)

---

*SMS is a software application that provides a simple and efficient way of collecting data from and storing programs to Card Storage Modules (CSM1 and MCR1), Storage Modules (SM192, SM716, SM4M, and SM16M) or a PC (PCMCIA) Card.*

*The software communicates with Storage Modules through an SC532 asynchronous adapter or through a datalogger (mixed array operating system only). The software communicates with PC cards using either Campbell Scientific's Card Storage Modules (CSM1 or MCR1) or the PC card slots installed in the computer. The software is provided with an extensive online help system.*

## 10.1 Overview

SMS is opened from the LoggerNet Toolbar's File | Open menu. By default, the button which launches SMS is not included Toolbar. However, it can be added by from the Options | Configure Buttons menu.

SMS provides easy communication with Campbell Scientific's external Storage Modules, card readers and dataloggers for collection and storage of both programs and data. Data can also be read directly from a PC card if the computer is fitted with a PC card slot and the PC operating system supports 32-bit VXD's.

---

### NOTE

The CSM1 Storage Module, the MCR1 Card Reader and a computer fitted with an internal PC Card reader all use SRAM PCMCIA cards which must be formatted to Campbell Scientific's specifications.

Windows NT, Windows 2000, and Windows XP do not support Microsoft's 32-bit Card Service VXD's. Therefore, SMS cannot communicate with PCMCIA card slots running under Windows NT, 2000 or XP.

---

There is extensive online help available either by pressing the F1 key or accessing help from the menu. The online help along with the user guide for the module you are using, will help you get the most out of your storage modules.

---

### NOTE

The CSM1, MCR1 and SM192/716 User Guides contain references to other Campbell Scientific software (CSMCOM and SMCOM). These programs are *not* required if you are using SMS.

---

## 10.2 Getting Started

When you start SMS the main window appears with tabbed configuration screens that give you access to the options you need for the type of storage module you have. An example screen for the SM192/716 module is shown.

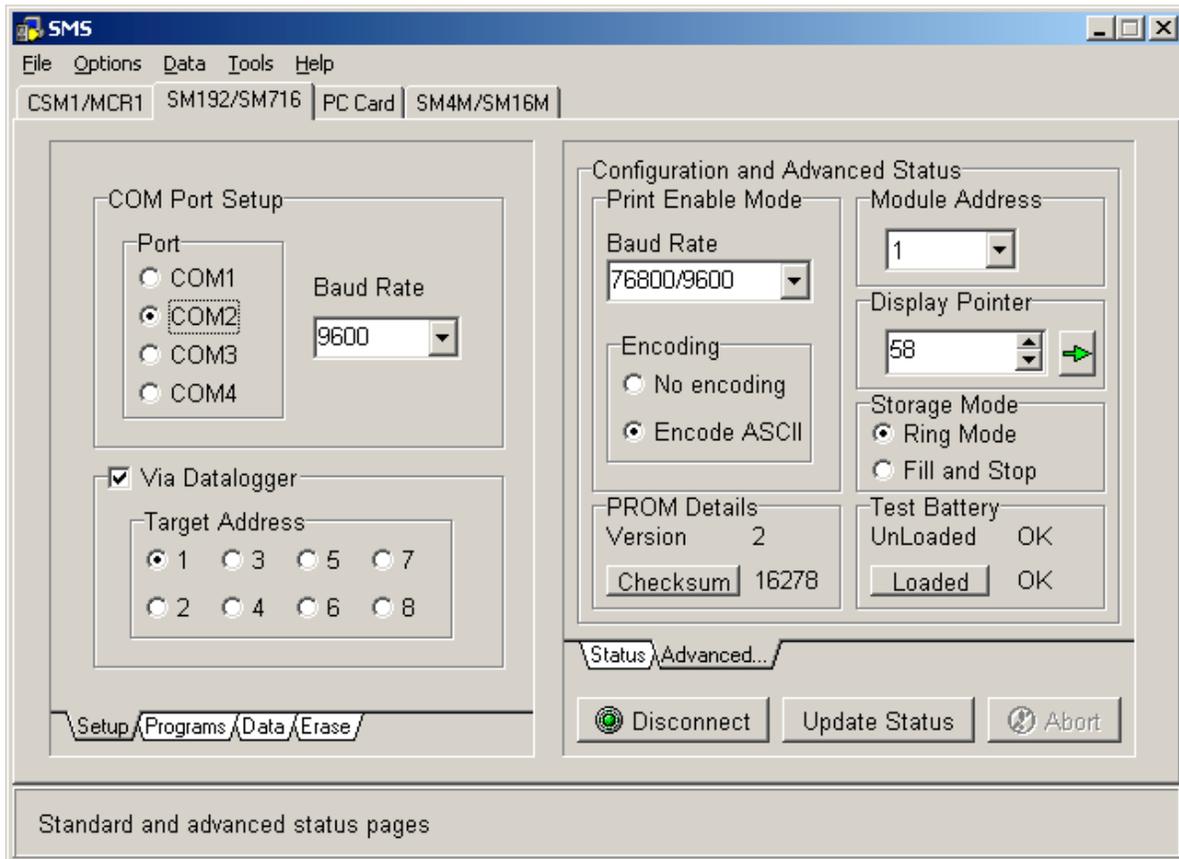


FIGURE 10.2-1. Typical SMS Screen

CSM1, SM192, SM176, SM4M, or SM16M Storage Modules usually require an SC532 interface, plus a 7026(standard serial) or SC25PS cable. A basic connection diagram is shown in Figure 10.2-2 below. If you are using an SC532A interface you will need a 9 pin to 9 pin serial cable instead of the 9 to 25. The Storage Module manuals give full details of connecting the modules to the computer.

---

**NOTE** The MCR1 Memory Card Reader does not require an SC532 interface.

---

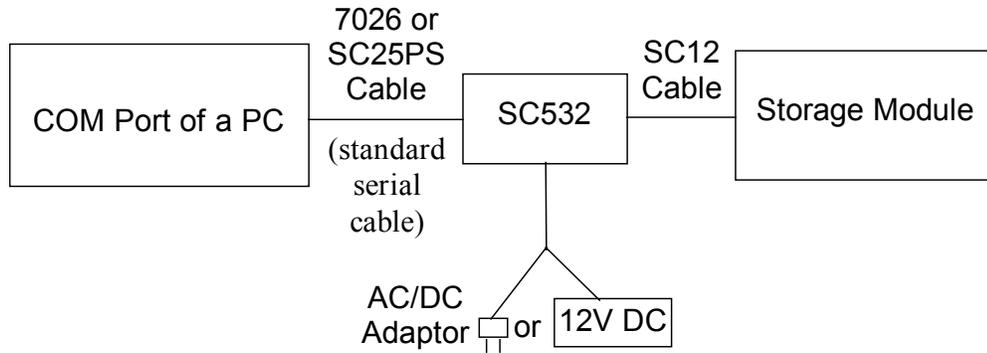


FIGURE 10.2-2. Connecting the Storage Module to a PC using an SC532 Interface

If your computer is fitted with a PC card reader supporting Card Services protocol, SMS can be used to interrogate the memory card in a similar way to interrogating a card in the CSM1/MCR1 Storage Module.

Start SMS by clicking the Stg Module button of the LoggerNet toolbar. The first time you use the program it will show the screen for the CSM1/MCR1 Card Storage Module as in the screen in Figure 13-3.

## 10.2.1 The Setup Screen

The setup screen shown in Figure 10.2-3 is used to set the Communications (COM) Port and Baud Rate before communications between the module and computer are established.

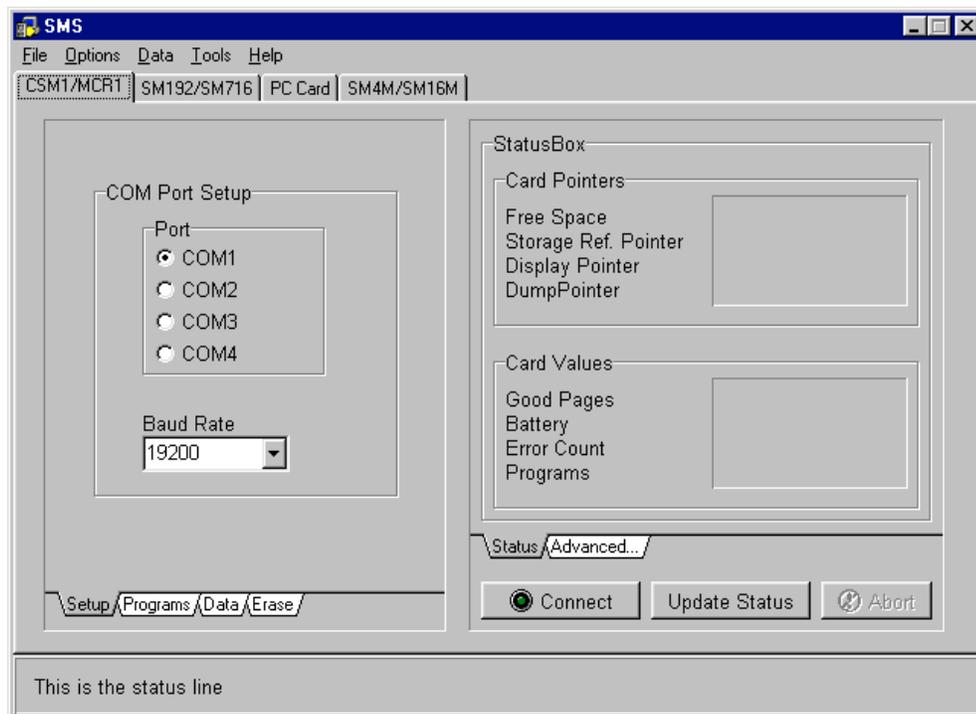


FIGURE 10.2-3. Typical SETUP Screen and StatusBox for the CSM1/MCR1 Module

## 10.2.2 Establishing Communications

For storage modules, the first step is to establish communications between your storage device and the computer. This involves selecting the correct communication port and baud rate for the device being used from the Setup screen.

If you have a computer fitted with a correctly configured card slot and your operating system is Win95/98/ME the PC Card screen will automatically show the available card slots you can use.

### 10.2.2.1 Selecting a COM Port

This section is applicable for Campbell Scientific's external Solid State Storage Modules such as the SM192/716 and SM4M/16M.

For an externally connected storage device you can use one of your computer's Communication (COM) ports for connecting the storage module to the computer. If you are not using COM1 for connection, you will need to select the port being used by clicking the appropriate check box in the Port Selection box. This will be COM1 or COM2 for most computers, but SMS can use COM3 and COM4 if your computer is equipped with these.

The physical implementation of the COM port is hidden from the software. The Windows environment provides a common interface via a COM port driver, with which SMS communicates. Sometimes, however, you might have a COM port which shares interrupts with other COM ports. This can be checked by examining the "Ports" (Settings/Advanced) setting in the Windows Control Panel. A COM port sharing an interrupt with another device can prevent Windows from performing high-speed communications using that port. This shared interrupt problem would, therefore, also affect operation of SMS.

---

**NOTE** If you appear to be experiencing communication problems when using external modules, always check the COM ports setting in the Windows Control Panel, using the "Ports" (Settings/Advanced) option, and check for any shared interrupts.

---

### 10.2.2.2 Baud Rate

The first time you use SMS with a storage module a default baud rate of 38,400bps is automatically selected for the CSM1/MCR1 and a baud rate of 19,200bps is set for the SM192/716 modules connected using an SC532. If you are connecting to a storage module via a datalogger, the baud rate will vary depending upon the datalogger chosen.

The default (maximum) baud rate (38,400bps for CSM1/MCR1 and 19,200bps for SM192/716) can be adjusted to your own requirements, depending on, among other things, the speed of your COM port and processor speed.

When SMS reads data from the storage device, it automatically performs a checksum test and requests that bad data be re-transmitted. If bad data is detected, the computer will beep.

If communication errors occur, it may help to reduce the baud rate. Possible reasons why good communications cannot be established using higher baud rates under Windows are:

- Using low-quality cards, or using generic drivers rather than the hardware specific drivers.
- Processor is loaded down by other applications (e.g., DOS programs set to exclusive mode or other operating Windows communication programs).
- A computer without a buffered UART (Universal Asynchronous Receive/Transmit) serial port (e.g., a 16550 UART), or without the FIFO (First In First Out) option enabled. (Other communications problems can occur if the computer does not actually have a buffered UART, but is using the Windows 95 default configuration with the FIFO option enabled.)
- A high level of communications activity on another port.
- Lack of computer memory, causing the computer to do a considerable amount of hard-disk paging. If the SMS communication routines are paged to disk and take too long to load back into memory, the buffer can overflow.

If the communication errors are reported as CE\_RXOVER errors, reducing the baud rate will help because it stops the buffer from filling up too quickly in situations where SMS is not able to respond to the incoming data.

### 10.2.2.3 Via Datalogger/Target Address (SM192/716 and SM4M/SM16M Modules Only)

When an SM192/716 or SM4M/SM16M Storage Module is connected with an SC12 cable to a datalogger which supports direct communications (CR10(X), CR500, CR510, or CR23X), you can establish communications using the “Via Datalogger” mode as explained below.

#### *Via Datalogger*

As an example, when an SM192/716 module is connected to a CR10(X), CR23X, CR500 or CR510 datalogger with an SC12 cable, direct communications can be established between the computer and the storage module (see Figure 10.2-4). Note, however, that the CR10(X), CR500, and CR510 dataloggers can only support a maximum baud rate of 9600.

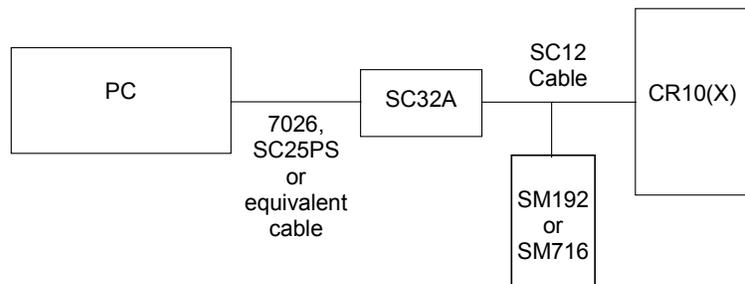


FIGURE 10.2-4. Connecting the SM192/716 to a PC and a CR10(X) Datalogger

SMS can automatically detect if a datalogger that supports direct communication is present and request that a connection be made to the Storage Module. In these circumstances the Via Datalogger check box can be used for connection. This sets the baud rate to 9600 and enables the Target Address check box. If this Via Datalogger is not selected and you attempt to connect to a storage module, SMS will detect the datalogger's presence and adjust the baud rate accordingly.

### **Target Address**

The Target Address selector is disabled unless you check the Via Datalogger box, because this is the only mode that simultaneously supports multiple storage modules.

An appropriate datalogger can be connected to up to eight storage modules simultaneously, each having a different address. The address selector is used to specify the address of the module to which you wish to connect. Note, however, that any module having a high address may answer to a call of a lower address if no module of the lower address is present.

For example, if only one storage module with an address of 8 is present, and you set the target address to 1, you will still be able to connect to the storage module. However, if your module had an address of 1 and your target address was set to 5, you could *not* connect.

The above shows that the target address need not represent the *actual* address of the Storage Module. The physical address of the Module is shown on the "Advanced" status page. Changing the physical module address will not change the selected target address, and, in fact, changing the physical address will not have any noticeable effect until you disconnect from the Storage Module.

---

#### **NOTE**

While up to eight SM4M or SM16M storage modules can be addressed, communications may be unreliable if more than four are connected to a datalogger at a time.

---

### **10.2.2.4 Connect/Disconnect**

After connecting your storage module to your computer and selecting the required COM port and baud rate, click the **Connect** button to establish communication between the computer and the storage module. If everything is satisfactory, a successful communications link will be established and the wording on the **Connect** button will change to **Disconnect**.

If your first attempt at connection fails and an error message "Cannot connect to CSM1" or "Communications Problems:..." appears, try to connect again (by clicking the **Connect** button). It may take some time (up to 30 seconds) to establish the connection. The most likely cause of a failed communication link is selection of the wrong COM port or a loose physical connection between the storage module and the computer.

## 10.3 Status Information

### 10.3.1 Status Box/Update Status

When successful communication has been established, a set of values will appear in the StatusBox at the right-hand-side of the screen. This screen shows the current status (values) for the card or module. Some typical values are shown in Figures 13-1 and 13-3 above.

The information in the main StatusBox is relatively straightforward and is similar for all modules. The SM192/716 and SM4M/SM16M modules have an additional Switch Settings information box that shows the settings made using the **ADVANCED SETTING** screen. Further information on each item in the **STATUS** screen can be obtained by clicking that item and pressing F1. Additional information is also available in the appropriate Storage Module Manual.

The status information can be updated at any time by clicking the **Update Status** button.

### 10.3.2 Advanced Status Information

The **ADVANCED STATUS** screen is different for each type of module, and provides extra information and configuration facilities.

#### 10.3.2.1 CSM1/MCR1

##### *Display Pointer*

The Display Location Pointer (DLP) indicates the location which holds the first value that will be read when extracting data from the card. The pointer can be positioned anywhere within the readable area of the card, either by typing the location directly into the window or by using the up/down arrows to increment the value. The green arrow moves the pointer to successive filemarks.

---

##### **NOTE**

If the Display Pointer points to a filemark or the start of a program, no data will be output in response to the "Get One" request. It must be moved past the filemark to get further data.

---

Further details on Filemarks and Pointers can be found in the relevant Storage Module Manual and the on-line help system.

##### *Card Format Type*

This is a format code for the card currently in use. It is an internal reference for use by Campbell Scientific and so has no significance for the end-user.

### **PROM Details**

The PROM checksum calculates a unique signature for that PROM version. For all storage devices the PROM checksum is displayed on the **ADVANCED** screen. At start-up the value may be displayed as a series of ??????. The true value can be shown by clicking the **Re-Calculate Checksum** button. Once displayed, this value is retained and will normally remain until you disconnect or remove power from the storage module. You can recheck the value at any time by clicking the **Re-Calculate Checksum** button.

### **10.3.2.2 SM192/SM716**

The SM192/716 **ADVANCED STATUS** screen is similar to the **CSM1** screen, but has additional configuration facilities. These settings configure the storage module, which has its own process that controls datalogger communications. These settings don't affect getting data out to the PC. These settings are listed below.

### **Print Enable Mode**

Data from 21X and CR7 dataloggers with older PROMs, and data from non-datalogger devices are stored in the Storage Module using the "Print Enable" mode. Full details of this storage method can be found in the SM192/716 Instruction Manual.

### **Baud Rate – Print Enable Mode**

This is the printer baud rate when data are collected by the Printer Enable (PE) method. For this method of collection you must ensure that the baud rate of the Storage Module and the datalogger are matched. Most Campbell Scientific dataloggers have selectable baud rates. You can select the baud rate using the Baud Rate setting box in the Print Enable Mode section. The SM192/716 Instruction Manual gives further details.

---

### **NOTE**

The 76,800/9600 setting is the default after an SM reset. In this setting the SM can automatically change from capturing data at 76,800 baud to capturing data at 9600 baud. Baud rate errors (baud rate mismatches between SM and datalogger) cause the SM to increment its error counter and enter a low power standby state.

---

### **Encoding**

These check boxes allow you to select how data are encoded in the Print Enable method of data storage:

*Encode ASCII* is the normal default setting. The Storage Module will encode incoming Printer Enable Method printable ASCII data and store it as Campbell Scientific Final Storage Format (binary) data.

*No Encoding* enables incoming data to be stored "as is", regardless of its format. This permits storage of raw ASCII data.

**NOTE**

---

Raw ASCII data will take up much more storage space than encoded data, and so it is generally recommended that data are always stored as binary.

---

**Module Address**

This shows the actual address of the Storage Module. Changing the actual address will *not* change the selected target address.

See Target Address for further details.

**Display Pointer**

This has exactly the same function as the Display Pointer for the CSM1/MCR1 (see above).

**Storage Mode**

Two methods of memory allocation for storage of data are provided – Ring Mode and Fill and Stop, as explained below.

**Ring Mode:** When the storage module is set to Ring Mode, data are stored sequentially. When all memory locations become full the ring memory “wraps around” and new data are then written over the oldest data stored in memory. To avoid data loss, retrieve all data from the storage module before this occurs.

**Fill and Stop:** When the storage module is set to Fill and Stop, data are stored sequentially, but, data storage stops when all memory locations become full.

When multiple storage modules are used, each should be assigned its own address and set to Fill and Stop. When one module is filled, data will then be stored in the module with the next highest address.

**Test Battery (Unloaded)**

The value shown under “Test Battery” is for the state of the battery when compared to a 2.5V DC reference without any load on the battery.

**Test Battery (Loaded)**

The battery can be checked when a 100 ohm load is connected for 1 second. This test shortens the overall battery life by approximately 3 minutes, but it is a much better test of the battery’s integrity. It is particularly important to carry out a Loaded Battery Test if the battery is more than four years old and you are intending to leave the Storage Module unattended to collect data for several months or you are leaving the module unattended to collect data in cold weather.

---

**CAUTION** Failure to check the battery can lead to data loss in the above conditions.

---

### 10.3.2.3 SM4M/SM16M

Many options on the SM4M/SM6M **ADVANCED STATUS** tab are similar to the other modules. Options unique to these two types of modules are explained below.

**Data Block Size** - The data block size is the physical size, in bytes, of flash memory blocks.

**Data Memory Size** - The data memory size is the total amount of memory, in bytes, for the storage module. Each low resolution data point uses 2 bytes; therefore, the SM4M can store 2 million data points and the SM16M can store 8 million data points (a high resolution data point uses 4 bytes and reduces the number of data points that can be stored in half).

**Max Program Size** - The Max Program Size is the amount of memory, in bytes, currently available for datalogger program storage.

**Program Area Size** - The Program Area Size is the total capacity that the module has for datalogger program storage.

## 10.4 Programs

Clicking the **PROGRAMS** tab brings up the **PROGRAM CONTROL OPTION** screen. The screen is very similar for the storage devices and the PC Card. It is essentially an interface for extracting programs, saving programs or clearing programs from the storage device.

The screen in Figure 10.4-1 shows the program selection check boxes and the program manipulation buttons. Datalogger programs can be stored in a card or module for subsequent downloading into a datalogger. The programs are stored in the same memory space available for data storage and are stored in up to eight memory locations. With the correct version of datalogger operating system, the program in location 8 can be automatically loaded, compiled and run in a datalogger on power-up of the datalogger. (See appropriate Storage Module Instruction Manual for more details.)

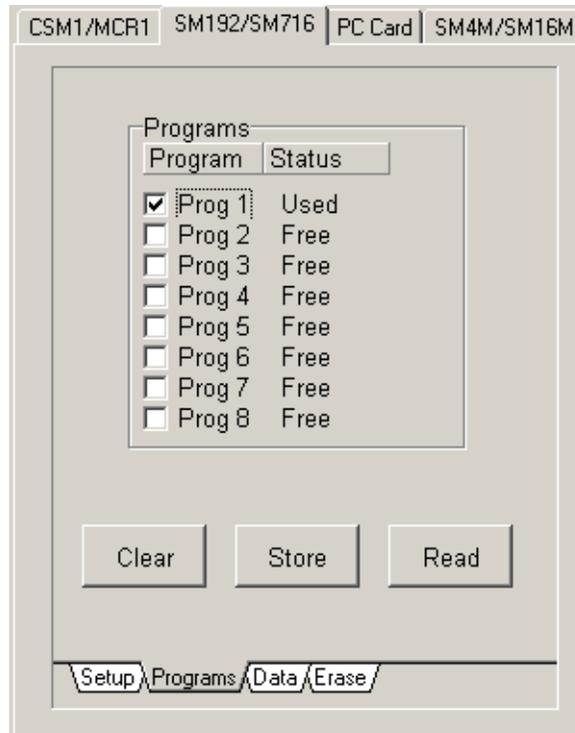


FIGURE 10.4-1. PROGRAMS Screen

### 10.4.1 Program Location

There are eight memory locations available. The Program Status box will either:

- Display a number indicating the starting position in memory of the program (CSM1/MCR1 and PC Card),
- Say “Used” for a program in a specific location (SM192/716 and SM4M/SM16M),
- Say “Free” for an unused location.

A program location is selected by clicking it to highlight its check box. Once you have selected the program location(s) you can carry out the desired action by clicking one of the three buttons.

### 10.4.2 Clear

Clicking the **Clear** button will clear the selected programs from the card or module memory. You will be asked to confirm this request before the program(s) are actually cleared.

### 10.4.3 Store

Clicking the **Store** button will allow you to store a program in the selected program location. If a program already exists in this location, you will be asked to confirm that it can be overwritten. Once a valid location has been selected, a Program Selection box appears showing any valid program files (.dld extension) on the current drive and directory. You can select any of these, or choose any other valid program file by browsing the directory structure. Once you have selected the required program, it will be written to the storage card or module. A Progress Indicator at the bottom right of the screen will confirm that the program is being written to the card or module.

### 10.4.4 Read

This is the reverse operation to the Store request. Clicking the **Read** button will allow you to read a program from the selected program location to disk. Once a location containing a program has been selected, clicking the **Read** button brings up a file selection menu for you to select the location in which to store the program on disk. A progress indicator will confirm that the program is being read from the card or module.

## 10.5 Data

Clicking the **DATA** tab will bring up the **DATA CONTROL OPTION** screen as shown in Figure 10.5-1, below.

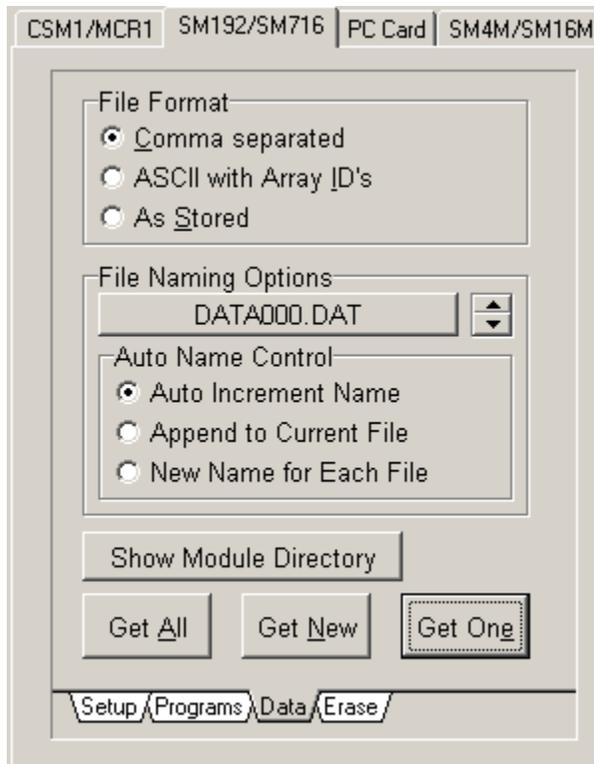


FIGURE 10-5.1. DATA CONTROL Screen

## 10.5.1 File Format

Before you attempt to “get” (collect) any data you should select your preferred file format.

### 10.5.1.1 Comma Separated

This format has array elements separated by commas. No element identifiers are included and all leading zeros and trailing zeros after the decimal point are removed. Each array is terminated with a carriage return and line feed.

Comma separated format is comparatively compact, can be read when printed, and can be imported into most data analysis packages (e.g., if you use a .CSV file extension, Microsoft Excel will auto-import the file).

An example is given below.

```
101,279,1413,12.68,24.75,75.3,25.42,75.1
101,279,1414,12.68,24.75,75.2,25.42,75.2
101,279,1415,12.68,24.77,75.4,25.49,75.1
101,279,1416,12.68,24.77,75.4,25.55,75.2
101,279,1417,12.68,24.78,75.4,25.42,75.2
101,279,1418,12.68,24.80,75.3,25.35,75.2
```

### 10.5.1.2 ASCII with Array IDs

This format is ASCII with Array Identifiers. Each data value is preceded by an identifier indicating its position in the array. This format requires the most disk space on the computer, and can be difficult to use in some data analysis packages.

An example of this format is:

```
01+0101. 02+0279. 03+1413. 04+12.68 05+24.75 06+075.3 07+25.42 08+075.1
01+0101. 02+0279. 03+1414. 04+12.68 05+24.75 06+075.2 07+25.42 08+075.2
01+0101. 02+0279. 03+1415. 04+12.68 05+24.77 06+075.4 07+25.49 08+075.1
01+0101. 02+0279. 03+1416. 04+12.68 05+24.77 06+075.4 07+25.55 08+075.2
01+0101. 02+0279. 03+1417. 04+12.68 05+24.78 06+075.4 07+25.42 08+075.2
01+0101. 02+0279. 03+1418. 04+12.68 05+24.80 06+075.3 07+25.35 08+075.2
```

This format is also ambiguous for arrays of 100 or more data values. It is therefore not recommended unless sending the raw data directly to a printer.

### 10.5.1.3 As Stored

When this option is used the software does not perform any decoding on the data in the card or module. Data are read out of the module and written to disk “as it is” (8-bit data).

If data has been stored from a datalogger in a binary format, this option writes the data to disk in the same format. This is the most compact format for data storage on disk, but the data needs to be decoded before it can be read. SPLIT (see Section 8.2) can be used to decode the data. If you wish to decode data with your own software, refer to the appropriate datalogger Operator’s Manual.

**NOTE** If the data was stored in the module in ASCII format, you should use the “As Stored” format to collect data from the module (see below). Storing data directly in ASCII format is inefficient in terms of use of the module memory, as printable ASCII requires five storage locations in the module (10 bytes) as compared to one (2 bytes) for binary data.

---

## 10.5.2 File Naming Options

Data files can be collected either individually or collectively. The File Naming Options box gives you control over how the extracted files are collected and named.

### 10.5.2.1 File name

Data files will normally be stored under a file name with the .DAT extension. When first using SMS, the button in the **FILE NAMING OPTION** screen will show the name DATA000.DAT. This name can be changed to any other name of your choice. By clicking the button, a standard Windows File Name box appears that will allow you to type in the name you want.

### 10.5.2.2 Auto Name Control

You may choose to Auto Increment the file name, to Append the data to a current file or to manually choose a New Name for each file saved.

#### ***Auto Increment Name***

When the software is first used, the program will always select a default file name and use automatic file incrementing. This means that the file name will initially be DATA000.DAT. Unless you change the default, further files will be saved in the sequence DATA001.DAT ... DATA999.DAT. This means that whenever you store new data files to disk, the name will automatically increment by adding one to the number at the end of the name.

If you are collecting multiple files (as in the Get All command), the file name will be incremented automatically as each file is collected. The program will scan the computer for files with the same base file name. If a file is found with a higher numeric suffix than the one you enter, the program will ask for confirmation to avoid the possibility of existing files being overwritten.

You may change the “base” file name by clicking the button showing the file name. The three-digit counter will be automatically added to the file name.

For example, if you chose a file name of ANDREW.DAT the program would convert it to ANDREW000.DAT so that auto incrementing could take place.

If you want completely unique names for each file, choose the “New name for each file” option.

### **Append to Current File**

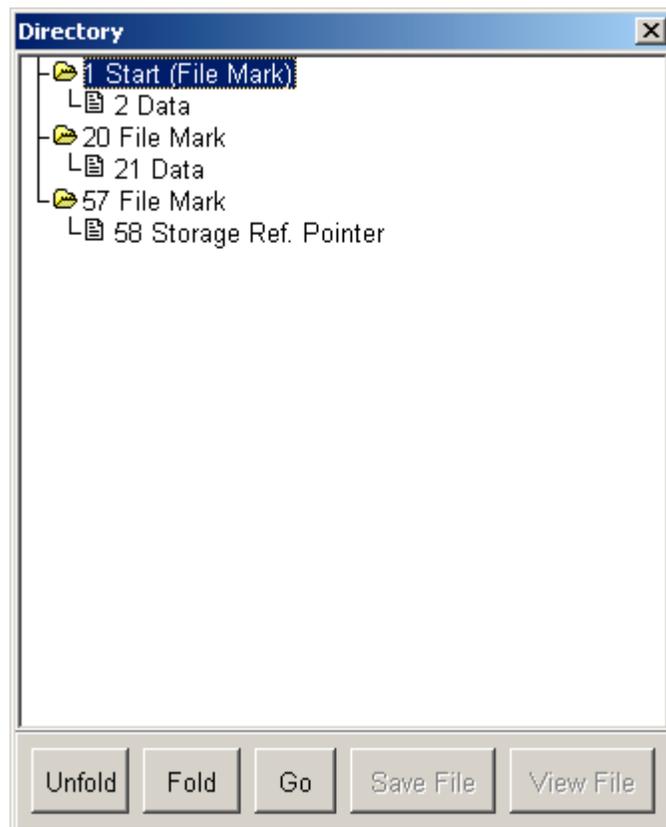
In this option files to be collected are added to the end of the current file (as indicated by the name on the **File Name** button). The new file will retain the current file name.

### **New Name for Each File**

By selecting this option you have the flexibility to choose a completely different name for each collected file. The names can be any name conforming to standard Windows convention. As the files are collected, you will be prompted to enter a valid file name for each individual file.

## **10.5.3 Show Card/Module Directory**

By clicking the **Show Card Directory** (CSM1/MCR1/PC Card) or the **Show Module Directory** (SM192/716 or SM4M/SM16M) buttons, a window showing the directory structure appears as shown below. Initially, the directory is in a concise, or folded, form showing just the locations of the filemarks. Click the **Unfold** button to expand the directory to show data file locations and other filemarks and pointers. Data can be read into a file and/or viewed directly from the **DIRECTORY** screen using the **Read File** and **View File** buttons. For a card or module with a large number of data files the Show Directory command may take several seconds to complete.



The Show Directory command is repeated on the menu bar. See the on-line help system for further details.

## 10.5.4 Get All

This option collects *all* valid data files in the card or module and stores them to the specified directory, starting with the file name shown in the File Name box. The way that the file is named is controlled from the **Auto Name Control** selection buttons (see above).

As the files are read to the disk, a Progress Indicator showing the file names is displayed at the bottom right of the display.

## 10.5.5 Get New

This option collects new data files, starting from the position of the storage module dump pointer. This pointer marks where data was last collected, and is automatically moved after each successful collection of data. This option can be used to collect any new data files stored in the card or module since data was last read. As the file is read to the disk a Progress Indicator showing the file name is displayed at the bottom right of the display.

See the relevant Storage Module Manuals and on-line help system for further details about Filemarks and Pointers in memory.

## 10.5.6 Get One

This option collects one data file starting at the specified location of the display pointer. You can position the display pointer to any starting position required by using the **ADVANCED STATUS INFORMATION** screen (see section 10.3.2).

This option is used mainly for re-collecting old data that has already been collected, or to recover data from a corrupted card.

As the file is saved to disk a Progress Indicator showing the file name is displayed at the bottom right of the display.

# 10.6 Erase

Clicking the **ERASE** tab will bring up the **ERASE OPTIONS** screen. This gives control over how data is erased or cleared from the card or module.

## 10.6.1 Erase Data

This option “erases” data by removing the references to the data files and erasing all filemarks. Only DATA from the card or module is affected. Any programs are temporarily off-loaded to disk, all pointers to data and programs on the card/module memory are erased and the pointers reset. The programs are then copied back to the card or module. Before any data is “erased” you will have the chance to cancel the option.

## 10.6.2 Erase Data and Programs

This option will completely “erase” (see above) all data and programs from the card or module. You will have the opportunity to cancel the option before the card/module is erased.

## 10.6.3 Erase and Test Card/Module

This option is similar to the Erase Data and Programs option except that a full integrity test is carried out on the card or module. This option takes considerably longer than the simple Erase Data option, and all memory location values are reset.

---

### CAUTION

Do not disconnect an SM4M/SM16M module from the SC532 or datalogger until the full memory test is complete. Doing so will corrupt the module’s flash memory and make it unusable until a full memory test is performed.

---

## 10.7 The Menu Bar

The Menu Bar appears along the top of all **SMS** screens with four options which can be accessed either by clicking with the mouse, or from the keyboard by pressing <Alt> + Underlined Character (F, O, D, T or H). Each option provides a further pull-down menu. The underscored character in the name is the “hot-key” for keyboard access.

### 10.7.1 File

The **File** pull-down menu gives three additional options:

#### Terminal Emulator

Clicking **T**erminal on the pull-down menu produces a **TERMINAL EMULATOR** screen. It allows you to communicate directly with the Storage Module and is provided so that you can issue storage module telecommunication commands manually. When the **TERMINAL EMULATOR** screen is opened, it automatically establishes a connection to the storage module. The storage module prompt (%) should be displayed while the connection is active.

Keyboard entries are transmitted from the COM port to the storage module, and storage module responses are displayed on the computer screen. Pressing <Esc> exits this mode and returns you to the main **SMS** screen. This option is seldom used because all normal operations can be accomplished easily by using other sections of the software. You are unlikely to need Terminal Mode unless you are trying to diagnose problems with a corrupted module.

Full details of direct module communication commands are given in Campbell Scientific’s Storage Module Instruction Manual.

### **Show Directory**

Clicking Show Directory produces exactly the same result as clicking the **Show Card/Module Directory** button (see Section 10.5.3).

### **Exit**

Clicking Exit on the Menu Bar will close all communications completely, exit the program, and return you to LoggerNet. Alternatively, you can exit the system by double clicking the small box at the top left-hand corner of the **SMS** screen or by clicking this box and choosing **C**lose from the pull-down menu, or by using the small “Exit” icon at the top right-hand corner of the screen. Do not use this option merely to close the pull-down menu and return to the **SMS** screen. To return to SMS, click the mouse anywhere on the **SMS** screen outside the pull-down menu box.

## **10.7.2 Options**

The options menu provides a pull-down menu with two options:

### **Enable Pop-Up Hints**

Clicking this option acts as a toggle. If Pop-Up Hints are already enabled, a check appears at the left hand side of the name. By clicking the name, the Pop-Up Hints in the MAIN PROGRAM screen can be disabled. An additional click will re-enable them.

### **Save Settings on Exit**

If you want to save all your set-up information when you leave SMS, ensure that Save Settings on Exit is checked. Again, this option may be enabled/disabled by clicking the mouse on the name.

## **10.7.3 Data**

Selecting Data from the Menu Bar provides a pull-down menu with five items. The first three—Get All, Get New and Get One are identical to the similarly worded buttons on the **SMS DATA** Screen (see Section 10.5).

The two remaining items on the pull-down menu are:

### **Insert Filemark**

Filemarks are used to separate data in the storage device. A Filemark is automatically placed in the device’s memory when it is connected to a power source (a datalogger or retrieval interface), when a datalogger compiles a program containing Instruction 96, or under program control. For example, if you retrieve data from one datalogger and disconnect the storage module, and then connect it to another datalogger, a Filemark is placed in the data when the second datalogger is connected. This mark follows the data from the first datalogger and precedes the data from the second.

### **Next Filemark**

The Next Filemark option moves the Display Pointer to the next filemark in the data in a similar way to clicking the green arrow in the **ADVANCED STATUS** screen (see Section 10.3.2).

## **10.7.4 Tools**

The Tools menu provides a way to access the tabs for each module when a mouse or other pointing device is unavailable. The Tools menu has six options: Setup, Programs, Data, Erase, Status, and Advanced. By selecting one of these options, you will be taken to that tab for the module to which you are connected.

## **10.7.5 Help**

The final option on the Menu Bar is Help. This provides a pull-down menu with five items connected with the SMS on-line help system and Windows help in general. The options are self-explanatory.

Note that the option “About...” will display the current version number of the software. This may be required if you need to contact Campbell Scientific for further information or support for SMS.

“About...” also indicates if any Card Services drivers have been found on your system.

## **10.8 Abort**

Most SMS operations can be stopped while in progress by clicking the **Abort** button. For example, if you click Abort while the software is attempting to establish communications between a storage module and computer, that operation will be aborted. Note that for some rapid operations, such as reading or storing small program or data files from a module, the operation may be completed before the abort option can take effect.



# Section 11. Clients Installed with LoggerNet

---

*LoggerNet has auxiliary clients installed with it, which are not accessible from the LoggerNet toolbar. CoraScript is available to configure and run LoggerNet from a command line. A useful feature of CoraScript is the ability to create a backup file of the network map, which can then be loaded back in to LoggerNet in the event that the network map is lost (irrecoverable hard disk crash, for instance).*

*RTMC Run-Time, which is the runtime engine for RTMC-created forms, is discussed in Section 5.*

## 11.1 CoraScript

### 11.1.1 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 CoraScript 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.

## 11.1.2 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.

### 11.1.2.1 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.

### 11.1.2.2 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 Setup screen 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.

### 11.1.2.3 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 file name 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.

### 11.1.2.4 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

### **11.1.2.5 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.

# Section 12. Utilities Installed with LoggerNet



Along with LoggerNet's server, clients and program editors, we also install several utilities. These are launched either from a separate shortcut in the program folder or from a command line calling the executable itself. These utilities include **Device Configurator**, a standalone application that uses a serial port to configure Campbell Scientific dataloggers and communications devices, **Transformer**, an application to convert a CR10X program written in Edlog to a CR1000 program in CRBasic, **TOB File Converter**, a tool to convert binary data files from a CRBasic datalogger to ASCII formats that can be loaded into third party software, and **Resource Monitor**, a tool that is provided for Windows 98 users to monitor resource use on their machines.

The Transformer application is discussed in the programming section of this manual (Section 7).

## 12.1 Device Configurator

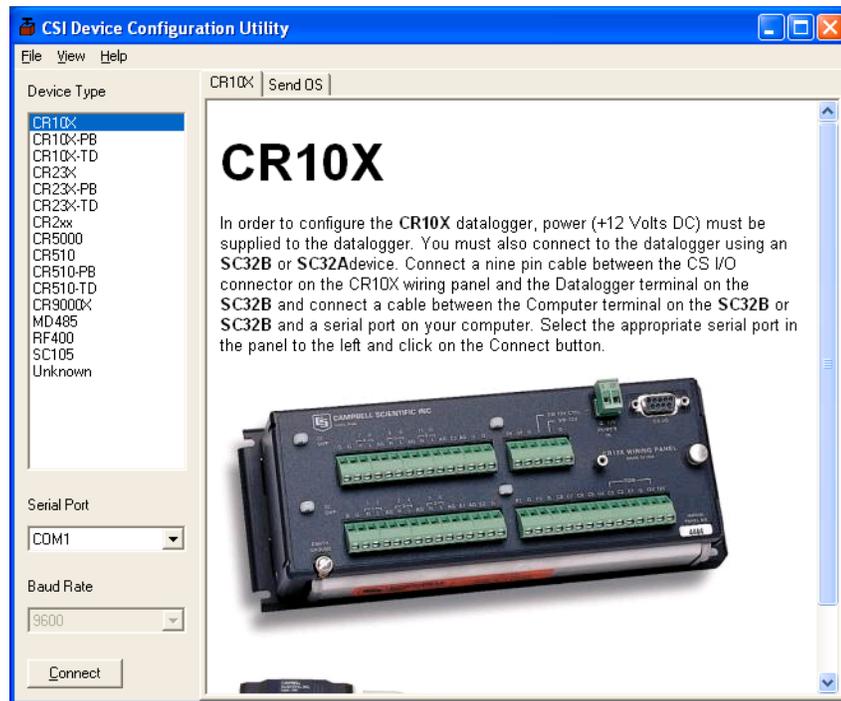
### 12.1.1 Overview

The Device Configuration Utility (DevConfig) sets up dataloggers and intelligent peripherals before those devices are deployed in the field and before these devices are added to networks in Campbell Scientific datalogger support software such as LoggerNet or PC400. Some key features of DevConfig include:

- To keep the process as simple as possible, DevConfig supports only direct serial connections between the PC and devices.
- DevConfig cannot only send operating systems to supported device types, but can also set datalogger clocks and send program files to dataloggers.
- DevConfig allows you to determine operating system types and versions, which can be very useful in classic dataloggers<sup>1</sup>, such as the CR10X, where the operating system version in the datalogger is not known.
- DevConfig provides a reporting facility where a summary of the current configuration of a device can be shown on the screen and printed. This configuration can also be saved to a file and used to restore the settings in the same or a replacement device.
- Some devices may not support the configuration protocol in DevConfig, but do allow configurations to be edited through the terminal emulation screen.

- Help for DevConfig is shown as prompts and explanations on its main screen. Help for the appropriate settings for a particular device can also be found in the user's manual for that device.
- Updates to DevConfig are available from Campbell Scientific's web site. These may be installed over top of older versions.

## 12.1.2 Main DevConfig Screen



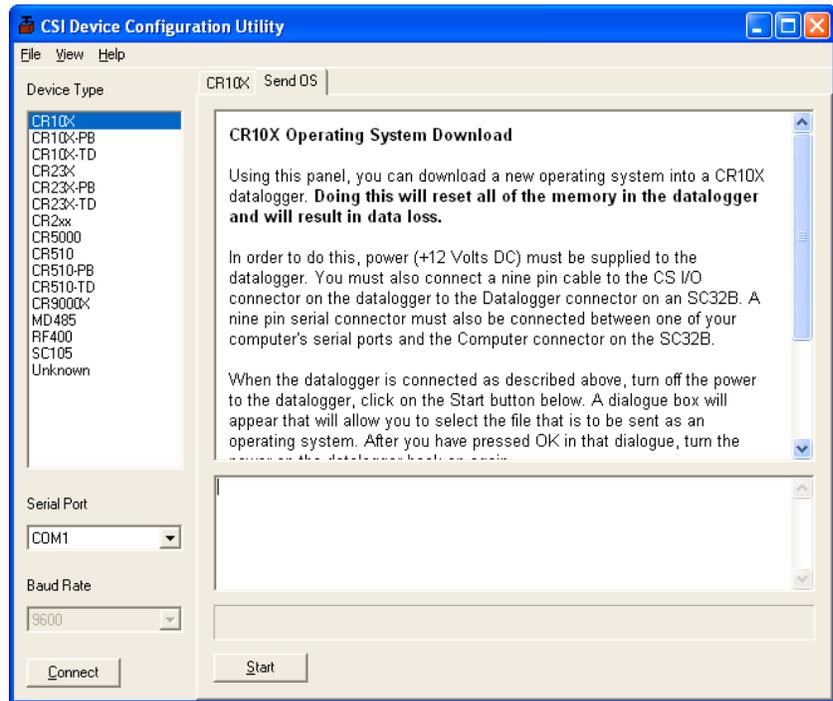
The DevConfig window is divided into two main sections: the device selection panel on the left side and tabs on the right side. After choosing a device on the left, you will then have a list of the serial ports (COM1, COM2, etc.) installed on your PC. You'll be offered a choice of baud rates only if the device supports more than one baud rate in its configuration protocol. The page for each device presents instructions about how to set up the device to communicate with DevConfig. Different device types will offer one or more tabs on the right.

When the user presses the Connect button, the device type, serial port, and baud rate selector controls become disabled and, if DevConfig is able to connect to the device, the button will change from "Connect" to "Disconnect". The tabs on the right side of the window will be replaced with tabs that represent the various operations that are available for that device in a connected state. These operations can vary from device to device and will be described in greater detail below.

## 12.1.3 Screens Applicable to All (or most) Devices

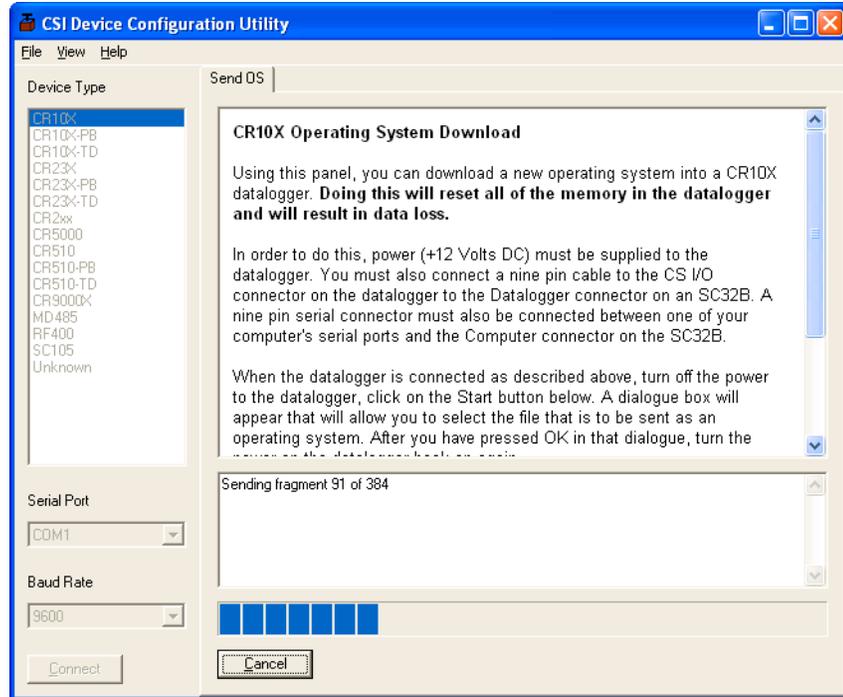
### 12.1.3.1 Send OS Tab - Downloading an Operating System

DevConfig can send operating systems to all Campbell Scientific devices with flash replaceable operating systems. An example for the CR10X is shown below:



The text at right describes any interface devices or cabling required to connect the PC to the device. Screens for other devices vary only in the text on the right side. This screen differs from other screens that are available in DevConfig in that it can be accessed from either a connected and disconnected state.

When you click the **Start** button, DevConfig offers a file open dialog box to prompt you for the operating system file (usually a \*.obj file). You may be required to cycle power to the device or press a special “program” button and when the device issues the appropriate prompts, DevConfig starts to send the operating system:



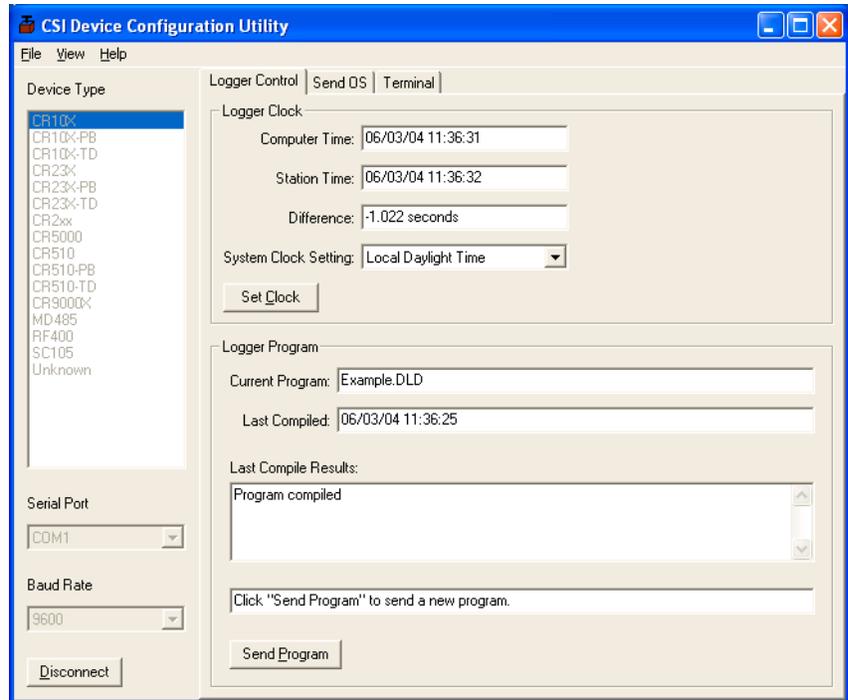
When the operating system has been sent to the device, a message dialog will appear similar to the one shown below:



The information in the dialog helps to corroborate the signature of the operating system sent. For devices such as the CR10X (especially those with extended memory) that can take a long time to reset following an OS download, text warns you against interrupting the memory test.

### 12.1.3.2 Logger Control Tab

The Logger Control tab for the CR10X mixed-array datalogger is shown below, though it is similar for all dataloggers.



The clock in the PC and the datalogger will be checked every second and the difference displayed. The **System Clock Setting** allows you to configure what offset, if any, should be used with respect to standard time (Local Daylight Time or UTC, Greenwich mean time). The value selected for this control will be remembered between sessions. Clicking the **Set Clock Button** will synchronize the station clock to the current computer system time.

**Current Program** displays the current program known to be running in the datalogger. CRBasic dataloggers provide this information. For other dataloggers, this value will be empty until a program is sent by DevConfig.

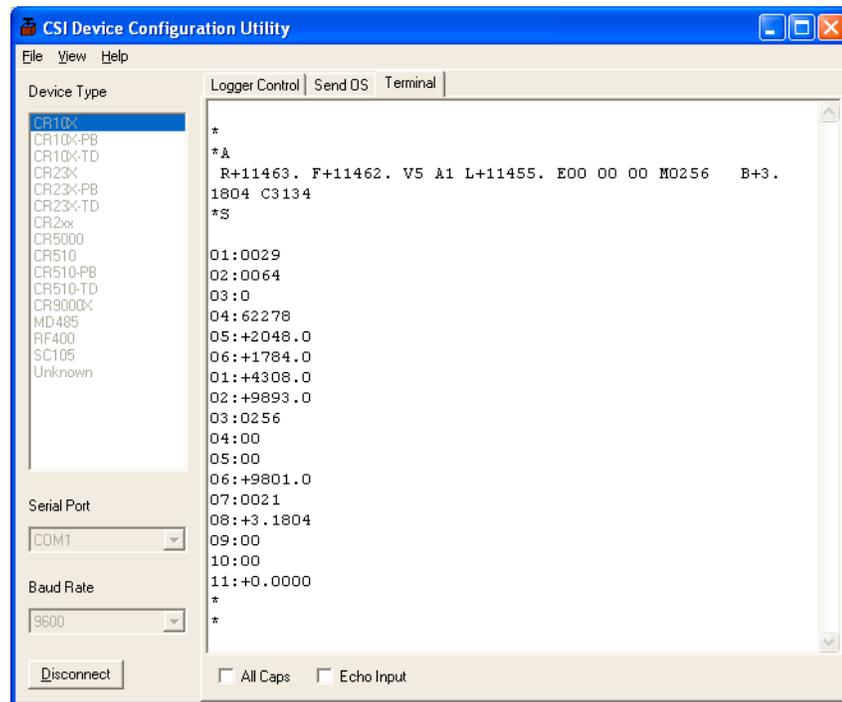
The **Last Compiled** field displays the time when the currently running program was last compiled by the datalogger. As with the Current Program field, this value will be read initially from the datalogger if it is available, otherwise, it will initially be empty.

**Last Compile Results** shows the compile results string as reported by the datalogger. Like the current program and last compiled fields, this control will be initially empty unless the datalogger supports the BMP5 Get Programming Statistics transaction.

The **Send Program** button presents an open file dialogue from which you can select a program file to be sent to the datalogger. The field above the button will be updated as the send operation progresses. When the program has been sent the Current Program, Last Compiled, and Last Compile Results fields will be filled in.

### 12.1.3.3 Terminal Tab

The **Terminal** tab will be available when the application is connected to any device type except the CR2xx (that device's terminal emulation is limited to work with SDI12 sensors). The Terminal tab offers a terminal emulator that can be useful in accessing settings or status information that are not exposed in other windows. For example, classic dataloggers with PakBus operating systems that are configured as routers contain routing tables that list the other PakBus nodes that are known to that datalogger. This routing table is only available through the \*D17 mode (see \*D descriptions in the datalogger's operators' manuals) using the keyboard/display or a terminal emulator. Another example is that the status table in mixed-array dataloggers (\*B) can also be accessed via an "S" command in terminal mode. This status information can provide important data for troubleshooting purposes.



The default for the Terminal tab is to only show characters that are returned from the device. However, if the **Echo Input** check box is enabled, the screen will also display the characters actually typed by the user.

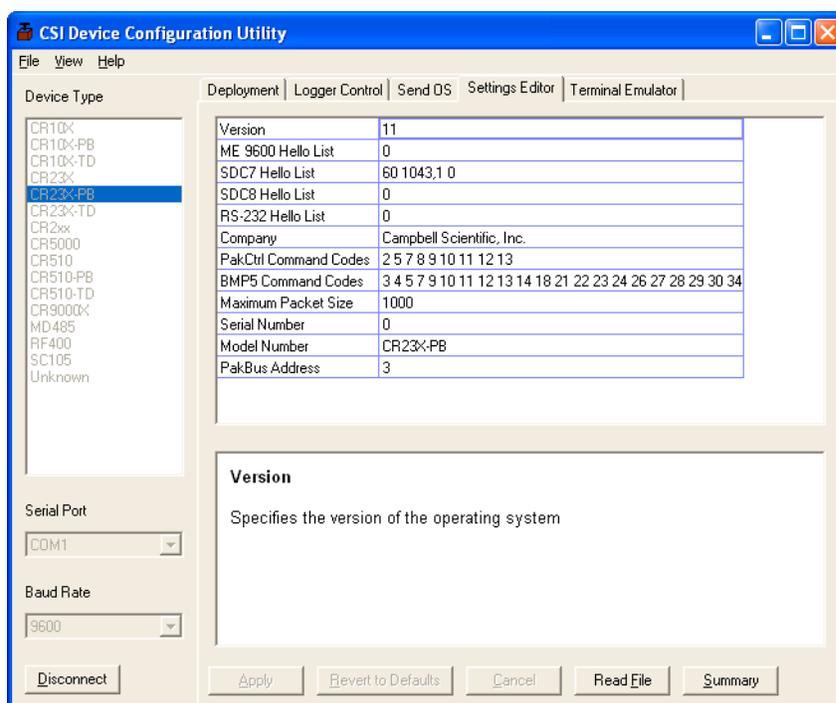
The **All Caps Check Box** controls whether the keyboard input will be forced to upper case before the characters are sent to the device. It will be disabled for some device types that require upper case input.

### 12.1.3.4 PakBus Dataloggers – Settings Editor Tab

PakBus dataloggers have a number of properties, referred to as “settings”, some of which are specific to the PakBus protocol. PakBus is discussed in more detail in the [PakBus Networking Guide](#) available from the Campbell Scientific website ([www.campbellsci.com](http://www.campbellsci.com)).

The **Settings Editor** tab provides access to most of the PakBus settings, however, on some dataloggers there is also a **Deployment** tab that can make configuring some of these settings a bit easier. The Deployment tab for an Edlog datalogger will be described later though many of the details are specific to other dataloggers.

The figure below shows the Settings Editor tab for a CR23X-PB.

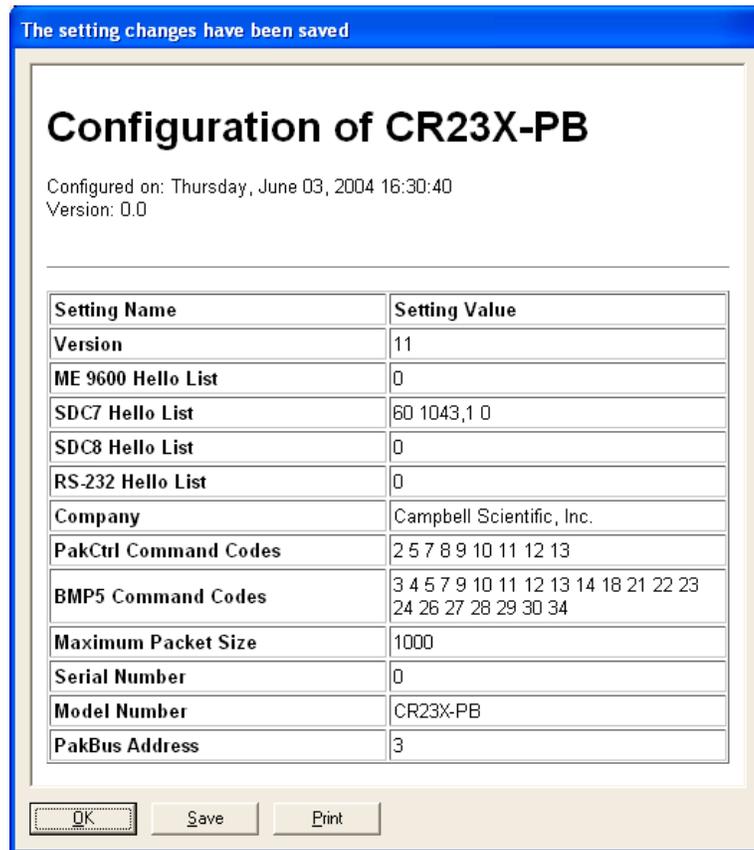


The top of the Settings Editor is a grid that allows the user to view and edit the settings for the device. The grid is divided into two columns with the setting name appearing in the left hand column and the setting value appearing in the right hand column. You can change the currently selected cell with the mouse or by using the up arrow and down arrow keys as well as the Page Up and Page Down keys. If you click in the setting names column, the value cell associated with that name will automatically be made active. You can edit a setting by selecting the value, pressing the F2 key or by double clicking on a value cell with the mouse. The grid will not allow read-only settings to be edited.

The bottom of the Settings Editor displays help for the setting that has focus on the top of the screen.

Once you have changed a setting, you can **Apply** them to the device or **Cancel**. These buttons will only become enabled after a setting has been changed. If the

device accepts the settings, a configuration summary dialogue will be shown that will give the user a chance to save and/or print the settings for the device:



Clicking the **Revert to Defaults** button on the Settings Editor will send a command to the device to revert to its factory default settings. The reverted values will not take effect until the final changes have been applied. This button will remain disabled if the device does not support the DevConfig protocol messages.

If, after changing a setting or clicking the **Summary** button, you clicked **Save** on the summary screen to save the configuration, you can use the **Read File** button to load those settings. The settings from the saved file are immediately sent to the device and, if they're accepted, you can then **Apply** them.

### 12.1.3.5 Device-specific Screens and Deployment Tab

The Device Configurator is especially useful for configuring PakBus device settings. Some settings are common to multiple devices; others are specific to a small group of devices or even a single type of device.

#### 12.1.3.5.1 Deployment Tab Settings Common to Many PakBus Dataloggers

Not all of the settings below are supported by all dataloggers. For more information on these settings, refer to the datalogger operating manual or the [PakBus Networking Guide](#).

**Serial Number** displays the serial number that is stored for the datalogger. For some dataloggers, this setting is set at the factory and cannot be edited.

**PakBus Address** allows you to set the PakBus address of the datalogger. The allowable range is between 1 and 4094. Each PakBus device should have a unique PakBus address. Addresses >3999 force other PakBus devices to respond regardless of their respective PakBus settings. See the [PakBus Networking Guide](#) for more information.

**Is Router** allows you to control whether the datalogger will act as a PakBus router.

**Station Name** displays the name that is set for this station.

**Default Router** specifies the address of a PakBus router to which the datalogger will send packets when it doesn't itself know the route to the destination.

**Selected Port** specifies the datalogger serial port to which the beacon interval and hello setting values will be applied.

**Beacon Interval** sets the interval (in seconds) on which the datalogger will broadcast beacon messages on the port specified by Selected Port.

**Verify Interval** specifies the interval (in seconds) at which the datalogger will expect to have received packets from neighbors on the port specified by Selected Port. A value of zero (default) indicates that the datalogger has no neighbor list for this port.

**Neighbors List**, or perhaps more appropriately thought of as the “expected neighbors list”, displays the list of addresses that this datalogger expects to find as neighbors on the port specified by Selected Port. As you select items in this list, the values of the **Begin** and **End** range controls will change to reflect the selected range. You can add multiple lists of neighbors on the same port.

**Begin and End Range** are used to enter a range of addresses that can either be added to or removed from the neighbors list for the port specified by Selected Port. As you manipulate these controls, the Add range and Remove Range buttons will be enabled or disabled depending on the relative values in the controls and whether the range is present in or overlaps with the list of address ranges already set up. These controls will be disabled if the **Verify Interval** value is set to zero.

**Add Range** will cause the range specified in the **Begin** and **End** range to be added to the list of neighbors to the datalogger on the port specified by Selected Port. This control will be disabled if the value of the **Verify Interval** is zero or if the end range value is less than the begin range value.

**Remove Range** will remove the range specified by the values of the **Begin** and **End** controls from the list of neighbors to the datalogger on the port specified by Selected Port. This control will be disabled if the range specified is not present in the list or if the value of **Verify Interval** is set to zero.

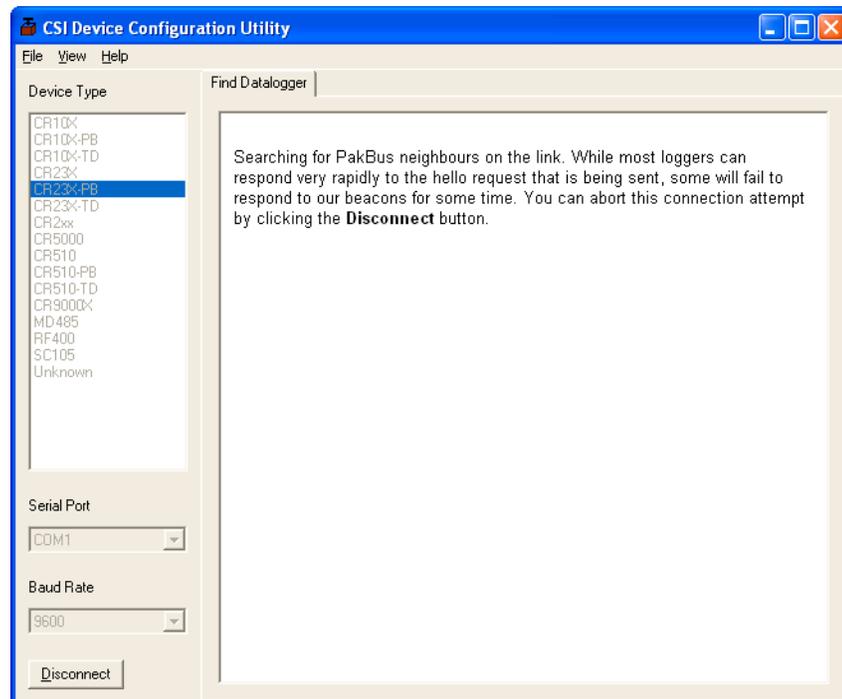
Help is displayed at the bottom of the Deployment tab. When you're finished, you must **Apply** to send the settings to the datalogger. The Summary window will appear and you can **Save** or **Print** the settings for your records or to use them as a template for another datalogger.

**Cancel** causes the datalogger to ignore the changes. **Read File** gives you the opportunity to load settings saved previously from this or another similar datalogger. If you load settings from a file, the changes will not actually be written to the datalogger until you click **Apply**.

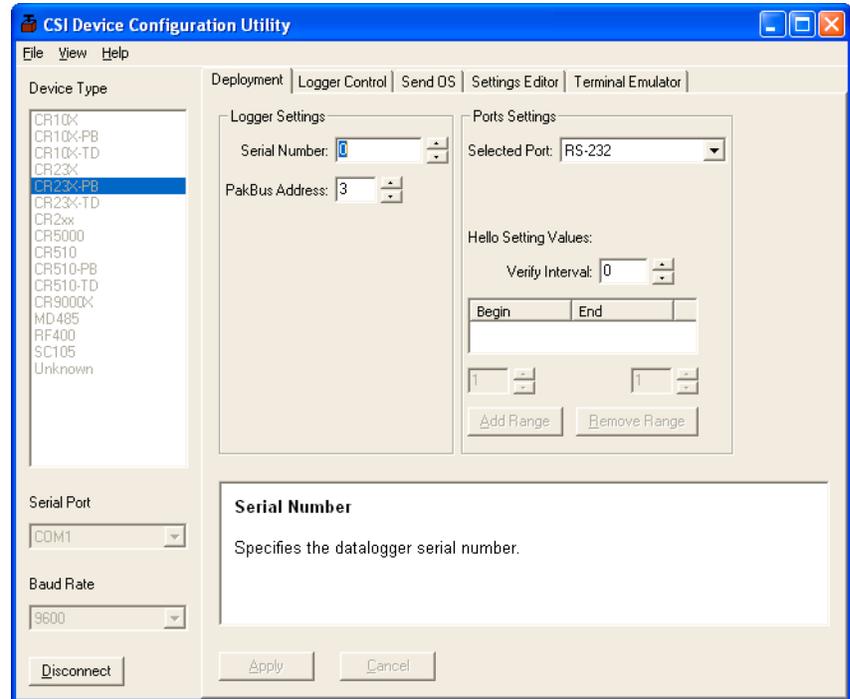
### 12.1.3.5.2 CR10X-PB, CR510-PB, and CR23X-PB Specific Screens

When you select the CR10X-PB, CR510-PB, or CR23X-PB device type, you'll see a description and picture of the datalogger with instructions on how to communicate with the datalogger from DevConfig. In addition, a Send OS table will be available that will allow the user to send an operating system to the datalogger.

Click the Connect button and these tabs will be replaced by a tab similar to that shown below:



This screen will be shown while DevConfig searches for a PakBus datalogger attached to the specified serial port. When the application has found the datalogger address and loaded its settings, this tab will be replaced with the following tabs:



The Logger Control, Send OS, Settings Editor, and Terminal Emulator tabs are described in Sections 12.1.3.

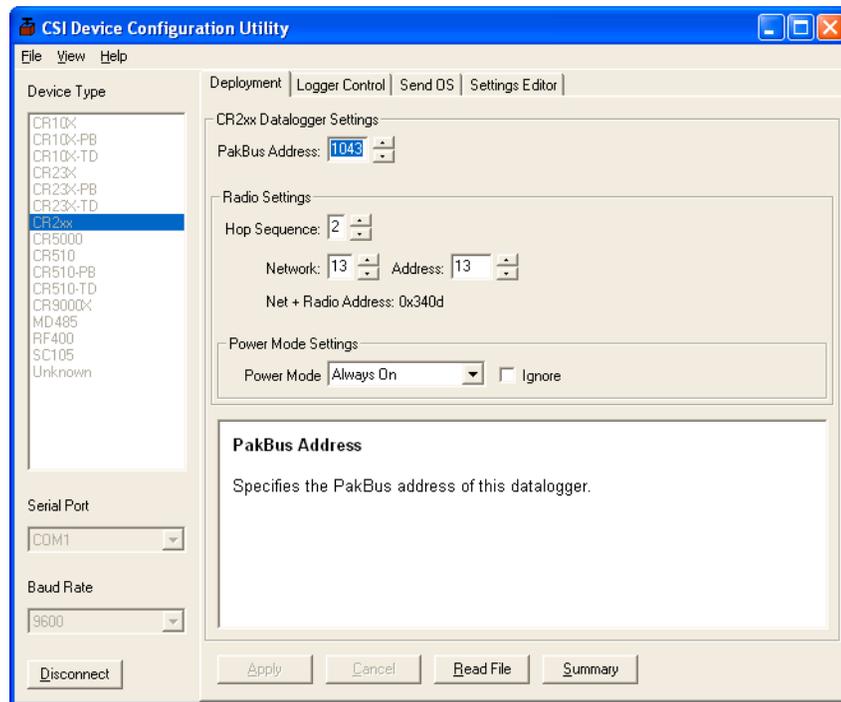
The **Deployment Tab** differs from the description in Section 12.1.3.5.1 in that:

- You can edit the **Serial Number**; this serial number can be read by program instructions and stored with final storage data to uniquely stamp the data files it creates.
- The **Station Name**, **Is Router**, and **Default Router Settings** are not available.
- The **Selected Port Beacon Interval** cannot be set (these dataloggers do not provide a way to do this other than through terminal emulation with \*D18 mode).
- In addition to the CS I/O ports listed in the **Selected Port** list, the CR23X-PB also has an RS-232 port that can be used to verify expected neighbors via the **Hello Setting Values**.

### 12.1.3.5.3 CR2xx Specific Screens

When you select the CR2xx device type, you'll see a description and picture of the datalogger with instructions on how to communicate with the datalogger from DevConfig. In addition, a Send OS tab will be available that will allow the user to send an operating system to the datalogger.

Click the Connect button and, once DevConfig has connected to the CR2xx datalogger, these tabs will be replaced by a screen with tabs similar to that shown below:



The Logger Control, Send OS, and Settings Editor tabs are described in Sections 12.1.3.

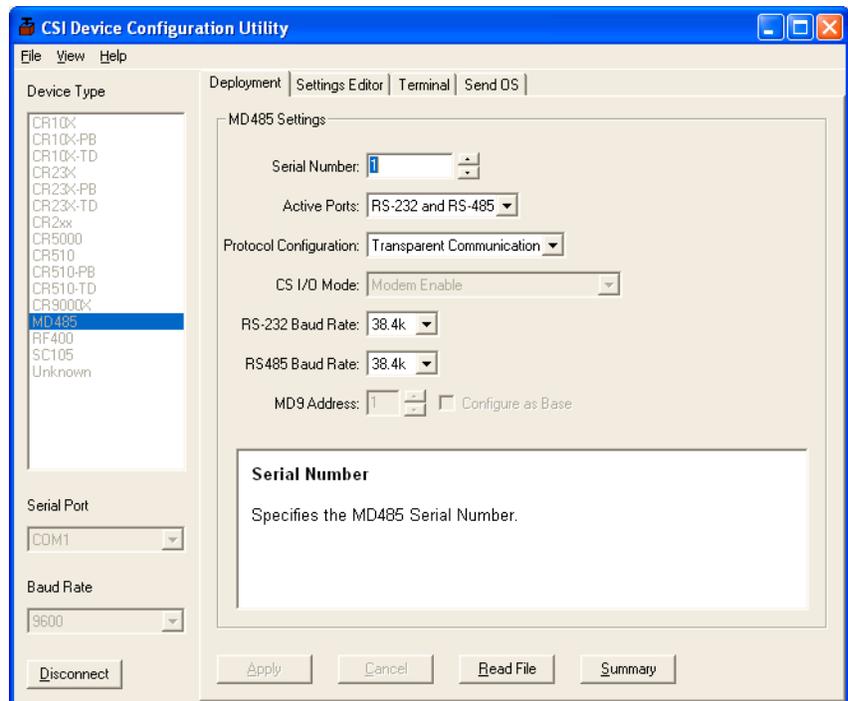
The **Deployment Tab** differs from the description in Section 12.1.3.5.1 in that:

- There is a spin control for setting the **PakBus Address**.
- The **Hop Sequence** sets the hopping pattern of the spread spectrum radio. It will be disabled if the datalogger is a CR200 (has no associated radio).
- The **Network** specifies the network to which the internal spread spectrum radio will belong. This value is actually the most significant six bits of the address that will be sent to the radio. This value will be disabled if the datalogger is a CR200 (has no associated radio).
- The **Address** specifies the radio address for the spread spectrum radio. This value is actually the least significant twelve bits of the address that is sent to the radio. This control will be disabled if the datalogger is a CR200 (has no associated radio).
- The **Power Mode** sets the duty cycle and other parameters of the spread spectrum radio. It will be disabled if the datalogger is a CR200 (has no associated radio) or if the **Ignore** check box is checked. In this latter mode, the datalogger will stay active all of the time rather than turning the receiver on and off periodically to save power.

### 12.1.3.5.4 MD485 Specific Screens

When you select the MD485 device type, you'll see a description and picture of the device with instructions on how to communicate with the device from DevConfig. In addition, a Send OS tab will be available that will allow the user to send an operating system to the device.

Press the **Program** button on the device and click the **Connect** button in DevConfig and, once DevConfig has connected to the MD485, these tabs will be replaced by a screen with tabs similar to that shown below:



The deployment panel and the settings editor will be initially disabled while DevConfig attempts to get the current settings from the MD485. If the device does not respond to the application's queries within fifteen seconds, the session will be cancelled and an error reported.

The Settings Editor, Terminal, and Send OS tabs are described in Section 2.1.3.

The **Deployment Tab** differs from the description in Section 12.1.3.5.1 in that:

- The **Serial Number** control allows the user to enter the serial number for the device. This needs to be entered by the user because the serial number setting will be lost when an operating system download operation takes place and it is not set at the time the device is manufactured.

- **Active Ports** allows you to specify which two of the three ports will be active. The choices are:
  - **CS I/O and RS-485** (disables the RS-232 Baud Rate, MD9 Address, and Configure as Base controls), typically used at a datalogger with a CS I/O port to connect it to an MD485 network.
  - **RS-232 and RS-485** (disables the CS I/O Mode, MD9 Address, and Configure As Base controls), typically used to connect a PC to the MD485 network.
  - **CS I/O and RS-232** (disables the Protocol Configuration, RS485 Baud Rate, MD9 Address, and Configure as Base controls), typically used to connect a datalogger directly to a PC's COM port, in lieu of the SC32A/B, in cases where you might want to use a mode other than modem enable (CSDC for PB dataloggers or SDC for mixed-array or TD dataloggers) between the PC and the datalogger.
  
- **Protocol Configuration** allows you to specify the protocol that the MD485 will use to communicate on its RS-485 port. It will be disabled if the value of the Active Ports controls is set to CS I/O and RS-232. The choices offered by this control are as follows:
  - **Transparent Communication**, typically used when there are only two MD485s in use, such as when needing a “direct connect” hard-wire link that is longer than the RS-232 voltages will support.
  - **PakBus Networking**, typically used when all of the devices on the MD485 network are PakBus devices with their own addresses.
  - **MD9 Emulation**, typically used with mixed-array dataloggers or a mix of mixed-array and table-based dataloggers where the addressing is to reside in the MD485.

*The value chosen for this control will affect the values offered in the CS I/O Mode control.*

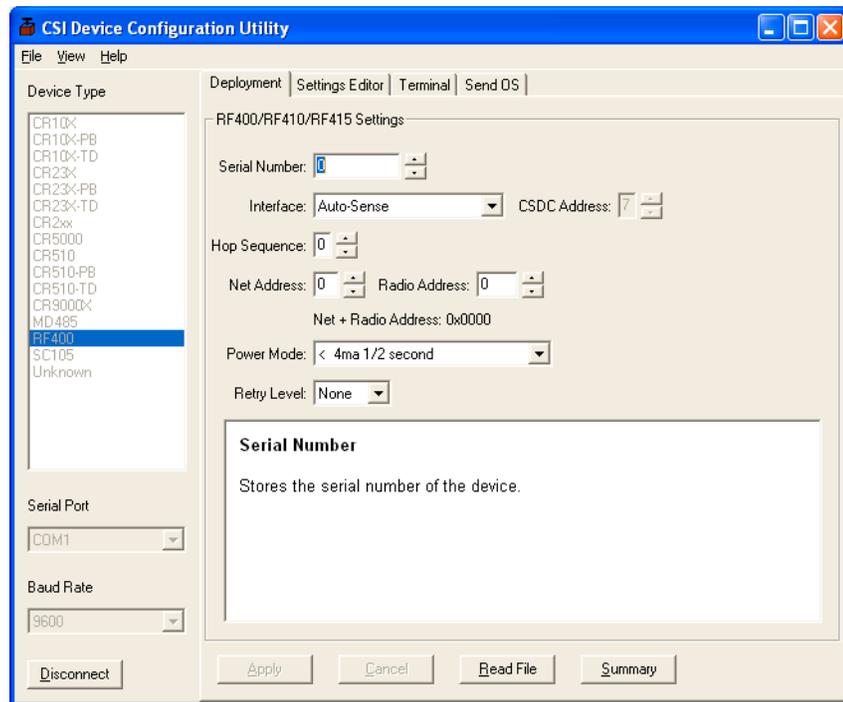
- **CS I/O Mode** allows you to specify how the CS I/O port will be driven. It will be disabled if **Active Ports** is set to RS-232 and RS-485. The values offered by this control are as follows:
  - **Modem Enable** (only available if Protocol Configuration is set to MD9 Emulation or Transparent Communication).
  - **SDC Address 7** (Only available if Protocol Configuration is set to Transparent Communication or PakBus Networking).
  - **SDC Address 8** (Only available if Protocol Configuration is set to Transparent Communication or PakBus Networking).

- **SDC Address 9** (Only available if Protocol Configuration is set to Transparent Communication or MD9 Emulation).
- **Modem to Modem w/o Datalogger** (Only available if Protocol Configuration is set to Transparent Communication or MD9 Emulation).
- **Addressed Print Device** (Only available if Protocol Configuration is set to Transparent Communication).
- **RS-232 Baud Rate** allows you to specify the baud rate that will be used on the RS-232 port. It will be disabled if Active Ports is set to CS I/O and RS-485.
- **RS-485 Baud Rate** allows you to specify the baud rate that will be used on the RS-485 port. It will be disabled if Active Ports is set to CS I/O and RS-232.
- **MD9 Address** allows you to specify the address of the device when the MD9 emulation protocol is being used on the RS-485 port. This control will be disabled if Active Ports is set to CS I/O and RS-232 or if Protocol Configuration is not set to MD9 Emulation. It will also be disabled if the Configure as Base check box is checked.
- Enabling the **Configure as Base** check box allows you to specify that the MD485 should have the address of an MD9 base unit (255) when the MD9 emulation protocol is used for the RS-485 port. This control will be disabled if Active Ports is set to CS I/O and RS-232 or if Protocol Configuration is not set to MD9 Emulation.

#### **12.1.3.5.5 RF400 Specific Screens**

When you select the RF400 device type, you'll see a description and picture of the device with instructions on how to communicate with the device from DevConfig. In addition, a Send OS tab will be available that will allow the user to send an operating system to the device.

Press the **Program** button on the device and click the **Connect** button in DevConfig and, once DevConfig has connected to the RF400, these tabs will be replaced by a screen with tabs similar to that shown below:



The Deployment, Settings Editor, and Terminal tabs will be initially disabled while DevConfig attempts to get the current settings from the RF400. If the device does not respond to the application's queries within fifteen seconds, the session will be cancelled and an error reported.

The Settings Editor, Terminal, and Send OS tabs are described in Section 12.1.3.

The **Deployment Tab** differs from the description in Section 12.1.3.5.1 in that:

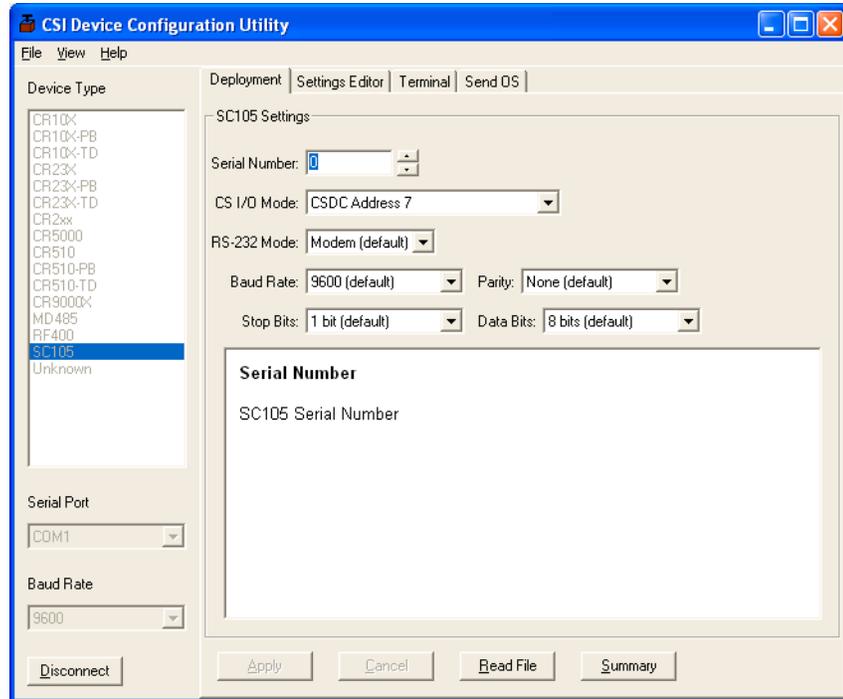
- The **Serial Number** control allows the user to enter the serial number for the device. This needs to be entered by the user because the serial number setting will be lost when an operating system download operation takes place and it is not set at the time the device is manufactured.
- The **Interface** drop down box allows you to specify the port and protocol that will be used for communication. The choices given are as follows:
  - **Auto-Sense** (the radio will automatically choose the active port based upon whether +5 volts is present on the CS I/O port. Typically used at remote mixed-array or TD dataloggers; this option is not compatible with PakBus dataloggers).

- **RS-232** (the radio will use the RS-232 port at 9600 baud with no parity, eight data bits, and one stop bit; typically used at the PC).
- **Datalogger Modem Enable** (the radio will use the CS I/O port and modem enable protocol; typically used at remote dataloggers).
- **Datalogger SDC** (the radio will use the CS I/O port and use an SDC protocol compatible with the mixed-array and TD classic dataloggers. This option is NOT compatible with PakBus dataloggers.)
- **Datalogger CSDC** (the radio will use the CS I/O port and will either use SDC7 or SDC8 addresses depending on the value of CSDC Address. This option is compatible only with PakBus dataloggers.)
- **COM2xx to RF400** (the radio detects a “ring” coming through the COM2xx phone modem and manages the ME line to the modem to let it connect; thereafter accepting commands from the PC to communicate with remote RF400s; typically used at a phone-to-RF base station with a PS512M or PS100 with A100 adapter to provide a CS I/O null modem).
- **CSDC Address** allows you to specify the concurrent SDC address (7 or 8) that will be used by the RF400 when the value of Interface is set to Datalogger CSDC.
- **Hop Sequence** allows you to specify the hopping sequence that will be used by the spread spectrum radio. All RF400s in the same network must use the same Hop Sequence.
- The **Net Address** specifies the network to which the internal spread spectrum radio will belong. This value is actually the most significant six bits of the address that will be sent to the radio. All RF400s in the same network must use the same Net Address.
- The **Radio Address** specifies the radio address for the spread spectrum radio. This value is actually the least significant twelve bits of the address that is sent to the radio. In PakBus networks, all RF400s should use the same Radio Address.
- The **Power Mode** sets the duty cycle and other parameters of the spread spectrum radio.
- **Retry Level** allows you to specify the retries that the spread spectrum radio will use when operating in point-to-point mode.

### 12.1.3.5.6 SC105 Specific Screens

When you select the SC105 device type, you'll see a description and picture of the device with instructions on how to communicate with the device from DevConfig. In addition, a Send OS tab will be available that will allow the user to send an operating system to the device.

Press the **Program** button on the device and click the **Connect** button in DevConfig and, once DevConfig has connected to the SC105, these tabs will be replaced by a screen with tabs similar to that shown below:



The Deployment, Settings Editor, and Terminal tabs will be initially disabled while DevConfig attempts to get the current settings from the RF400. If the device does not respond to the application's queries within fifteen seconds, the session will be cancelled and an error reported.

The Settings Editor, Terminal, and Send OS tabs are described in Section 12.1.3.

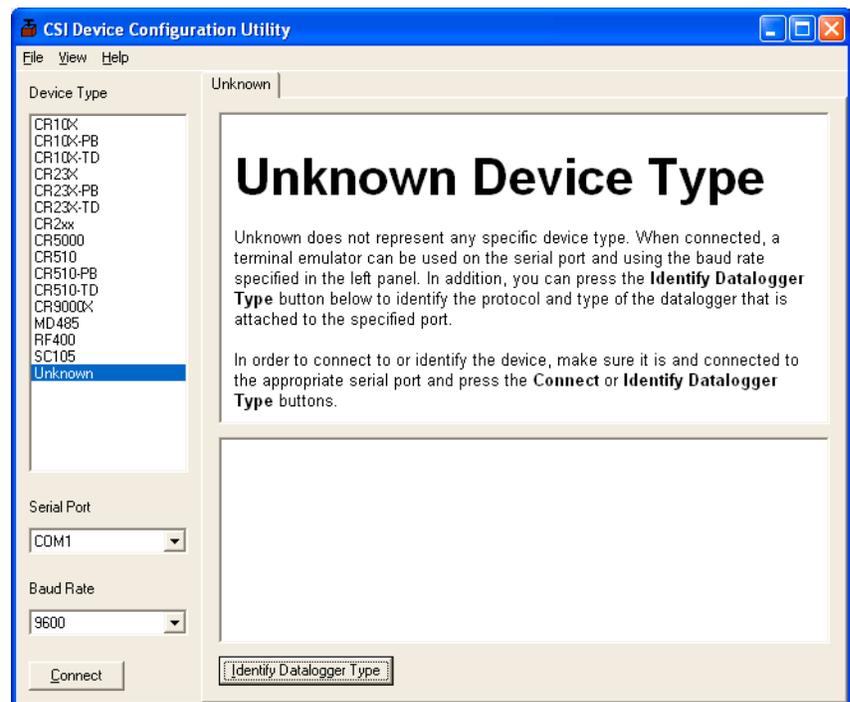
The **Deployment Tab** differs from the description in Section 12.1.3.5.1 in that:

- **Serial Number** allows you to specify the serial number for the SC105.
- **CS I/O Mode** allows you to specify the protocol that will be used on the CS I/O port. The choices include the following:
  - **Modem Enable** (default).
  - **CSDC Address 7** (compatible only with PakBus dataloggers).

- **CSDC Address 8** (compatible only with PakBus dataloggers).
- **SDC Address 9** (compatible only with CR10X, 23X, and CR510 mixed-array dataloggers).
- **Addressed Print Device** (output only; e.g., to a printer).
- **RS-232 Mode** allows you to specify the manner in which the RS232 port will be driven. Choices include:
  - **Modem** (the default, typically used for third party RS-232 compatible modems).
  - **PC or PDA** (for connecting to a computer or personal digital assistant).
  - **FreeWave™ Radio** (for connecting to a FreeWave™ spread spectrum radio on its RS-232 port).
- **Baud Rate, Parity, Stop Bits, and Data Bits** controls allow you to specify these settings to be used on the RS232 port, depending on the requirements of the RS-232 device. This control will be disabled when the RS-232 Mode control is set to PC or PDA.

#### 12.1.3.5.7 The Unknown Device Type

When the Unknown device type is selected, a panel will be shown in the tab control similar to that shown below:



Clicking **Connect** puts DevConfig into Terminal emulation mode on the Serial Port and at the Baud Rate selected.

When you click on **Identify Datalogger Type**, DevConfig will attempt to identify the type of device that is connected on the specified serial port. It will attempt to communicate using each of the datalogger protocols (mixed-array, table-data, and PakBus) in turn. If it fails to get any answer to any of these attempts, the baud rate will be automatically changed and the various protocols will be attempted again. When DevConfig recognizes the response from the device and the device type is one of the supported types, that device type will automatically be selected.

## 12.2 TOB File Converter

TOB files are binary files that were collected from a PC card installed in a CR9000 or CR5000 datalogger. TOB files can be opened in View or post-processed using Split. You can also process them through the TOB File Converter to convert them to a TOA5 (ASCII) format. The converter will also let you convert a file to TOB1 format.

The converter is started from the Window's Start menu, Programs | LoggerNet | Utilities | TOB File Converter. To begin a conversion, first select the type of file that should be created: TOA5 or TOB1. Next, press the Select File and Convert button. A standard file dialog box will be displayed from which to select the TOB file. After a file is selected and Open is pressed, another dialog box will be displayed in which the new file to be created will be stored. When this file is selected and Save is pressed, the file will automatically be converted. The name of the newly created file will be displayed in the Files Created field of the application.

If Process File Marks is selected, multiple files will be generated from the data stored on the card. You can choose whether or not to include records numbers in the new file by selecting or clearing the Include Record Numbers checkbox.

## 12.3 Resource Monitor

The Resource Monitor is started from the Window's Start menu, Programs | LoggerNet | Utilities | Resource Monitor. It is a tool that is provided for Windows 98 users to monitor GDI and User resources on their computers.

As mentioned in the Requirements section of this manual, LoggerNet can be under Windows 98 operating systems. However, users may experience some performance issues, including resource limitations which in turn might limit the number of applications that can be opened at one time. The Resource Monitor will display the amount of GDI and User resources, from 0 to 100 percent, being consumed by the running applications.

# **Section 13. Optional Client Applications Available for LoggerNet**

---

*Several client applications are available that are compatible with LoggerNet. Many of these allow remote access to the data in the LoggerNet data cache, or provide a way to post process that data.*

*Client applications include LoggerNet Data, RTMC-RT, RTDM, Baler, OPC Server, and the LoggerNet SDKs.*

*For the client applications that allow remote access to the LoggerNet data cache, LoggerNet must be configured to allow connection from remote clients.*

## **13.1 Allowing Remote Connections to the LoggerNet Server**

LoggerNet is a 32-bit client/server application, and therefore, the server can run on one computer while a client application can be run on a separate computer attached to the same network. Campbell Scientific offers client applications for LoggerNet that take advantage of this remote access capability. If Allow Remote Connections is enabled, you can run LoggerNet on one computer, and use a remote client application to display data remotely on a different computer or save a copy of the data on the remote computer. If remote connections are denied, data access from a remote computer is not possible.

Remote Connections is enabled from the LoggerNet Toolbar's Options menu, Allow Remote Connections. When a check mark appears beside the menu text, Remote Connections is enabled. Conversely, when a check mark does not appear, Remote Connections will be denied. This menu option is a toggle; select it once to change its state. After the change is made, LoggerNet must be restarted for the change to take affect.

Though this may be a desirable feature, enabling Allow Remote Connections also makes your LoggerNet network configuration vulnerable to changes by other parties on the network. LoggerNet comes with a command line utility called CoraScript (Section 11). This utility can be used to create a back-up script of your network and troubleshoot problems. However, it is possible that another user could use CoraScript to make unwanted changes to the datalogger network, including changing the data collection schedule or deleting devices entirely. We therefore recommend that this feature be disabled, unless it is essential that the data be made available to a remote computer running a remote client application.

## 13.2 LoggerNet Data

LoggerNet Data is a suite of LoggerNet Client applications. The applications include RTMC Development version, RTMC Runtime version, Split, View, and DataFiler.

The first four of these applications is discussed elsewhere in this manual. The following section describes the use of the DataFiler.

### 13.2.1 Data Filer

DataFiler is a LoggerNetData application that is used to retrieve data from the LoggerNet data cache, and save the data to a file. It provides a means for a user to manually retrieve and store ASCII data on a remote PC, which can then be used for further analysis.

DataFiler is not shipped with the main LoggerNet application because it is redundant to the file creation functions already provided in LoggerNet; it must be purchased separately as part of the LoggerNetData software suite.

#### 13.2.1.1 DataFiler Requirements

DataFiler is an application that is capable of accessing the data in the LoggerNet data cache and storing that data to a file. The DataFiler can run on the same computer as the LoggerNet software, but more commonly it connects to a LoggerNet server computer over a TCP/IP connection. The LoggerNet server must be configured to allow remote connections; this is set up during the installation of the LoggerNet software (refer to Section 13.1 for additional information on allowing remote connections).

Because DataFiler retrieves data from LoggerNet's data cache (and not the datalogger directly), LoggerNet must first collect the data from the datalogger before it is available for use by the DataFiler. Data collection in LoggerNet can be performed manually by a user or automatically by setting up a data collection schedule.

For information on collecting data from a datalogger, refer to Section 5. For a description of LoggerNet's data cache, refer to Appendix D.

#### 13.2.1.2 Using the DataFiler

##### 13.2.1.2.1 Connecting to a Computer Running the LoggerNet Server Software

When DataFiler is first opened, it will prompt you for the Server Name, User Name, and Password of the LoggerNet server:

**Server Name** - The name of the computer on which the LoggerNet software is running. This must be the valid name of an existing computer or a TCP/IP address (in the form ###.###.###.### consisting of the IP network number, ###.###.###, and the host number, ###). If the LoggerNet server resides on the same computer as the DataFiler application, you can simply type in LocalHost for the server name.

**User Name** - Your user name on the LoggerNet server.

**Password** - Your password for the LoggerNet server.

---

**NOTE** The **User Name** and **Password** fields are required only if security has been set up on the LoggerNet server to which you are trying to connect.

---

Each time you start the DataFiler, you will be prompted to enter this information. However, the **Automatically log in with this information** check box can be selected to skip this window and use the information from the last session.

To specify a different LoggerNet server, select the **File | Select Server** menu option.

#### 13.2.1.2.2 *Setting Up the DataFiler*

Once connection to the LoggerNet server has been established, a list of dataloggers set up in LoggerNet will be displayed in the **Stations** field (left side of the window). To retrieve data for a particular station, use the mouse pointer to select the datalogger then set up the **Collection Options** (explained below), select one or more tables to be collected, and press the **Start Collection** button. The retrieved data will be stored to the directory and file name shown in the **File Name** field. The directory or file name can be changed by highlighting the table and pressing the **Change File Name** button.

Tip: Quickly choose all tables for the highlighted datalogger by selecting the <b>Select All</b> check box.
---

#### 13.2.1.2.3 *Collection Options*

##### *Collect Mode*

This option is used to specify what data will be retrieved from the LoggerNet data cache and stored on the remote computer by the DataFiler:

**All the Data** - Retrieves all records from the selected tables.

**Data Since Last Collection** - Retrieves all uncollected records from the selected tables.

**Newest Number of Records** - Retrieves a specific number of records from the selected tables by backing up the number of records entered in the **Number of Records** field and retrieving all data forward.

**Specific Records** - Allows you to specify a beginning record number and the number of records to collect after that record. The range of records to retrieve is specified by completing the **Starting Record #** and **Number of Records** fields.

**Data from Selected Date and Time** - Allows you to specify a span of time for data collection. When this option is selected, the **Starting Date/Time** and **Ending Date/Time** fields will be enabled.

### ***File Mode***

This option is used to determine how data will be stored in relation to existing data files with the same name:

**Append to End of File** - Adds new data to the end of the existing data file.

**Overwrite Existing File** - Replaces the existing file with a newly created file.

**Create New File** - Renames the existing file with a \*.bak extension, and stores the new data with the specified file name. Subsequent \*.bak files will be named \*.bak1, \*.bak2, etc. The most recently \*.bak file will have the highest number.

### ***File Format***

This option is used to determine the format in which the data file will be saved:

**TOA5** - Data is stored in a comma separated format. Header information for each of the columns is included, along with field names, units of measure (if they are available), and output processing types (average, sample, total, etc.).

**CSV** - Data is stored in a comma separated format, without any header information. This format is easily imported into spreadsheet applications.

### ***Starting Record Information***

The Starting Record Information is applicable if the Collect Mode is "Newest Number of Records" or "Specific Records".

For **Newest Number of Records**, enter a value into the **Number of Records** field. Data collection will include the number of records specified, prior to and including the last record stored (i.e., back up X number of records from the last record stored, and collect all records from there).

For **Specific Records**, enter values into the **Starting Record #** and **Number of Records** fields. The Starting Record is the first record that will be collected from the datalogger; data collection will continue until the number of records specified have been received.

### ***Starting Date/Time and Ending Date/Time***

The **Starting Date/Time** and **Ending Date/Time** fields are used when the Collect Mode is "Data from Selected Date and Time". The two fields are used to specify a range of records to collect, based on the records' time stamps.

To complete a **date** field, type in a date directly or click the arrow to the right of the field to display a calendar from which to choose a date. To complete a **time** field, type in the time directly or use the arrows to the right of the field to increase or decrease the highlighted time value.

### 13.2.1.3 The Collected Data

After data is collected for one or more tables, a Summary window will show the table name in the datalogger, the number of records stored, the first and last timestamps of the collected data, and the first and last record numbers of the collected data.

The stored file can be viewed by pressing the **View Data File** button. The DataFiler uses LoggerNet's View utility to display the ASCII file.

### 13.2.1.4 Determining the Data Available in the Data Cache

When a datalogger is selected in the **Stations** list, you can press the **View Data Info** button to display a Data Information table that indicates the number of records and range of record numbers in the LoggerNet data cache for each table in the datalogger. These are the records that are available for collection and storage by the DataFiler. The Data Information table is retrieved from LoggerNet when the window is opened. It can be updated by pressing the **Refresh** button.

**Table Name** - The name of the data storage table in the datalogger.

**# of Records** - The number of records in LoggerNet's data cache for the table. By default, the size of the data cache for each datalogger table is set to two times the size of the table in the datalogger. Once a datalogger table in the data cache has reached its defined size, the oldest record is deleted from the data cache when the newest one is written.

**Earliest Timestamp** - The time stamp of the first record in the data cache.

**Latest Timestamp** - The time stamp of the last record in the data cache.

**Earliest Record #** - The record number of the first record in the data cache.

**Latest Record #** - The record number of the last record in the data cache.

---

#### NOTE

Because the data cache is updated based on data collection from the datalogger, there could be additional records stored in the datalogger's memory which have not yet been retrieved to the data cache.

---

### 13.2.1.5 Record Number Anomalies

Under certain circumstances it may appear there is a problem with the number of records and their record numbers reflected by the Data Information table. It is possible for the oldest record to have a record number higher than the newest record. This is due to a combination of events.

Tables in dataloggers are configured as ring memory. Eventually, they will fill and the oldest records will be overwritten with newer ones. The LoggerNet data cache, too, is configured as ring memory, but sized to hold twice the number of records that can be stored in the datalogger (default size). When the datalogger compiles its program, it starts with record number 0; therefore, if something causes the datalogger to recompile its program (such as sending a program to the datalogger or using a keyboard display to alter the program

slightly) all of its tables will start with record number 0 again. Therefore, the record numbers reflected in the Data Information table may appear to be incorrect.

As an example, if the datalogger's internal table size were 100 records, LoggerNet's cache would be sized at 200 records. If both had rung around and LoggerNet's cache now held record numbers 201-400 and someone re-sent the same program to the datalogger, LoggerNet would not clear its data cache, but would continue to store the new records. These record numbers, however, would start at 0. After a short while as the new records were put into the data cache and old ones overwritten, the earliest record in the data cache might be 251 while the newest record number might be 50. In the data file, however, data would appear in correct sequence ordered by date/time stamps.

#### **13.2.1.6 Communication Status**

A box in the lower right corner of the DataFiler's window provides an indication of the DataFiler's connection with the LoggerNet server. When the DataFiler is in communication with the LoggerNet server, the box will appear green and the IP address or computer name (e.g., LocalHost) will be displayed. If communication with the server is lost (for instance, if LoggerNet is closed), the box will appear red with the text "No Connection". If communication is lost but the DataFiler is attempting to reconnect, the box will appear blue with the text "Attempting Connection".

### **13.3 RTMC Run-time**

RTMC Run-time is an application that allows you to remotely run real-time graphic display screens that have been created in the RTMC Development version. RTMC Run-time is discussed in Section 5 of this manual.

### **13.4 RTDM**

RTDM is a data display package that lets you design graphical screens that display data from data files collected by LoggerNet or from the LoggerNet data cache. It includes the ability to output image files (jpls, png, or gif) from the data display, with command line capability so that the files can be uploaded to a web site via a third-party RTP client.

Use of RTDM is not covered in this manual. Product literature and a user's manual can be downloaded from our web site at [www.campbellsci.com](http://www.campbellsci.com).

### **13.5 Baler**

Baler is a LoggerNet client application that allows you to set up a schedule on which data will be retrieved from the LoggerNet server and saved to a file on a remote computer. Tables are selected for baling from the dataloggers in the LoggerNet network. The Baler will create a separate file for each selected table. The data can be saved in TOAC1, TOA5, TOB, or CSV format.

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 stored, and once "baling" is started, data is saved in interval-sized packets or "bales".

Baler is not covered in this manual. Baler comes with a separate user's manual. Product literature can be downloaded from our web site at [www.campbellsci.com](http://www.campbellsci.com).

## 13.6 OPC Server

The OPC Server is a LoggerNET client that makes OPC data available to third-party OPC clients. OPC is an acronym for "OLE (Object Linking and Embedding) for Process Control". It is a set of industry standards, based on Microsoft's OLE technology, designed to provide a common interface between automation and control hardware and software. The OPC specifications were developed by a group of organizations involved in the automation and control industries in conjunction with Microsoft Corporation. CSIOPCServer is a server application developed by Campbell Scientific to provide data from its dataloggers, via LoggerNet data acquisition and management software, in an OPC format to other applications. The CSIOPCServer supports OPC Data Access Specification 2.05a.

The OPC Server is not covered in this manual. The OPC Server comes with a separate user's manual. Product literature can be downloaded from our web site at [www.campbellsci.com](http://www.campbellsci.com).

## 13.7 Software Development Kit

The LoggerNET-SDK Software Development Kit offers a flexible and powerful programming tool to easily create innovative applications that communicate with our dataloggers. Applications developed using the SDK require a LoggerNet license for use. PC-SDK features three ActiveX® controls that dovetail into an integrated development environment such as Microsoft's Visual Basic® or Borland's Delphi®.

LoggerNet-SDK's controls encapsulate some of the fundamental tasks users want to perform with our dataloggers. The "CsiBrokerMap" control browses the current datalogger network to display station and data table information. The "CsiDataLogger" control checks and sets the datalogger's clock, sends and receives datalogger programs, collects data from the datalogger, toggles ports and flags, and sets input location variables. You can use the "CsiDataSource" control to retrieve the logger's historical or realtime data from the LoggerNet server's cache. PC-SDK includes a Beginner's Guide, a Programmer's Reference, and examples of how to build applications with each ActiveX control using both Visual Basic and Delphi.

The LoggerNet SDK is not covered in this manual. The SDK comes with a separate user's manual. Product literature can be downloaded from our web site at [www.campbellsci.com](http://www.campbellsci.com).



# Section 14. 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 4 if you need general information on adding devices to the device map.

---

## 14.1 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 collection intervals and the communication time required between the computer and the RF base.

LoggerNet 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. LoggerNet will also initiate a call for any datalogger operations such as connect in the Connect screen or get table definitions in the Setup screen.

### 14.1.1 Setup

The device map set up in the Setup Screen 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.

## 14.1.2 Operational Considerations

### 14.1.2.1 Scheduled Data Collection

The intervals for scheduled data collection need to be set up to allow time for the communications link to be established. Keep in mind that the phone modem requires 15-20 seconds to dial and establish communication. Also each link in the RF connection requires 3-5 seconds to link to the next device. This means that a phone to RF network with 2 repeaters could take over 30 seconds to establish the link and start collecting data. The amount of time required for collecting the data will vary, depending upon the amount of data to be collected and the speed and integrity of the communications link. Some time is also required to close down the link before another station can be contacted. Make sure the collection schedule allows enough time for each station to be contacted before contacting the first station again.

### 14.1.2.2 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.

### 14.1.2.3 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 Setup.

### 14.1.2.4 Max Time Online

The Max Time Online setting is used to force LoggerNet to terminate a connection if the maximum time limit is exceeded. This is used to prevent a phone modem from getting stuck online and incurring extra long distance phone charges or inhibiting scheduled data collection from other stations. In phone to RF systems this value should be long enough to allow data collection for the anticipated amount of data. If the Max Time Online is reached, LoggerNet will force the connection to close, even if it is in the middle of collecting data.

## 14.1.3 Attaching a Datalogger to the RF Base

Connecting a datalogger at the RF Base in a Phone to RF system requires some additional configuration. The operational considerations are the same as for the standard Phone to RF network. (See section 14.1.2.)

### 14.1.3.1 Hardware Setup

The RF modem in the RF Base has to be configured to work in Synchronous Device Communication (SDC) mode. This is done by changing the 9<sup>th</sup> DIP switch inside the RF Base modem to a 0 or closed. This will allow the datalogger to pass communication to the RF Base.

---

**NOTE** SDC mode cannot be used with 21X or CR7 dataloggers or any of the table based dataloggers.

---

The datalogger and the RF base must both be connected to the remote phone modem on the same 9 pin ribbon cable.

### 14.1.3.2 Network Setup in LoggerNet

Setting up a datalogger at the RF base in a Phone to RF system requires that the datalogger be connected as a child of the remote phone modem, and the RF Base be connected as a child of the datalogger. Then the RF network is connected to the RF Base. An example of this type of configuration is shown below.




---

**NOTE** In Phone to RF systems with a datalogger at the RF base, you should set all three levels of datalogger security so the computer has to unlock the datalogger before getting data. This will prevent a situation where the computer can be getting data from the wrong datalogger when a connection to a remote station fails.

---

## 14.2 Phone to MD9

### 14.2.1 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.

## 14.2.2 Operational Considerations

### 14.2.2.1 Scheduled Data Collection

The intervals for scheduled data collection need to set up to allow time for the communications link to be established. Keep in mind that the phone modem requires 15-20 seconds to dial and establish communication. Some time is also required to close down the link before another station can be contacted. Make sure the collection schedule allows enough time for each station to be contacted before contacting the first station again.

### 14.2.2.2 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 Setup for the MD9 remote modem must match its hardware configuration.

### 14.2.2.3 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.

### 14.2.2.4 Max Time Online

The Max Time Online setting is used to force LoggerNet to terminate a connection if the maximum time limit is exceeded. This is used to prevent a phone modem from getting stuck online and incurring extra long distance phone charges or inhibiting scheduled data collection from other stations. In

phone to MD9 systems this value should be long enough to allow data collection for the anticipated amount of data. If the Max Time Online is reached, LoggerNet will force the connection to close, even if it is in the middle of collecting data.

### 14.2.2.5 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.

## 14.3 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.

### 14.3.1 Setup

The device map set up in the Setup screen for a TCP/IP to RF link would look similar to the communications network below.



To begin, add an IPPort to the device map if one does not exist. Add an RF base modem to the IPPort, 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.

### 14.3.2 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).

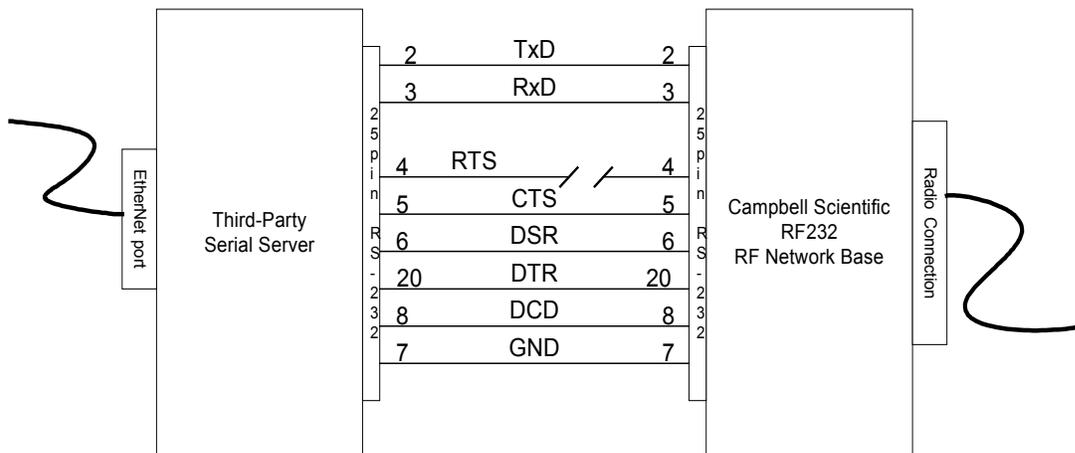
**Max Time online** – This should be set to zero (disabled) for all of the stations in the RF network. Otherwise, when the communications link is dropped because this value is exceeded, communication will be re-attempted immediately. Forcing the connection offline and back on quickly causes errors because not enough time is allowed for the serial server to reset the TCP/IP socket.

### 14.3.3 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.

When connecting the RF Base to a Serial Server or the RS232 connector of the NL100 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. Do not use leading zeros for any of the numbers in the IP address. An address of 198.192.035.002 would cause an error in the network configuration.

**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 Setup for the IPPort device. For example, if the port ID was specified to be 3201, using the IP address above the entry in Setup 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.



# 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 Terminal Emulator and Data Table Monitor that can be useful in troubleshooting LoggerNet problems.*

## 15.1 What's Changed?

When things stop working the most important thing to ask yourself is: “What’s changed?” A new computer, new software (especially non-Campbell Scientific software), different communication peripheral such as a new modem, new datalogger program, etc., can all interrupt communications. If you can’t deduce the nature of the problem, go back to the original configuration. If a phone telecommunications or socket link starts to fail for no apparent reason, ask someone else to try it from a different computer. Try swapping out components – one at a time – from a link in the network that you know works or from spare equipment that you are certain is functioning correctly. Sometimes the smallest thing – a cable or a new PC utility program – can cause widespread havoc. If using TCP/IP or cellular telephone communications, check with the network administrator to see if anything changed that coincided with the loss of communications.

## 15.2 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.

### 15.2.1 Starting LoggerNet and Connecting to the Server

**Problem:** Cannot start LoggerNet on a laptop computer. (Error message “LoggerNet was unable to start the communications server.” Socket Failure creation 10047.”)

**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 non-existent dial-up service with a fixed IP address. There is a description of this procedure in Section 1.2.

**Problem:** Message indicating Server Connection Lost.

**Remedy :** This message indicates that the main communications software has stopped responding to the user interface screens. You need to close down all of the applications along with the Toolbar and start over.

## 15.2.2 Socket Errors

The LoggerNet Server uses TCP/IP sockets for communications. Various problems can occur with these socket connections. Some of the most common errors and remedies are listed below.

### Maximum Number of Sockets Open

The Windows operating system has limits on the number of socket connections that can be held open. For Windows NT/2000/XP this number is about 5000. For most operations this should be more than enough to cover the open applications that use sockets. One situation that does cause problems is using the IPPorts to communicate with dataloggers where the socket is being opened and closed quickly. For example if you have 20 stations on IPPorts and you do normal data collection every 5 seconds, 20 new sockets are created every 5 seconds. The normal lifetime of the created socket is about 2 minutes leaving over 500 active sockets at a time. If there are other applications that use sockets, it is possible to exceed the allowed number of sockets.

To work around this problem, either slow down the rate of data collection, or keep the stations on line. There is an application called Managecomms.exe in the LoggerNet program file directory. The documentation for this program is also included in an Adobe Acrobat PDF file. Use this program to maintain an active connection to all of the stations so the sockets are open all the time.

Note that if you are using Windows 95, 98, or ME, the number of sockets defaults to about 100.

### Socket Error Messages

When you get an error message that says Socket Error and a number, check the chart below for the type of error that occurred and what to do about it. Note that these error messages can show up either in pop up error boxes or as part of the LoggerNet Communications log.

Socket Error Number	Message Meaning	User Response to Message
10013	<i>Permission Denied.</i> The requested socket connection has refused the connection.	This is normally a network type of issue. Check with your computer network operator.
10024	<i>Too many open files.</i> Too many open sockets for the applications running.	This can occur when you have many applications that are using sockets running at the same time.
10047	<i>Address family not supported by protocol family.</i> The socket being addressed does not support the type of connection being attempted.	This message shows up when the LoggerNet Toolbar comes up but the server did not come up because TCP/IP is not installed on the computer. Install TCP/IP and restart LoggerNet. (Section 1.2)
10055	<i>No buffer space available.</i> Cannot create more temporary sockets.	The operating system cannot create any more socket connection. See the text above about Maximum Number of Sockets Open.
10058	<i>Cannot send after socket shutdown.</i> A message was sent to a socket that has been closed.	This would be an indication that an application is not communicating well with the server. Check the application.
10060	<i>Connection timed out.</i>	Either the server has crashed and is not responding or the application did not maintain the connection to the server. Try restarting LoggerNet. This message can also be seen in connection with the NL100 LAN interface.
10061	<i>Connection refused.</i> The LoggerNet server or an NL100 refused to allow the socket connection.	This is normally associated with the NL100 and occurs because the last connection did not have enough time to close before a new connection is requested. Slow down the low level polling delay interval.
10065	<i>No route to host.</i> The application is trying to connect to a host address that isn't in the routing table.	This occurs with remote connections to a LoggerNet server running on another computer. The requested host name can't be found.

### 15.2.3 Data Collection Issues

**Problem:** Scheduled data collection is enabled but no data is being saved in the data files, or data is not updating on numeric or graphical display.

**Remedy 1:** Make sure that communications are enabled for the datalogger and all the devices in the communications link.

**Remedy 2:** For table-based dataloggers, make sure tables are included for collection and the table definitions are current. For array-based dataloggers, make sure the correct final storage areas are included for collection.

**Remedy 3:** Check communication state on Status Monitor. An indication of Primary or Secondary retry indicates that LoggerNet is waiting for the next collect time. If the station is waiting in Secondary Retry mode, open the Connect Screen and click the Collect Now button. This will force collection from the datalogger and will return the datalogger to the normal collection state.

Invalid Table Defs indicates that LoggerNet does not have a current copy of the datalogger table definitions so you have to Get Table Defs from the Setup screen.

Network Paused indicates that data collection for the entire network has been suspended. So look on the Status Monitor for the Pause Schedule check box.

## 15.3 Application Screen Problems

**Problem:** Table-based datalogger program won't compile and returns an E43 error.

**Remedy:** There is an interval mismatch between the P84 instructions and the program scan rate. If the P84 intervals are not multiples of the scan rate, the program won't compile and will return this error.

## 15.4 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 in the Setup Screen for the datalogger to a smaller number.

**Remedy 2:** On noisy phone links try lowering the max baud rate for the datalogger on the Setup screen.

**Remedy 3:** On RF networks, make sure that the extra response time is sufficient for the reply to come back, especially if there are

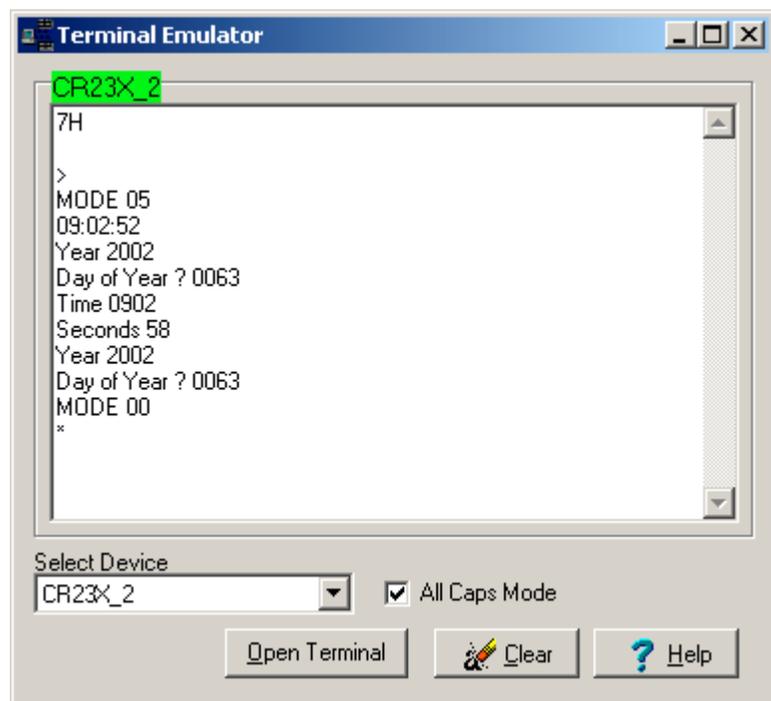
repeaters in the network. Use the minimum value necessary to make the link function; usually less than 5 seconds.

## 15.5 Terminal Emulator to Test Communications

Terminal Emulator is a utility to test communications with the devices in the datalogger network. Terminal Emulator is accessed from the Test menu of the Toolbar. The operation of a ComPort and the connection to a phone modem can be tested.

The Terminal Emulator utility is available from the main toolbar to help troubleshoot communications problems. When you choose a device with the Select Device field, the Terminal Emulator will attempt to establish communications with that device. The Terminal Emulator will use the lowest baud rate among all of the devices involved in the link. For example if choosing a COM port, the baud rate will typically be 115,200 baud and LoggerNet simply opens the port. For a phone modem, the baud rate will be set to the value in the Setup screen for that phone modem, the COM port will be opened and the DTR line will be asserted to enable the phone modem. For a datalogger on a phone link, the baud rate will be set for the lower of the phone modem or datalogger baud rates in the Setup screen, AND LoggerNet will try to dial the phone modem and get a prompt from the datalogger. You can also use terminal emulation to send commands to the dataloggers.

When the Terminal Emulator screen comes up as shown click the drop down arrow to the right of the Select Device box to choose the device from the list of devices in the network map. The correct baud rate for the link is automatically set. The characters you type in the window are sent as ASCII text to the selected device. The options that are available from this screen depend on the device you select.



### Dataloggers

The example above shows a terminal emulation session with a datalogger. Once you have selected the datalogger click Open Terminal to start communications. Array-based dataloggers require you to type in the letters 7H (2178H for CR7 and 21X) and press Enter to establish terminal emulation mode. Table-based dataloggers are ready for terminal emulation when they are first selected. Just press Enter. For a definition of commands that are available in terminal emulation mode see the datalogger operators manual or prompt sheet. On CR5000 dataloggers type H when the prompt comes up and a list of options will be displayed.

---

**NOTE**

Use caution while in terminal emulation mode. You can change or disable operation of the datalogger with these commands.

---

### ComPort

You can use the Terminal Emulator to perform a communications test on a ComPort. To perform a feedback test, select the ComPort and click Open Terminal. Then connect the Transmit and Receive lines (2 and 3) of the serial port cable using a small screw driver or paper clip. Click in the window to get the cursor and type some characters on the computer keyboard. If the characters are echoed back to the screen, the ComPort is working.

The characters on the screen can be cleared by clicking Clear.

### Phone Modems

Selecting a phone modem and clicking Open Terminal will allow you to send ASCII characters to the phone modem. This can also be used to test communications to the phone modem and the initialization strings used to set up and configure it. Get the information for the available commands and format from the modem manufacturer.

An example of how to troubleshoot a datalogger on a phone link might follow this sequence.

Say that you cannot communicate with a datalogger and you don't hear the phone modem dial the number. You could wonder if you are using the correct COM port. Disconnect the serial cable from the phone modem, select the COM port, and click Open Terminal. Then, shorting pins 2 and 3 at the end of the serial cable, type characters and, if you've chosen the correct COM port, you should see those characters echoed to the screen. (Note: the RS232 protocol allows any pins on a cable to be shorted without damaging the computer. Pins 2 and 3 are the second and third pins from the upper left on the top row when looking at the male end of the cable, with the long row of pins on top. This is true for either 9-pin or 25-pin cables.)

Once you have established that you have the correct COM port, you may hear the modem dialing, but the datalogger doesn't connect. Perhaps the modem is not dialing correctly or using an appropriate initialization string. You can work with the modem by selecting it in Terminal Emulator and clicking Open Terminal. LoggerNet will open the COM port and raise DTR to enable the

modem. You can then type characters to be sent to the modem. For example, assume you have a Hayes-compatible modem and an array-based datalogger on a phone link with the phone number “752-7779”. You could test the link with the following sequence:

Type	ATH <Enter>	To hang up the modem. You should see an “OK” on the screen sent by the modem. If you do not, perhaps there is no modem attached to that COM port or perhaps the modem is not powered on.
Type	AT&F <Enter>	To put a typical Hayes-style modem back to the factory defaults. You should see an “OK” echoed by the modem. If you do not, perhaps it’s not a Hayes-compatible modem. For example, many U.S. Robotics modems require “AT&F1”. Older 1200 baud modems may require “ATZ”.
Type	AT&C1&D2 <Enter>	To cause the modem to use hardware flow control and report the loss of the carrier when the datalogger hangs up its modem. You should see “OK” appear on the screen. If you see “ERROR”, then the modem doesn’t recognize one or both of the “&C1” or “&D2” commands.
Type	ATV1 <Enter>	To force the modem to follow the serial port rate rather than try to connect at the fastest rate the remote modem will support. You should see “OK” appear on the screen. If you see “ERROR”, then the modem doesn’t recognize this command.
Type	ATDT752777 9 <Enter>	To dial, using tone dialing, a datalogger at “7527779”. You should hear a dial tone, followed by a series of beeps and tones, followed by what sounds like white noise or scratching sounds and screeches. When the modems connect, you should see the word, “CONNEC T”, appear on the screen, perhaps followed by information about the speed and type of connection. If you don’t hear a dial tone, perhaps you haven’t plugged in the telephone line or the line is not operational. If you hear the dial tone and the beeps of the phone number being dialed, but there’s no ring on the other end, perhaps the number isn’t valid or you didn’t “get an outside line”. If a person answers, perhaps you have the wrong phone number. If it rings, and answers with tones and screeches, but you don’t get a “CONNECT” message, perhaps you have dialed a fax number, or other data modem, but not one with a datalogger attached or the datalogger is the wrong type.

Type	<Enter> <Enter> <Enter> about 2-3 seconds apart	To send carriage return characters to the array-based datalogger. If the datalogger recognizes these characters, it will send back an asterisk, “ * ” for every <Enter> keystroke. If it does not, perhaps it’s not an array-based datalogger, or you’ve chosen a baud rate that’s too high for the datalogger or the quality of the phone line is too poor to support that baud rate.
Type	A <Enter>	You should see a string of characters from the datalogger that report its status, including final storage pointers, memory size, perhaps lithium battery voltage, and internal error counters. (See Section 5 in the datalogger manual for more information on telecommunication commands.) If you do not, then the phone line may be too poor to support communications. A long series of nonsense characters usually indicates electrical noise in the vicinity of the telephone cable that the modems are interpreting as high and low digital signals and reporting them as characters.
Type	E <Enter>	To hang up the datalogger, which causes it to turn off its phone modem, which in turn causes loss of the carrier signal between the modems. After a few seconds you should see “NO CARRIER” reported by your base modem.
Type	ATH <Enter>	To hang up your base modem.

If the modem you select in LoggerNet’s Setup screen doesn’t work, check to make sure you’ve selected the correct modem, that it’s powered up and that the phone line is working. You may have to adjust the baud rate. If you still have trouble, you may need to consult your modem manual for the appropriate initialization strings.

Perhaps you can communicate with other dataloggers on this phone line and with this base modem, but there’s one remote datalogger that’s problematic. You could try communicating with that datalogger in remote keyboard mode. Using the example above, choose instead the datalogger in the Terminal Emulator and click Open Terminal. You should hear the phone modem dial, followed by screeches as the modems negotiate a connection, followed by “CONNECT”, etc., and then a response from the datalogger. Pressing <Enter> for an array-based logger should return an asterisk “ \* ”. Typing “A” <Enter> should return the status line. Type “7H” (“2718H” for 21X or CR7X dataloggers) to put the datalogger in remote keyboard mode. From there, you can enter commands much like from a keyboard/display handheld interface. Pressing “\*6” followed by several <Enter> keys should cause the datalogger to report its input locations. If all you get from some of these commands is “MODE”, perhaps the datalogger has security set. See your datalogger manual for other remote keyboard commands. You may also call your Campbell Scientific application engineer for more help on troubleshooting links.

**NOTE**

---

Using remote keyboard mode can result in loss of programs, data, or the ability to further communicate with a datalogger over a remote link, for example, by altering security settings or changing a program leading to memory resets or powering down a cellular phone. Remember that keystrokes entered may not reach the datalogger intact. That is, the datalogger may not receive what you send.

---

## 15.6 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 poor line-of-sight or other environmental factors. There are a number 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.

### 15.6.1 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. (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.
3. Check that the radio is set for the right frequency. With a programmable radio, verify that the correct frequency and other settings are set properly. If the radio is crystal based there should be a 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 for a clear line-of-sight 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  $\Omega$ .

One simple, but very effective, technique is to swap out components. Use components from a part of the network that you know is working, and swap them out one at a time to isolate a faulty hardware component.

## 15.6.2 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 watt meter is needed such as the Bird Electronic Corporation's Model 4304A Wattmeter. Proper connectors are also needed to place the watt meter 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 push-button 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 watt meter in series between the radio and antenna cable. Set the watt meter to the 15-Watt range, or the next highest watt meter setting, and point the directional arrow first toward the antenna cable to measure forward power ( $W_f$ ). Initiate radio communication, let the watt meter stabilize, and record the watt meter reading. Reverse the directional arrow so it is pointing back toward the radio, initiate radio communication, let the watt meter stabilize, and record the watt meter 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.

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.) Pull gently on the cable to make sure the connectors are still attached securely.
- 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 power transmission is good. However, be sure the antenna is oriented properly.

While at the station, check the voltage on the 12 V 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.

### 15.6.3 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 inadequate 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.
- Improper antenna alignment.
- Outside interference on the channel frequency from another source.

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 with a way to talk to each other.

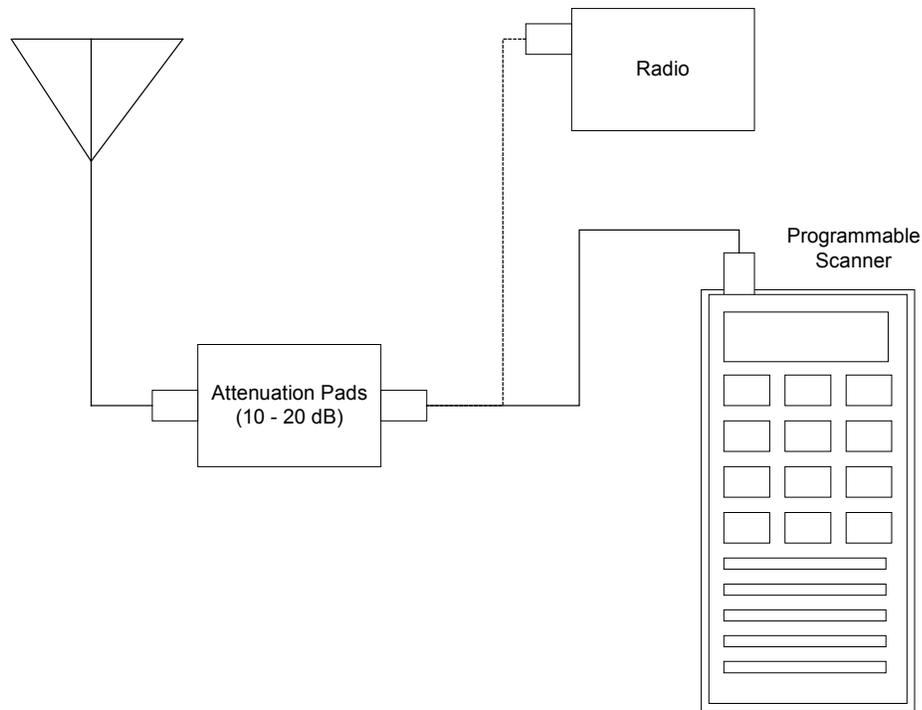
If the carrier detect light is coming on at the RF base station radio, but communication quality is poor or not being set up properly, there may be a marginal or low signal power inherent in the RF link. In this case, it is a good idea to do a signal power check with attenuation pads for each sub-link 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, a signal can be detected on the radios with a power greater than  $-115$  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, are inserted into the link and subsequently the radio will not detect the signal, 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 detect the signal, 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 ambient RF noise is just cut out. 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. 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 push-button 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.

---

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.

## 15.7 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 by 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.

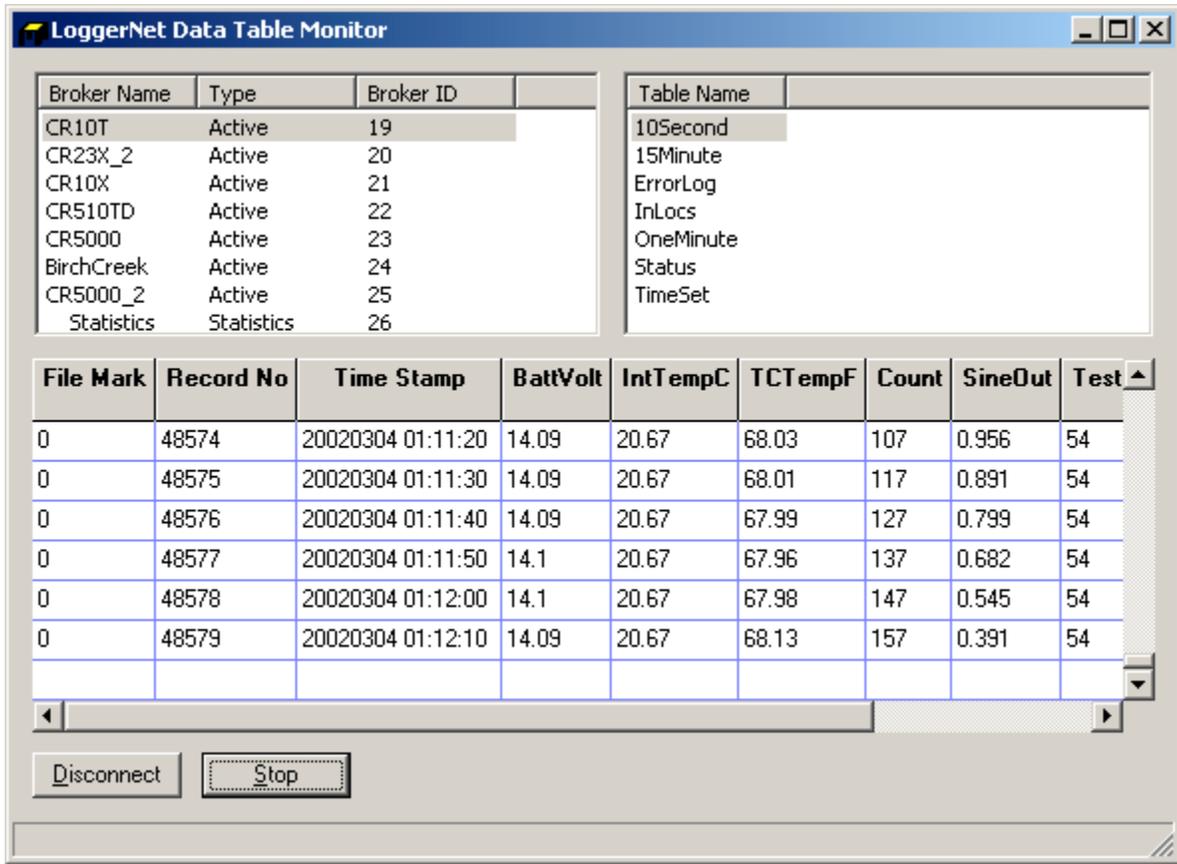
---

### CAUTION

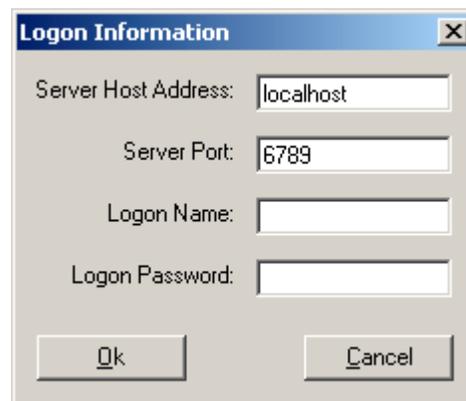
One caution about the data file created by Data Table Monitor—there are no limits to size or longevity. If you plan to use the export to file feature on a regular basis, make sure to either restart Data Table Monitor (which overwrites the exported file) or delete the files periodically. The data export can easily be restarted by clicking the Start button. This will delete the old file and start a new one.

---

To start Data Table Monitor open Windows Explorer and go to the Program Files\CampbellSci\LoggerNet directory. Double click the Tablemon2.exe file. The utility will start with a screen similar to the one shown below.



Click the Connect button to connect to the LoggerNet server. The dialog box shown below will be displayed. If you are working on the same computer where LoggerNet is running leave the default Server Host Address as localhost. The Server Port number should also be 6789. The Logon Name and Logon Password are only used with versions of LoggerNet that support security. To connect to LoggerNet on another computer, enter the computer network name or IP address as the Server Host Address. When you click OK a list of the dataloggers in the network will be shown in the upper left window.



Selecting a datalogger will list the names of the data tables or array IDs in the datalogger. Note that if data collection has not been set up and enabled in the Setup Screen, no data will be coming into the data cache. Data Table Monitor

can only display and output data from the data cache. Data Table Monitor displays and outputs all the data points from an array or table.

Click the Start button to bring up the Start Advise Options dialog. This dialog gives you choices about which records to display and the data file in which to store them.



**Start Option:** This selects the starting point for the data to be displayed and output to the file.

- **At Record:** 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.
- **At Time:** This option allows a selection of the starting position based on the timestamp in the data. The time and date are set in the Begin Date field. All of the records available after this timestamp are output.
- **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.
- **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.

- **Relative to Newest:** This option starts from the most recent record collected. The Offset from Newest specifies how much time to go back from the current write index. For example, an offset of 10 with a setting of minutes will get the last 10 minutes of data collected.
- **At Offset from Newest:** This option allows you to specify how many records back from the current write index to go. A setting of 10 in the Start Offset box will display the last 10 records collected.

The Start File Mark, Start Record Number, Start Offset, Begin Date, and Offset from Newest edit boxes are used only with the corresponding start options above. For each option selected, the appropriate boxes are enabled.

Once the start options have been set, click 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.

#### **Ordering Option:**

- **Collected:** 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.
- **Logged 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.
- **Logged 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.
- **Real Time:** the most recent data is always sent out starting with the last record stored. This will not provide a complete data set.

---

#### **NOTE**

Holes in the data only apply to table-based dataloggers that are using data advise collection. This option is supported in version 1.1 of LoggerNet and will be supported in future versions of LoggerNet 2. Only the Collected and Real Time options apply for array-based dataloggers.

---

Set the output file directory and name in the Export File box. The Browse button will bring up a Windows Save As dialog box to select the file name and directory.

Click OK to begin displaying the data.

## 15.8 Troubleshooting PakBus Communications

**Problem:** LoggerNet can't communicate with in-range PakBus datalogger

(PC-RF400~~~RF400-CR510PB~~~ RF400-CR510PB~~~CR205)

**Possible reason 1:** LoggerNet's PakBus Address for datalogger doesn't match datalogger's PakBus Address.

**Remedy 1:** Make them match.

**Possible reason 2:** An RF400 Series radio is set to a different Hopping Sequence, Network Address, Radio Address, or Standby Mode.

**Remedy 2:** Set both radios exactly the same in the above parameters. ('base' radio's Active Interface is typically Auto Sense, remote radio is typically CSDC 7)

**Possible reason 3:** No PakBusPort in device map between root and datalogger.

**Remedy 3:** Add PakBusPort.

**Possible reason 4:** No PakBus OS in datalogger

**Remedy 4:** If you have two in-range routers using neighbor filters, in order for them to discover one another you must list each of them as a potential neighbor in the other's neighbor filter.

**Problem:** LoggerNet can't communicate via datalogger-router to a certain remote datalogger.

(PC-RF400~~~RF400-CR10XPB~~~RF400-CR510~~~CR205)

**Possible reason 1:** A datalogger router has insufficient \*D15 max nodes, max neighbors, or max routers configured.

**Remedy 1:** Increase the \*D15 numbers.

\*D15 settings of 000x, 6, 6, 6, 0 are reasonable for a router in a network of under a half dozen nodes

\*D15 settings allocate memory similar to \*A. It is a good idea, when configuring \*D15 settings, to leave 'room to grow.' Changing \*D15 settings later on could result in loss of data as \*0 is entered to compile new settings.

**Possible reason 2:** Last datalogger router has no means configured of discovering the remote datalogger.

**Remedy 2:** Configure the datalogger router with either a neighbor filter or a beacon. Make sure Neighbor Filter potential neighbors

include the remote datalogger's address. Whether you set up a beacon or Neighbor Filter make sure the port so configured matches the communications device port configuration. For example, if the selected neighbor filter port is "17", make sure that the RF400 Active Interface is "CSDC 7."

**Possible reason 3:** An RF400 Series radio is set to a different Hopping Sequence, Network Address, Radio Address, or Standby Mode.

**Remedy 3:** Set all network radios exactly the same in the above parameters. Network RF400s' Active Interfaces may vary from node to node, however, they will typically be configured for CSDC 7 or 8 except for a 'base' radio which is typically AutoSense or M.E.. Dataloggers automatically detect the RF400's port (Active Interface) for packet communications, however, the potential neighbor hello port or beacon port must be configured to match the RF400's Active Interface, or no discovery of neighbors will take place.

**Possible reason 4:** A CR200 Series has just received an OS download resetting network address, radio address, hopping sequence, and radio power mode to defaults.

**Remedy 4:** Configure CR205 settings to agree with network.

**Possible reason 5:** The two routers in the path to the CR205 have neighbor filters and at least one of the neighbor filters doesn't list the other as a potential neighbor.

**Remedy 5:** If you have two in-range routers using neighbor filters, in order for them to discover one another you must list each of them as a potential neighbor in the other's neighbor filter.

**Problem:** Can't LoggerNet communicate with datalogger-router in network?

(PC-RF400~~~RF400-CR10XPB~~~CR205)

**Possible reason:** The datalogger-router has more than one M.E. peripheral cabled to it.

**Remedy:** Connect only one M.E. device to a datalogger (or change one M.E. peripheral to CSDC 8 port).

**Problem:** Changed P190 port type and it no longer communicates with remote.

**Possible reason:** The Active Interface of the communications device (for example, RF400 Series) no longer matches the P190 port.

**Remedy:** Make the communications device Active Interface agree with P190 Parameter 1.

**Problem:** Rapid spurious communications lasting a few seconds at a time between devices in RF400 network.

**Possible reason:** Two network devices have the same PakBus Address.

**Remedy:** Change one of the duplicate PakBus Addresses. Make all addresses unique throughout the network.

**Problem:** In P193 network, certain CR200 Series devices don't transfer data.

**Possible reason:** The Master datalogger's \*D15 setting configures too few max nodes, max neighbors, and max routers.

**Remedy:** Change Master \*D15 settings for max nodes, max neighbors, and max routers to larger numbers reflecting the network size.

\*D15 settings allocate memory similar to \*A. It is a good idea, when configuring \*D15 settings, to leave 'room to grow.' Changing \*D15 settings later on could result in loss of data.

# ***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** – See Mixed-array Datalogger.

## **B**

**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 typically 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 frequency (RF) site is accessed through and a child of 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. Email programs running on individual PCs are typically client applications that connect to an email server program running on a computer at an Internet Service Provider to receive and send email messages.

**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.

**Communication Server** – The software (typically packaged as a DLL) that provides the communications functions within other software such as PC200W, PC400, or LoggerNet.

**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, monitor pulse signals, control Synchronous Devices for Measurement (SDM), or used as data input/output connections for SDI-12 sensors.

**CoraScript** – A command line interpreter client to the LoggerNet server 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** – Any of the Edlog dataloggers with table-data operating systems become “TD” dataloggers, including the CR10T, CR510-TD, CR10X-TD, and CR23X-TD.

**CRBasic** - The programming language used for CR200, CR5000 or CR9000 dataloggers. Short Cut or the CRBasic Editor are used to create program files for these dataloggers.

**CRBasic Datalogger** – A CR200/205, CR5000 or CR9000 datalogger. Sometimes referred to as “CRx000 dataloggers.”

**CRx000 Datalogger** – Generally, a CR200/205, CR5000 or CR9000 datalogger. More correctly referred to as “CRBasic dataloggers.”

## D

**Data Advise (Datalogger)** – A mutual agreement between the communication 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 communication 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 communication server. This data is stored in binary files on the hard disk of the computer where the server is running.

**Data Collection** - Getting a copy of the data stored in the datalogger and saving it in the communication server's data cache (compare to Data Retrieval).

**Data Point** - A data value that is sent to Final Storage as the result of an Output Instruction. A group of data points output at the same time makes up a record in a data table.

**Data Retrieval** – Sending a copy of the data from the communication server's data cache 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** – A field in the status table that shows the number of days before any of the tables using automatic record allocation are filled.

**DevConfig** – Short for “Device Configurator”, a software application that provides a graphical user interface to configure settings in dataloggers and communications peripherals. Available in PC400, LoggerNet, and as a stand-alone application from the Campbell Scientific website. (Supplants CSOS.EXE, PakCom, and stand-alone terminal emulators.)

**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** - An ASCII file that can be sent to program an Edlog 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 file name extension. A program must be sent to the datalogger before the datalogger will begin to collect data.

## E

**Edlog** - Campbell Scientific's software application used to create new or edit existing datalogger programs. Edlog supports all of the programming capabilities in the dataloggers it supports. (Program generators such as Short Cut are necessarily more limited in the features they can support.)

**Edlog Datalogger** – Any of the dataloggers, 21X, CR7, CR10, CR500, CR10X, CR510, or CR23X. The default operating system for these dataloggers is a mixed-array configuration. Some of these, specifically the last three, can have alternative operating systems installed by users. These include mixed-array, table-data (TD), or PakBus (PB) operating systems.

**EEPROM** - Electrically erasable programmable 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 and DevConfig).

**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** - Sensors utilizing electrical bridge circuits require a precise electrical voltage to be applied. The excitation channels, marked as E1, E2, etc., on the datalogger wiring panel, provide this required precision 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 area or 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 (use analog ground if available)
- a power return path (do NOT use analog ground for power return)
- a connection for cable shield wire to help reduce electrical noise (do not use analog ground for shield wires, also known as drain wires)

## 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** – When using Data Advise, the communications server always gets the most recent data records, so if there are more records to be returned than can fit in one packet 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 collects these holes only if that option is enabled. 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 data records missing from the data cache but possibly still in the datalogger. 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 communication server software is running.

## I

**INI Files** - Configuration files that are used to preserve the last known setups or states 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 (such as a running total and number of samples for an average calculation), 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.

**LDEP** – Logger Data Export Protocol, a protocol and client application that provides for data distribution from the communications server to a third party application through a standard TCP/IP socket. Installed with LoggerNet; see the associated PDF file for more information. Requires record-specific acknowledgements for record flow control. See LDMP.

**LDMP** – Logger Data Monitoring Protocol, a protocol and client application that provides for data distribution from the communications server to a third party application through a standard TCP/IP socket. Installed with LoggerNet; see the associated PDF file for more information. Requires very simple acknowledgements for record flow control. See LDEP.

**Log Files** - Text files that are stored on the computer's hard drive that record activity. They contain information about communications between the communications server and other devices in the datalogger network. Log files are typically used for troubleshooting purposes. LoggerNet has four types of log files: Transaction, Communications Status, Object State, and Low Level I/O. Refer to Appendix C or the help within the LogTool (in PC400 click the Tools | LogTool menu item) application 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** – Values stored by the datalogger in an Input Location after reading an electronic signal from a sensor and converting the raw signal into meaningful units.

**Mixed-array** – Dataloggers with mixed-array operating systems save output in a common area of 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". "Mixed-array dataloggers" typically save all information that is directed to output storage to the same area of datalogger memory (as opposed to table-based dataloggers that always store different output processing intervals to separate tables in datalogger memory). Data retrieved by the PC must be processed by PC software to separate the data based on the array IDs.

**Modem** – From “modulator-demodulator”; a device used to transmit and receive digital data over normally analog communications lines, such 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 that performs an analog-to-digital conversion which permits its attached computer to use the data.

## N

**Net Description** – Description of dataloggers and communications devices that form the datalogger network. Created using the EZWizard in PC400 or Setup screen in LoggerNet to communicate with the various dataloggers.

**Node** – Part of the description of a datalogger network. Each node represents a device that the communications server will dial through or communicate with individually. 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 at which the datalogger writes data to Final Storage. The output interval is defined by Instruction 84 in Edlog (for table-based dataloggers) or the instructions that set the output flag high in mixed-array 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. See the operator's manual or programming software for output processing options available for each datalogger model.

**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 or PakBus devices that are communicating. Each packet can contain data, messages, programming, etc. Usually contains addressing and routing information.

**PakBus** – A packet-based and packet-switched networking protocol used by newer dataloggers. PakBus allows for robust transmission of commands and data, dynamic routing between PakBus devices, and peer-to-peer communications (such as when one datalogger needs to control another datalogger without involving the PC).

**Parameter** - Number or code which helps 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. Some communications devices, such as RF232T radio bases or repeaters can also 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 typically 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 in Edlog dataloggers can be separated into two tables, each having its own execution interval. A third table is available for programming subroutines that may be called by instructions in Tables 1 or 2. Programming in CRBasic dataloggers can be separated into different "scans". The length of the program tables or scans is constrained only by the total memory available for programming.

**PROM** - Programmable Read-Only Memory - integrated circuit chips that are used to store the Operating System (OS) in the CR10T datalogger and some other communications peripherals. The PROM can be replaced to install a new operating system (also 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" - 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. Some Edlog dataloggers (CR10X, CR510, and CR23X) and all CRBasic dataloggers have battery backups that maintain the clock even when 12V 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) in TD dataloggers or the DataTable declaration in CRBasic dataloggers. The individual fields within each record are determined by the Output Processing instructions following the instruction that created the data table.

**RecNbr** - An entry in a table that shows the sequential record number in the table.

**Remote Site** – Typically where a 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. Most remote datalogger sites with radios can act as repeaters.

**Retries** – When a transaction or communication between two devices or programs fails, the transaction or communication can often be triggered to repeat until it succeeds.

**Retrieval** - (see Data Retrieval).

**RF** – Radio Frequency.

**RTDM** – Real Time Data Monitor software. A very sophisticated graphical data display application that gets data from either data files or the communication server's data cache. RTDM is a stand-alone application.

**RTMC** – Real Time Monitoring and Control software. A client application to the communications server that displays data from the server's data cache (only) and updates as new data is collected. RTMC is relatively easy to set up, and ships with LoggerNet.

**RTMS** – Real-Time Monitoring Software. A software application designed by Campbell Scientific for fast real-time data acquisition. RTMS included both communications and graphical display features and was designed for IBM's OS/2 PC operating system and replaced by LoggerNet, RTMC and RTDM.

## 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 code entered into the datalogger either directly with a keypad or via the datalogger's program to prevent unauthorized access to datalogger settings, programs, and data.

**Server** – Also “communication server”, 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. PC200W and PC400 also use the communication server but in a more limited configuration.

**Short Cut** – A program generator application that ships with PC400, LoggerNet, and is available as a stand-alone product from the Campbell Scientific website. Short Cut does not require knowledge of individual program instructions. Users need only know what kind of datalogger and sensors they're using and decide what output they require. Short Cut generates

the program for them. (Contrast a “program generator” with the full-featured “program editors”, Edlog and CRBasic Editor.)

**Signature** – Number calculated to verify both sequence and validity of bytes within a packet or block of memory.

**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.

**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. Not to be confused with datalogger serial numbers, PakBus addresses, or addresses set in communications peripherals such as RF or MD9 modems.

**Storage** - An entry in the status table that shows the number of final storage locations available.

**Synchronous** - The transmission of data between devices occurs as groups 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 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. "Table-based" includes both "TD" table-data and "PB" PakBus versions of the Edlog dataloggers as well as the CRBasic dataloggers.

**Table Definitions** – List of data available from a table-based datalogger. The datalogger supplies this list on request. The tables are determined by the datalogger program. The LoggerNet server must have a current version of the table definitions to collect data from the datalogger.

**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 the clock in a datalogger requires 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. CRBasic dataloggers use variables for all measurements, processing values, including variables to be used in Boolean form as “high” or “low”.

## 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 have built-in wiring panels. Others, such as the CR10X, 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. Campbell Scientific File Formats

---

*Campbell Scientific, Inc. uses different formats for data in datalogger memory, external PC cards, datalogger communication software, and PC files. The data formats written to PC files by LoggerNet are written by default as .DAT files. The following sections will focus on the format of these PC files, discuss the data formats that exist in the datalogger and on PC cards, and describe methods for converting binary data formats.*

## B.1 PC File Data Formats

The type of data file generated by LoggerNet depends on the type of datalogger from which data are being collected. Mixed-array dataloggers such as the CR10X include the option to output comma separated files, ASCII printable files, and binary files. Data from table-data dataloggers are output as ASCII table files with no header, TOAC11 files, TOA5 files, or TOB1 binary files. Data from PC cards in CR1000, CR5000 and CR9000 dataloggers are output as TOB2 or TOB3 binary format.

### B.1.1 Comma Separated

This file format describes values that are separated with a simple comma. Also known as comma-delimited files, this data file format has the following features:

- Data from multiple arrays or intervals can be included in the same file.
- Data are formatted in as little space as possible. Values are printed with all extraneous formatting such as that used in the printable ASCII format removed.
- Commas separate all values.

An example of a comma separated file:

```
108,2002,7,1528,58,.17365
112,2002,7,1528,58,.98481
108,2002,7,1528,59,.19081
112,2002,7,1528,59,.98163
108,2002,7,1529,0,.20791
112,2002,7,1529,0,.97815
```

## B.1.2 ASCII Printable

Files in ASCII Printable format have the following features:

- Data are printed in fixed field widths.
- The length of each line of text in the file will not exceed 79 characters.
- Data from multiple arrays can be included in the file.
- A two-digit column number and sign precede each value in a column.

An example of a printable ASCII file:

```
01+0112. 02+2002. 03+0007. 04+1456. 05+43.000 06-.99619
01+0115. 02+2002. 03+0007. 04+1456. 05+43.000 06-.99619 07+0.0000 08+0.0000
09+0.0000 10+0.0000 11+0.0000 12+0.0000
01+0108. 02+2002. 03+0007. 04+1456. 05+44.00 06+.06979
01+0112. 02+2002. 03+0007. 04+1456. 05+44.000 06-.99756
01+0115. 02+2002. 03+0007. 04+1456. 05+44.000 06-.99756 07+0.0000 08+0.0000
09+0.0000 10+0.0000 11+0.0000 12+0.0000
```

## B.1.3 TOACI1

This file format was originally introduced to support data coming from table-data dataloggers. This format has the following features:

- The file includes a header that contains the following:
  - The file format type, the station name, and the table name.
  - The field name for each of the data values.
- Each record in the file is assigned a timestamp and record number. The record number is a logged sequence number that is assigned by the datalogger.
- The data values are formatted as comma-separated text suitable for importing with little modification into most spreadsheet and database applications.
- Each TOACI1 file contains data from only one table.

An example of a TOACI1 file with a header and data values:

```
"TOACI1", "gold", "one_min"
"TMSTAMP", "RECNR", "temp_deg_f_AVG", "meas1", "meas2"
"2001-12-30 19:16:00", 18002, 69.05, 3000, 1500
"2001-12-30 19:17:00", 18003, 69.06, 3001, 1499
"2001-12-30 19:18:00", 18004, 69.06, 3002, 1498
```

## B.1.4 TOA5

TOA5 is a text-based file format similar to TOAC11 but with additional information in the header. This format has the following features:

- Contains a text header that provides the following information:
  - The file format type, the station name, the datalogger type, the serial number, the OS version, the DLD name, the DLD signature, and the table name.
  - The field name for each of the data values.
  - The units for each field as determined by the datalogger program.
  - The processing performed in the datalogger to produce each value in the table.
- A timestamp and record number can optionally be included as part of the record data. If the timestamp is present, it will be formatted with sub-second resolution.
- Data values are formatted as comma separated text suitable for importing into spreadsheet or database applications.
- Each TOA5 file contains data from only one table.

An example showing a TOA5 file containing the optional timestamp and record number:

```
"TOA5","CR1000","CR1000","1031","CR1000.Std.00.60","CPU:Test.CR1","4062","Test"
"TIMESTAMP","RECORD","batt_volt_Min","PTemp"
"TS","RN","Volts","C"
"","","Min","Smp"
"2004-11-11 15:03:45",0,13.7,24.92
"2004-11-11 15:04:00",1,13.7,24.95
"2004-11-11 15:04:15",2,13.7,24.98
```

## B.1.5 TOB1

TOB1 files can be generated by LoggerNet when outputting data files to the PC. This binary file format is typically only used when it is essential to minimize the file size or when other software requires this format. It has the following structure:

ASCII header line 1
ASCII header line2
ASCII header line3
ASCII header line 4
ASCII header line5
Binary Records

An example of a TOB1 ASCII header:

```
"TOB1", "STATION", "CR9000", "1000", "1.00", "CPU:BIG.DLD", "25871", "VALUES"  
"SECONDS", "NANOSECONDS", "RECORD", "Array(1)", "Array(2)", "Fast", "my_string"  
", "", "RN", "mVolts", "mVolts", "mVolts"  
", "", "", "Smp", "Smp", "Smp"  
"ULONG", "ULONG", "ULONG", "IEEE4", "IEEE4", "FP2", "ASCII(25) "
```

Header line one describes the file environment with the following eight fields:

- Data file type (TOB1).
- Station name (STATION).
- Model name of the datalogger (CR9000).
- Serial number of the datalogger (1000).
- Operating system on the datalogger (1.00).
- Name of the program running in the datalogger (CPU:BIB.DLD).
- Signature of the program running in the datalogger (25871).
- The name of the datalogger table (VALUES).

Header line two consists of a set of comma-delimited strings identifying the names of the fields in the table of the datalogger program.

Header line three describes the units associated with each field in the record. Units are optional and are specified in the datalogger program, if included. If no units are provided in the program, then an empty string placeholder is left in this line for that specific field.

Header line four describes the processing performed in the datalogger to produce the value for each field in the record; for example, sample, average, min, max, etc. If there is no known processing for a field, that field will be assigned an empty processing string. There will be one value on this line for each field name given on header line two.

Header line five describes the data type for each field and supports the following values: IEEE4, IEEE8, FP2, ULONG, LONG, SecNano, BOOL, and ASCII(len).

Each data record following the header is a sequence of binary values. The length of each value is determined by the data type assigned to it in header line five and the length of the entire record is the sum of the individual data value lengths. There are no characters that separate records so the application that reads the TOB1 file must understand the file header so that the record length can be calculated.

The timestamp and record number for each record are an optional output in a TOB1 file. If these elements are present, a "SECONDS", "NANOSECONDS", and "RECORD" column will be generated as names in the field list of header line two.

## B.2 Datalogger Data Formats

### B.2.1 TOB2 or TOB3

TOB2 and TOB3 files are created when data are retrieved from external PC cards on dataloggers such as the CR9000, CR5000, and CR1000. The TOB2 file format has been replaced in new dataloggers by the TOB3 file format. TOB3 format is similar to TOB2 in most respects but differs from TOB2 in the following ways:

- Frame headers in TOB3 are twelve bytes long rather than eight bytes long. The additional four bytes contain an unsigned integer with the least significant byte written first to identify the record number for the first record in the frame.
- In the TOB3 format, the offset field in the major frame footer no longer represents the number of frames back to the last minor frame. This information is used in TOB2 to help accelerate searching for data but is not considered to be necessary in TOB3 because of the presence of the record number in the frame header.

The TOB2 or TOB3 binary file format has the following structure with each header line terminated with a carriage return and line feed (CRLF):

ASCII Header Line 1		
ASCII Header Line 2		
ASCII Header Line 3		
ASCII Header Line 4		
ASCII Header Line 5		
ASCII Header Line 6		
Frame Header	Frame Body n	Frame Footer

Header line one describes the file environment with the following fields:

- Data file type (TOB2 or TOB3).
- Station name.
- Model name of the datalogger.
- Serial number of the datalogger.
- Operating system on the datalogger.
- Name of the program running in the datalogger.
- Signature of the program running in the datalogger.
- The time that the file was created.

Header line two contains:

- The name of the table as declared in the datalogger program.
- The non-timestamped record interval.
- The data frame size.
- The intended table size.
- The validation stamp.
- The frame time resolution.

Header line three describes the names for each field in a table record as determined by the datalogger program.

Header line four describes the units associated with each field in the record. Units are optional and are specified in the datalogger program, if they are included. If no units are provided in the program, then an empty string placeholder is placed in this line for that specific field.

Header line five describes the processing performed in the datalogger to produce the value for each field in the record; for example, sample, average, min, max, etc. If there is no known processing for a field, that field will be assigned an empty processing string. There will be one value on this line for each field name given on header line three.

Header line six defines the data types for each field in the record and supports the following values: IEEE4, FP2, ULONG, LONG, SecNano, and ASCII(len).

TOB2 frame headers are eight bytes long and hold the timestamp for the first record in the frame. TOB3 frame headers are twelve bytes long and contain the same timestamp information but also add a four-byte unsigned integer that represents the beginning record number for that frame.

The frame data begins immediately following the frame header and consists of zero or more data records. Each record contains one data point for each of the field names identified in header line three. The data type and implied size of these data points are identified by the data types list given by header line six.

The frame footer makes up the last four bytes of the frame.

## B.3 Binary Data Value Types

When data is written in datalogger memory or in binary data files each value must be assigned a particular data type. These data types describe the format of the data.

### **B.3.1 FP2 (2 Byte Low Resolution Format)**

A two-byte floating-point number format created by Campbell Scientific, Inc. and used to store low-resolution values. Appendix C of the CR10X datalogger manual describes this format in greater detail. Basically, this format consists of a single sign bit, a two-bit negative decimal exponent, and a 13-bit mantissa.

### **B.3.2 FP4 (4 Byte High Resolution Format)**

A four-byte floating-point number format created by Campbell Scientific, Inc. and used for input location values as well as high-resolution final storage values. This format consists of a single sign bit, a seven-bit base-two exponent, and a 24-bit mantissa.

### **B.3.3 IEEE4**

A standard four-byte floating-point number format used for certain values within a record. This format consists of a single sign bit, an eight-bit binary exponent, and a 23-bit mantissa.

### **B.3.4 IEEE8**

A standard eight-byte floating-point number format used for certain values within a record. This format consists of a single sign bit, an 11-bit exponent, and a 52-bit mantissa.

## **B.4 Converting Binary File Formats**

Campbell Scientific dataloggers not only use the previously mentioned binary file formats but users may also choose to use and access these binary formats on the PC. Binary files may be output as data files to the PC by LoggerNet to save hard disk space or to accommodate a user that is only interested in using binary files in an application. In addition, PC cards that are written by the datalogger will contain binary files that can be accessed directly by the PC.

Binary files cannot be interpreted through mere visual inspection. Therefore, binary file processing tools are available to read and convert these binary data files to ASCII text. These conversion tools are Split, View, TOB Convert, and TOB32.EXE.

### **B.4.1 Split**

Split has the capability of reading TOB1, TOB2, and TOB3 files and displaying data from those files in ASCII format. The output parameters are user specified and Split generates a file containing the converted ASCII format values.

### **B.4.2 View**

View converts TOB1, TOB2 and TOB3 data files and generates a new TOA5 file. Once the TOA5 file is generated, View can display the converted data. In

that sense, View is a software tool that combines the conversion of binary files to TOA5 with the ability to view the data once the file has been converted.

### B.4.3 TOB Convert

The TOB Convert program is available as a shortcut from the LoggerNet folder in the Windows Start Menu at Start | Programs | LoggerNet | Utilities. TOB Convert can convert TOB1, TOB2, and TOB3 files to TOA5 file format or can convert TOA5 files to TOB1 file format. First, choose the file format to create. Then, select the file to convert and name the file that will be created by the conversion. The software converts the data and generates the new file.

### B.4.4 TOB32.EXE

The TOB32.EXE command line utility is installed by default in the LoggerNet program directory at C:\Program Files\Campbellsci\Loggernet\tob32.exe and works in a similar manner to the TOB Convert software. Command line switches are used to determine the new file format that will be created. Some of the basic switches available are listed below:

-h or -? | Help

-a | ASCII (TOA5) Generates CSI Table Oriented ASCII version 5 format files

-b | Binary (TOB1) Generates CSI Table Oriented Binary version 1 format files

Some examples using these switches include:

tob32.exe -a mydata.dat (converts mydata.dat to TOA5 format)

tob32.exe -b mydata.dat (converts mydata.dat to TOB1 format)

# **Appendix C. 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, CR1000, CR5000, CR200, and CR9000.*

## **C.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.

### **C.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 execution interval in calculating the length. Since the datalogger tries to make the tables fill up at the same time, with programs using short execution intervals these event driven tables may take up most of the memory leaving very little for the other, longer interval, automatically allocated data tables.

---

## C.1.2 CR5000/CR1000/CR9000 Memory for Programs and Data Storage

The datalogger memory for the CR5000, CR1000, 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/CR1000/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, with programs using short execution intervals these event driven tables may take up most of the memory leaving very little for other, longer interval, automatically allocated data tables.

---

### C.1.3 CR200 Series Dataloggers

CR200 Series dataloggers are similar to the CR5000/CR1000/CR9000 dataloggers regarding the format of final storage. Data is stored in final storage tables that are made up of records and fields. As with the other table-based dataloggers, the user can specify the number of records for each table, or table-size can be determined by the datalogger. And as with the other dataloggers, the size of event driven tables should always be entered by the user, or else the datalogger will calculate the amount of memory for the table based on the execution interval.

The CR200 Series dataloggers have a Public table in which the current scan's measurements are held. However, the CR200 Series does not have the ability to store multiple programs and program processing takes place in a linear fashion, similar to the TD family of dataloggers (e.g., each programming instruction is performed sequentially; one instruction must finish before the datalogger proceeds with the next).

The CR200 Series dataloggers do not have an on-board compiler. Programs must be precompiled by LoggerNet, or when using the CRBasic Editor, before they are sent to the datalogger. Refer to Section 5.1.5.2 for additional information about sending programs to CR200 dataloggers.

## C.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.

**NOTE**

---

PakBus versions of the CR10X family of dataloggers generally use the same programming instructions as TD table-based dataloggers.

---

## C.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. Check the Options | DLD File Labels setting to ensure the "Include first \_\_\_ Input location labels" field is not set to 0 if the option is selected.
7. Save and compile the program, correcting any errors that may be found by the compiler.

## C.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.
- **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 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. )

## C.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. The record

number will continue to increment, and the oldest record will “drop off” the top.

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 two 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. If automatically allocated, event driven tables in the later versions of the operating systems of CR10X-TD type dataloggers are assumed to have one record stored per execution interval in calculating the length.

In CR5000/CR1000/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, if you let the datalogger automatically allocate table sizes these event driven tables may 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. 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.

## C.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, CR1000, CR9000, and CR200 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 typical fields that are given in the Status table. Not all fields will be present or applicable for all dataloggers. See the datalogger operator's manual for specifics.

**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>Table B-1. Example of Status Table Entries (CR10T)</b>	
<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.
<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>Table B-1. Example of Status Table Entries (CR10T)</b>	
<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 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>Table B-1. Example of Status Table Entries (CR10T)</b>	
<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 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 D. Software Organization***

---

## **D.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 handles connections from all the user interface screens simultaneously, allowing many different views and ways to access the data collected from the dataloggers. In addition to running on the same computer with LoggerNet, the LoggerNetData applications can be run on other computers connected to the LoggerNet computer over a local area network (LAN).

LoggerNet 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.

## **D.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. The data cache files are stored in addition to the output data files.

### **D.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 table data dataloggers. For array based dataloggers the array definitions are contained in a final storage label file that is associated with a datalogger. This table or array information is used to set up equivalent tables and data arrays for data storage in the data cache. The size of the areas set up in the data cache is dependent on the size of final storage in the datalogger.

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.

## D.2.2 Operation

Normal data collection from the datalogger is done with polling 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 poll request to the datalogger to get all of the data stored in the selected tables since the last poll. The tables to be collected are specified by the user in the Setup screen.

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 there is a gap in the record numbers because the data table filled and overwrote the requested 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 Collect Now from the Connect Screen. When a manual poll is done the data from the datalogger is saved in the output data file and is also put into the data cache.

## D.2.3 Retrieving Data from the Cache

Once the data has been stored in the data cache it is retrieved by the applications such as the graphical and numerical displays that request the data by datalogger, table or array, 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.

## D.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.

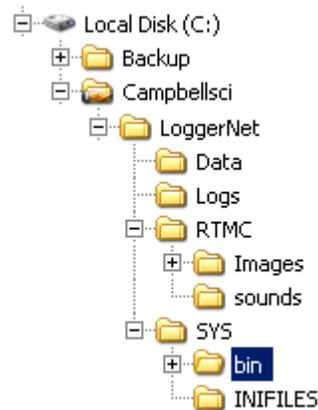
There are a number of things that could cause datalogger table definitions to change. A new program may have been downloaded to the datalogger, or the keyboard display may have been used to manually make changes to the datalogger program.

**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 the Collection State of Status Monitor 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. LoggerNet will save the existing output data file with a modified name and create a new output data file.

## D.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.

### D.3.1 C:\CampbellSci\LoggerNet Directory (Working Directory)

The C:\CampbellSci\LoggerNet working directory includes four subdirectories: Data, Logs, RTMC and SYS. The c:\CampbellSci\LoggerNet directory is used for storing user-created files, data files, and status information files from LoggerNet.

By default all user created datalogger programs and the data files collected with scheduled data collection are stored in the c:\CampbellSci\LoggerNet directory.

The files collected using the Custom Collect on the Connect Screen are stored by default in the C:\CampbellSci\LoggerNet\Data subdirectory to avoid confusion with the main data collection files.

The Logs directory contains the LoggerNet operational logs if they are being saved to disk. The RTMC directory contains the images, sounds and working files for RTMC.

The Infiles subdirectory under C:\CampbellSci\LoggerNet\SYS contains all configuration files for the Setup Screen, Connect Screen, Status Monitor and other applications.

The C:\CampbellSci\LoggerNet\SYS\bin subdirectory contains configuration files for the devices in your datalogger network (\*\_CONF), and the modem configuration file (modem.ini). This subdirectory has a C:\CampbellSci\LoggerNet\SYS\bin\Data subdirectory that is used to store the Data Cache for the devices in the network.

---

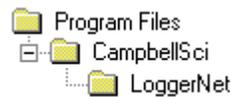
**NOTE**

Changing or removing any of the files in the C:\CampbellSci\LoggerNet\SYS\bin directory can cause loss of data, a system crash or destruction of the network map. There are no user editable files in this directory.

---

System administrators concerned about security and system integrity should use Windows NT and its directory access tools to control access to the working directories.

### D.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.

### D.3.3 Backing Up Critical Information

Since after installation, no files are stored in the programs directory (C:\Program Files\CampbellSci\LoggerNet) all you need to back up is the working directory, c:\CampbellSci\LoggerNet. We recommend that you back up this entire directory, with all of its files and subdirectories, on a regular basis.

# Appendix E. Log Files

---

## E.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 6.2.3 for information on saving the logs to disk.

### E.1.1 Log Categories

The LoggerNet server logs events in four different kinds of logs 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. This is the most readable of the logs and contains event messages that are meaningful to most users. 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 by software developers 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.

### E.1.2 Enabling Log Files

Use Status Monitor (Edit |Log Settings) to enable logging of events to files. If enabled, the server will write log records to text files in the log file 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 are 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.

## E.1.3 Log File Message Formats

### E.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:

**Timestamp** - 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.

### E.1.3.2 Transaction Log Format

Each record in the transaction log includes at least two fields in addition to the timestamp and device name:

**Message Type Code** - Identifies the type of event that has occurred. This is a number that corresponds to the description immediately following. If this log is being read by a software program, a number is very easy to use for comparison when looking for specific message types.

**Message Type Description** - Text that describes the message type code.

The following table is a list of the different messages that can appear in the transaction log, some of the optional parameters and what the message means. Where appropriate, a suggested response to the message is provided.

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 LoggerNet 1.1)	
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.
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.

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
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\bin). 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 Setup, check that communications are enabled in Setup for all the devices in the path.
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.	

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
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)	
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.	

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
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 normal operations.

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
				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 Status Monitor. 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.
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.
74	Device renamed		The name of a device in the network was changed.	

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
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 "Collect Now" button on the Connect screen.	
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.

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
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.
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	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 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.	
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.	

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
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 CoraScript 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	Call-back begin		A device has called in to the server starting the call-back response.	
116	Call-back stopped		A datalogger that called in to the server with call-back is hanging up.	
117	Client logged off	The login name of the client; The reason the session was closed.	A client application has closed or lost the connection to the server.	
118	Table size reduced during creation	The name of the table that was resized; The original specified size of the table; The new size of the table.	The size of the table in the data cache was reduced because there was not enough computer disk space to create it, or the file would have exceeded the 2 Gbyte size limit.	Reduce the size of the tables in the datalogger program or get more hard disk storage space for the computer.

### Transaction Log Example

```
"2002-03-06 11:30:01.075","CR510TD","94","Collect area poll complete","TimeSet","0"
"2002-03-06 11:30:01.325","CR510TD","41","Records received","TenSecond","242747","242751"
"2002-03-06 11:30:01.325","CR510TD","94","Collect area poll complete","TenSecond","12"
"2002-03-06 11:30:01.325","CR510TD","97","Scheduled polling complete"
"2002-03-06 11:30:04.670","CR23X_2","93","Final storage values received","final_storage_1","72"
"2002-03-06 11:30:04.670","CR23X_2","94","Collect area poll complete","final_storage_1","72"
"2002-03-06 11:30:04.700","CR23X_2","94","Collect area poll complete","ports_and_flags","0"
"2002-03-06 11:30:04.700","CR23X_2","97","Scheduled polling complete"
```

```

"2002-03-06 11:30:30.097","CR510TD","13","Clock checked","20020306 11:28:32.455"
"2002-03-06 11:31:00.010","CR23X_2","96","Scheduled poll started"
"2002-03-06 11:31:00.010","CR510TD","96","Scheduled poll started"
"2002-03-06 11:31:00.010","CR23X_2","92","Collect area poll started","final_storage_1"
"2002-03-06 11:31:00.010","CR23X_2","92","Collect area poll started","ports_and_flags"
"2002-03-06 11:31:00.010","CR510TD","92","Collect area poll started","ErrorLog"
"2002-03-06 11:31:00.010","CR510TD","92","Collect area poll started","Hourly"
"2002-03-06 11:31:00.010","CR510TD","92","Collect area poll started","OneMinute"
"2002-03-06 11:31:00.010","CR510TD","92","Collect area poll started","TenSecond"
"2002-03-06 11:31:00.010","CR510TD","92","Collect area poll started","TimeSet"
"2002-03-06 11:31:00.100","CR510TD","13","Clock checked","20020306 11:29:02.454"
"2002-03-06 11:31:00.290","CR510TD","41","Records received","ErrorLog","5","5"
"2002-03-06 11:31:00.290","CR510TD","94","Collect area poll complete","ErrorLog","0"
"2002-03-06 11:31:00.530","CR510TD","41","Records received","Hourly","673","673"
"2002-03-06 11:31:00.530","CR510TD","94","Collect area poll complete","Hourly","0"
"2002-03-06 11:31:00.731","CR510TD","41","Records received","OneMinute","40459","40459"
"2002-03-06 11:31:00.731","CR510TD","94","Collect area poll complete","OneMinute","3"
"2002-03-06 11:31:00.961","CR510TD","41","Records received","TenSecond","242758","242758"

```

### E.1.3.3 Communications Status Log Format

Each record in the communications status log includes two fields in addition to the timestamp 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.

#### 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 BMP1 communication framing packet was sent and the response received from the device. (CR10X-TD table based type devices)	
Classic;;Cmd	The listed command was sent to an array based datalogger.	For a list of the commands and their meanings see the datalogger operator's manual.
BMP1 packet received	A BMP1 packet was received from the device. (CR10X-TD type devices only)	
RPC packet exchanged	A BMP3 packet was exchanged. (CR5000, CR9000 dataloggers only)	
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.	
Unable to Locate Serial synch byte	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 set up as a table based datalogger in the network map.

### E.1.3.4 Object State Log Format

The object state log includes two fields in addition to the timestamp 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 F. Importing Files into Excel

Data files saved by LoggerNet can be imported into a spreadsheet program for analysis or manipulation. Instructions are given below for importing a comma separated file into Microsoft Excel.

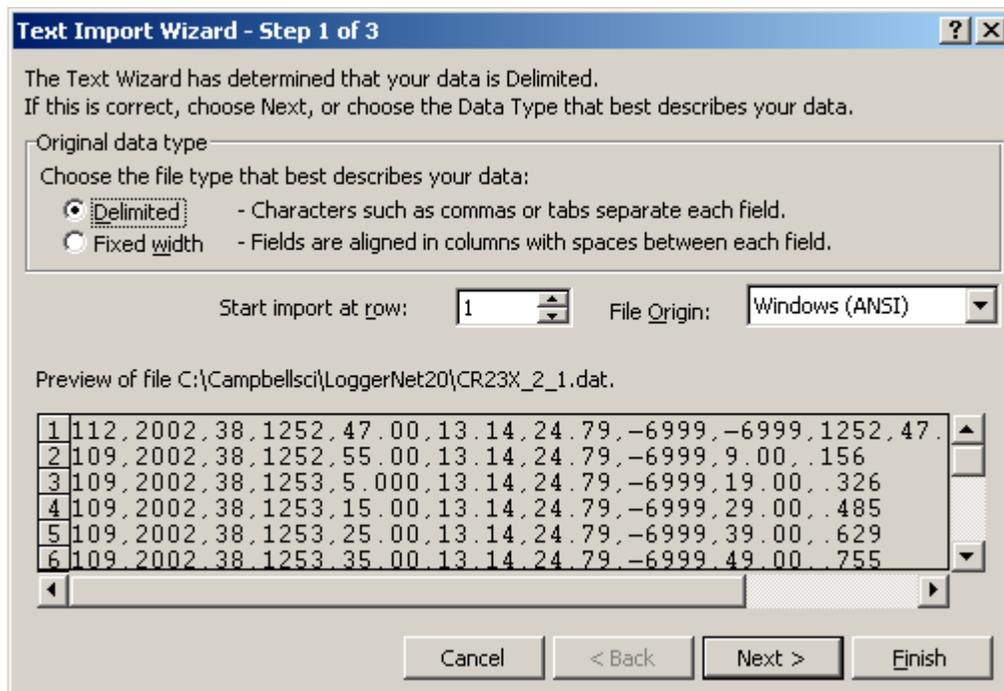
From the Excel menu, select File | Open. Browse for the \*.dat file that you want to import. Excel will recognize the file as not being in an xls format, and will invoke the Text Import Wizard. The Text Import Wizard consists of three steps, each having its own window.

## F.1 Array-Based Data File Import

Array-based data files are typically comma separated with the first column of each line being an array ID. If time and date are included in the arrays, they may be in columns 2 through 5, with some of the columns optional. Excel will easily read the comma separated file. Getting the different time columns pulled together into one timestamp field takes a little manipulation.

### Step 1 of 3

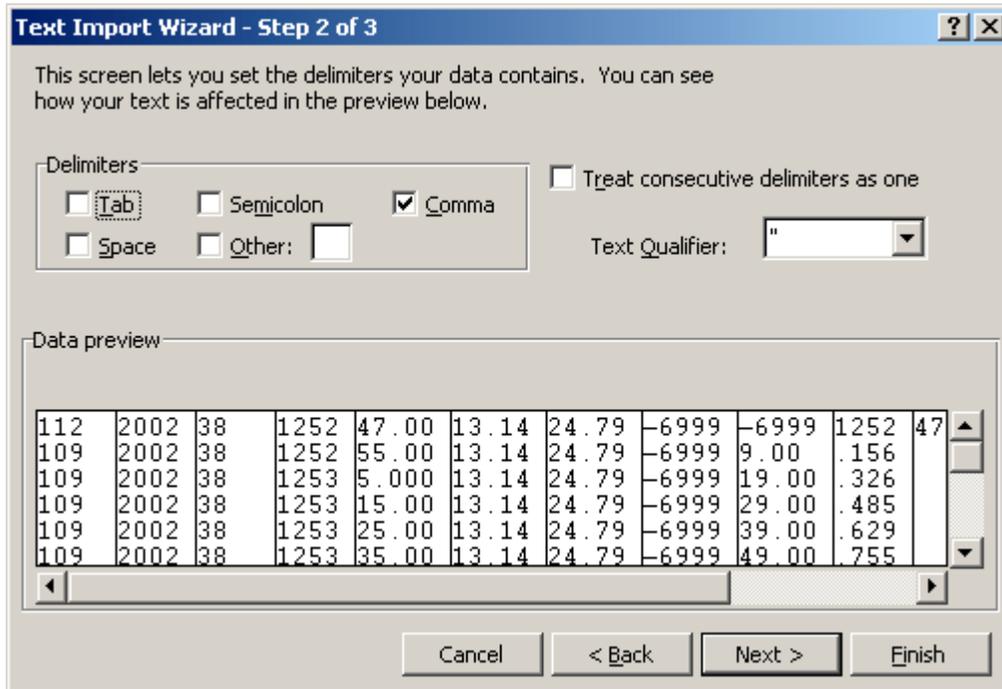
Select the Delimited option from the Original Data Type group box. Using the arrow buttons to the right of the Start Import at Row field, select the number of the first row of data to be imported. Select the Next button.



## Step 2 of 3

From the Delimiters group box, select Comma and Space. The Comma option directs Excel to place each data value, which is separated by a comma, into a separate column. The Space option will separate the Date and the Time into two columns.

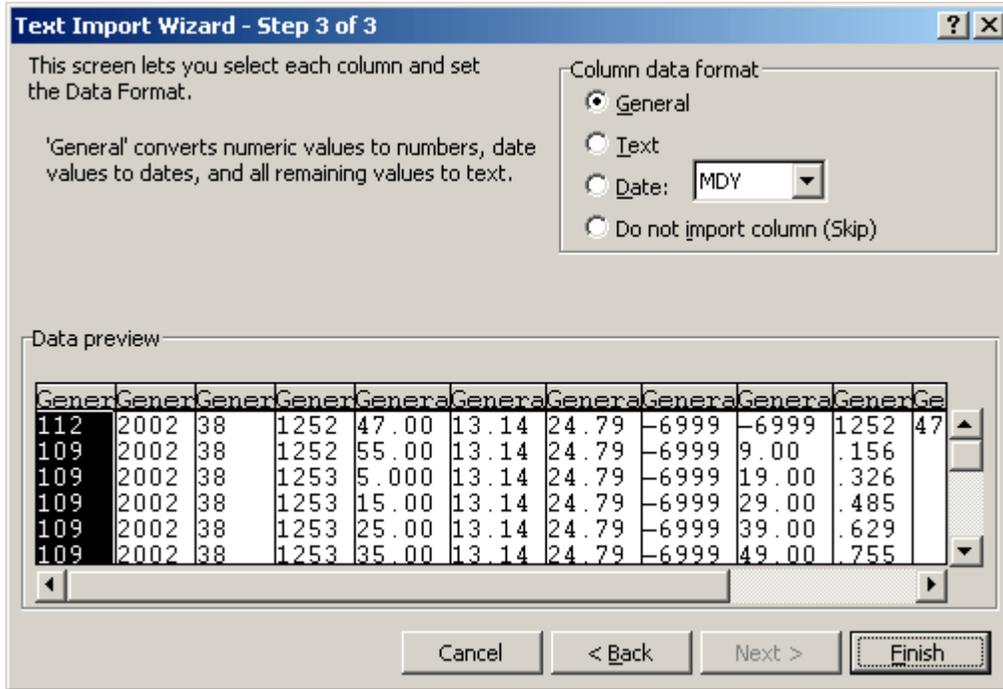
From the Text Qualifiers list box, select None. Select the Next button.



### 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.



### Converting to Excel Format Date and Time

Once the data file has been imported into Excel, the time fields are still displayed as comma separated numbers such as Year, Day of Year, and Hours/Minutes in HHMM format.

Split can take array-based data files and convert the year, day of year, and hours/minutes fields into a standard timestamp format that Excel will read directly. See Section 8.2.3.1.

You can also enter formulas as described below to convert the timestamp fields in the array data to the decimal format used by Excel. Microsoft’s database (MS Access) and spreadsheet (MS Excel) programs store dates and times as real numbers, where the integer portion of the number represents the number of days since some base date (usually January 1, 1900), and the fractional portion represents the time of day. For example, June 1, 2000 at 10:00 a.m. would be stored as “36678.41667.”

This formula will take the comma separated date and time fields and convert them to the decimal date use by Excel. The variables shown in brackets [ ] should be replaced by the cell location for that data. If you don’t have all of the date or time elements in your data, you can replace that part of the equation

with a number (e.g. [Year] = 2002), or for Hours, Minutes, and Seconds leave that part of the formula out.

$$([Year]-1900)*365+1+Int((([Year]-1901)/4)+[Day]+Int([HHMM]/100)/24+([HHMM]/100-Int([HHMM]/100))*100/60/24+[Sec]/60/60/24$$

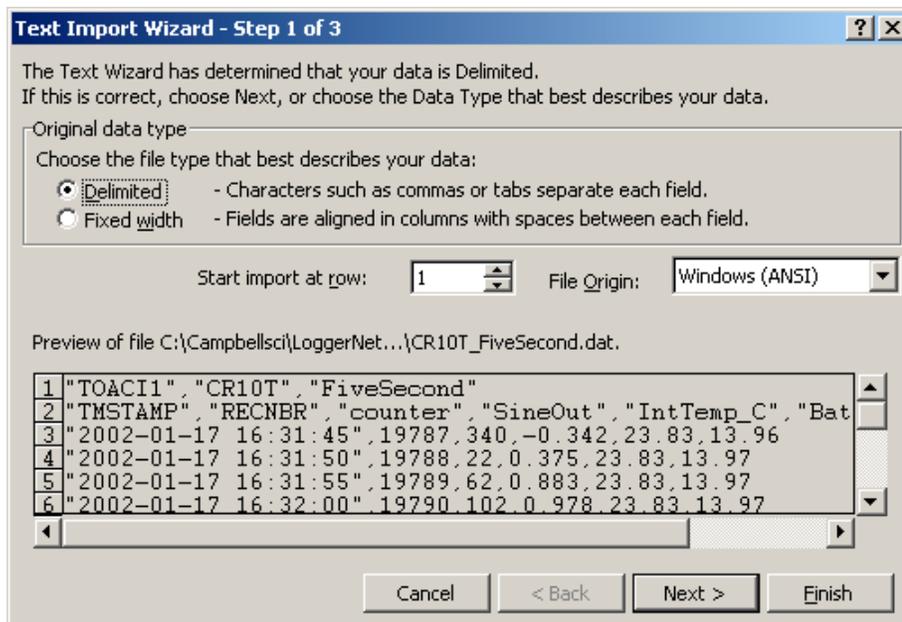
Section of formula:	Results:
$[Year]-1900)*365+1+Int((([Year]-1901)/4$	Delivers the number of days since the beginning of the 20 <sup>th</sup> century through the end of last year taking into account leap years
$+ [Day]$	Adds the Julian date to the above
$+Int([HHMM]/100/24$	Converts the hour portion of the HHMM time field to a fraction of a day
$+([HHMM]/100-Int([HHMM]/100))*100/60/24$	Converts the minutes portion of the HHMM time field to a fraction of a day
$+ [Sec]/60/60/24$	Converts the seconds field in to a fraction of a day

Once you have entered the formula for one cell you can apply it to multiple cells using Excel's Fill function. Selecting the cells and using Format Cells can set the display format of the timestamp.

## F.2 Table-Based Data File Import

### Step 1 of 3

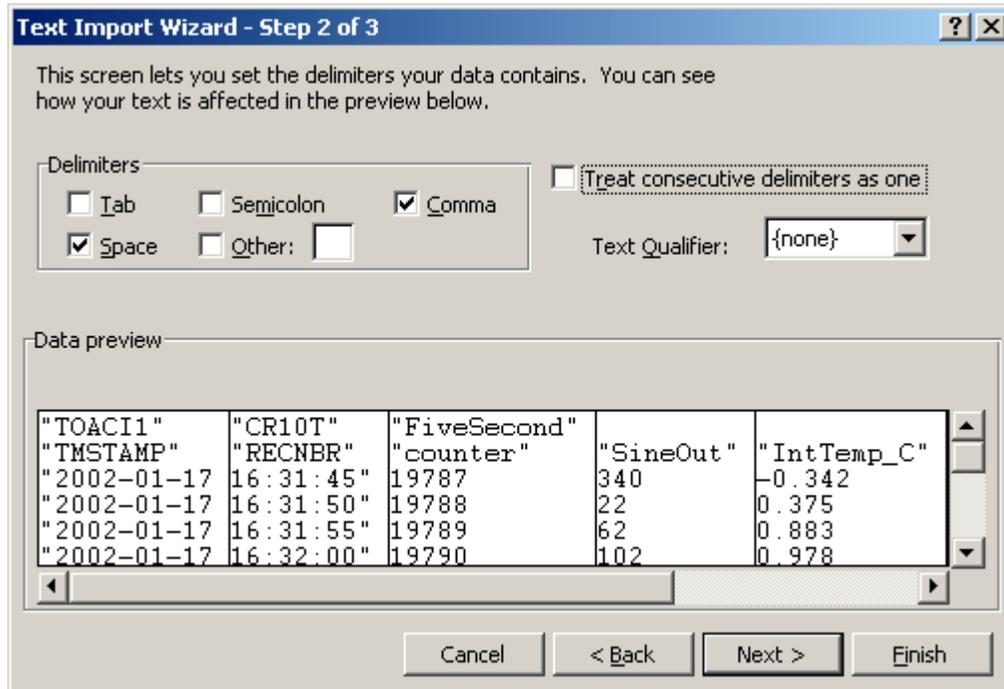
Select the Delimited option from the Original Data Type group box. Using the arrow buttons to the right of the Start Import at Row field, select the number of the first row of data to be imported. If your data file has headers included, you can import those or start the import at the first row of data (typically row 3). Select the Next button.



## Step 2 of 3

From the Delimiters group box, select Comma and Space. The Comma option directs Excel to place each data value, which is separated by a comma, into a separate column. The Space option will separate the Date and the Time into two columns.

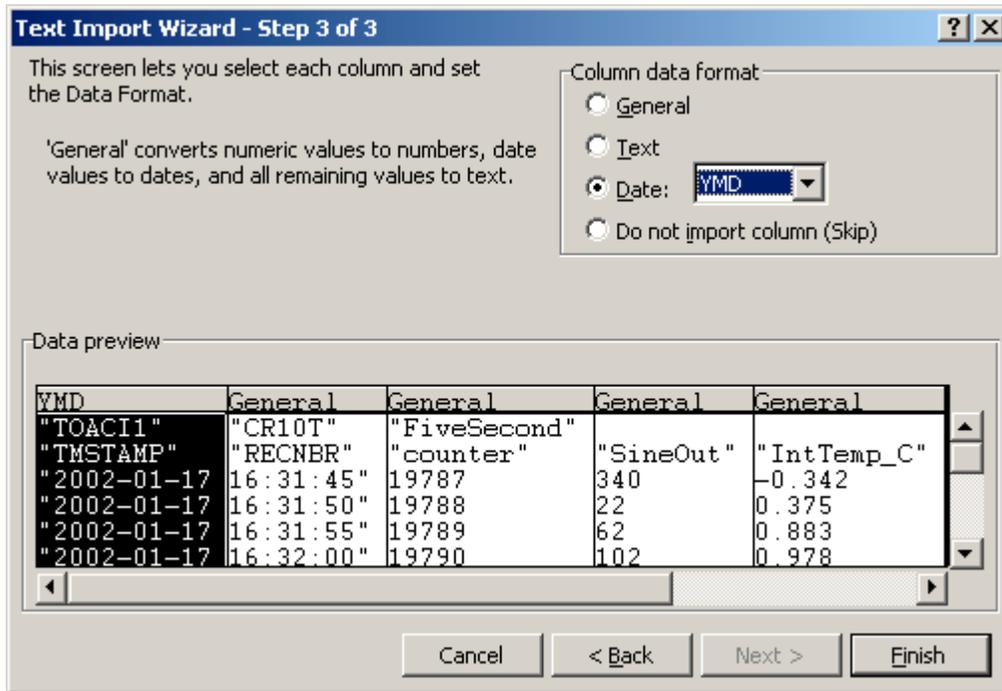
From the Text Qualifiers list box, select None. Select the Next button.



### Step 3 of 3

A quick look at the columns of data is provided in the Data Preview group box. Highlight the column with the year/month/day and from the Column Data Format group box, select the Date option. From the drop down list box to the right of this option select the YMD format.

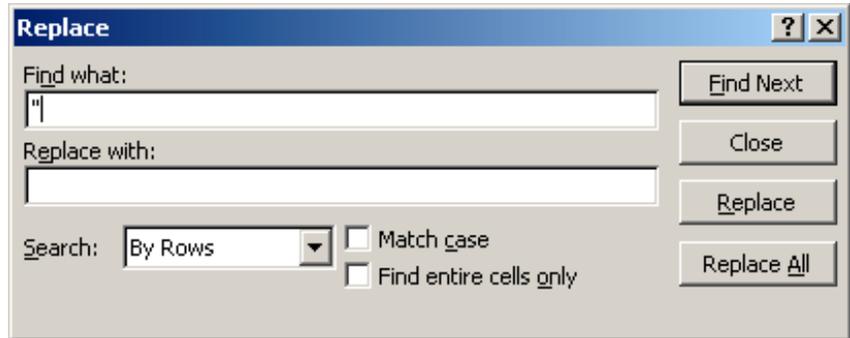
To complete the import, select the Finish button.



As imported, the Date and Time fields have a quotation mark in the field.

"TOACI1"	"CR10T"	"FiveSecond"				
"TMSTAMP"	"RECNR"	"counter"	"SineOut"	"IntTemp_C"	"BattVolt"	
"2002-01-17"	16:31:45"	19787	340	-0.342	23.83	13.96
"2002-01-17"	16:31:50"	19788	22	0.375	23.83	13.97
"2002-01-17"	16:31:55"	19789	62	0.883	23.83	13.97
"2002-01-17"	16:32:00"	19790	102	0.978	23.83	13.97
"2002-01-17"	16:32:05"	19791	142	0.616	23.83	13.96
"2002-01-17"	16:32:10"	19792	182	-0.035	23.83	13.97

The quotation marks can be removed by using Excel's Search and Replace feature. From the Excel menu, select Edit | Replace. In the Find What field, type in a quotation mark ("). Leave the Replace With field blank, and select the Replace All button.



If headers have been imported with the data, the column headings will be off by one since the date and time have been imported as two separate fields. The headers can be highlighted and moved one cell to the right to correct this.





## **Campbell Scientific Companies**

---

### **Campbell Scientific, Inc. (CSI)**

815 West 1800 North  
Logan, Utah 84321  
UNITED STATES  
[www.campbellsci.com](http://www.campbellsci.com)  
[info@campbellsci.com](mailto:info@campbellsci.com)

### **Campbell Scientific Africa Pty. Ltd. (CSAf)**

PO Box 2450  
Somerset West 7129  
SOUTH AFRICA  
[www.csafrica.co.za](http://www.csafrica.co.za)  
[sales@csafrica.co.za](mailto:sales@csafrica.co.za)

### **Campbell Scientific Australia Pty. Ltd. (CSA)**

PO Box 444  
Thuringowa Central  
QLD 4812 AUSTRALIA  
[www.campbellsci.com.au](http://www.campbellsci.com.au)  
[info@campbellsci.com.au](mailto:info@campbellsci.com.au)

### **Campbell Scientific do Brazil Ltda. (CSB)**

Rua Luisa Crapsi Orsi, 15 Butantã  
CEP: 005543-000 São Paulo SP BRAZIL  
[www.campbellsci.com.br](http://www.campbellsci.com.br)  
[suporte@campbellsci.com.br](mailto:suporte@campbellsci.com.br)

### **Campbell Scientific Canada Corp. (CSC)**

11564 - 149th Street NW  
Edmonton, Alberta T5M 1W7  
CANADA  
[www.campbellsci.ca](http://www.campbellsci.ca)  
[dataloggers@campbellsci.ca](mailto:dataloggers@campbellsci.ca)

### **Campbell Scientific Ltd. (CSL)**

Campbell Park  
80 Hathern Road  
Shepshed, Loughborough LE12 9GX  
UNITED KINGDOM  
[www.campbellsci.co.uk](http://www.campbellsci.co.uk)  
[sales@campbellsci.co.uk](mailto:sales@campbellsci.co.uk)

### **Campbell Scientific Ltd. (France)**

Miniparc du Verger - Bat. H  
1, rue de Terre Neuve - Les Ulis  
91967 COURTABOEUF CEDEX  
FRANCE  
[www.campbellsci.fr](http://www.campbellsci.fr)  
[campbell.scientific@wanadoo.fr](mailto:campbell.scientific@wanadoo.fr)

### **Campbell Scientific Spain, S. L.**

Psg. Font 14, local 8  
08013 Barcelona  
SPAIN  
[www.campbellsci.es](http://www.campbellsci.es)  
[info@campbellsci.es](mailto:info@campbellsci.es)