LOGGERNET USER'S MANUAL

Version 2.0

REVISION: 3/02

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

11. View 11-1

12. Real-Time Monitor and Control 12-1

13. Storage Module Software (SMS) 13-1

Appendices

G. Importing Files into Excel G-1

Tables

Figures

Section 1. Introduction

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 Microsoft Windows 95, 98, Windows NT version 4.0, and Windows 2000.

LoggerNet version 2.0 is the first software product that supports communication and data collection for *all* of the Campbell Scientific dataloggers. Array-based dataloggers are supported including the CR500, CR510, CR10, CR10X, 21X, CR23X, and CR7 along with the table-based dataloggers including the CR5000, CR9000, CR510-TD, CR10T, CR10X-TD, and the CR23X-TD.

The LoggerNet software is written using an advanced "client-server" architecture. The server is a software program that 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. This is similar to the public library that connects to the information providers (authors and publishers) and brings the data (books) to a central repository (library) where the clients (patrons) can retrieve the data.

Launching the Toolbar automatically starts the server. The buttons on the Toolbar launch applications that may connect to the server and allow you to set up the network or view the collected data. This software design allows for easier future customizations and new features since the core communications software does not have to change. It also allows simultaneous access by many computers to the same data without burdening the datalogger network with redundant communications.

The LoggerNet applications include a set of tools for you to work with the datalogger network and retrieve the data.

- Network configuration tools allow you define and configure the dataloggers in the network, how they are connected to the computer, and what data should be collected. You can also connect in real time to work with individual dataloggers.

- Editors allow the creation and modification of datalogger programs.

- Other applications allow you to view and process the data collected from the dataloggers.

- A set of miscellaneous utilities let you work with other Campbell Scientific hardware such as storage modules.

One significant benefit of the software design is that some client applications (RTMC and Cora Script) will not have to run on the same computer as the server. These applications can be run on any computer that is connected to the main computer by an Ethernet network connection. Some examples of these

networks are Local Area Network (LAN), Wide Area Network (WAN), or the Internet.

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

A word about Windows conventions.

For the past 15 years Microsoft has been working to establish a standard for the operation of graphical user interfaces. There are numerous conventions and expectations about the way a software program looks and behaves running under Microsoft Windows. With LoggerNet, Campbell Scientific has adopted as many of these conventions as reasonable.

This manual describes a collection of screens, dialogs and functions to work with a network of dataloggers. As with most Windows based software there is almost always more than one way to access the function you want. We encourage you to look around and experiment with different options to find which methods work best for you.

To keep the manual as concise and readable as possible, we will not always list all of the methods for getting to every function. Typically each function will have two methods and some will have as many as four.

The most common methods for doing things are:

Menus – At the top of most windows menu items are accessed either by a left mouse click, or using a hot key combination (e.g. , Alt+F opens the File menu). When the menu is opened the desired item can be selected by clicking on it, or using arrow keys to go down to it and pressing the Enter key, or typing the underlined letter.

If there is an arrow next to the menu item, there is a submenu with more choices. Clicking or selecting the item will bring up the submenu. By convention menu items that bring up dialog boxes or windows will be followed by an ellipsis (…). Other items execute functions directly or can be switched on or off. Some menu items show a check mark if a function is enabled and no check mark if disabled.

Buttons – Buttons are an obvious way to access a function. They are normally used for the functions that need to be called frequently or are very important. Clicking on the button executes the function or brings up another window.

Button functions can also be accessed from the keyboard using the tab key. On each screen one button or text area is selected or "has the focus". The name of the selected item is surrounded by a dotted line. A selected text edit box can be changed by typing. A selected check box can be toggled by pressing the space bar. A selected button can be clicked by pressing the Enter key. The tab key is used to move the focus to each of the buttons and controls on the screen.

Right Click Menus – Many areas have pop-up menus that bring up frequently used tasks or provide shortcuts. Just right click over an area and then left click on the menu item you want.

Hot Keys or Keyboard Shortcuts – Many of the menus and buttons can be accessed using Hot Keys. An underlined letter identifies the hot key for a button or function. To get to a menu or execute a function on a button hold down the Alt key and type the underlined letter in the menu name or the button text.

Pop-**Up Hints** – 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 the text is hidden, the full text will appear in the hint.

Section 2. System Requirements

2.1 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 computer configuration for running LoggerNet is Windows NT or Windows 2000 because they offer the most stable operating environment. LoggerNet will also run on Windows 95 and Windows 98. All installations require at least a Pentium II or equivalent processor, a minimum of 32 MB of RAM, a minimum of 45 MB free space on the hard disk, and TCP/IP support installed.

2.2 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 95, 98, or NT system. For Windows 2000 the same things need to be set up, but they are accessed in different ways. See the documentation and help for Windows 2000 to add a dial-up connection and associate it with TCP/IP.

NOTE
> Before beginning this procedure make sure that you have your Windows installation CD-ROM (or floppy disks as appropriate) handy.

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

1.   Click on the Start button and select Settings | Control Panel.

2.   When the Control Panel window comes up double click on the Add/Remove Programs icon.

3.   Select the Windows Setup tab.

4.   Select Communications and click on the Details button.

5.   On the Communications options screen click the box by "Dial-Up Networking" (Win 98/95) or "Phone Dialer" (NT). 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 on the Network icon.

10. In the list box on the Configuration tab (Win95/98) or Protocols tab (NT) of the Network window which comes up, see if there is an entry TCP/IP -> Dial-Up Adapter or TCP/IP protocol. If this entry exists, cancel and skip the next steps.

11. Click on the Add button. In the Select Network Component Type window which comes up select Protocol or TCP/IP protocol and click on the Add or OK button.

12. When the Select Network Protocol window comes up select Microsoft under Manufacturers:, and TCP/IP under Network Protocols:. Click OK.

Section 3. Installation

3.1 CD-ROM Installation

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

1. Put the installation CD in the CD-ROM drive. The install application should come up automatically. Skip to step 3. If the install does not start, then from the Windows system menu, select Start | Run.

2. Type D:\Disk1\Setup.exe in the Open field or use the Browse button to access the CD-ROM drive and select the setup executable in the Disk1 folder.

3. This activates the LoggerNet Installation Utility. Follow the prompts on the screen to complete the installation.

Items are added to your computer's Start menu under Programs | LoggerNet that start the Toolbar and some other selected utilities. If the default directories are used, LoggerNet executable files and help files are placed in the C:\Program Files\CampbellSci\LoggerNet directory. The directory C:\CampbellSci\LoggerNet is a working directory and contains the user's programs and data files, along with files maintained by LoggerNet such as the binary data cache and configuration files.

3.2 Upgrade Notes

If you are upgrading from an installation of PC208W version 3.0 or greater, you may want to convert your current network description from PC208W format to the LoggerNet format. The 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 description. It is also a good idea to make a backup of the PC208W files. The explanation for the use of Convert is in Section 5.6 on network setup.

Section 4. The LoggerNet Toolbar

This section provides an overview of the LoggerNet Toolbar and the associated applications. Each of the applications is explained in detail in subsequent sections.

4.1 The 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.

NOTE   When LoggerNet first starts, automated data collection is delayed for several seconds. This allows the user time to pause collection on the Status Monitor if desired.

To start LoggerNet double click the icon that was placed on your desktop. You can also go to the start menu of the computer and under Programs | LoggerNet select LoggerNet. This will bring up the Toolbar, as shown below, and start the server.
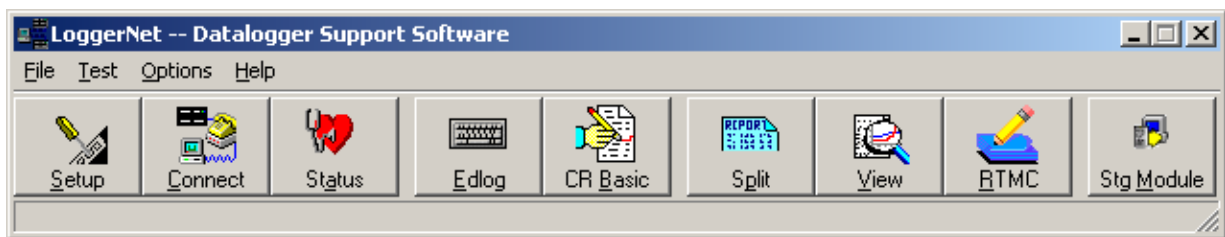
Clicking on any of the Toolbar buttons will bring up the screen for that function. An alternate way to bring up screens to use the File | Open menu at the top of the Toolbar. The menu provides access to all of the available screens, including those that may not be displayed as buttons on the Toolbar.

The appearance of the Toolbar can be changed from a horizontal layout as shown, to a vertical layout. This is done by dragging the right side of the Toolbar as far as it will go to the left, and then releasing the mouse button. To return to the horizontal layout, drag the right side of the Toolbar to the right. Dragging the bottom of the toolbar up in horizontal view will change to a toolbar with just the buttons and icons with no text.

The user can choose which buttons to display or hide by going to the Options | Modify Buttons menu item. All of the functions available under the File | Open menu can be shown or hidden on the Toolbar. Any functions not shown on the Toolbar can still be accessed through the File menu.

To keep the Toolbar from being hidden behind other Windows applications, under the Options menu you can select Stay On Top. If this option has a check

mark beside it, it is enabled, and the Toolbar will stay in the foreground even though other Windows applications are active.

To exit LoggerNet and shut down the server, either click on the X in the upper right hand corner of the Toolbar, or under the File menu select Exit.

4.1.1 Test Menu

4.1.1.1 Communication Test

The Toolbar can also be used to test communication with each datalogger in the communications network to confirm the integrity of the computer-datalogger network communications link.

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

Selecting Test | Communication Test from the main menu will bring up a screen similar to following dialog box:
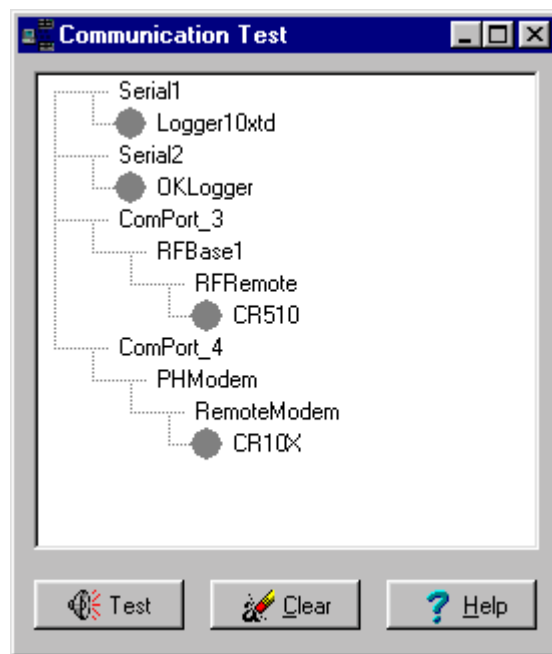
All devices in the network are displayed in the window. Each datalogger has a gray circle beside it. Once the communications test is run, the color of the circle will reflect the state of the communications link to the datalogger.

Select a datalogger to be tested by clicking on the circle next to the datalogger. When a datalogger is selected the circle beside it will turn black. Continue clicking on the dataloggers you want to test. Clicking a second time on a datalogger will deselect it. To deselect all the dataloggers click on the Clear button.

To begin testing, click the Test button. As a device is tested, the status icon to its left will change colors. Green signifies a good communications link, and red is critical. When a device is being tested the icon will be yellow. If the status is unknown the datalogger icon will remain gray.

4.1.1.2 Terminal Emulation

Terminal Emulation is a way to test communications links. The device is selected by choosing from the devices in the Select Device list. This utility allows you to type characters to send out over the communications link and will display any response from devices on that link. This can also be used for terminal emulation with dataloggers. For more information on using this capability see the troubleshooting section (14).

4.1.2 Options Menu

Configure Buttons – This option allows you to choose which buttons will be displayed on the Toolbar. If there are applications or screens that you don't use, the Toolbar can be simplified by unchecking the buttons you don't want to see.
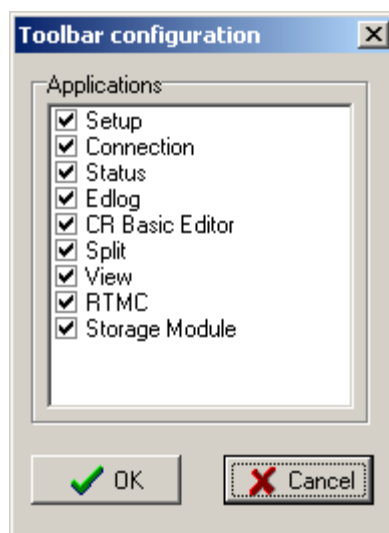
Stay on Top – Clicking on this menu option turns the Stay on Top attribute on or off. If this option is selected, the Toolbar will stay in front of any other windows displayed on the computer. This can be a benefit so the Toolbar doesn't get hidden behind other applications.

4.2 LoggerNet Applications

4.2.1 Setup (Section 5)

The Setup screen is used to add dataloggers to the network, define the communications paths between the computer and the dataloggers, choose what data should be collected, and set up an automatic collection schedule. Data collection and schedule information are set up separately for each datalogger.

The collected data is stored on the computer's hard drive in data files. Setup can also create and configure tasks that can be triggered by data collection or on a schedule. LoggerNet can also automatically check the datalogger's clock, comparing it with the computer's clock, and set it if it exceeds a specified variation.

The Task Master included in the Setup Screen allows you to trigger data collection, as well as execute batch file scripts and programs based on a variety of data collection events. Tasks may also be scheduled based on time intervals or the completion of another task.

4.2.2 Connect Screen (Section 6)

The Connect Screen 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.

4.2.3 Status Monitor (Section 7)

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 depicted by a status icon for each device. Columns can be set up to display information on communications quality and data collection. For troubleshooting purposes, windows are available to view operational log messages for the server as well as the low-level communication between the datalogger and the server. Task status can also be monitored.

4.2.4 Edlog (Section 8)

Edlog is a tool to create and edit datalogger programs for all Campbell Scientific dataloggers except the CR5000 and CR9000. Instructions are included 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.

4.2.5 CRBasic Editor (Section 9)

The CRBasic Editor is a tool to create and edit programs for the CR5000 and CR9000 dataloggers. Instructions are included for sensor measurement, program and peripheral control, and data storage.

4.2.6 Split (Section 10)

Split is used to separate mixed array data files into individual files based on the array ID. Split can also be used to create files in custom formats for use in

reports or as input to other data applications. Split has many uses and lends itself to experimentation.

4.2.7 View (Section 11)

View is used to look at data files. 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.

4.2.8 Real Time Monitor and Control (Section 12)

Real-Time Monitor and Control (RTMC) creates and displays real-time graphic display screens using the data collected from the dataloggers by LoggerNet. Once the display screen is configured it can be run on any computer that has a network connection to the LoggerNet computer. Users can customize graphic displays including graphs, tables, dials, alarms, digital values and other graphic elements. These displays automatically update when LoggerNet collects new data.

4.2.9 Storage Module Software (Section 13)

The Storage Module Software (SMS) handles configuration, data retrieval and datalogger program management with 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 95/98) a PCMCIA card slot installed on the computer.

4.3 Getting Help for LoggerNet Applications

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

A troubleshooting guide is provided in Section 14 of this manual. If you are unable to resolve your problem after reviewing the above noted resources, contact your Campbell Scientific Representative or Campbell Scientific directly.

4.4 LoggerNet Operations and Backup Procedures

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

4.4.1 Backing up Data

As with any computer system that contains important information, the data stored in the LoggerNet data files should be backed up to a secure archive or transferred to another system 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 as shown below. The default directory name is:

c:\CampbellSci\LoggerNet

📁 **LoggerNet**

The collected data files and datalogger programs. Most user files will be in this directory.

📁 **Logs**

The logs of server operations and communication.

📁 **RTMC**

The graphic and sound resource files for RTMC.

📁 **SYS\bin**

<u>LoggerNet Server working directory</u> – these files are controlled by the server and should not be disturbed by the user except when backing up the system. Tampering with these directories or files can cause unpredictable consequences and result in the loss of

📁 **Inifiles**

Initialization files created by various clients.

This will back up all of the working files for the server and the client applications along with any datalogger program files. For a detailed description of the directories and the files see Section 4.5.6.

Note that the above directories assume that the installation used the default directory structure suggested by the install utility. If a different working

directory was used, then the files will be contained in the same set of subdirectories, under the main working directory.

The maximum interval for backups depends primarily on the amount of data maintained in the datalogger memory. The datalogger's final storage is configured as ring memory that will overwrite itself once 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.

(There is a function available in Cora Script to save a script file containing all the commands to rebuild the network map and restore the configuration settings for all of the devices. This can be used 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 Cora Script see Appendix F.)

4.4.2 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 device configuration files. The data cache files contain all of the data that has been collected from the dataloggers by the LoggerNet server. These files are kept open (or active) as long as data is being stored to the file.

The configuration files contain information about each device in the datalogger network, including collection schedules, device settings, and other parameters. These files are written to frequently to make sure that they reflect the current state and configuration of each device. The configuration files are only opened as needed.

If computer system power is lost while the LoggerNet server is writing data to the active files, the files can become corrupted, making the files inaccessible to the server. This is particularly a problem for Windows 95 and 98 machines using the FAT32 disk file formatting. Windows NT and Windows 2000 offer the choice of NTFS that provides a greater protection for this type of event. Thus, Windows NT and Windows 2000 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.

4.4.3 Program Crashes

If the communication server crashes, there is a possibility that files can be corrupted. This is much less likely than problems due to power loss since the computer operating system remains in control and can close the files left open by the failed program. Again this is handled better with Windows NT and Windows 2000 than on Windows 95 and 98. If, after a program crash, the server does not run properly, you may need to restore the data from backup.

If you have problems restarting the LoggerNet server after a program crash or it crashes as soon as it starts, on Windows NT and 2000 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 on "End Process"; you will be asked to confirm the end process.

4.4.4 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.

Once all the files have been copied, you can restart the LoggerNet server and let the server start data collection for all of the stations. If all of the stations are using scheduled data collection, the server will automatically call at the next scheduled collection interval, and collect as much of the missing data as the dataloggers have available. For any dataloggers not scheduled for data collection you can use the Connect Screen application to get data collection up to date.

4.4.5 Directory and File Descriptions

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

c:\CampbellSci\LoggerNet	The main working directory. The datalogger programs created by the user and the collected data files are in this directory.
\Logs	The server operational and low level log files.
\RTMC\Images	A collection of graphics objects to use in creating a real-time display.
\RTMC\sounds	Wave files for use in audible alarms.
\SYS\IniFiles	Initialization files for client applications.
\SYS\bin	Application server working directory.
\Logger1	Directory with the last logger program file sent.
\Logger2	Directory with the last logger program file sent.
\Data	Directory containing station data cache files.
\2	Directory containing data cache for station 2.
\3	Directory containing data cache for station 3.

Section 5. Setup Screen

The Setup Screen provides a way to create and maintain a network of dataloggers. The datalogger network map shows all of the devices and communications links to reach the datalogger stations. The settings for all of the devices are displayed and can be modified on the configuration tabs.

The Setup Screen also provides the configuration for tasks that can be associated with data collection.

5.1 Setting Up a Datalogger Network

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

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

- Direct Connect – Simple serial communications typically on demand and close to the computer, or where devices such as an SC32A, short haul modems or RF400 spread spectrum radios are used in a configuration that requires no dialing or addressing to appear "transparent".

- Phone Modem – Connection from a phone modem at the computer to a datalogger attached to a remote phone modem. Cell phone communication is also supported.

- Radio Frequency (RF) – Connection over RF using antennas for line of sight communications.

- Multi-drop Networked Direct Connect – Direct connection over dedicated cables.

- TCP/IP Ethernet or Internet – Connection over a computer local area network or over the Internet using TCP/IP modems at the datalogger.

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).
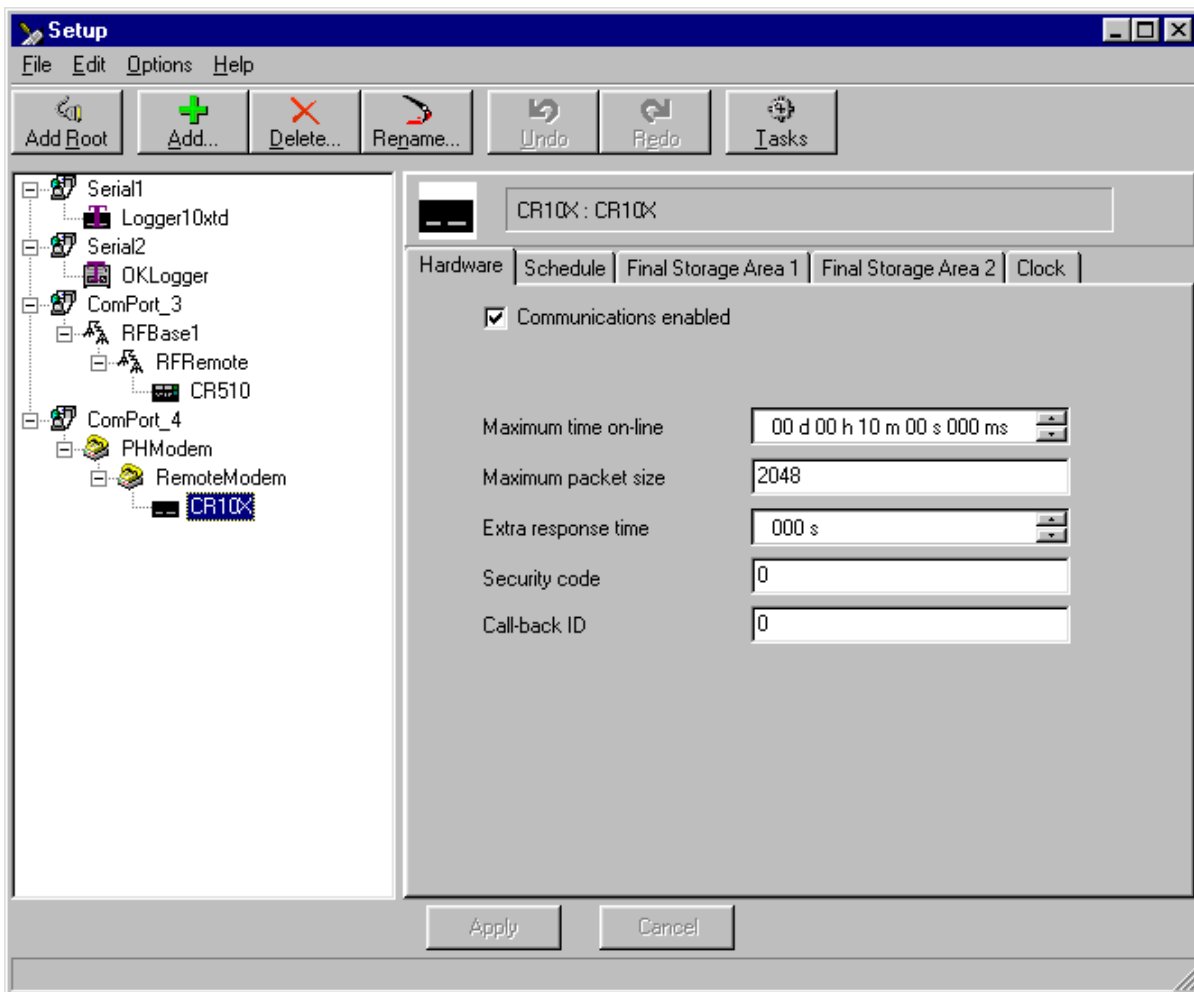
5.1.1 Adding Devices to the Network

Begin adding devices to the device map in the order that they appear in your communications link. Let's assume that your server computer is connected to the datalogger via a telephone modem. You would first add a 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 on the Add Root button, or choose Edit | Add Root. Once the ComPort is in place you can click on the Add button or choose Edit | Add from the menu to bring up the Add Device window.
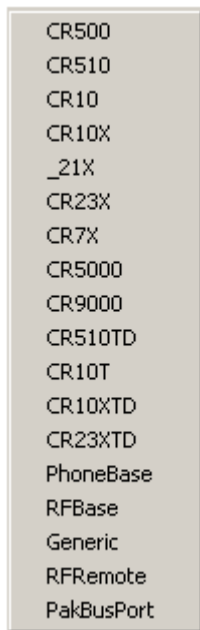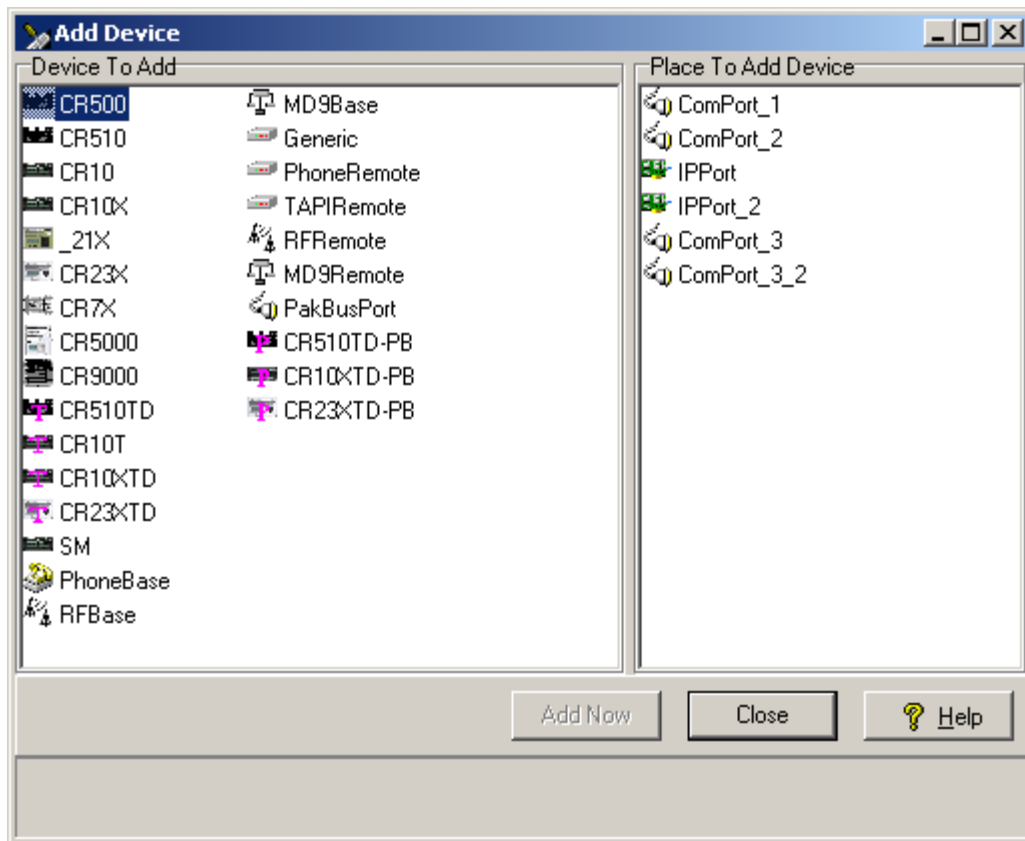
When you select an item from the left side of the Add Device window, valid connections will be displayed in the right-hand column. Highlight the device where you want to attach the new device and click the Add Now button. Continue to add devices in this manner until your network map is complete.

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 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 on a ComPort, only valid connections for ComPorts will be presented.

To delete a device from the network map select the device and either click on 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.

Once all devices are added to the device map, complete the tabs associated with each device. Refer to Section 5.2 for information on setting up devices.

5.1.2 Applying Changes, Undo and Redo

The device map is not saved or entered in LoggerNet until you click on 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 on the Undo button will roll back each change in reverse order to the original 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 using the Undo button.

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

5.1.3 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.

5.2 Device Configuration Settings

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

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

5.2.1 Serial Port

The serial port has only a Hardware tab to configure.

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

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.

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 Com 1 through Com 12. (If you need a ComPort above 12 or the ports on your computer are set up with other names, you will need to use Cora Script to edit this setting. See Appendix F.)

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 breaking the communications link if there is no response from the serial port. Additional time may be needed in instances where the communications link is noisy or network traffic is heavy. If extra response time is needed, it is typically set to 1 or 2 seconds.

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

5.2.2 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.

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.

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.

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 123.456.789.1:1024.

NOTE   When entering the IP address, do not use leading zeros for the address numbers. For example use 123.456.2.34 instead of 123.456.002.034. The software doesn't handle leading zeros.

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 IPPort. Additional time may be needed in instances where the communications link is noisy or network traffic is heavy.

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

5.2.3 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.

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.

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.

TAPI Line – Select the modem you want to use for communication. The modems listed are defined by Windows as part of 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 will have to change the settings for the different applications.

NOTE   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.

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 IPPort. Additional time may be needed in instances where the communications link is noisy or network traffic is heavy.

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

5.2.4 PakBusPort

The PakBusPort is a device that supports a new communication protocol under development at Campbell Scientific. This protocol has not been completed and should be considered experimental. Talk to an application engineer at Campbell Scientific for more information.

5.2.5 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, and whether to automatically update the datalogger's clock.

5.2.5.1 Hardware Tab

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

Maximum Time On-line - A time limit can be set for the length of time 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.

NOTE	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.

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

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

<table>
<tr><td>**NOTE**</td><td>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.</td></tr>
</table>

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.

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 terminal emulation 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 dataloggers.)

<table>
<tr><td>**NOTE**</td><td>Refer to your datalogger operator's manual for complete information on its security functions.</td></tr>
</table>

Call-back ID - 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. Call-back is a mode supported by some dataloggers where an instruction in the datalogger program can initiate a call to the computer. When LoggerNet receives the call it turns around and contacts the datalogger.

5.2.5.2 Final Storage Area 1 and 2 (Array-based dataloggers only)

Array-based dataloggers include the CR7, 21X, CR500, CR510, CR10, CR10X, 21X, CR23X, and CR7. When the datalogger program stores data in an array-based 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 filename 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.

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 will 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, imported to 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 imported to 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 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 Call** - 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 Call is checked, LoggerNet will keep the datalogger on line until it has caught up with the collected data. If Collect All on First Call is not checked, the first call to the datalogger will get the number of arrays specified. 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 most 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.

5.2.5.3 Data Files Tab (Table-based dataloggers only)

Table-based dataloggers include the CR10T, CR510TD, CR10X-TD, CR23X-TD, CR5000, and CR9000. Data output to final storage is stored as records in tables. The data collection 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 you need to click on 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. By default all of the final storage tables along with the Errorlog and Timeset tables are selected for collection. For a description of the built-in tables see Appendix B.3.

The individual tables can be highlighted by clicking on 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 on a table name will toggle collection of that table on or off.

Output File Name - This setting defines the filename and path for the output data file that contains the data collected from the datalogger. Clicking on the browse button (…) at the right of the box will allow you to choose another directory or filename 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.

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.

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 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 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 (no header) - The data is output with timestamp and data values separated by commas with no header.

> ASCII, TOACI1 - 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, 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, TOB1 - The data is stored in Table Oriented Binary format type 1.

Get Table Definitions - This button will cause LoggerNet to get the table definitions from the datalogger. 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. For a description of data tables see Appendix B.3.

5.2.5.4 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 on 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 communication when the schedule is enabled and applied. This setting is also used with the Collection Interval to determine the time data collection will be done.

NOTE	Unlike PC208W, 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 just normal and primary collection schedules 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.

5.2.5.5 Clock Check/Set Tab

Automated Clock Check Enabled - Enable this to compare the datalogger's clock to the server PC's clock based on the schedule defined by the other

parameters on this tab. If the datalogger's time differs from the server's time by more than a specified amount, the datalogger's clock will be set to the server's time.

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

Setting up a clock check may not be desirable. It is possible to end up with missing data or duplicate data if the datalogger's clock is set forward or backward enough to skip or duplicate a data storage event. Special consideration should be given if the PC clock automatically adjusts for Daylight Savings Time.

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

Time Zone Offset - A value can be entered into this field to set an offset for the datalogger's clock from the server's clock. Positive values set the logger clock as if it were in a more eastern time zone. This may be useful if the server and the datalogger are in different time zones.

Initial Date - The initial date field is used to define the date on which the first clock check will occur. If the date entered in this field has already passed, the datalogger's clock will be checked at the next scheduled 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 24 hours, the datalogger's clock will be checked daily, based on the initial date and time.

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

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

5.2.6 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.

Hardware Tab

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.

5.2.7 RFRemote

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

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

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.

RF Options

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 EOL (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 SOL (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.

5.2.8 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.

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.

5.2.9 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.

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.

5.2.10 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 telephone modem has only the Hardware tab. This device must be properly installed and configured in the operating system to use one of the computer's ComPorts before it can be used by LoggerNet.

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.

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.

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.

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

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

5.2.11 PhoneRemote

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

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

Phone Number/Delay Field - This field is used to enter the telephone numbers or dialing strings for the remote modem. To add a number to the list simply click on 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 on the number and use backspace.

Multiple entries are allowed to accommodate entry of access codes, credit card numbers, or other numbers that may be needed for communication to the remote modem and its attached device. As an entry is created a new line is added for additional entries as needed. The first 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.

5.3 Setting the Clock

A datalogger's Clock tab can be used to define a schedule at which an automatic clock check will be performed (refer to Section 5.2.3.5). Based on the information entered for this tab, 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.

NOTE   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.

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

NOTE   Changing the computer system clock while 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.

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.

NOTE   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.

5.4 Setting Up Scheduled Data Collection

5.4.1 Data Collection Scheduling Considerations

For any data to be available for viewing in any of the clients, the data has to be collected from the dataloggers. As the data is collected from the dataloggers it is placed in data files according to the settings specified for that station. The data is also kept in data storage areas by LoggerNet for use by the display screens such as the Numeric Display and Graph screens.

NOTE   Input locations and Public variables don't have to be scheduled for collection to be displayed on the Numeric Display or the Graph.

For data to be collected it has to be selected for collection on the Data File or Final Storage tab in the Setup screen for each datalogger (Section 5.2.3). If the tables or final storage are not selected for collection, no data will be stored or displayed on the display screens.

5.4.1.1 Intervals

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

5.4.1.1.1 Datalogger Program Intervals

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

- **Program Execution Interval** – The execution, or scan, interval determines how often the datalogger carries out the instructions in the datalogger program. It is specified in seconds and determines the fundamental rate at which data is available. In typical programs the sensor readings are taken at this rate and the values are stored in corresponding Input Locations or variables. This execution interval is the fastest that data measurements can be updated and data stored. (Depending on how the program is written, sensor readings may occur at specified intervals and not on every program execution.)

- **Table Storage Interval 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 from a data table.

5.4.1.1.2 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 (Section 5.2.3). If the collection interval set up is faster than the rate at which data is being stored to a data table by the datalogger program, data will not be collected every call, but only at intervals when new data is available.

If data collection is enabled for input location data (Inlocs) on CR10X-TD family dataloggers or Public tables on CR5000 or CR9000 type dataloggers, the current values will be collected every time a scheduled collection occurs, whether or not the values have been updated.

5.4.1.1.3 Communications Path Considerations

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

Phone modems require anywhere from 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 additional 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 any manual 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.

5.4.2 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 (Section 5.2.3.2 or 5.2.3.2). The Schedule tab (refer to Section 5.2.5.4) 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 internal storage, which is accessible to the Numeric Display and Graph.

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   For table-based dataloggers, if no table names appear on the Data Files Tab, click on 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 normally begin immediately.

The Status Monitor screen (Section 7) can be used to ensure that data collection is occurring on the defined schedule. If data is not being collected, check the following:

- The Scheduled Collection Enabled box on the 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 enabled.

- Unsuccessful attempts to communicate with the datalogger may exhaust the number of Primary Retries specified so that the Secondary Retry interval is in effect. Check the date and time listed for the next data collection in the 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

5.5 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.

Clicking on the Tasks button or selecting Tasks from the Options menu brings up the Task Master window. This window shows all of the dataloggers in the network and any tasks that have been defined. If a task is attached to a datalogger as Task_2 in the example below, the task execution will be based on a datalogger collection event. Task_1 is not linked with a datalogger and will run as a scheduled event.
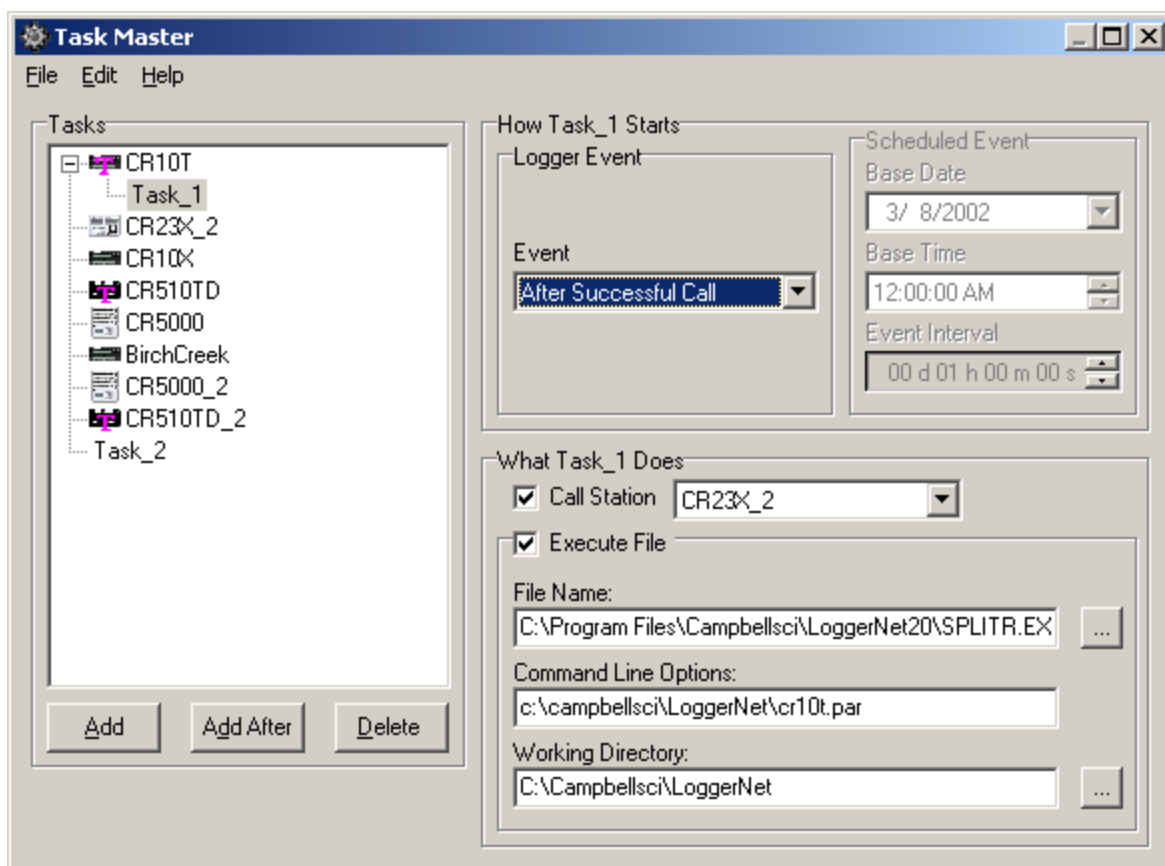
5.5.1 Adding Tasks

To add a task that will run based on a data collection event for a datalogger, select the datalogger by clicking on it. Then click on 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 a datalogger. A task linked to another task has no start options but will execute the specified action following the completion of the linked 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 on 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 on 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.

5.5.2 Logger Event Tasks

There are eight data collection events that can be selected to trigger a task linked to a datalogger. Clicking on 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 you flexibility in deciding when a linked task should be run.
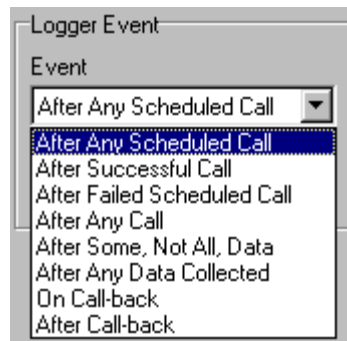
- **After Any Scheduled Call** – The task will run each time scheduled data collection is attempted. This will run whether or not the attempt succeeded or data was collected.

- **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.

5.5.3 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, time and any offset for the task execution. The interval specifies the time between task executions. In the example shown the task will start on March 12,2002 and run once an hour at 15 minutes past the hour.

5.5.4 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 either initiate a call to a datalogger or 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 the configuration set for scheduled data collection. Click on the triangle next to the drop down list for a list of all the dataloggers in the network.

Execute File will run the listed program or script file. Click on the button to the right of the **File Name** to browse the computer for applications or scripts. Any file types that are registered with Windows such as audio, text or graphic files will bring up the associated application. This will allow you to put in a wave file that will automatically announce the trigger condition.

Any command line parameters that are needed for the program can be listed on the second line as **Command Line Options**.

If the application needs to have a working directory specified for other resource files, the directory path is specified in the **Working Directory** box. Clicking on the Browse button (…)to the right will bring up a directory list to choose from existing directories.

5.6 Convert from PC208W Network

The convert utility 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. To run convert, shut down PC208W and bring up LoggerNet. You should also make a backup copy of the PC208W files.

Make sure that the network map in LoggerNet 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 after you delete all the devices or they will all come back.)

```
Convert .DND File                                              [X]

Please select a .dnd file to load into LoggerNet

C:\PC208W\PC208W.dnd                                        [...]

          ✓ OK          ✗ Cancel          ? Help
```
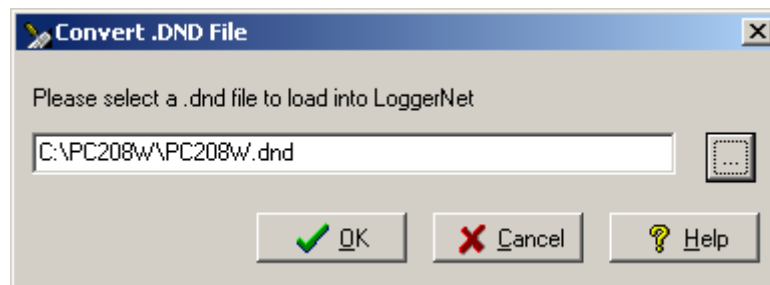
Select the Convert from PC208W Network from the Options menu on the Setup screen. A pop-up message will remind you that data collection should be up to date. Then the dialog box shown above will come up allowing you to choose the PC208W network description file (pc208w.dnd) you want to convert. You can either type in the path or click on 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 by default have the name and path as shown in the dialog box above. Once you have found the network description file to convert, click on 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. The data collected by PC208W will remain in the PC208W directory and is not brought over to LoggerNet. LoggerNet will start data collection and create new files and data storage based on the scheduled collection settings for the dataloggers.

Section 6. Connect

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.

6.1 Connecting to the Datalogger

The Connect screen works with and displays data from only one datalogger at a time. You must select and connect to the datalogger before the other functions of the Connect screen become available. 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 clicking on the check box Sort Alphabetically. All the station names will be listed alphabetically to make it easier to find a specific station.
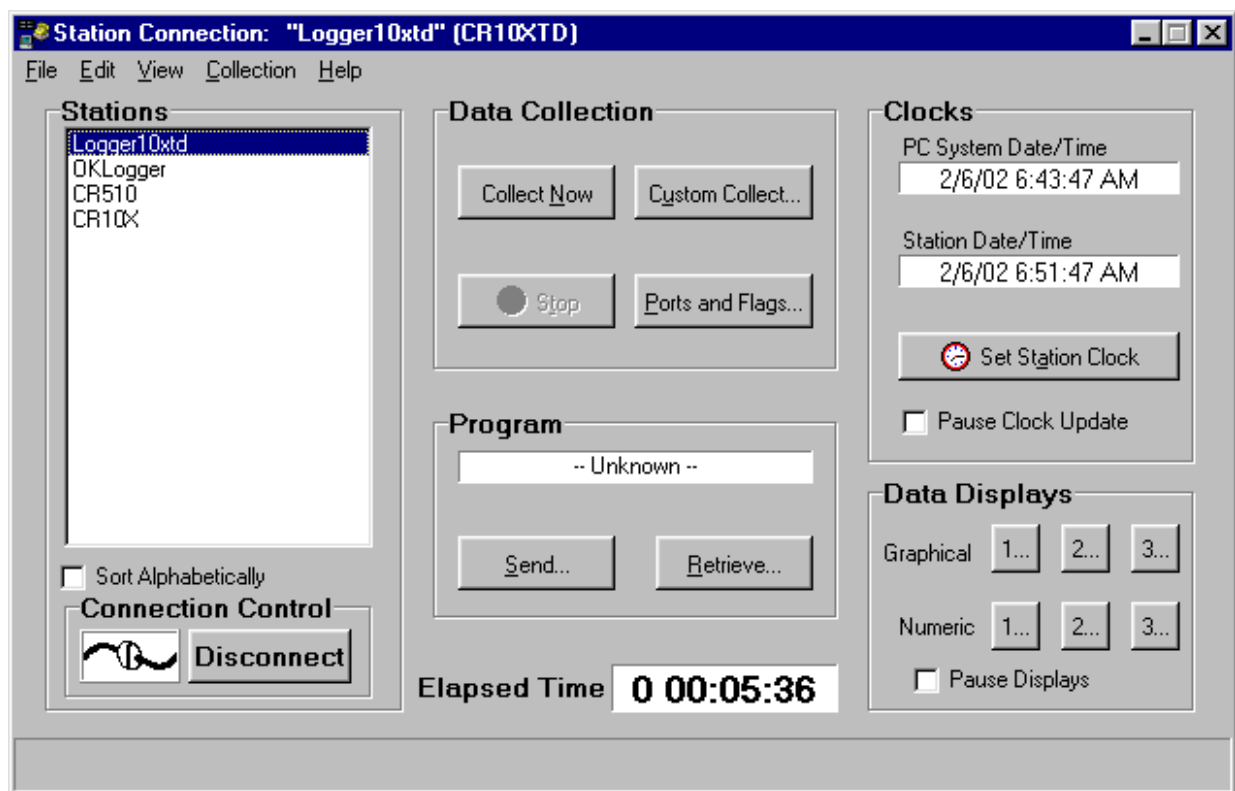
When you click on the Connect button the animated graphic will indicate the connection state. It will show that LoggerNet is trying to establish the connection; the two connectors join together when the connection is made. You can also connect to the datalogger by double clicking on the datalogger name 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. The connection takes priority over other communication so 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 the 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 on the button that now reads Disconnect. To select another datalogger you have to disconnect from the first one. Double clicking on another datalogger will disconnect from the first datalogger and select the new one without prompting. (You will need to click the Connect button to connect to the new datalogger.)

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.

6.2 Data Collection

The Data Collection section of the Connect screen allows you to control some aspects of collecting data from the datalogger. You can force a scheduled collection to occur or initiate a custom data collection. This is also where you have access to view and set ports and flags.

6.2.1 Collect Now/ Stop Collection

The Collect Now function is the equivalent of doing a scheduled data collection for the datalogger without waiting for the scheduled time. Clicking on 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 the graph; or before data files are copied for processing.

Once you have started data collection with Collect Now, you can stop it by clicking on the Stop button or 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.

6.2.2 Custom Collection for Array-based Dataloggers

Clicking on the Custom Collection button while connected to an array-based 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.
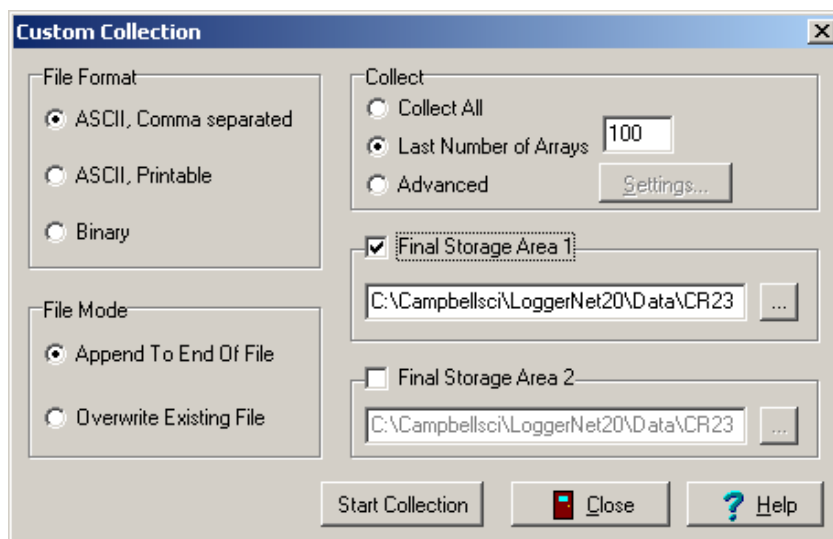
NOTE

While retrieving data from the datalogger using Custom Collection, scheduled data collection will be suspended. The 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.

File Format

−   **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 in tab separated columns. 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
```

−   **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 processing with Split.

File Mode

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

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.

– **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.

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 on the browse button (…) next to the filename will bring up the file select dialog.

6.2.3 Custom Collection for Table-based Dataloggers

Clicking on the Custom Collection button while connected to a table-based datalogger brings up the dialog box shown below. The options you can specify include the file format; whether to overwrite, append or create a new file; how much data to collect; 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.

Collect Options

– **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.

– **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 run. 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.

Starting/Ending Date and Time

The Starting and Ending Date and Time is 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 selects the file format for the output data file.

− **TOACI1** Table Oriented ASCII type 1 format. This format has a two-line header giving the station and table name along with the labels for each of the data values. The data is written one record per line in ASCII text with the timestamp at the beginning of each line. This file can be opened in LoggerNet View, a text editor, processed using Split, or brought into a spreadsheet application. The example below shows the data from a OneMinute table in a CR510TD datalogger.

```
"TOACI1","CR510TD","OneMinute"
"TMSTAMP","RECNBR","BattVolt_Time_MIN","RainTip_TOT"
"2002-02-13 16:29:00",10519,13.94,"2002-02-13 16:28:07",0
"2002-02-13 16:30:00",10520,13.94,"2002-02-13 16:29:07",0
"2002-02-13 16:31:00",10521,13.94,"2002-02-13 16:30:09",0
"2002-02-13 16:32:00",10522,13.94,"2002-02-13 16:31:06",0
"2002-02-13 16:33:00",10523,13.94,"2002-02-13 16:32:06",0
"2002-02-13 16:34:00",10524,13.94,"2002-02-13 16:33:09",0
"2002-02-13 16:35:00",10525,13.94,"2002-02-13 16:34:06",0
"2002-02-13 16:36:00",10526,13.94,"2002-02-13 16:35:07",0
"2002-02-13 16:37:00",10527,13.94,"2002-02-13 16:36:08",0
```

− **TOA5** Table Oriented ASCII type 5 format. This format has a multi-line header including the TOACI1 information along with unit specifiers and an indication of the output process used for each of the data values. The data is written one record per line in comma separated ASCII text with the timestamp at the beginning of each line. This file can be opened in

LoggerNet View, a text editor, processed using Split, or brought into a spreadsheet application.

```
"TOA5","CR5000,"","","","","","","OneMin"
"TIMESTAMP","RECORD","bat_Avg","temp_Avg","counter_Avg","curve_Avg"
"TS","RN","","","",""
"","","Avg","Avg","Avg","Avg"
"2002-02-13 16:29:00",11300,13.05,24.5,178.9,0.013
"2002-02-13 16:30:00",11301,13.05,24.5,182,-0.018
"2002-02-13 16:31:00",11302,13.05,24.5,178.3,0.013
"2002-02-13 16:32:00",11303,13.05,24.5,180,-0.001
"2002-02-13 16:33:00",11304,13.05,24.5,181.5,-0.012
"2002-02-13 16:34:00",11305,13.05,24.5,178.1,0.018
```

− **TOB1** Table Oriented Binary type 1 format. This is a compressed binary format that can be converted by View or processed using Split.

Data Collection

− **Select Tables** - Tables are selected by clicking the check box next to the table name. The Select All checkbox provides a shortcut to select all the available tables in the datalogger.

− **Change File Name** will bring up a file selection dialog to choose the directory and filename for the highlighted table. To change the name of the file, highlight the table by clicking on the table name. Then click on Change File Name for the file dialog. Each data table will be saved in its own file. By default the filename is the name of the station and the name of the table. Right click on the file name will allow you to apply the directory for the selected file to all of the files.

NOTE   You can have all the file stored to the same directory by changing the directory for one of the files and then right click on the file and choose Apply Directory to All. The directories for all of the files will then point to the same place.

− **Reset Tables** (CR5000 and CR9000 dataloggers only) lets you clear out the data and reinitialize the data tables in the datalogger and the data cache. Clicking on Reset Tables will bring up a list of the tables defined in the datalogger. Select the tables you want to reset (or click in the Select All checkbox to get all of the tables) and click on 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 (App. C) for the selected tables. Make sure you don't need the stored data or the data has been saved to a file.**

6.2.4 Ports and Flags

The Ports and Flags window shows the current state of the ports and flags for the datalogger. If the flag is enabled or the port is high, the button next to the name appears green; if the flag or port is low, the button appears black.
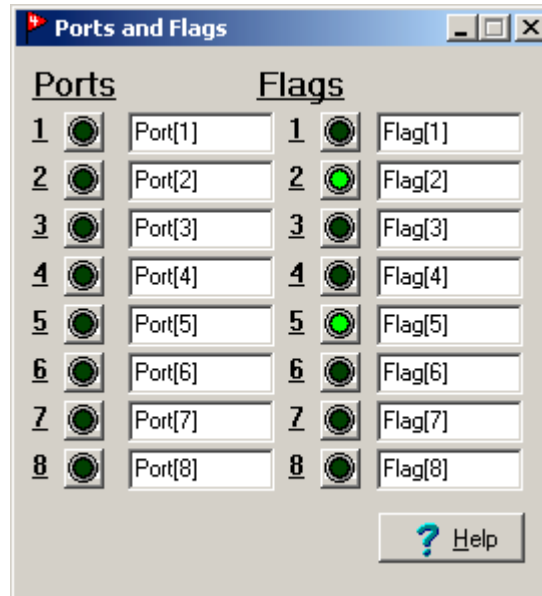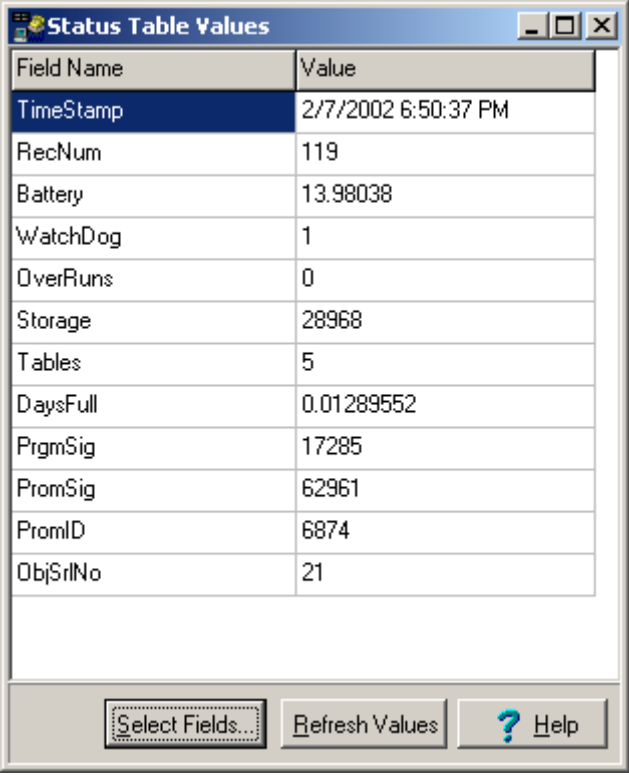
Click on the button next to the flag or port to turn it on or off. (Remember that ports can only be set if they are configured as outputs in the datalogger program.) You can customize the names by clicking on the name and typing the desired text.

6.2.5 View Status Table (Table Dataloggers Only)

To view the status table for a table-based datalogger select View | Status Table from the menu on the Connect screen. A window similar to the one below will be displayed. Click on 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 on 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 Section B.3.

Status Table Values	
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

Select Fields...   Refresh Values   ? Help

6.3 Datalogger Clock

Once the connection to the datalogger is established, the datalogger clock is checked continuously and displayed along with the computer clock as updates are received. The clock update can be paused by clicking the checkbox 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 on 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.

An automatic scheduled clock check can be set up in the Setup screen (Section 5.2.3.5).

NOTE   If a manual clock set does not change the datalogger clock, check the allowed deviation specified in the Setup screen for the datalogger's automatic clock check (Section 5.2.3.5). If the difference in clocks is less than the specified deviation, a manual clock set will not have any effect.

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

6.4 Program Management

The Program section on the Connect screen is used to send or retrieve datalogger programs from dataloggers in the network.

6.4.1 Programs Created Using Edlog

After a program is created and compiled using Edlog (Section 8.2) Program Send on the Connect screen is used to transfer it to the datalogger. To transfer a program, first connect to a datalogger, then click on 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 filters the displayed files showing only Edlog files (*.dld).

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 array dataloggers data is not lost if the memory configuration does not change, table-based dataloggers clear 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 (Section 5.2.3.1). Otherwise, access to the datalogger may be limited or completely denied.

The program running in the datalogger can be retrieved and saved to a file by connecting to the datalogger and clicking on the Retrieve button. You will be prompted for a name and directory to store the retrieved file. A retrieved *.dld file can be imported into Edlog for editing by using Edlog's Document DLD feature (Section 8.2.4). This is useful if the original file has been corrupted, lost, or erased. Programs cannot be reliably retrieved over some slow communications links.

6.4.2 Programs Created Using CRBasic Editor

Working with CR5000 and CR9000 dataloggers provides additional options when programming the dataloggers and managing the datalogger programs. These dataloggers have a built in file system much like a computer hard disk. Multiple files can be stored in the datalogger memory, including data files and datalogger programs. The user can choose a program to run, stop execution of the active program, or select a program to be run on power-up of the datalogger.

The Connect screen Program Send for the CR5000 and CR9000 dataloggers works similarly to the other dataloggers.

After a program is created using the CRBasic Editor (Section 9.2) Program Send on the Connect screen can be used to transfer it to the datalogger. To transfer a program, first connect to a datalogger, then click on 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 filters the displayed files showing only CRBasic files (*.cr5 or *.cr9).

After selecting a datalogger program file a warning will appear to remind you that data will be lost when the new program is sent. 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 message from the datalogger indicates a successful compile the progress box will close.

The active program running in the datalogger can be retrieved and saved to a file by clicking on the Retrieve button. You will be prompted for a name and directory to store the retrieved file. Program files retrieved from a CR5000 or CR9000 can be opened directly in the CRBasic Editor. This is useful if the original file has been corrupted, lost, or erased.

An alternate way to send a program to these dataloggers is to use the File Control dialog that comes up by selecting from the menu View | File Control (see Section 6.6 for details).

6.5 Data Displays

The Connect screen Data Displays provide display screens to monitor numeric values or view graphs of the collected data. 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. Active scheduled collection will supply updated data for these displays automatically as it is received from the datalogger.

NOTE   Input locations, ports and flags for array-based and CR10X-TD type dataloggers, and public variables for the CR5000/CR9000 are automatically retrieved from the datalogger when selected for display. These do not have to be added to scheduled data collection to be viewed.

Updates to the displays can be suspended by clicking the check box Pause Displays. This will stop all the updates to the 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.

6.5.1 Graphical Display Screens

Data values collected from the datalogger can be plotted on a line graph in the Graphical Display. Up to three Graphical Display screens can be active. The settings and selected data values are retained as part of the graph settings even when the graph or Connect screens are closed. The settings for each datalogger are saved independently so a different datalogger will have different settings.

Clicking on one of the three numbered buttons next to the Graphical display area will bring up a Graphical Display screen. An example of one of these screens is shown below. If a Graphical Display screen is already active but hidden behind other windows, clicking on its number will bring it to the front.

NOTE	Final storage data must be collected by LoggerNet before it can be graphed.

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.
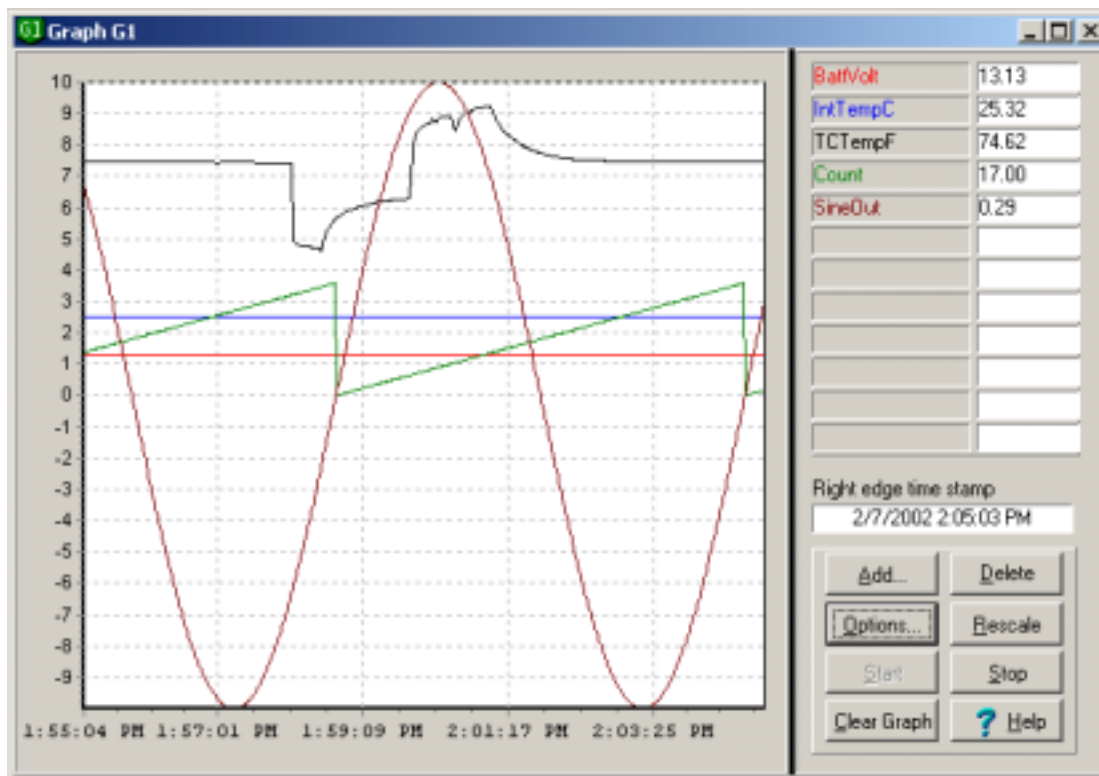
6.5.1.1 Add

When first opened the Graphical Display is initially blank; the fields to be plotted must be selected. Press the Add button to bring up the Available Fields dialog box that lists the data fields for the available datalogger tables or final storage. 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 on the data field names. Multiple data fields can be selected by holding down the Shift or Ctrl key while clicking additional names.

The selected data fields can be added to the graph either by clicking on the Paste button to enter them on the graph starting with the selected cell, or dragging the selected fields to the display cells on the graph. An entire table or array can be added by dragging the table onto the cells, or by selecting the table and clicking on Paste.

Up to twelve data fields can be graphed simultaneously. The Available Fields dialog can be kept in front of other windows by clicking the Stay on Top check box.

For array-based dataloggers that have an associated datalogger program, the final storage labels and input locations names are available to select and add to the graph. If the datalogger doesn't have an associated file, the input locations can still be graphed by clicking on the Select Input Locations button at the bottom of the screen. The numbers of the input locations to graph 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 graph: input locations 2,3,4,6,14,15,16, and 17.
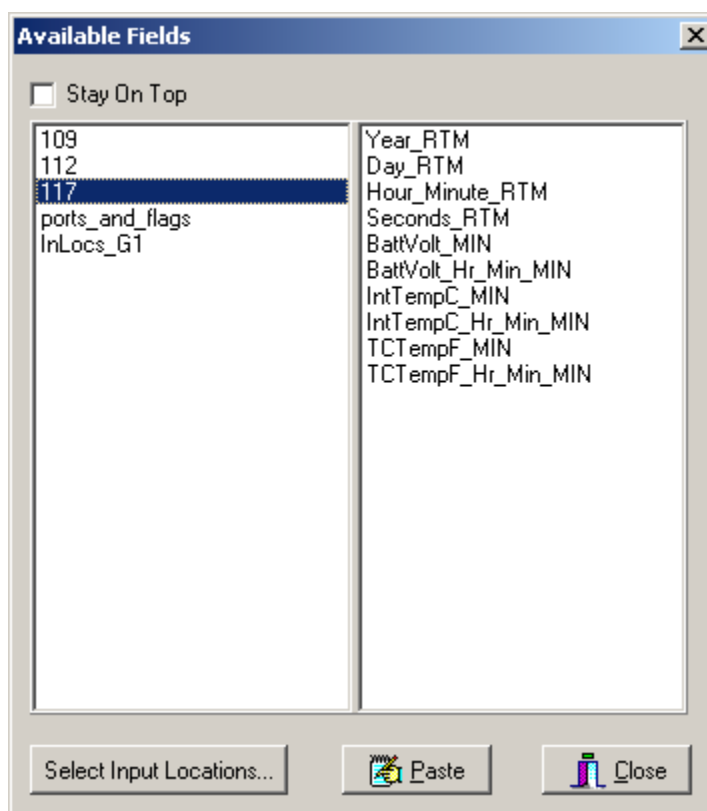
Once data fields have been added, the Graphical Display will start automatically. If there is historical data for the selected fields in the data cache, it will be displayed on the graph. Note that Input Location or Public table data don't have history stored in the data cache; therefore, no historical data will be displayed.

6.5.1.2 Delete

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.

6.5.1.3 Graph Options

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

6.5.1.3.1 Graph Scaling Options

The Graph 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 Ten 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. (Section 6.5.1.3.5) The scales are only shown on the graph if there are data fields linked with them.
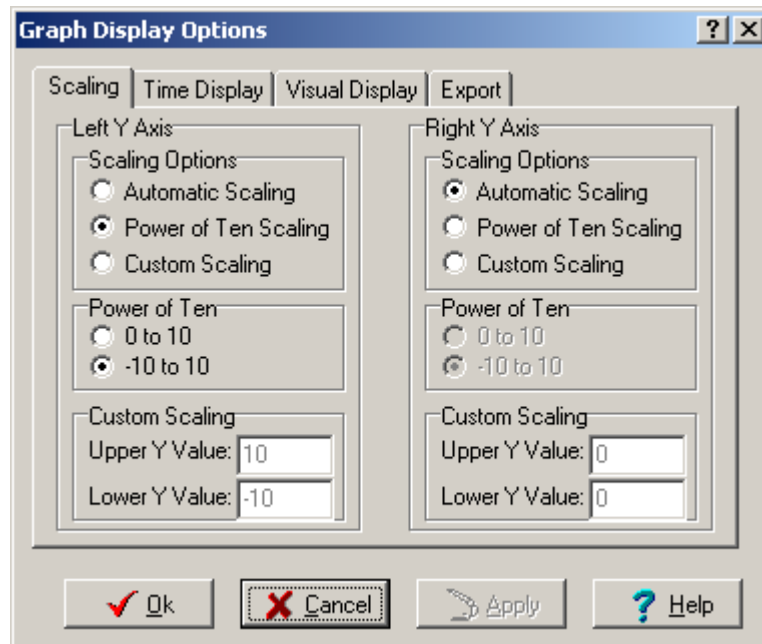
Scaling Options

– **Automatic Scaling** will adjust 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 Ten Scaling** applies a Power of Ten 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. This option will scale the data values by factors of ten so they will fit between 0 to 10, or -10 to 10, depending upon the option chosen. When this option is in effect, pressing the Rescale Button on the Graph tab 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.

6.5.1.3.2 Graph Time Display Options

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.

Select Update Interval defines how often the Input Locations or Public Table data are updated on the graph. A shorter interval here will increase the amount of communication taking place with the datalogger. Final storage data will only update when collected by scheduled data collection.

There are two **Time Axis Display** options for the X axis. You can choose to Show Time and Date, or Display Just Time.
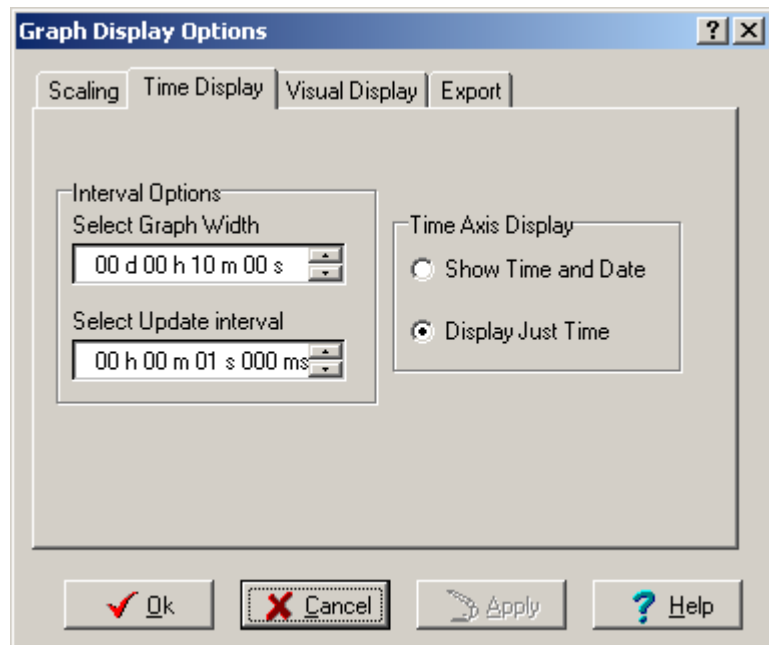
6.5.1.3.3 Graph Visual Display Options

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

−   **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.

−   **Left / Right Grid Color** selects the color of the grid lines that go with the left or right scale axis.

Trace Options selects whether the trace will be displayed with the line plot, the actual data points or both. The line color, width and style for the trace are set in Cell Options (Section 6.5.1.3.5).

Title allows you to place a descriptive title on the graph. Enter the text of the title in the text box, then click on the Show Title checkbox to make the title visible. Click on the Font button to choose the font style, size and color for the title.
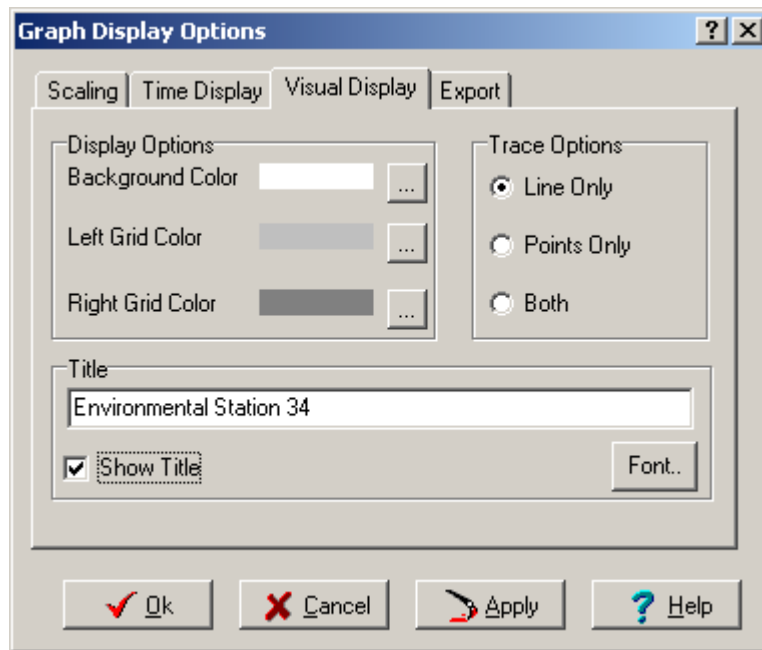
6.5.1.3.4 Graph Export Options

Graph Export will save an image of the displayed graph to the clipboard, a graphics file, or will print the graph to a printer. The file export can be saved either to a bitmap (*.bmp) or a windows metafile (*.wmf) format. Most graphic file applications can import one or both of these formats.

The Margins setting provides for a border area around the graph. These are specified in % of the total linear measurement. The size of the saved graph

image depends on the size of the graph on the screen. If the graph is maximized then the saved file will save the large image.
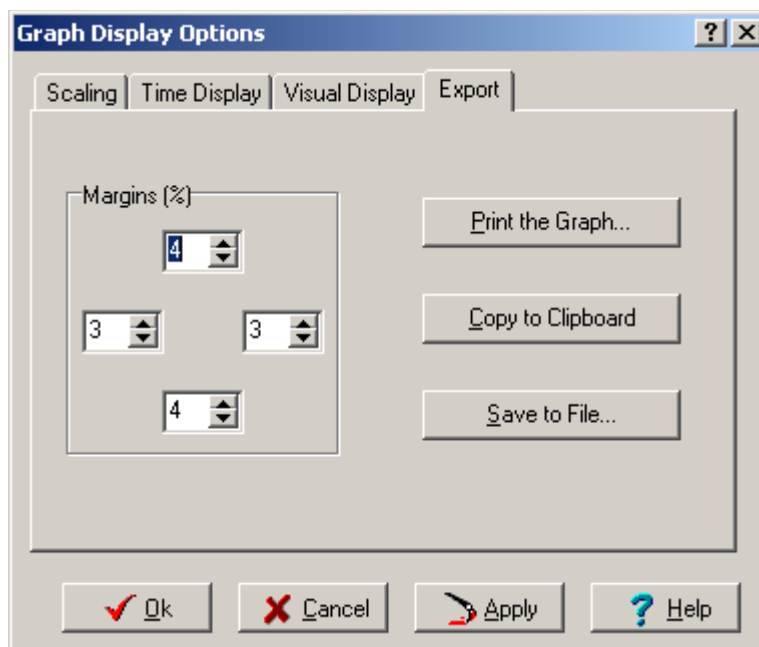
6.5.1.3.5 Data Field Display Options

The display options for the data fields can only be accessed using the right click menu. Right click one of the data fields on the graph and the following menu will appear:

Cell Options brings up the dialog box shown below. This allows the configuration of various trace options.
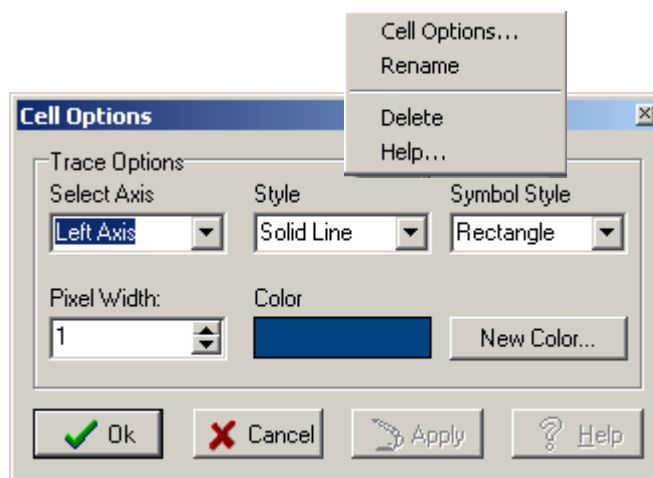
Select Axis sets whether the data trace is linked to the left or right axis. The displayed graph will depend on the axis settings for the selected scale.

Style selects the style for the line.

Symbol Style selects the appearance of the individual data points.

Pixel Width sets the width of the trace line. 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.

6.5.1.4 Rescale Graph

Rescale Graph is only used for Power of Ten Scaling. Sometimes the scale factor for a field is fixed before the field has reached a maximum. This may cause the data to be displayed partially or completely off the Graphical Display. Clicking on the Rescale Graph button causes the graph to re-evaluate the scale factors for all the fields and adjust them as necessary to keep them within the limits.

6.5.1.5 Start / Stop Graphing

While the graph is running, any new data received from the datalogger will be automatically plotted on the graph and the graph scrolled forward. The graphing can be stopped to capture some event to a graphics file by clicking on the Stop button. Clicking the Start button will start the graph again, including filling in historical data values for final storage data.

6.5.1.6 Clear Graph

Clear Graph clears all the trace plots from the Graphical Display area. The graphing of data continues at the right side of the graph as data is collected. To graph the historical data again, click the stop and then the start button.

6.5.2 Numeric Display Screen

Data values collected from the datalogger can be displayed in numerical format on the Numeric Display. Up to three Numerical Display screens can be active. The settings and selected data values are retained as part of the display settings even when the display or Connect screens are closed. The settings for each datalogger are saved independently so a different datalogger will have different settings.

Clicking on one of the three numbered boxes next to the Numerical display area will bring up a Numerical Display screen. An example of one of these screens is shown below. If a Numerical Display screen is already active but hidden behind other windows, clicking on its number will bring it to the front.

NOTE	Final storage data must be collected by LoggerNet before it can be displayed.

Numeric Display N1							
Column 1		**Column 2**		**Column 3**		**Column 4**	
TimeStamp	2:58:55 PM			BattVolt		InputLocation_1	
RecNum	756.00	RecNum	4788.00	IntTempC		InputLocation_2	
Year_RTM	2002.00	flags(1)	True	TCTempF		InputLocation_3	
Day_RTM	38.00	flags(2)	False	Count		InputLocation_4	
Hour_Minute_RTM	1458.00	flags(3)	False	SineOut		InputLocation_5	
Seconds_RTM	55.00						
BattVolt	13.12						
IntTempC	25.39						
TCTempF	88.20						
Count	9.00						
SineOut	0.16						

Delete   Add...   Options...   ? Help

6.5.2.1 Add

When first opened the Numeric Display is initially blank; the fields to be displayed must be selected. Press the Add button to bring up the Available Fields dialog box that lists the data fields for the available datalogger tables or final storage arrays. 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 on 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 on the table name.

The selected data fields can be added to the display either by clicking on the Paste button to enter them on the display starting at the selected cell, or dragging the selected fields to the display cells.

Up to sixty data fields can be displayed simultaneously. The Available Fields dialog can be kept in front of other windows by clicking the Stay on Top checkbox.

For array-based 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, the input locations can still be displayed by clicking on 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. The frequency of the update is set under cell options in the Update Interval. Setting the Update Interval for one cell applies to all cells on the display.
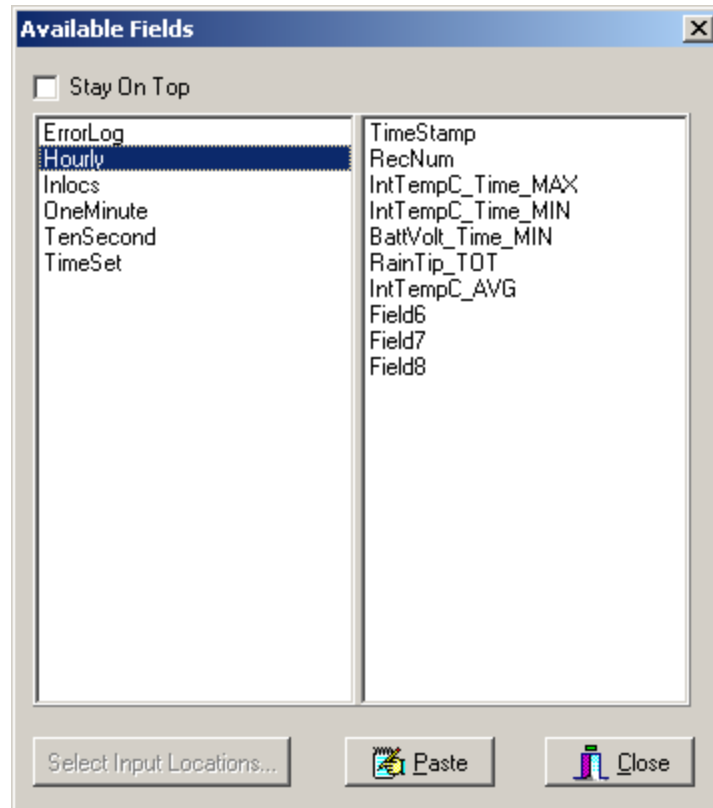
6.5.2.2 Delete

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.

6.5.2.3 Display Options

To customize the Numeric Display, select one or more cells and click on the Options button. The Cell Options dialog box allows you to set up how the data should appear and set visual alarms.

Display sets up the appearance of the numbers being displayed

–   **Number of Decimal Places to Display** sets the number of digits to display to the right of the decimal point. This can be used to show greater precision or present a cleaner appearance, depending on the data need.

–   **Set Data Justification** sets whether the data will be displayed at the right or left side of the display box.

– **Update Interval** controls how often LoggerNet will get updated Input Locations or Public table values from the datalogger if those fields are not part of scheduled data collection. Setting this interval for any cell applies to all fields on the Numeric Display.

There are limits to the number of Input Locations that can be brought over from some of the dataloggers. Older array-based dataloggers can bring in 62 input location values at a time. The newer array-based dataloggers can bring in 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 and CR9000 dataloggers don't have practical limits for the number of Public Variables that can be requested or displayed.

Both array-based and CR5000/9000 dataloggers will retrieve only the requested input locations or public variables. CR10X-TD type dataloggers must retrieve the whole input locations record in one packet. If more than 500 input locations are used, the record cannot be collected.
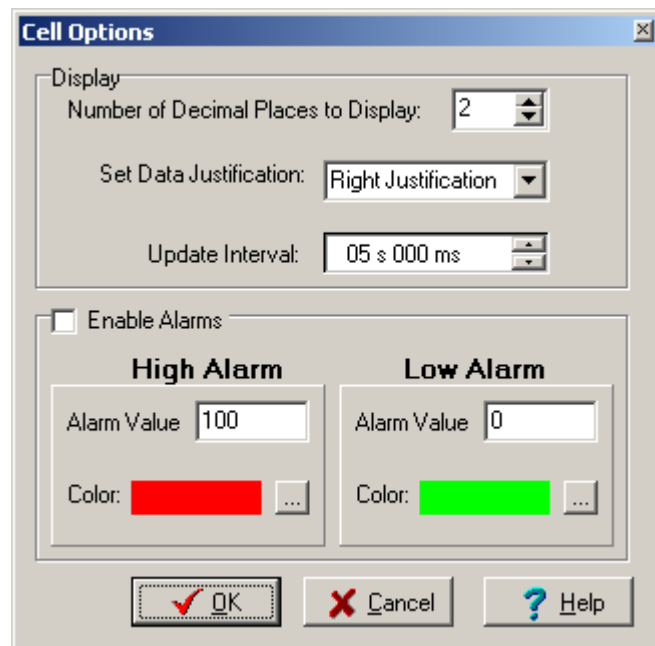
Alarms can be set to turn the display field a different color depending on the value of a data point. Click on the Enable Alarms check box to turn on the alarm feature. The alarm levels can be set for a cell by entering a value in the High Alarm Value field and the Low Alarm Value field. When the data value displayed in the cell exceeds the limits set, the cell color will change based on the colors selected for each of the alarm values.

6.6 File Control for CR5000 and CR9000

To access File Control for the CR5000 and CR9000 dataloggers select View | File Control from the menu. This will bring up the File Control window as shown below.

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

The run attributes 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 Run Now attribute is the active program. Run on Start-up is the program that will run when the datalogger powers up. Run Always is a combination of both attributes set.
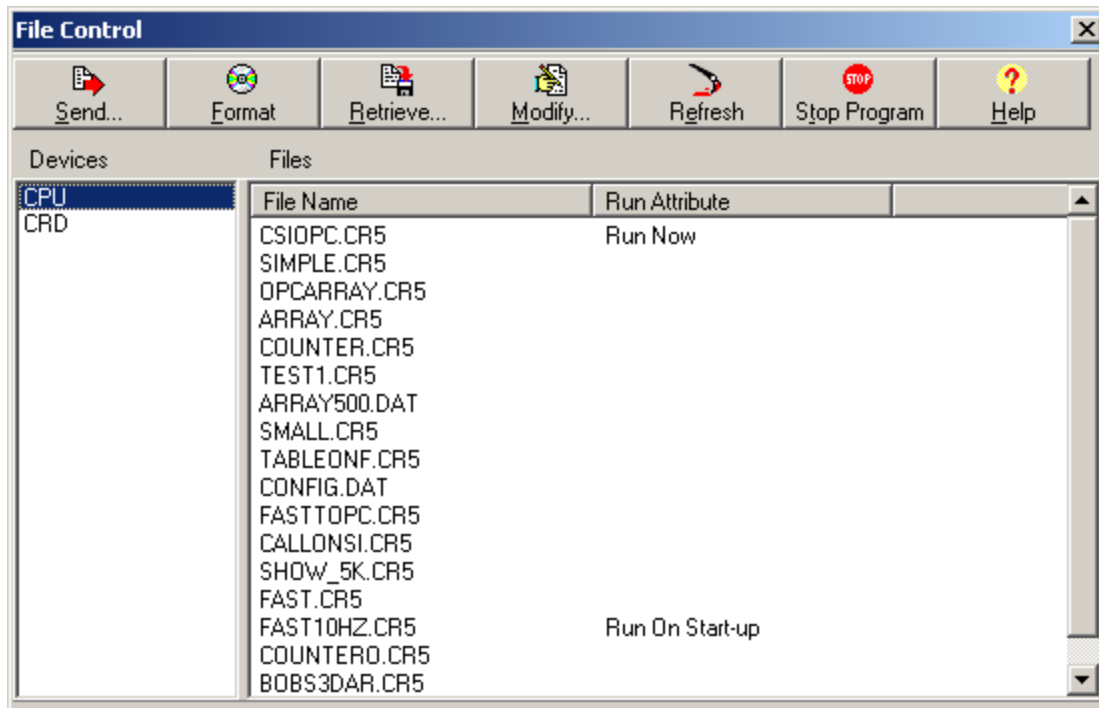
There are several options to work with the files and directories on the CR5000 and CR9000 dataloggers.

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

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

NOTE   Because of the number of special operations that run in the background that don't happen in a normal file transfer, a datalogger operating system has to be sent to the datalogger using the Send Program button on the Program area of the Connect screen. PC9000 can also be used to transfer a datalogger operating system.

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.

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.

Modify brings up a dialog box to change the attributes of the selected file. The device name and filename appear at the top of the dialog screen. Run Now will make the selected file the active datalogger program. Run on Start-up will run the selected program the next time power is cycled to the datalogger. Both attributes checked is Run Always.
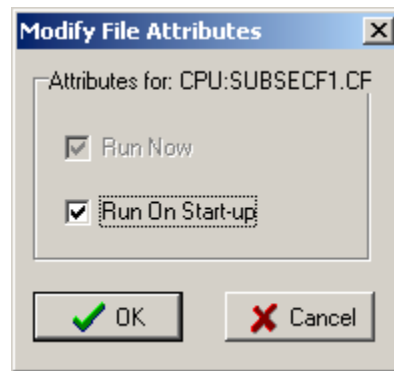
Refresh will update the list of files for the selected device. This is used when the files on the datalogger have changed because of files being created by the datalogger or file attributes being changed by the datalogger under program control.

Stop Program halts execution of the currently running datalogger program.

6.7 View 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 terminal emulation screen select View | Terminal Emulator from the menu.

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

Section 7. Status Monitor

The Status Monitor screen provides a way to monitor communications statistics for the datalogger network. Statistics are displayed for data collection attempts and communication failures. There are also functions available to monitor the log messages for the computer and the tasks that have been set up.

7.1 Main Screen

The main screen displays the network map showing the devices in the network and the statistics associated with each device. The network map is in the leftmost column. The name of each device is displayed next to a status icon that indicates the current communication state for that device.

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

The display defaults with the most commonly monitored statistics for each device and can be customized by the user. For a description of the statistics and their meaning see Section 7.2.1.

Above the display area is a row of buttons providing quick access to many of the functions available in the Status Monitor. 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. The current computer time is displayed in the lower right hand corner.

The Status Monitor functions can also be accessed through the menus at the top of the screen.

7.2 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 screens such communication log monitors or task status. Most of these functions can be accessed in more than one way. Each of the access methods is described for the functions.

7.2.1 Select 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.
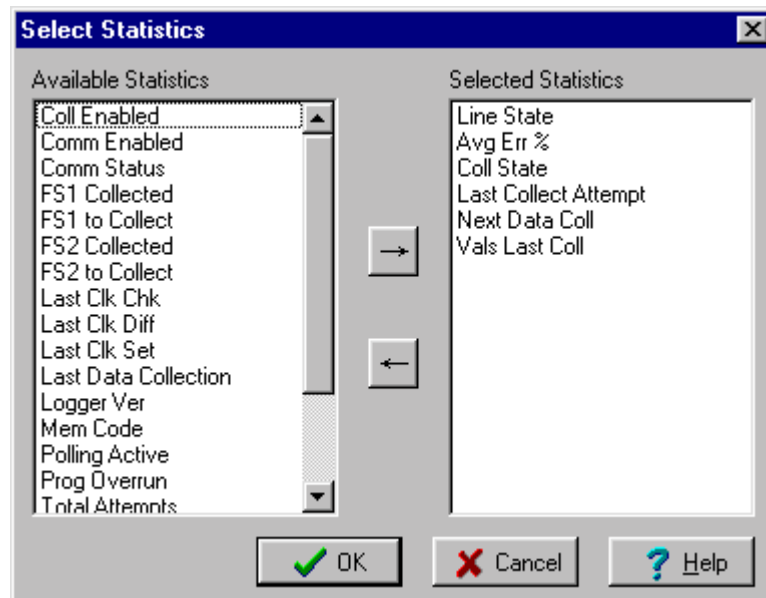
Entries in the Available Statistics field will not be displayed on the main screen. Entries in the Selected Statistics field will be displayed on the screen. The arrow buttons are used to move entries between the two columns. Alternately, an entry can be moved from one column to the other by double clicking 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 Error %)** - 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 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.

  - Communications 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.

- **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.

- **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.

    - Not-applicable - LoggerNet cannot communicate with this device directly. Table-based RF Modems is one example.
    - 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.

- **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.

- **Memory Code (Mem Code)** – (Array-based 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)** – (Array-based 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.

- **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.

- **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)** – (Array-based 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 terminal emulation. See the datalogger operator's manual for more information.

7.2.2 Toggle Collection On/Off

This function 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 on the button or selecting Options | Toggle Collection On/Off.

7.2.3 Reset Retries

Resets the cumulative attempts and retries along with the average error rate for the selected device. Reset Retries is done either from the button or selecting Options | Reset Retries.

7.2.4 Collect Data/Stop Collect

Collect Data 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 Stop Collect 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 on the buttons or on the Options menu.

7.2.5 Server Logs

Server Logs brings up the LoggerNet log message viewer. You can enable the view for each of the logs by clicking the View check box for the log you want to see. The windows can be resized and the scrolling display can be paused by clicking the Pause check box.

There are three logs kept by LoggerNet that track the operation of the server, communications with the dataloggers and data collection. A brief description of these logs and their messages is found in section 7.3. A more detailed description of the messages and their meanings is found in Appendix D.

7.2.6 View I/O

View I/O brings up a Low Level Input/Output monitor for examining serial communications between LoggerNet and a root level device. Both hexadecimal and ASCII representations of the serial communications with a timestamp are seen in the log message window. The R and T at the beginning of each line represent a message received or transmitted by LoggerNet. The scrolling display can be paused by clicking the Pause check box.

The Low Level I/O is brought up by selecting a root level device and clicking on the View I/O button, or selecting Options | View I/O. A description of the Low Level I/O log is outlined in section 7.4.

7.2.7 Task Status

Task Status brings up a window, similar to the one shown below, to monitor the status and operation of tasks that have been defined in the Task Manager.
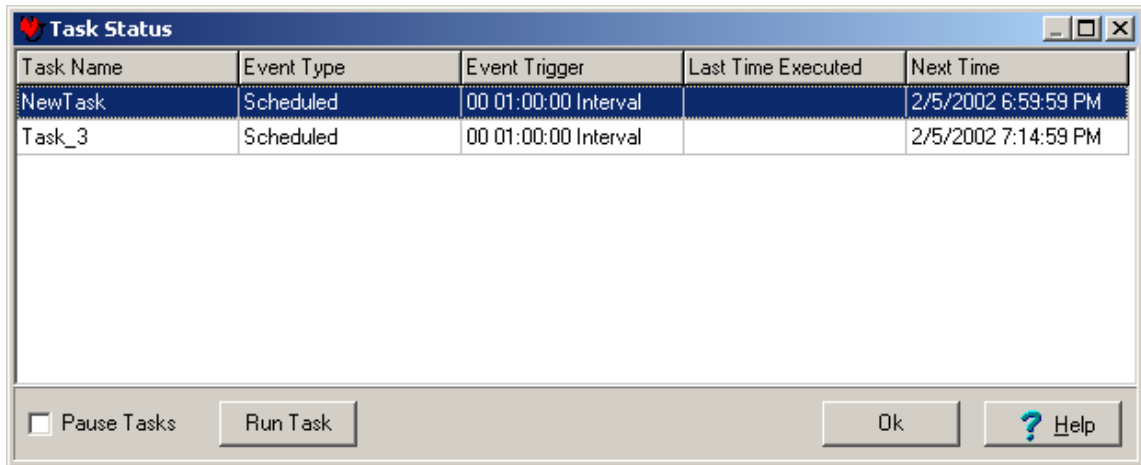
The window shows the type of task, the last time it was run, the interval if it is scheduled or the event that triggers it if it depends on data collection. The Run Task button will run the selected task and can be used to test the operation of a task. There is also a check box that will suspend the execution of all tasks.

7.2.8 Show Stations Only

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 is enabled or disabled by clicking on the check box at the bottom of the Status Monitor screen or selecting Options | Show Stations Only.

7.2.9 Pause Schedule

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 is enabled or disabled by clicking on the check box at the bottom of the Status Monitor screen or selecting Options | Pause Schedule.

7.2.10 Log Settings

Log Settings allows you to control whether the logs are saved to files along with the size and number of log files that are created. One set of log files is created for each of the three LoggerNet logs. If Save to Disk is enabled 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 the next log file created will overwrite the oldest file.

Selecting Edit | Log Settings brings up the following dialog box.
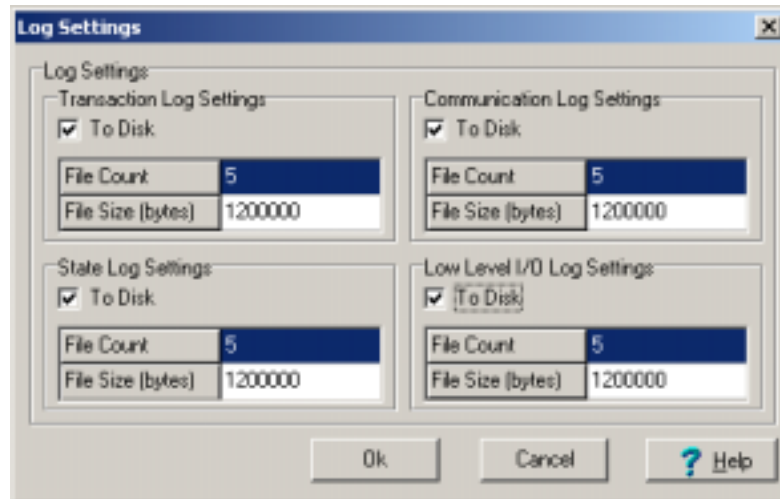
The following settings are used enable saving the logs to disk as well as to control the number and size of the log files.

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

File Count – This setting determines the number of log files to be saved to disk for this type of log. The server will store up to the number specified before overwriting the oldest log.

File Size – This setting determines how big the log file is allowed to grow before being saved to a 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…).

7.3 Monitoring Operational Logs

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

Each log is initially configured in the Status Monitor screen (refer to Section 7.2.10).

In the Status Monitor screen, clicking on the Server Logs button will bring up the LoggerNet logs display. Selecting View Server Messages on the Options menu will also bring up the log display. Individual log displays can be turned on or off by selecting the View check box, or temporarily paused by selecting the Pause check box.

NOTE

Examples of the log files, and the details of the messages can be found in Appendix D.
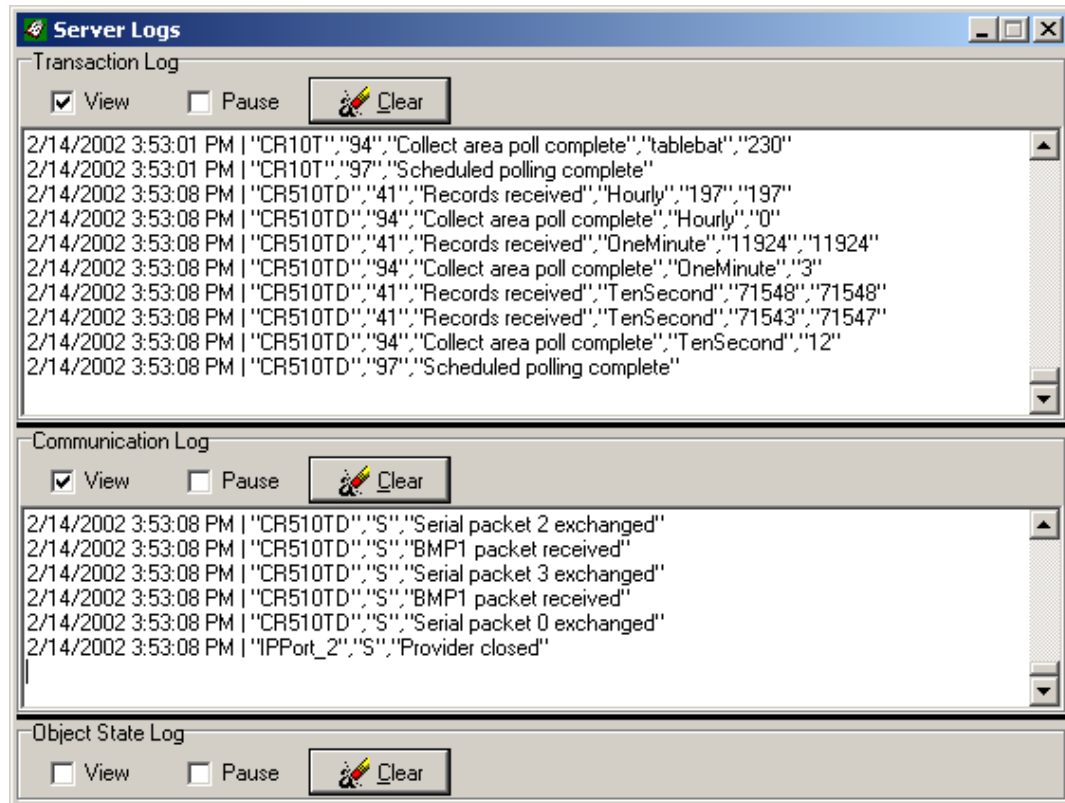
7.3.1 Transaction Log (TRAN$.LOG)

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

7.3.2 Communication Log (COMMS$.LOG)

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

7.3.3 Object State Log (STATE$.LOG)

This log is used for troubleshooting an object in the datalogger network. The information in this log conveys the state of an object at a given time. When saved to disk, the file name for the log will be state$.log, where $ denotes the active log, and state1.log, state2.log, etc. denote historical logs.

7.4 Monitoring Low Level I/O

A low-level log is available for each root level device used for LoggerNet communications. The low-level log shows all of the low-level transactions being passed between the server and the devices in the datalogger network over a communications port. As with the operational logs, this log's initial settings are specified using the Edit | Log Setting on the Status Monitor screen.

The low-level log is displayed by highlighting the root level device in the device map, and clicking the View I/O button or from the menu selecting Options | View I/O.

The names for the files that are saved start with io$ and include the name of the device. For example io$SerialPort_1 log. A separate log file is kept for each root device. The active log is identified with the $. Once the maximum size has been reached, the file is saved and the $ is replaced by a sequential number.

Each line in the log includes a time stamp. An R indicates information that has been received from a device in the network; a T indicates information that is being transmitted to a device.

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

Section 8. Datalogger Program Creation with Edlog

This section provides information on memory allocation and programming for Campbell Scientific's CR7, CR10, 21X, CR500, CR510, CR10X, CR23X, CR510-TD, CR10X-TD, CR23X-TD and CR10T dataloggers. See Section 9 for information about the CR5000 and CR9000 CRBasic program editor.

8.1 Overview

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

Precompiler - Edlog precompiles the program to check for errors and to create the file that is downloaded to the datalogger. The precompiler will catch most of the errors. The errors that the precompiler misses will be caught by the datalogger when the program is compiled. The download file (*.DLD) is stripped of comments to make it more compact. During the precompile step, a Program Trace Information file (*.PTI), which provides an estimate of program execution time, is also created (Section 8.1.1.1). For array-based dataloggers the precompiler also creates a Final Storage Label file (*.FSL) to supply labels for final storage values to be used by other software applications.

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

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

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

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

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

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

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

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

TempF=TempC*1.8+32

8.1.1 Creating a New Edlog Program

To create a new datalogger program, choose File | New from the Edlog menu and select the datalogger type from the dialog box. A template similar to the one shown below appears.
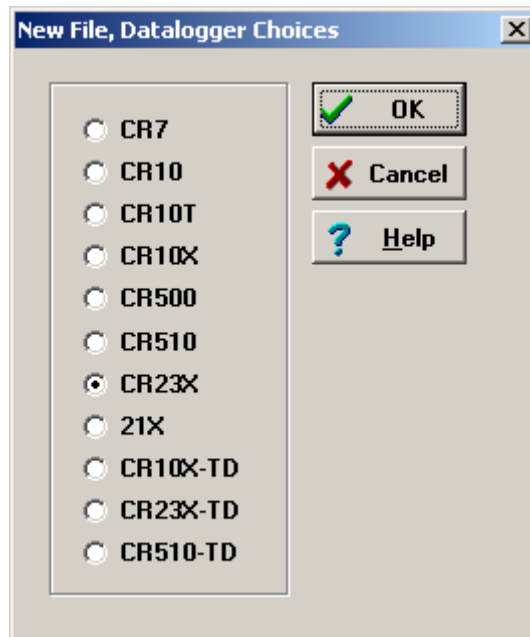
Select the datalogger you are using from the list and click OK. A blank program template will come up as shown below for a CR10X.

```
Edlog 32 (CR10X) - [NONAME1.csi]                    _ □ ×
File  Edit  Search  Compile  Display  Options  Window  Help  _ 8 ×
;{CR10X}
;
*Table 1 Program
  01: 0.0000     Execution Interval (seconds)

*Table 2 Program
  02: 0.0000     Execution Interval (seconds)

*Table 3 Subroutines

End Program

1:1
```
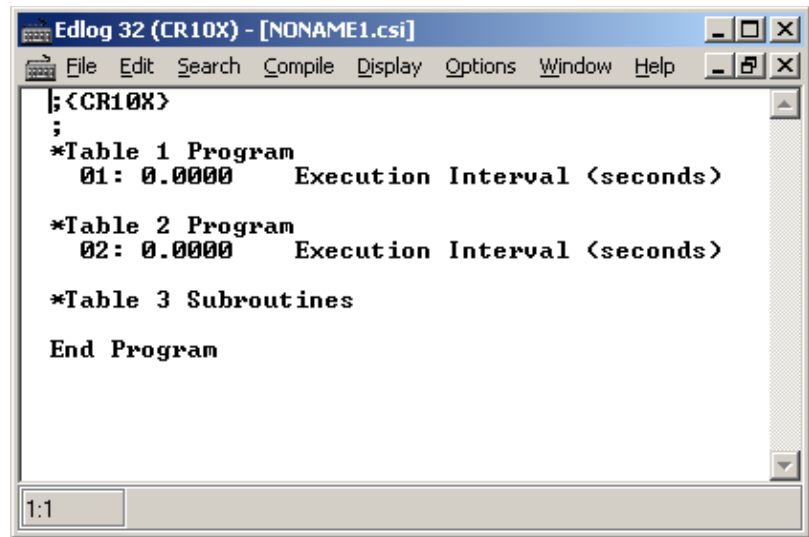
The first line of text identifies the type of datalogger program to be written. This is followed by a comment line and the Program Table Headers and Execution Interval fields. The Program Table Headers and Execution Interval fields are protected text that cannot be deleted or commented out. When the cursor is moved to the Execution Interval line, the field for the execution interval is highlighted. A numeric value must be entered or the instructions in the table will never be executed.

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

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

When the program is complete, select File | Save from the Edlog menu. A standard file dialog box will appear in which to type a file name. Edlog supports long filenames for the datalogger programs. Use descriptive names to help document the program's function. After saving the file, you will be prompted to compile the program. When a program is precompiled the code will be checked for errors. After precompiling, the datalogger program can be sent to the datalogger using the Connect screen (Section 6.4).

8.1.1.1 Program Structure

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

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

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

- Measure Sensors – In this first section put all the instructions that get data from the sensors attached to the datalogger. The sensor readings are stored in input locations, ready for the next section.

- Process Measurements – In this section do all the calculations and data processing to prepare the data for output.

- Control – Do any control of external hardware or devices.

- Output Data – Check to see if it is time, or a condition exists, to trigger output data to be saved in final storage.

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

8.1.1.2 Edlog File Types

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

- *.CSI - The CSI file is what the user actually edits. When an Edlog program is saved, Edlog automatically adds a CSI extension to the program's name. Existing CSI files can be edited by selecting File | Open. Long file names are supported. Although CSI files are ASCII files they require a particular format, so editing the files with some other text editor can corrupt the Edlog programs so that they no longer compile.

- *.DLD - When Edlog compiles a program (*.CSI), a DLD file is created. This is the file that is downloaded to the datalogger, (and also the type of file that is retrieved from the datalogger). If an existing program file is edited and compiled, the old DLD file will be overwritten by the new file. A CSI file can be created from a DLD by choosing File | Document DLD File.

- *.PTI - PTI files show the execution times for each instruction, block (e.g., subroutine), and program table, as well as the estimated number of final storage locations used per day. The execution times are estimates. PTI files do not account for If commands, Else commands, or repetitions of loops. For some instructions, the execution times are listed as 0. This

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

- *.FSL – Final Storage Label files contain the final storage labels for the data values in the output data records. The FSL file is used by LoggerNet to allow the Connect screen and RTDM to select final storage data for real-time displays. This file is also 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.

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

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

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

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

8.1.1.3 Inserting Instructions into the Program

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

- Select File | Insert Instruction from the Edlog menu.

- Press Shift+Insert on the keyboard.

- Right click on a blank line and select Insert Instruction from the menu.

- Type the instruction number onto a blank line and press enter.

The first three options will invoke the Insert Instruction dialog box.
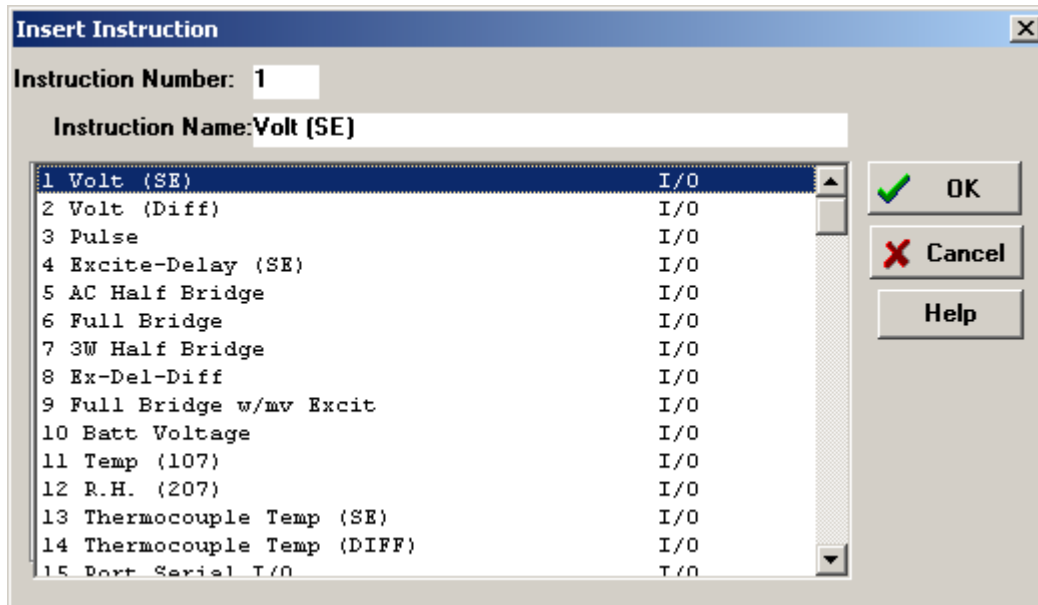
To insert an instruction into the program, select and then choose OK, or double click the entry in the list. If you need more information on an instruction, select the instruction and click on the Help button.

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

8.1.1.4 Entering Parameters for the Instructions

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

- Select the parameter with your mouse and press the right button. This brings up a dialog box from which to select a value or a pop up description of what should be entered.

- With your cursor anywhere within the instruction, press <F1>. This opens the help system to a detailed description of the instruction and parameters.

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

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

Data Entry Warnings

Edlog has a Data Entry Warning function that is accessed from the Options | Editor menu item. By default, the Data Entry Warning is enabled. When the

Data Entry Warning is active, a warning is displayed immediately after an invalid input has been entered for an instruction's parameter. The warning lists the valid inputs. A valid input must be entered before advancing to the next parameter.

8.1.1.5 Program Comments

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

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

- Select a block of text, press the right mouse button, and select "comment" or "uncomment" from the right button menu.

- Select File | Comment or File | Uncomment from the Edlog toolbar.

- Select a block of text and press <ctrl>+n to comment text (or <shift><ctrl>+n to uncomment text).

- Press <End> to automatically insert a semi-colon to the right of the protected text of an instruction or parameter, and type the desired comment.

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

8.1.1.6 Expressions

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

TempF=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 8.2-1 below).

- Numbers and labels that appear immediately after a function must be enclosed in parentheses.

- Several operators and/or functions can be used in one expression. Operations and functions that are enclosed in parentheses are calculated first; the inner most parentheses are evaluated first.

- To continue an expression to the next line, end the first line with an underscore (_).

TABLE 8.1-1. Operators and Functions	
Operators	
*	Multiply
/	Divide
+	add
-	subtract
^	Raise to the power of; enclose negative values in parentheses
@	modulo divide
E	scientific notation; 6e-1=0.6
Functions	
COS	cosine; angle in degrees
SIN	sine; angle in degrees
TAN	Tangent; angle in degrees
COTAN	cotangent; angle in degrees
ARCTAN	arctangent; angle in degrees
ARCSIN	arcsine; angle in degrees
ARCCOS	arccosine; angle in degrees
ARCCOT	arccotangent; angle in degrees
SQRT	square root
LN	natural logarithm
EXP	exponent of e; EXP(2) = e^2
RCP	reciprocal; RCP(4) = 1/4 = 0.25
ABS	absolute value
FRAC	takes the fraction portion; FRAC(2.78)=.78
INT	takes the integer portion; INT(2.78)=2

Below are examples of valid expressions:

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

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

```
Execution Interval = 10 sec

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

1:  Temperature (107) (P11)
  01:         1          REPS
  02:         2          Channel
  03:         1          Excitation Channel
  04:         2          Loc [TempC]
  05:         1          Mult
  06:         0          Offset



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

TempF = (TempC*1.8)+32
```

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

```
1:  Temperature, 107 (P11)
  01:         1
  02:         2
  03:         1
  04:         2
  05:         1.0
  06:         0.0

2:  Z=X (P31)
  01:         2
  02:         5

3:  Z=F (P30)
  01:         1.8
  02:         0
  03:         3

4:  Z=X*Y (P36)
  01:         3
  02:         5
  03:         5

5:  Z=F (P30)
  01:         32
  02:         0
  03:         3

6:  Z=X+Y (P33)
  01:         3
  02:         5
  03:         6
```

Errors That Can Occur With Expressions

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

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

Variable Name Expected

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

For Example:

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

$$5=el*(Vee+en)$$
$$(lambda) = COS(theta)$$
$$10-(zee/2)=bee$$

These are correct ways of entering the above expressions:

$$five=el*(Vee+en)$$
$$lambda = COS(theta)$$
$$bee=10-(zee/2)$$

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 (see section 2.3.3.2), this error message occurs.

New Line Expected

Indicates one of the following situations:

(1) An expression contains more than one equal sign.

(2) There is no operator between two sets of parentheses.

For Example:

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

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

These are correct ways of entering the above expressions:

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

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

For Example:

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

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

These are correct ways of entering the above expressions:

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

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

For Example:

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

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

These are correct ways of entering the above expressions:

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

Equal Sign Expected

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

For Example:

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

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

These are correct ways of entering the above expressions:

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

8.1.2 Editing an Existing Program

To edit an existing file, load it into Edlog by choosing File | Open from the Edlog menu. Changes can be made as desired and then the file can be saved and compiled under the same (File | Save) or a new name (File | Save As). Table 8.2-2 provides a list of keystrokes that can be used in editing programs and moving around in Edlog.

TABLE 8.1-2. Editor Keystrokes	
PgUp	Page Up
PgDn	Page Down
Up Arrow	Move Up One Line
Down Arrow	Move Down One Line
Right Arrow	Move One Character Right
Left Arrow	Move One Character Left
<Ctrl> Home	Move Cursor to Beginning of File
<Ctrl> End	Move Cursor to End of File
<Ctrl> PgUp	Move Cursor to Top of Screen
<Ctrl> PgDn	Move Cursor to Bottom of Screen
<Enter>	Move to Next Field or Create New Line
<Shift> Ins	Select an Instruction from a Dialog Box
<Ctrl> Right Arrow	Move Instruction 1 Tab Right (Cursor on Parameter)
<Ctrl> Left Arrow	Move Instruction 1 Tab left (Cursor on Parameter) or Move from Input Location label to Input Location number.
<Ctrl> n	Comment out a Line or Instruction
<Shift> <ctrl> n	Uncomment a Line or Instruction
<End>	Move to end of line, Add a comment if on an Instruction
<Ctrl>C	Copy selected text
<Ctrl> X	Cut selected text
<Ctrl>V	Paste clipboard
Del	Delete character to right or selected text
<Shift> Del	Delete the Instruction or Line Under the Cursor
<Esc>	Close Dialog Box

8.1.2.1 Editing Comments, Instructions, and Expressions

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

8.1.2.2 Cut, Copy, Paste, and Clipboard Options

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

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

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

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

8.1.3 Library Files

Library files can be created to store portions of programs, which can then be inserted into a different program. Library files are useful if you want to write different programs for the same sensor set, or it 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 filename. To insert a library file in a program, move the cursor to the desired insertion point and select Edit | Insert Library File.

NOTE

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

8.1.4 Documenting a DLD File

As noted in Section 8.1.1.2, the CSI file is the file created by Edlog that is used to generate the DLD code and other files. If for some reason your CSI file is missing, you can import the DLD file into Edlog to create another editable CSI file. From the Edlog menu select File | Document DLD. Select the DLD file to be imported and remember to save the file to create a new CSI file.

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

8.1.5 Display Options

8.1.5.1 Graphical Toolbar

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

Toolbar
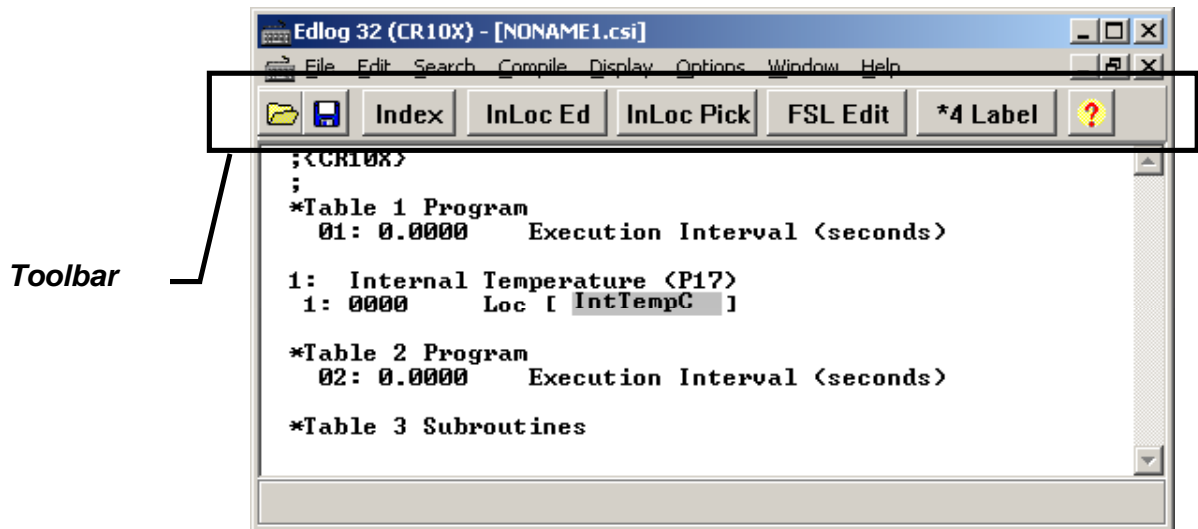
📂	Open a new file.
💾	Save the current file to disk and optionally precompile the program.
Index	Index the parameter that is selected (for information on indexing, refer to your datalogger operator's manual).
InLoc Ed	Invoke the input location editor (for a discussion on input locations, see Section 8.3).
InLoc Pick	Display the input location list; allows the user to select and insert an input location automatically into a parameter.
FSL Edit	Invoke the final storage label editor (for more information on editing final storage labels see Section 8.4).
*4 Label	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.
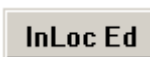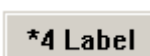
8.1.5.2 Renumbering the Instructions

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

8.1.5.3 Compress VIEW

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

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

8.1.5.4 Indention

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

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

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

8.2 Input Locations

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

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

8.2.1 Entering Input Locations

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

You may prefer to enter all input locations into the Edlog program before writing the program. This makes all the labels available from the input location pick list, and can help reduce programming errors because of typos. See section 8.2.3.

Labels can have up to 9 characters for array-based dataloggers and 14 characters for table-based dataloggers. The first character must be a letter. The allowed characters are letters, numbers, and the underscore character (_). The following labels are reserved for expressions and should not be entered by the user: CSI_R, CSI_2, CSI_3,... CSI_95.

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

8.2.2 Repetitions

Many input/output and output processing instructions have a repetitions parameter. Repetitions (REPS) allow one programming instruction to measure several identical sensors or to process data from several Input Locations. When REPS are greater than 1, the Input Locations are assigned consecutive numbers (e.g., with REPS of 2 and LOC of 5, the Input Locations are 5 and 6). Each rep label is the initial label with a "_" and the next consecutive number (i.e., with 3 REPS and a label of "data" the labels for each REP are: data_1, data_2, and data_3).

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

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

```
10:  Temperature (107) (P11)
  01:          1          REPS
  02:          2          Channel
  03:          1          Excitation Channel
  04:          1          Loc [TempC]
  05:          1          Mult
  06:          0          Offset

11:  Battery, Volt (P10)
  01:          2          Loc [BatteryV]

12:  If time is (P92)
  01:          0          minutes into interval
  02:          60         minute interval
  03:          10         Set high Flag 0(output)

13:  Sample (P70)
  01:          2          Reps
  02:          1          Loc [TempC]
```

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

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

8.2.3 Input Location Editor

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

Location number

Label

Program Use:
R Read
W Written to
M Manually marked

Input Location Editor   □ ▣ ✕

Edit Help

Addr	Name	Flags	# Reads	# Writes	Blocks
1	[RefTemp]	RW-	1	1	----- ----- ---
2	[TC_1]	-W-	0	1	Start ----- ---
3	[TC_2]	-W-	0	1	----- Member ---
4	[TC_3]	-W-	0	1	----- Member ---
5	[TC_4]	W	0	1	Member
6	[TC_5]	-W-	0	1	----- ------ End
7	[_____]	---	0	0	----- ------ ---
8	[_____]	---	0	0	----- ------ ---
9	[_____]	---	0	0	----- ------ ---
10	[_____]	---	0	0	----- ------ ---
11	[_____]	---	0	0	----- ------ ---
12	[_____]	---	0	0	----- ------ ---

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 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.

8.2.4 Input Location Anomalies

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

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

- Instruction 15, Serial I/O with Control Port

- Instruction 23, Burst Measurement

- Instruction 49, Spatial Maximum

- Instruction 50, Spatial Minimum

- Instruction 54, Block Move

- Instruction 75, Histogram

- Instruction 80, Store Area

- Instruction 81, Rainflow Histogram

- Instruction 100, TDR Measurement

- Instruction 101, SDM-INT8

- Instruction 105, SDI-12 Recorder

- Instruction 106, SDI-12 Sensor

- Instruction 113, SDM-SIO4

- Instruction 118, SDM CAN

- Instruction 119, TDR100

- Instruction 120, Data Transfer to TGT

- Instruction 127, HDR Goes Status and Diagnostics

- Instruction 128, SHEF Data Transfer to TGT

- Instruction 189,

- Instructions P190-199 PakBus control

- Indexed input locations in a loop

See Edlog Help for each instruction to get a derailed description of input location usage. You can also refer to the datalogger user's manual for more information on these instructions.
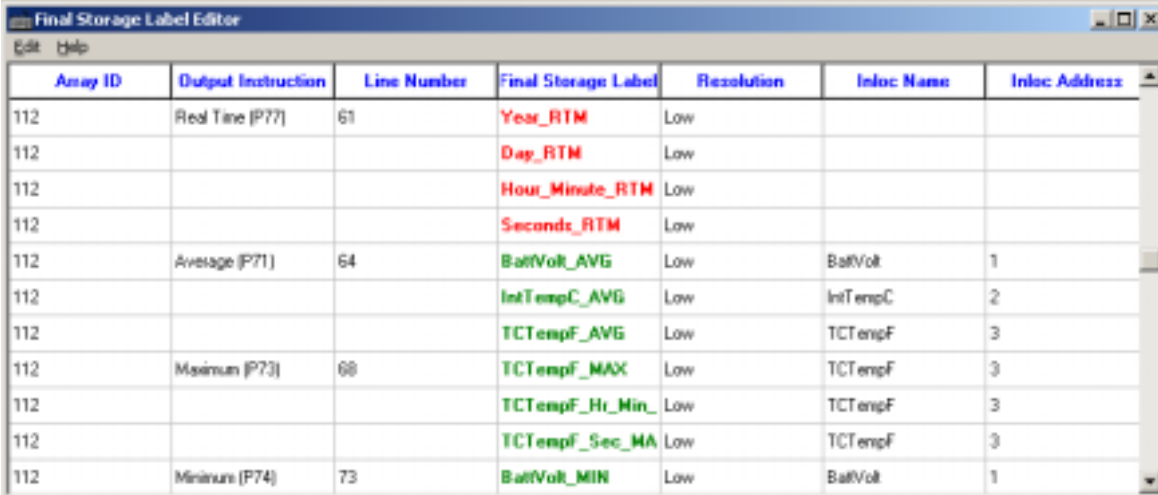
When these instructions are used in a program, the Toggle Manual feature can be used to manually mark Input Locations for use by the program.

8.3 Final Storage Labels

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

For array-based dataloggers the final storage labels are stored in an *.FSL file when the program is compiled. For table-based dataloggers the final storage labels are included as part of the datalogger program in the *.DLD file. LoggerNet gets the final storage labels as part of the table definitions from the datalogger. Split, Graph and Numeric Display as well as RTDM, and View use the final storage labels.

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

Final Storage Label Editor

Edit   Help

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

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

− Array ID or Table Name identifies the set of output data instructions the data is associated with. For array-based dataloggers the array ID is at the beginning of each output record. In table-based dataloggers, the table name shows the name of the table where the data values will be stored.

− Output Instruction lists the output instruction that was used to output the data value.

− Line Number is the line number in the Edlog program for the output instruction.

− Final Storage Label is the label that is associated with this final storage value. Red labels are associated with automatically created data entries such as time stamps and record numbers. The red labels cannot be changed with the Final Storage Label Editor. The green labels are associated with user programmed sensor data. To change the label, click in the box and type in the new label.

− Resolution shows whether the data will be stored in low or high resolution. (High resolution stores data as a 4-byte floating point number, Low resolution uses a 2-byte number)

− Inloc Name is the label of the input location that the final storage data is based on.

− Inloc Address is the numeric label for the input location used for the final storage data value.

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

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

Section 9. Datalogger Program Creation with CRBasic Editor

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

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

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

9.1 Overview

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

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

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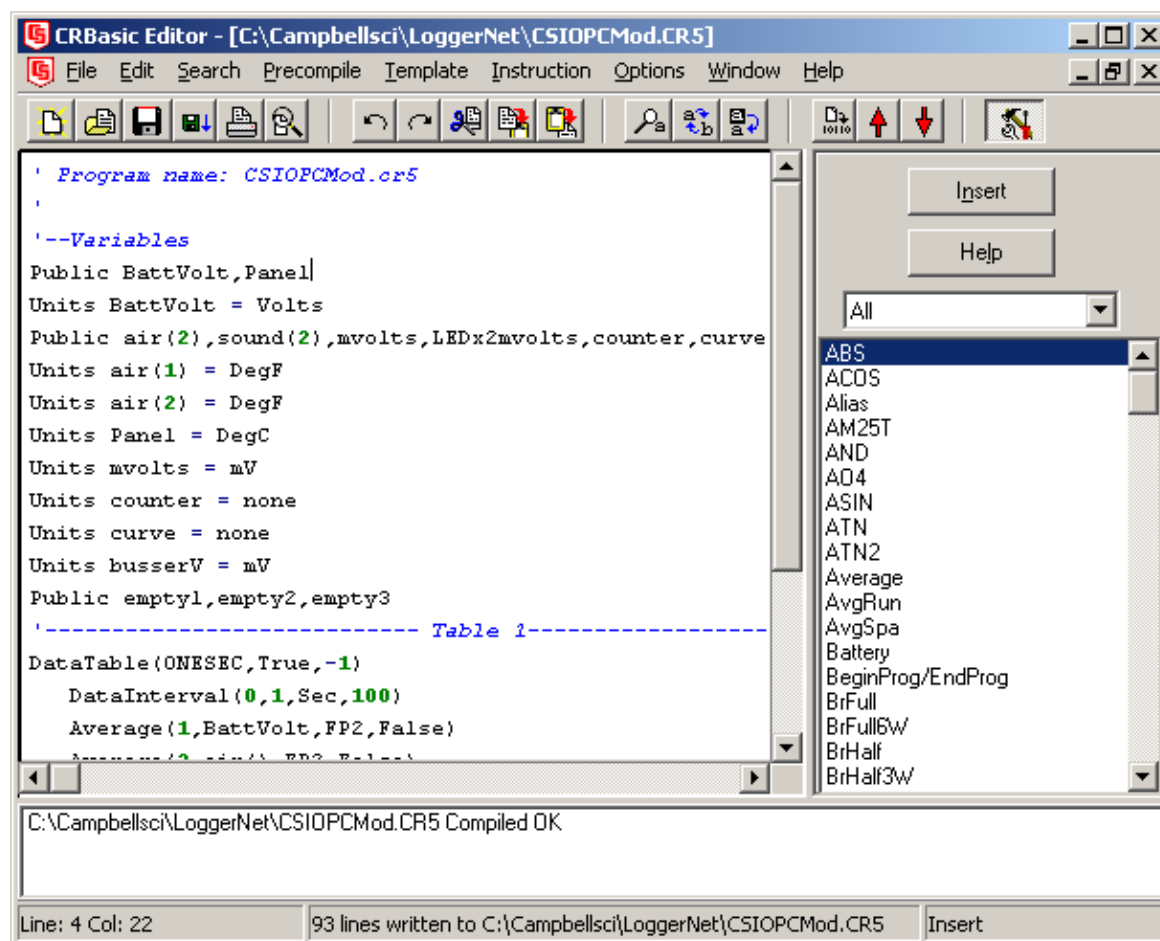
9.1.1 Inserting Instructions

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

You can filter the list of instructions available in the Instruction Panel by clicking the drop down arrow to the right of the text box above the list. This will allow you to display only instructions of a specific type such as Measurement/Control or Declarations. This provides a smaller list to select from and makes it easier to find the instruction you want. Switch back to All to see all of the instructions available. You can create custom instruction filter lists as described in Section 9.1.7.

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

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

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

9.1.2 Parameter Dialog Box

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

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

Short Cuts for Editing the Parameters

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

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

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

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

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

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

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

Changing Default Parameters Values for an Instruction

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

9.1.3 Right Click Functionality

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

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

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

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

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

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

- **Comment/Uncomment Block**: Only one of these options will be available, depending upon the status of the highlighted text. If the text has been marked as a Comment, you can choose to uncomment it. If the text is not commented, you can chose to make it into a comment. Commented text has a single quote (') at the beginning of the line. Comments are ignored by the datalogger's compiler.

- **Decrease/Increase Indent**: You can increase or decrease the indention of the selected text. The spacing is increased or decreased by one.

- **Cut/Copy/Paste/Delete**: Standard editing functions can be accessed through this menu.

9.1.4 Toolbar

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

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

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

Save – Saves any changes to the currently opened program. If this a new program and has not been saved yet, a Save As dialog will prompt you for the filename and location to save the file.

Save and Download – Saves any changes to the currently opened program and opens PC9000 to send the file to the datalogger. If the CRBasic editor can't find PC9000 you will be prompted to browse for it.

Print – Prints the program listing of the currently open program to the printer.

Print Preview – Opens a Print Preview screen that will show what the program will look like when printed. You can check and set the margins and printer options.

Undo – Each time the Undo button is clicked it will step back through the last changes made to the program.

Redo – Cancels the undo and steps forward restoring the changes.

Cut – Removes the selected part of the program and puts it on the clipboard to be pasted elsewhere.

Copy – Places a copy of the selected part of the program on the clipboard to be pasted elsewhere.

Paste – Inserts a copy of the contents of the clipboard into the program at the cursor location.

Find – Brings up a Find dialog to specify a text string to search for in the program listing. Click on the Find Next button or press F3 to go to successive occurrences of the text.

Replace – Brings up a Find and Replace dialog that allows you to specify a text string to search for and a text string to replace it with. You can replace all occurrences of the text or check them one at a time to make sure they should be replaced.

Find Next – Finds the next occurrence of the text string specified in the Find dialog.

Compile – Starts the compiler to check the current program for errors and consistency. Compile results and errors will be displayed in the message area at the bottom of the screen.

Previous Error – Moves the cursor to the part of the program where the previous error was identified.

Next Error – Moves the cursor to the part of the program where the next error was identified.

Instruction Panel – Controls whether the Instruction Panel is displayed. Hiding the Instruction Panel can allow more room in the window to view the program listing.

9.1.5 Compile

Compile is a function provided by the CRBasic Editor to help the programmer catch problems with the datalogger program. Compile is available from the toolbar and the Compile menu.

When the Compile function is invoked, the CRBasic Editor checks the program for syntax errors and other inconsistencies. The results of the check will be displayed in a message window at the bottom of the main window. If an error can be traced to a specific line in the program, the line number will be listed before the error. You can double click an error preceded by a line number and that line will be highlighted in the program editing window. To move the highlight to the next error in the program, press the Next Error button or choose Next Error from the Compile menu. To move the highlight to the previous error in the program, press the Previous Error button or choose Previous Error from the Compile menu.

The error window can be closed by selecting the Close Message Window menu item.

9.1.6 Templates

The use of templates can be a powerful way to quickly create a set of similar datalogger programs. All or part of a program can be saved so that it can be used when creating new programs. These files are called templates. The Template menu provides access to create and use templates.

Save File as Template - Saves the comments and instructions in the active file as a template. To save part of a program as a template, copy the selected part to a new program file and then Save File as Template.

Save as Default Template - Saves the comments and instructions in the active file as a template that will be used each time File | New is selected for that type of datalogger.

Save as Initial Template - Saves the comments and instructions in the active file as a template that will be used each time the CRBasic Editor is started. If Display Last Window Used option is selected, the initial template will not be used.

Delete - When selected, a list of all dataloggers is displayed. Select a datalogger to invoke a dialog box containing a list of saved templates. A template can then be highlighted and deleted from disk.

(Datalogger Types) - When a datalogger type is selected, a list of all templates is displayed.

NOTE

Template files are associated with a specific datalogger type. Templates for a CR5000 cannot be used for CR9000 programming and vice versa. Each datalogger has its own set of instructions that may be different than the other.

9.1.7 CRBasic Editor Options

This menu item allows you to specify the files used in the CRBasic Editor and customize its look and syntax highlighting.
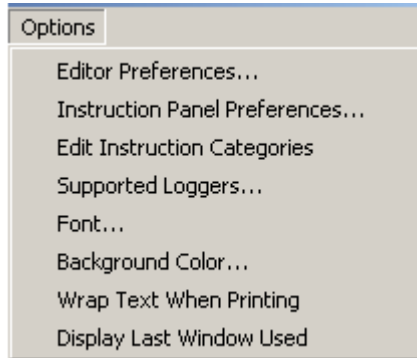
Editor Preferences sets up the appearance options for the text instructions and the behavior of popup hints.

The **Editor** tab allows the user to toggle the popup hints for instructions on or off, set the amount of time the cursor must hover over the instruction before the popup hint occurs, and the background color of the popup hint. This is also used to set the number of spaces used when the CRBasic Editor automatically indents an instruction.
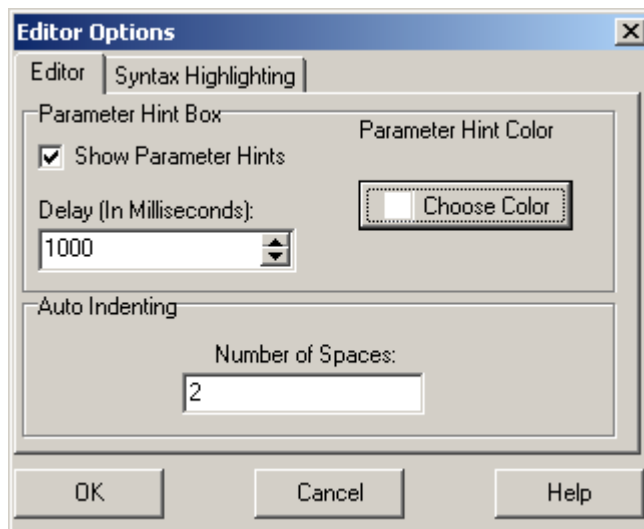
The **Syntax Highlighting** tab sets up the appearance of different text elements in the program using different font styles and colors. Comments, Instruction Names, and other text elements can each have a different appearance. You can customize the appearance of the text by giving normal text, keywords, comments, operators, and numbers each a different font style and color to make the program easier to read and edit. Text colors and styles can be disabled by clearing the Syntax Highlighting check box.
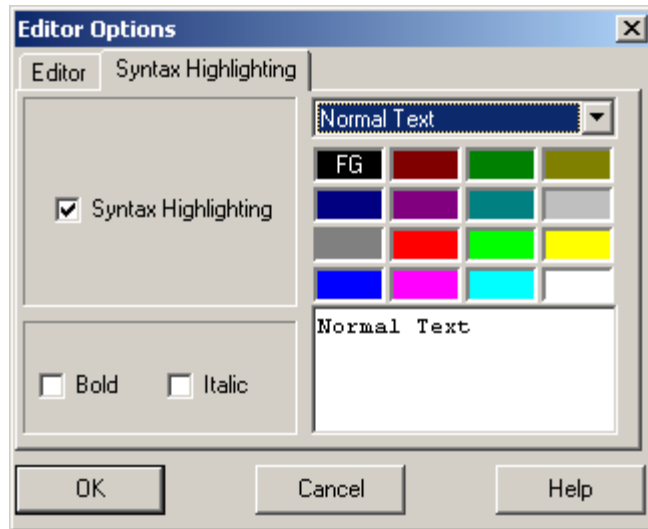
Instruction Panel Preferences allows the user to define how the insert function will behave in the program.
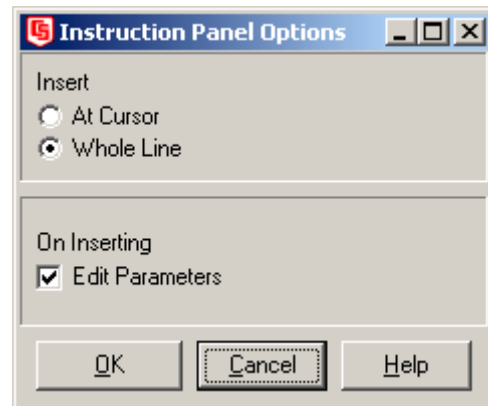
− **At Cursor/Whole Line**: When At Cursor is selected, inserted instructions will appear at the cursor location. When Whole Line is selected, a new line will be created for the instruction. (Note that the At Cursor option may cause you to unintentionally insert an instruction in the middle of another instruction.)

− **On Inserting**: When Edit Parameters is selected, the Edit Parameters dialog box will appear when an instruction is added to the program. When it is not selected, the instruction will be inserted directly, using the default parameters.

Edit Instruction Categories allows the user to create custom categories. (Note: The default categories cannot be edited or deleted.)
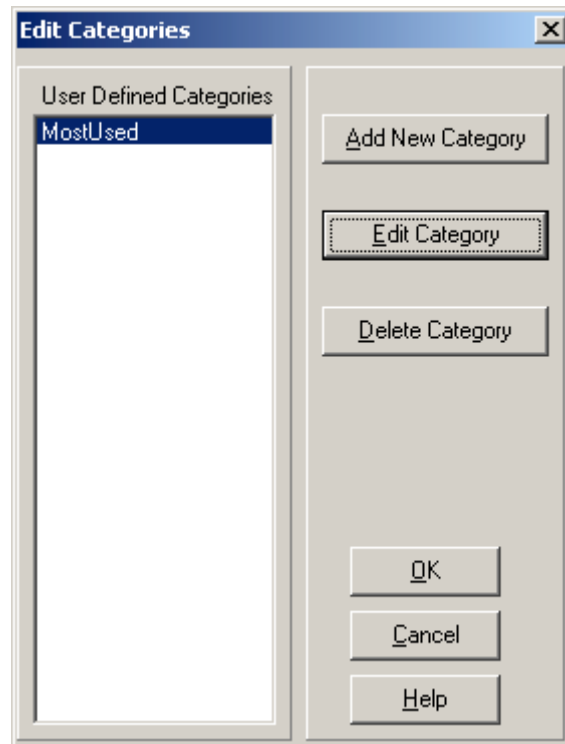
In the Edit Categories dialog shown above, a new category MostUsed has been added. Clicking on the Edit Category button will bring up the Select Instructions dialog that is shown below. This will allow the user to display a filtered instruction list containing only those instructions most often used. The instructions to be displayed are selected in the Select Instruction screen shown below. Select the check box next to the instructions you want in the user category from each list of instruction types. Press OK to save the list.
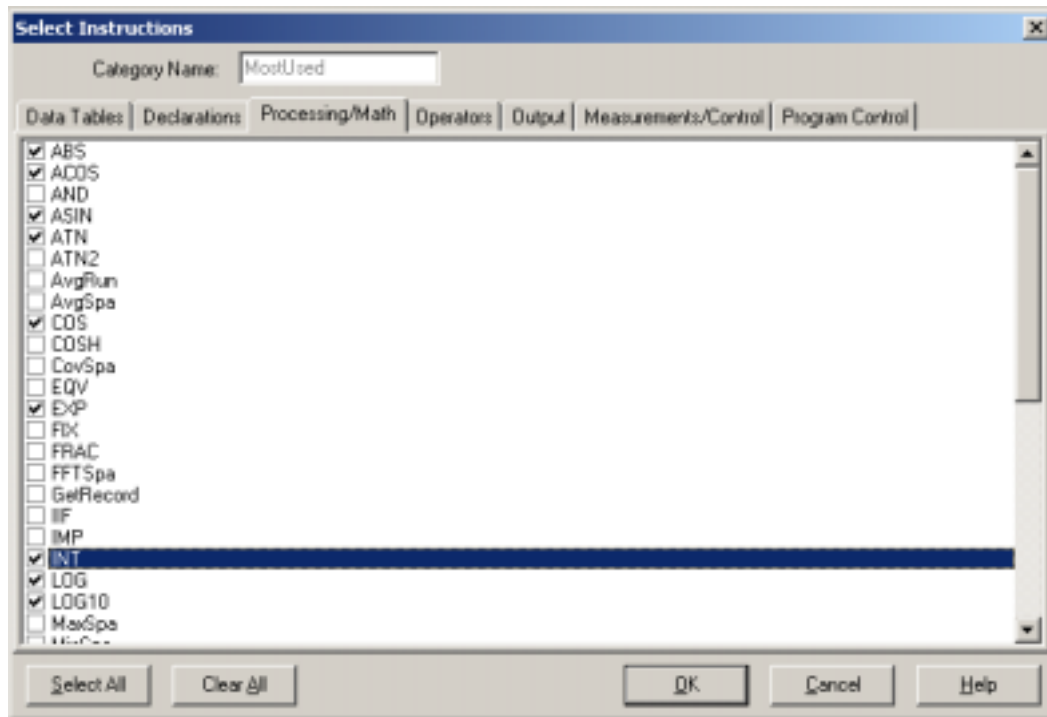
Supported Loggers allows you to set up additional dataloggers to be programmed using the CRBasic Editor. In the dialog window shown below the editor is set up for CR5000 and CR9000 dataloggers. When new dataloggers are added the language, help and compiler files can be linked to the CRBasic editor in this window.
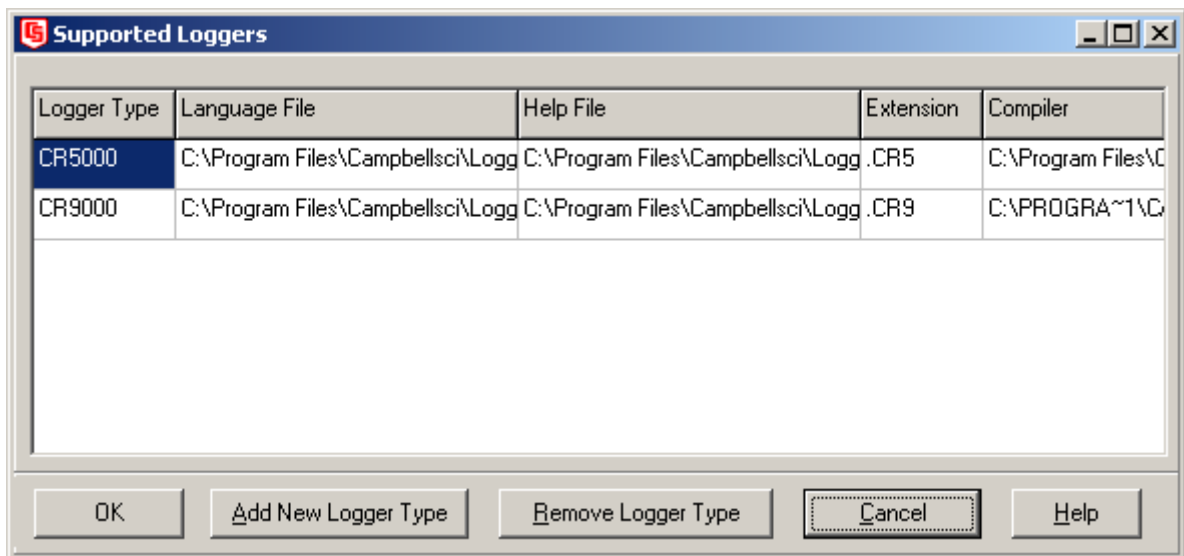
Font brings up a font selection dialog to select the font typeface and size for the text in the CRBasic Editor. Font style and color are set under Editor Preferences.

Background Color brings up the color selection dialog to set the color of the CRBasic program window.

Wrap Text When Printing - When this option is selected, long lines that extend past the right margin will be wrapped to the next line. This option affects printing, as well as the Print Preview mode. A check mark will appear next to the option in the menu when it is selected.

Display Last Window Used - When this option is enabled, the program that was active in the CRBasic Editor when it was last closed will be visible when the Editor is reopened. If this option is disabled, no program will be loaded when the Editor is opened. A check mark will appear next to the option in the menu when it is selected.

9.1.8 Available Help Information

Pressing the Help button at the bottom right of the Parameter dialog box will bring up detailed help topic for the instruction being edited. Pressing F1 when your cursor is within a parameter field will bring up help only on that parameter. Some fields also have text in the Comments column, which provides a short description of the option that has been selected for the parameter.

9.2 CRBasic Programming

CRBasic is a programming language that has some similarities to a structured BASIC. There are special instructions for making measurements and for creating tables of output data. The results of all measurements are assigned variables (given names). Mathematical operations are written out much as they would be algebraically. This section describes a program, its syntax, structure, and sequence.

9.2.1 Programming Sequence

The structure of a datalogger program requires that variables and subroutines be defined before they can be used. The best way to do this is to put all the variable declarations and output table definitions at the beginning, followed by the subroutines, and then the program. Below is the typical layout of a program. Note that the online help has example code for each instruction to demonstrate the use of the instruction in a program.

Declarations	*Make a list of what to measure and calculate.*
Declare constants	*Within this list, include the fixed constants used,*
Declare Public variables	*Indicate the values that the user is able to view while the program is running,*
Dimension variables	*the number of each measurement that will be made,*
Define Aliases	*and specific names for any of the measurements.*
Define data tables	*Describe, in detail, tables of data that will be saved from the experiment.*
Process/store trigger	*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?*
Table size	*Set the size of the table in RAM.*
Other on-line storage devices	*Should the data also be sent to the external storage?*
Processing of Data	*What data are to be output (current value, average, maximum, minimum, etc.).*
Define Subroutines	*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.*
Program	*The program section defines the action of datalogging.*
Set scan interval	*The scan sets the interval for a series of measurements.*
Measurements	*Enter the measurements to make.*
Processing	*Enter any additional processing with the measurements.*
Call Data Table(s)	*The Data Table must be called to process output data.*
Initiate controls	*Check measurements and Initiate controls if necessary.*
NextScan	*Loop back (and wait if necessary) for the next scan.*
End Program	

9.2.2 Program Declarations

Variables must be declared before they can be used in the program. Variables declared as Public can be monitored by LoggerNet using the Numeric Monitor screen or the Graphical Display. Variables declared using Dim cannot be displayed. Variables assigned to a fixed value are used as constants.

For example, in a CRBasic program there may be more than one temperature (or other) measurements. Rather than have many different names, a *variable array* with one name and many elements may be used. A thermocouple temperature might be called TCTemp. With an array of 20 elements the names of the individual temperatures are TCTemp(1), TCTemp(2), TCTemp(3), ... TCTemp(20). The array notation allows compact code to perform operations on all the variables. For example, to convert ten temperatures in a variable array from C to F:

```
For I=1 to 10
        TCTemp(I)=TCTemp(I)*1.8+32
Next I
```

Aliases can also be created that will allow an element of an array or another data result to be referred to by a different name.

9.2.3 Mathematical Expressions

Mathematical expressions can be entered algebraically into program code to perform processing on measurements, to be used for logical evaluation, or to be used in place of some parameters.

As an example of **Measurement Processing,** to convert a thermocouple measurement from degrees Celsius to degrees Fahrenheit, you could use the following expression:

TCTempF=TCTemp(1)*1.8+32

Logical Evaluation expressions could be used to determine the flow of a program:

```
If TCTemp(1) > 100 Then
Call Subroutine1
Else
'enter code for main program
End If
```

Many parameters will allow the entry of expressions. In the following example, the DataTable will be triggered, and therefore data stored, if TCTemp(1)>100.

DataTable(TempTable, TCTemp(1)>100, 5000)

9.2.4 Measurement and Output Processing Instructions

Measurement instructions are procedures that set up the measurement hardware to make a measurement and place the results in a variable or a variable array. Output processing instructions are procedures that store the results of measurements or calculated values. Output processing includes averaging, saving maximum or minimum, standard deviation, FFT, etc.

The instructions for making measurements and outputting data are not found in a standard basic language. The instructions Campbell Scientific has created for these operations are in the form of procedures. The procedure has a keyword name and a series of parameters that contain the information needed to complete the procedure. For example, the instruction for measuring the temperature of the CR5000 input panel is:

PanelTemp *(Dest, Integ)*

PanelTemp is the keyword name of the instruction. The two parameters associated with PanelTemp are: *Dest*ination, the name of the variable in which to put the temperature; and *Integ*ration, the length of time to integrate the measurement. To place the panel temperature in the variable RefTemp (using a 250 microsecond measurement integration time) the code is:

PanelTemp(RefTemp, 250)

The use of these instructions should become clearer as we go through an introductory example in Section 9.3.

9.2.5 Inserting Comments Into Program

It is often useful to provide comments in your datalogger program so that when you review the program at a later date, you will know what each section of code does. Comments can be inserted into the program by preceding the text with a single quote. When the program is compiled, the datalogger compiler will ignore any text that is preceded by a single quote. A comment can be placed at the beginning of a line or it can be placed after program code. If Syntax Highlighting is enabled (Options | Editor Preferences | Syntax Highlighting), commented text will appear formatted differently than other lines of code.

```
'CR5000

'The following program is used to measure
'4 thermocouples


'VARIABLE DECLARATION
Dim TCTemp(4)                          'Dimension TC measurement variable
Alias TCTemp(1)=EngineCoolantT         'Rename variables
Alias TCTemp(2)=BrakeFluidT
Alias TCTemp(3)=ManifoldT
Alias TCTemp(4)=CabinT
```

In the sample code above, the commented text will be ignored by the datalogger compiler.

9.3 Example Program

The following program will serve as a programming example in this section to illustrate the concepts and program structure.

```
Const RevDiff=1
Const Del=0
Const Integ=250                    Declare constants
Const Mult=1
Const Offset=0

Public RefTemp                     Declare public variables ,
Public TC(6)                       dimension array, and        Declarations
Units RefTemp=degC                 declare units.
Units TC=DegC

DataTable (Temp,1,2000)
 DataInterval (0,100,mSec,10)
 Average (1,RefTemp,FP2,0)         Define Data Table
 Average (6,TC(),FP2,0)
EndTable

BeginProg
 Scan (10,mSec,3,0)
  PanelTemp (RefTemp, 250)
  TCDiff (TC(),6,mV20C ,1,TypeT,RefTemp,RevDiff,Del,Integ,Mult,Offset)   Measure   Scan loop
  CallTable Temp                                                         Call Data Table
 NextScan
EndProg
```

9.3.1 Data Tables

Data storage follows a fixed structure in the datalogger in order to optimize the time and space required. Data are stored in tables such as:

TOA5	StnName	Temp						
TIMESTAMP	RECORD	RefTemp_Avg	TC_Avg(1)	TC_Avg(2)	TC_Avg(3)	TC_Avg(4)	TC_Avg(5)	TC_Avg(6)
TS	RN	degC	DegC	degC	degC	degC	degC	degC
		Avg	Avg	Avg	Avg	Avg	Avg	Avg
1995-02-16 15:15:04.61	278822	31.08	24.23	25.12	26.8	24.14	24.47	23.76
1995-02-16 15:15:04.62	278823	31.07	24.23	25.13	26.82	24.15	24.45	23.8
1995-02-16 15:15:04.63	278824	31.07	24.2	25.09	26.8	24.11	24.45	23.75
1995-02-16 15:15:04.64	278825	31.07	24.21	25.1	26.77	24.13	24.39	23.76

The user's program determines the values that are output and their sequence. The datalogger automatically assigns names to each field in the data table. In the above table, TIMESTAMP, RECORD, RefTemp_Avg, and TC_Avg(1) are fieldnames. The fieldnames are a combination of the variable name (or alias if one exists) and a three letter mnemonic for the processing instruction that

outputs the data. Alternatively, the FieldNames instruction can be used to override the default names.

The data table header may also have a row that lists units for the output values. The units must be declared for the datalogger to fill this row out (e.g., Units RefTemp = degC). The units are strictly for the user's documentation; the datalogger makes no checks on their accuracy.

The above table is the result of the data table description in the example program:

```
DataTable (Temp,1,2000)
        DataInterval(0,10,msec,10)
        Average(1,RefTemp,fp2,0)
        Average(6,TC(1),fp2,0)
EndTable
```

All data table descriptions begin with **DataTable** and end with **EndTable**. Within the description are instructions that tell what to output and that can modify the conditions under which output occurs.

DataTable(*Name, Trigger, Size*)
DataTable (Temp,1,2000)

The DataTable instruction has three parameters: a user specified name for the table, a trigger condition, and the size to make the table in CR5000 RAM. The trigger condition may be a variable, expression, or constant. The trigger is true if it is not equal to 0. Data are output if the trigger is true and there are no other conditions to be met. No output occurs if the trigger is false (=0). The example creates a table name Temp, outputs any time other conditions are met, and retains 2000 records in RAM.

DataInterval(*TintoInt, Interval, Units, Lapses*)
DataInterval(0,10,msec,10)

DataInterval is an instruction that modifies the conditions under which data are stored. The four parameters are the time into the interval, the interval on which data are stored, the units for time, and the number of lapses or gaps in the interval to track. The example outputs at 0 time into (on) the interval relative to real time, the interval is 10 milliseconds, and the table will keep track of 10 lapses. The DataInterval instruction reduces the memory required for the data table because the time of each record can be calculated from the interval and the time of the most recent record stored.

NOTE   Event driven tables should have a fixed size rather than allowing them to be allocated automatically. Event driven tables that are automatically allocated are assumed to have one record stored per second in calculating the length. Since the datalogger tries to make the tables fill up at the same time, these event driven tables will take up most of the memory leaving very little for the other, longer interval, automatically allocated data tables.

The output processing instructions included in a data table declaration determine the values output in the table. The table must be called by the program in order for the output processing to take place. That is, each time a new measurement is made, the data table is called. When the table is called, the output processing instructions within the table process the current inputs. If the trigger conditions for the data table are true, the processed values are output to the data table. In the example below, several averages are output.

Average(*Reps, Source, DataType, DisableVar*)
Average(1,RefTemp,fp2,0)
Average(6,TC(1),fp2,0)

Average is an output processing instruction that will output the average of a variable over the output interval. The parameters are repetitions (the number of elements in an array to calculate averages for), the Source variable or array to average, the data format to store the result in (Table 9.3-1), and a disable variable that allows excluding readings from the average if conditions are not met. A reading will not be included in the average if the disable variable is not equal to 0; the example has 0 entered for the disable variable so all readings are included in the average.

TABLE 9.3-1. Formats for Output Data				
Code	Data Format	Size	Range	Resolution
FP2	Campbell Scientific floating point	2 bytes	±7999	13 bits (about 4 digits)
IEEE4	IEEE four byte floating point	4 bytes	1.8 E -38 to 1.7 E 38	24 bits (about 7 digits)
LONG	4 byte Signed Integer	4 bytes	-2,147,483,648 to +2,147,483,647	1 bit (1)

9.3.2 The Scan -- Measurement Timing and Processing

Once the measurements and calculations have been listed and the output tables defined, the program itself may be relatively short. The executable program begins with BeginProg and ends with EndProg. The measurements, processing, and calls to output tables bracketed by the Scan and NextScan instructions determine the sequence and timing of the datalogging.

```
BeginProg
Scan(1,MSEC,3,0)
        PanelTemp(RefTemp, 250)
        TCDiff(TC(),6,mV50,4,1,TypeT,RefTemp,RevDiff,Del,Integ,Mult,Offset)
        CallTable Temp
NextScan
EndProg
```

The Scan instruction determines how frequently the measurements within the scan are made:

Scan*(Interval, Units, BufferOption, Count)*
Scan(1,MSEC,3,0)

The Scan instruction has four parameters. The Interval is the interval between scans. Units are the time units for the interval. The maximum scan interval is one minute. The BufferSize is the size (in the number of scans) of a buffer in RAM that holds the raw results of measurements. Using a buffer allows the processing in the scan to at times lag behind the measurements without affecting the measurement timing (see the scan instruction in the help for more details). *Count* is the number of scans to make before proceeding to the instruction following NextScan. A count of 0 means to continue looping forever (or until ExitScan). In the example the scan is 1 millisecond, three scans are buffered, and the measurements and output continue indefinitely.

9.4 Numerical Entries

In addition to entering regular base 10 numbers there are 3 additional ways to represent numbers in a program: scientific notation, binary, and hexadecimal (Table 9.4-1).

TABLE 9.4-1. Formats for Entering Numbers in CRBasic		
Format	Example	Value
Standard	6.832	6.832
Scientific notation	5.67E-8	5.67×10^{-8}
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.

9.5 Logical Expression Evaluation

9.5.1 What is True?

Several different words get used to describe a condition or the result of a test. The expression, X>5, is either **true** or **false**. However, when describing the state of a port or flag, **on** or **off** or **high** or **low** is more intuitive. In CRBasic there are a number of conditional tests or instruction parameters, the result of which may be described with one of the words in Table 9.5-1. The datalogger evaluates the test or parameter as a number; 0 is false, not equal to 0 is true.

TABLE 9.5-1. 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	$\neq 0$	0
Digital port	5 Volts	0 Volts

9.5.2 Expression Evaluation

Conditional tests require the datalogger to evaluate an expression and take one path if the expression is true and another if the expression is false. For example:

If X>=5 then Y=0

will set the variable Y to 0 if X is greater than or equal to 5.

The datalogger will also evaluate multiple expressions linked with **and** or **or**. For example:

If X>=5 and Z=2 then Y=0

will set Y=0 only if both X>=5 and Z=2 are true.

If X>=5 or Z=2 then Y=0

will set Y=0 if either X>=5 or Z=2 is true (see And and Or in the help). A condition can include multiple **and** and **or** links.

9.5.3 Numeric Results of Expression Evaluation

The datalogger's expression evaluator evaluates an expression and returns a number. A conditional statement uses the number to decide which way to branch. The conditional statement is false if the number is 0 and true if the number is not 0. For example:

If 6 then Y=0,

is always true, Y will be set to 0 any time the conditional statement is executed.

If 0 then Y=0

is always false, Y will never be set to 0 by this conditional statement.

The expression evaluator evaluates the expression, X>=5, and returns -1, if the expression is true, and 0, if the expression is false.

W=(X>Y)

will set W equal to -1 if X>Y or will set W equal to 0 if X<=Y.

The datalogger uses -1 rather than some other non-zero number because the **and** and **or** operators are the same for logical statements and binary bitwise comparisons. The number -1 is expressed in binary with all bits equal to 1, the number 0 has all bits equal to 0. When -1 is anded with any other number the result is the other number, ensuring that if the other number is non-zero (true), the result will be non-zero .

9.6 Flags

Any variable can be used as a flag as far as logical tests in CRBasic are concerned. If the value of the variable is non-zero the flag is high. If the value of the variable is 0 the flag is low. LoggerNet looks for the variable array with the name **Flag** when the option to display flag status is used in one of the real time screens.

9.7 Parameter Types

Instruction parameters allow different types of inputs. These types are listed below and specifically identified in the description of the parameter in the following sections or in CRBasic help.

> Constant
> Variable
> Variable or Array
> Constant, Variable, or Expression
> Constant, Variable, Array, or Expression
> Name
> Name or list of Names
> Variable, or Expression
> Variable, Array, or Expression

Table 9.7-1 lists the maximum length and allowed characters for the names for Variables, Arrays, Constants, etc.

TABLE 9.7-1. Rules for Names		
Name for	**Maximum Length (number of characters)**	**Allowed characters**
Variable or Array	16	Letters A-Z, upper or lower
Constant	16	case, underscore "_", and
Alias	16	numbers 0-9. The name must
Data Table Name	8	start with a letter. CRBasic is
Field name	16	not case sensitive.

9.7.1 Expressions in Parameters

Many parameters allow the entry of expressions. If an expression is a comparison, it will return -1 if the comparison is true and 0 if it is false (Section 9.5.3). An example of the use of this is in the DataTable instruction where the trigger condition can be entered as an expression. Suppose the variable TC(1) is a thermocouple temperature:

DataTable(Name, TrigVar, Size)
DataTable(Temp, TC(1)>100, 5000)

Entering the trigger as the expression, TC(1)>100, will cause the trigger to be true and data to be stored whenever the temperature TC(1) is greater than 100.

9.7.2 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()
```

9.8 Program Access to Data Tables

Data stored in a table can be accessed from within the program. The format used is:

Tablename.Fieldname(fieldname index,records back)

Where *Tablename* is the name of the table in which the desired value is stored. *Fieldname* is the name of the field in the table. The fieldname is always an array even if it consists of only one variable; the *fieldname index* must always be specified. *Records back* is the number of records back in the data table from the current time (1 is the most recent record stored, 2 is the record stored prior to the most recent). For example, the expression:

Tdiff=Temp.TC_Avg(1,1)–Temp.TC_Avg(1,101)

could be used in the example program (Section 4.3) to calculate the change in the 10 ms average temperature of the first thermocouple between the most recent average and the one that occurred a second (100 x 10 ms) earlier.

In addition to accessing the data actually output in a table, there are some pseudo fields related to the data table that can be retrieved:

Tablename.record(1,n) = the record number of the record output n records ago.

Tablename.output(1,1) = 1 if data were output to the table the last time the table was called, = 0 if data were not output.

Tablename.timestamp(m,n) = element m of the timestamp output n records ago where:

       timestamp(1,n) = microseconds since 1990
       timestamp(2,n) = microseconds into the current year
       timestamp(3,n) = microseconds into the current month
       timestamp(4,n) = microseconds into the current day
       timestamp(5,n) = microseconds into the current hour
       timestamp(6,n) = microseconds into the current minute
       timestamp(7,n) = microseconds into the current second

Tablename.eventend(1,1) is only valid for a data table using the DataEvent instruction, *Tablename*.eventend(1,1) = 1 if the last record of an event occurred the last time the table was called, = 0 if the data table did not store a record or if it is in the middle of an event.

NOTE | The values of Tablename.output(1,1) and Tablename.eventend (1,1) are only updated when the tables are called.

Section 10. Split

Split is a tool that works with output data files collected from Campbell Scientific dataloggers. It can be used to separate data collected on a particular interval from a data file containing data output at several different intervals. Split can read data files from table based and array based dataloggers.

Split can be used to create reports based on collected data by filtering data based on time or conditions. It can separate statistics, calculations, file reformatting, data quality checking (limit testing), and table generation with report and column headings. It can also handle the time synchronization and merging of up to eight data files

10.1 Functional Overview

Split is a tool to aid the use and analysis of data collected from Campbell Scientific dataloggers. Its name comes from its function of splitting specific data from a larger data file.

In addition to splitting out 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 single 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 (Section 10.3.1) must be formatted in Printable ASCII, Comma Separated ASCII, Field Formatted ASCII, Final Storage (Binary) Format, Table Oriented ASCII (TOACI1 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 between the file formats used by different dataloggers. For example a Table Oriented ASCII file can be converted to the same type of Comma Separated ASCII format used in array based datalogger data files. This is useful to convert table based data files to work with applications that were written to work with array based files.

Output files generated by Split can be Field Formatted (default), Comma Separated ASCII, or Printable ASCII.

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 Input File.

10.2 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 station was named BirchCreek when telecommunications were set up so the data are stored in the file BirchCreek_1.dat. 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 10.2-1)

99	Year_RT	Day_RTM	Hour_Mi	BattVol	BattVol	BattVol	Encl_RH	Encl_RH	LogrTem	LogrT
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 10.2-1. 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 10.2-2).
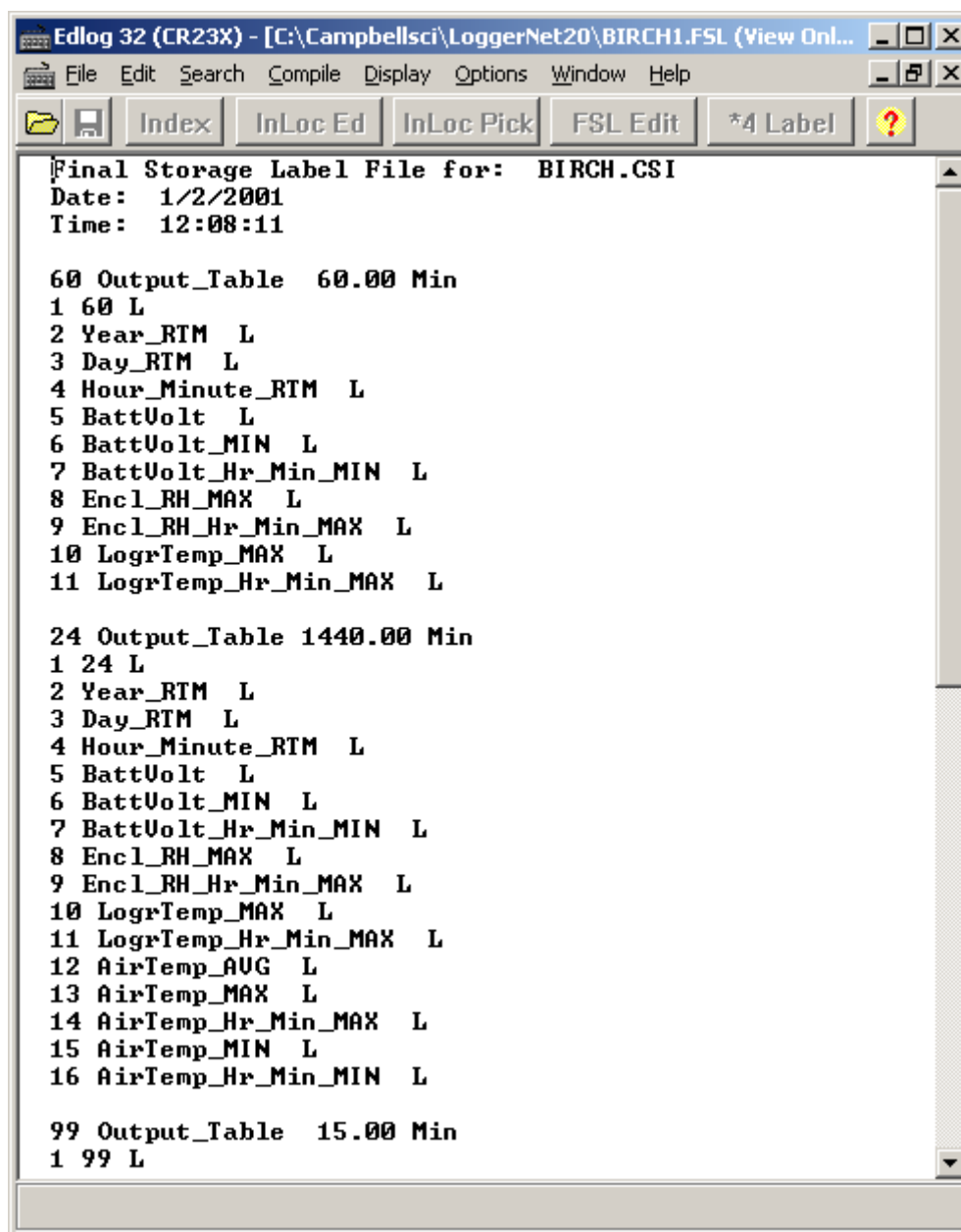
FIGURE 10.2-2. FSL file
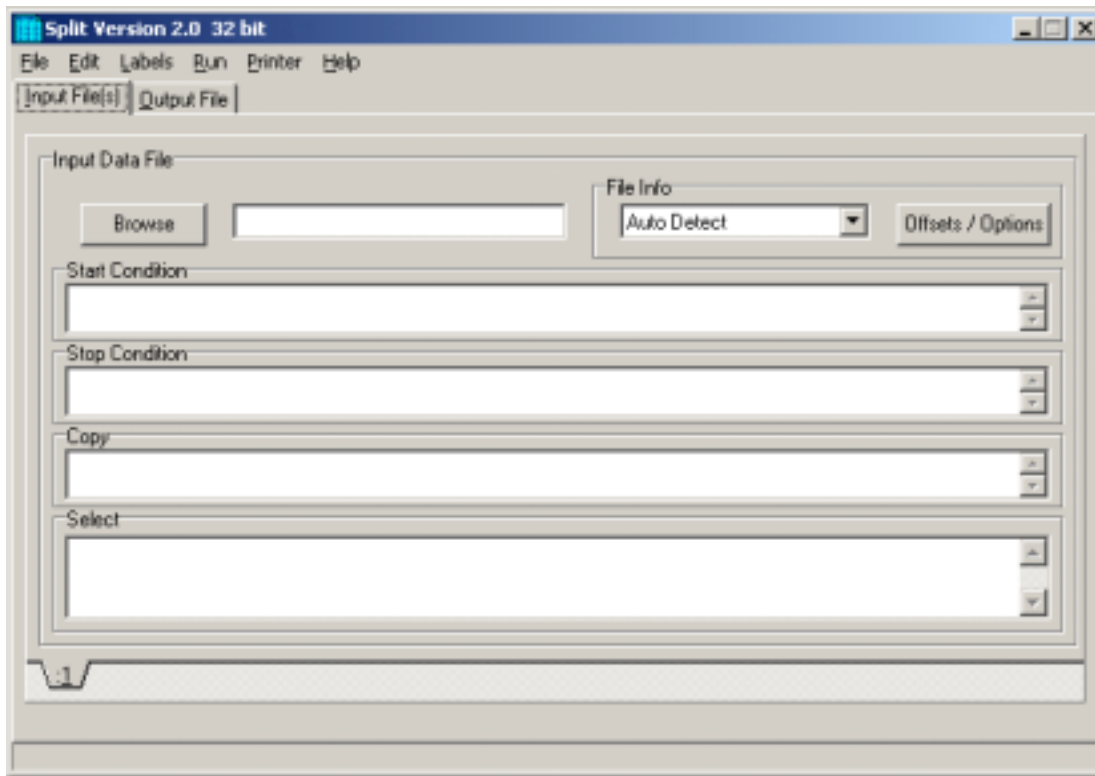
FIGURE 10.2-3. Input File Template

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 we only need to specify the input filename, copy condition, and the data to select. Split allows specific 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_1.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 on 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 10.2-4).
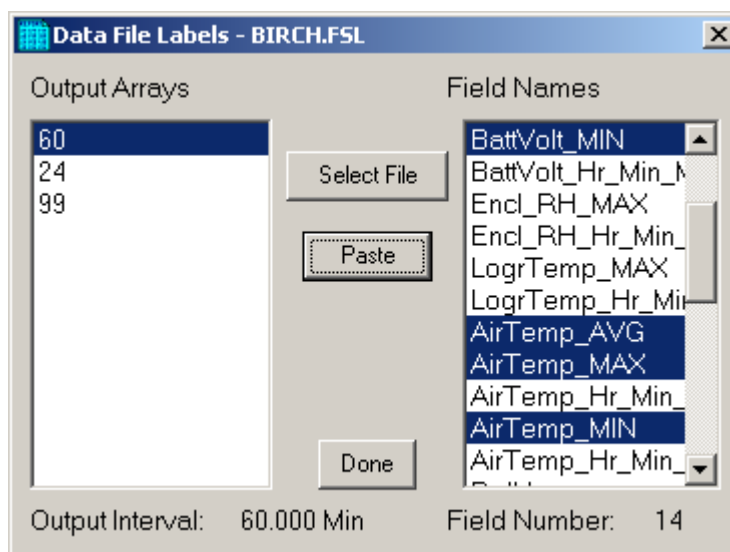
FIGURE 10.2-4. DATA FILE LABELS Screen
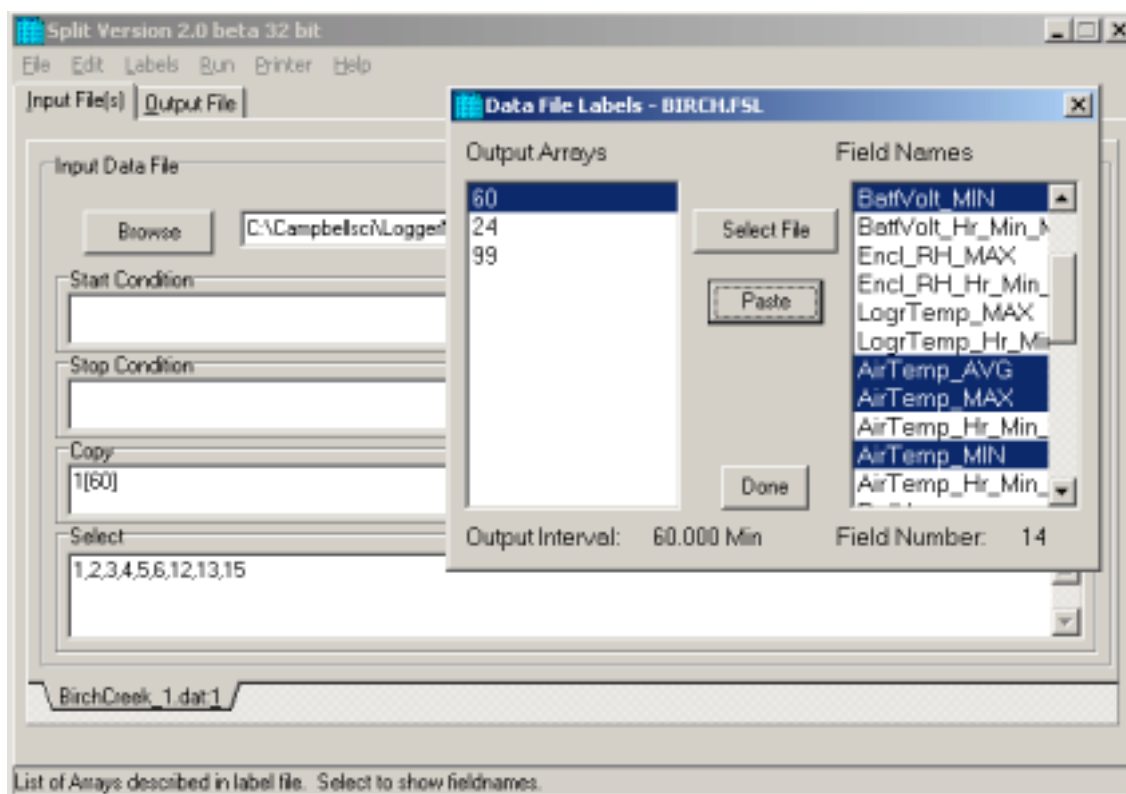
FIGURE 10.2-5. Pasting Values into Split

In this example we want the hourly data (note the Output Interval at the bottom of the Data File Label window) so we click on array 60. To paste the desired values from this array into the Select box, we select the field names while

holding down the <ctrl> key. All of the values could be selected by clicking
on the first one and holding the mouse button down, and dragging to the end.
Once the values you want have been selected click on Paste (Figure 10.2.5).

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 Filename box or in Start/Stop
condition, you will get the error message "Cannot Paste There".

The Paste operation put 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 we must specify the Output File. (Without one specified, Split will run
and display results but no output file will be created.) Click on the **OUTPUT
FILE** tab. We 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.

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 10 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
10.2-6).
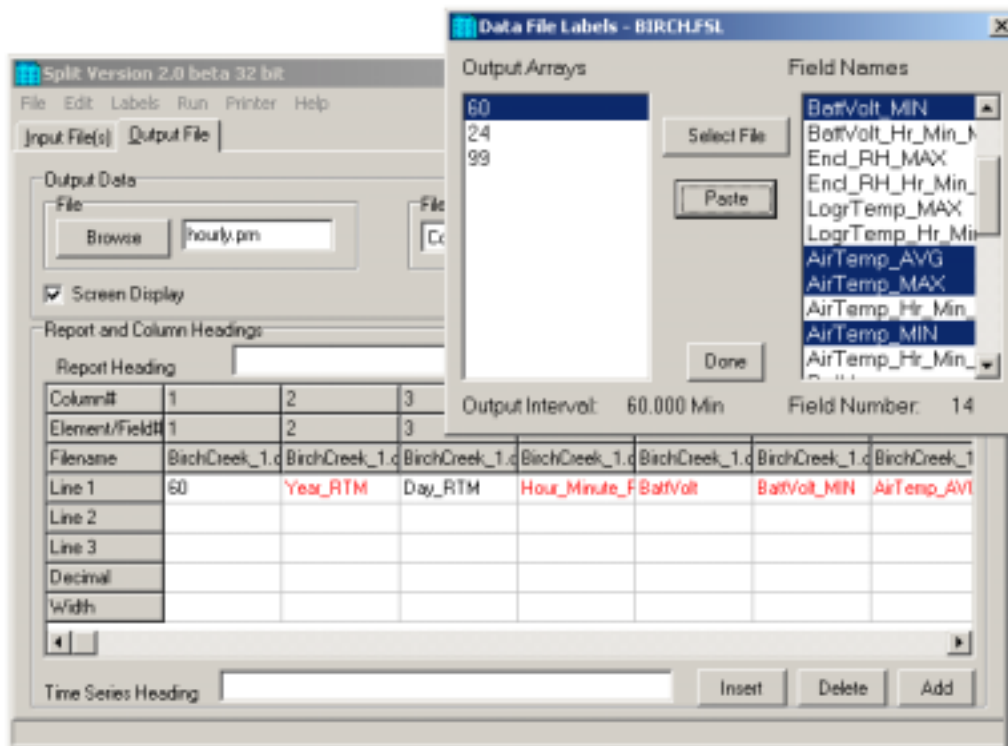
FIGURE 10.2-6. Pasting Labels Into Split

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
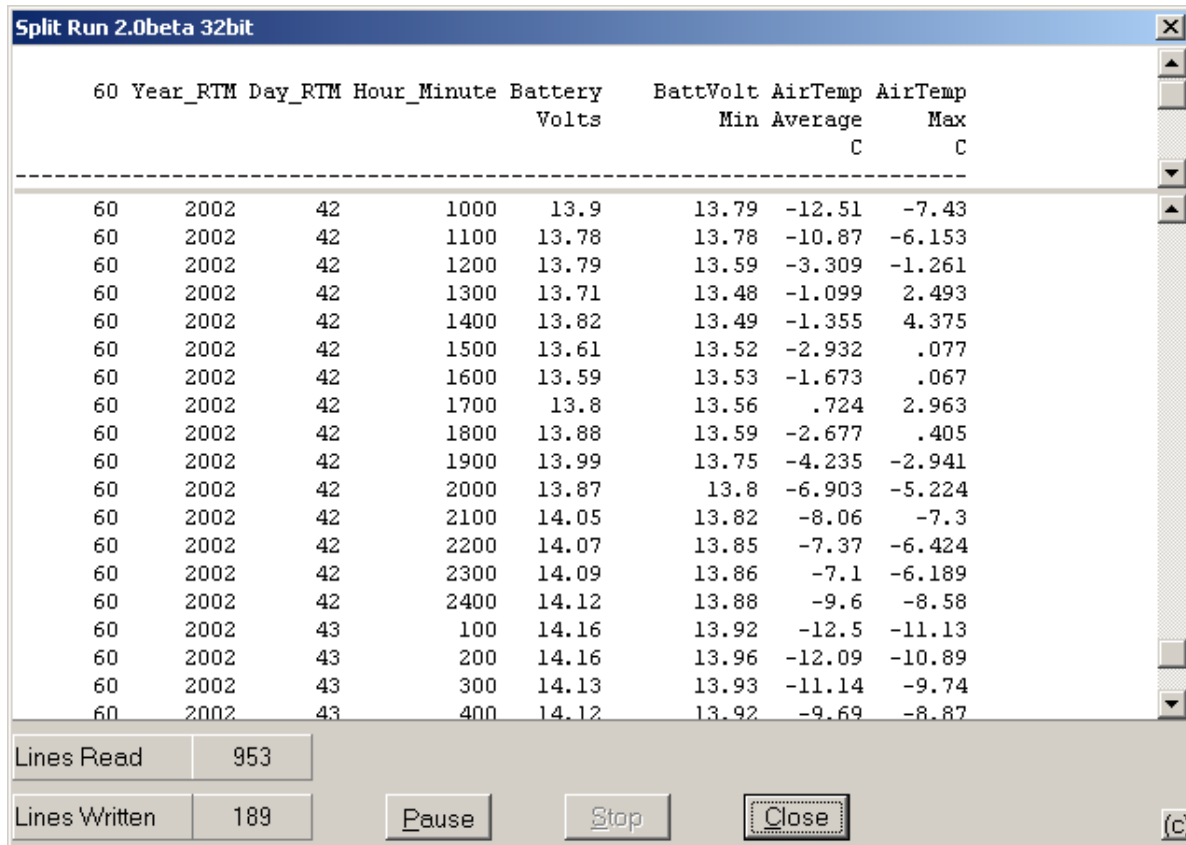
FIGURE 10.2-7. Edited Headings

Maximum column heading width is one less than the number entered in the column width field. 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 10.3-7) using the same editing technique.

For table based data files the timestamp is normally the first column and is a quoted text string ("2002-02-26 10:30:00"). Split uses these time stamps but to display them in the output you will need to change the column width for the first column to at least 24.

Now we are ready to run Split. Select RUN | GO and the hourly data will be split out and stored in hourly.prn. The results are displayed on the screen as shown in Figure 10.2-8.

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 10.2-8. Split Run Showing Hourly Data

10.3 Split Parameter File Entries

10.3.1 Input Files

The name of the Input File is entered in the space to the right of the **Browse** button. The default directory is defined by the working directory for LoggerNet. If the input file is not in the default directory, use the **Browse** button to find the input file.

Input Files must be formatted in Comma Separated ASCII, Final Storage (Binary) Format, Field Formatted ASCII (Split default output format), Printable ASCII, Table Oriented ASCII (TOA) or Raw A/D data (refer to special Burst Mode Instruction in 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 use the Data Label browser you can convert the binary file using the TOB2 File converter in under Programs | LoggerNet | Utilities.

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

Table 10.3-1 provides an example of Printable ASCII, Comma Separated, and Field Formatted 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 10.3-1 is used for example purposes in the following sections.

TABLE 10.3-1. Comma Separated, Field Formatted, Printable ASCII, and Table Oriented ASCII Input File Format Types

COMMA SEPARATED

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

FIELD FORMATTED

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

PRINTABLE ASCII

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

"TOACI1","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
```

A maximum of eight input files may be opened by Split. Additional input files are opened with the EDIT | ADD DATA FILE menu option. Split defaults to 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 templates will be generated. To change from one template to the other, click on 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.
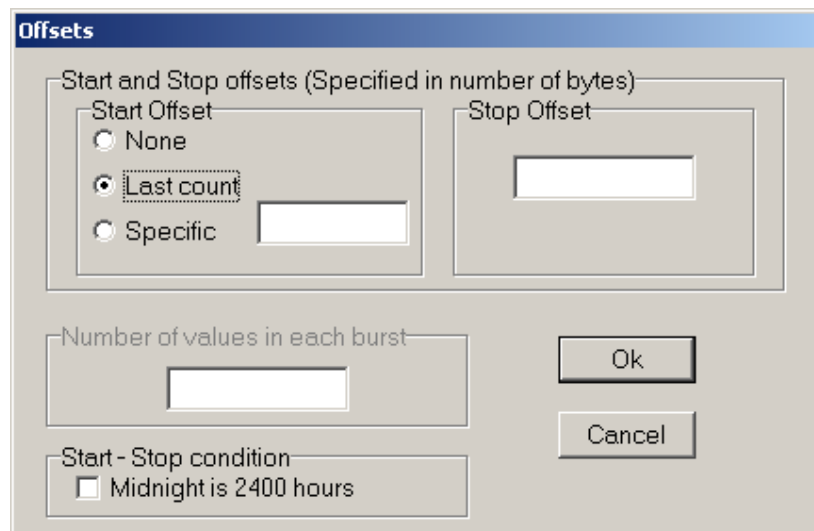
Example 1

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 on the **Offsets** button and selecting the Last Count option (see Example 1). 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 using the scheduled data collection option of LOGGERNET.

CAUTION   When using the Last Count option, if the Start and Stop
                  Conditions (Sections 10.3.2, 10.3.3) are specified, they
                  must exist in the newly appended data or Split will never
                  begin execution.

By selecting the Specific option and entering a number, Split will "seek" that
position in the file. This option saves time by starting part way through a large
data file. The number specifies the number of characters 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. Beginning and
ending points may be specified.

Example 2

Example 3

In Example 2, Split will skip the first 256 characters before it begins processing the data in Input File. In Example 3, Split will skip the first 256 characters before beginning and stop execution on character 1024.

In most instances, Split automatically recognizes the type of data file it is reading by using Auto Select in the File Info field. However, there are two exceptions:

- **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.

This option works with all subsequent commands except with the Transpose option (Section 10.3.1) or the Select elements option (Section 10.3.5).

10.3.2 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.

For example, the data in Table 10.3-1 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.

NOTE   Timestamps from table data files can be used as described in section 10.3.1. (For example, : :1[1500]: would start at 1500 hours.)

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.

NOTE   The element must match this start value exactly to trigger the start condition.

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_i 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, the wind direction must be between 200 to 275 degrees, inclusive.

10.3.2.1 Time Synchronization

The time synchronize function is useful when data are missing from files or several files of data need to be merged together. The files are synchronized according to time. This function synchronizes according to day, hrmn (hour-minute), and/or seconds. The syntax used to identify the time elements for array data is:

$$e_i[day]:e_i[hrmn]:e_i[seconds]$$

Referring to Table 10.3-1, to identify the day of year, type:

$$2[189]::$$

for hrmn type:

$$:3[1200]:$$

and seconds are expressed as:

$$::4[5]$$

A single colon is assumed to be between day and hrmn (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. For example 4[60] in the Copy line will create a synchronized file containing the data from the input files that occurred on 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.

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).

Table-based datalogger

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). As noted above in the array-based example, the Start condition is entered for both input files, but the copy condition is entered only for the 1 minute data file.

Start Condition = 1:1[-1]:1[1200]:1:

Copy Condition = 1:1:1[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:). 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). When no starting time is specified, Split assumes the starting time to be 0 and inserts carriage return line feeds (CRLF) at the start of the Output File. The number of CRLFs equals the time between 0 and the time of the first array selected, divided by the interval specified in the Copy condition.

When time synchronizing, if multiple input files are given starting times, Split starts the Output at the earliest specified starting time. 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.

10.3.2.2 Starting Relative to PC Time

Included in the time synchronize function is the ability to start relative to the current PC TIME (computer time). This feature allows a .PAR file to be run on 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.

As an expanded example, assume that LoggerNet is used in scheduled data collection mode that automatically appends data to an archive file. SplitR is executed using the "After Call do" option in the Task Master. In this case the frequency of data collection and data reduction is the same. Time values in the data file (day, hrmn, 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.

CAUTION	Split will not start reading if the exact specified starting time cannot be found. 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.

10.3.2.3 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.

10.3.3 Stop Condition

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. Logical "and" and "or" statements can be used when specifying the Stop Condition (Section 10.3.2).

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 array 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 on several diskettes.

The C and F commands alter the meaning of the Stop Condition.

10.3.3.1 "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).

To use the C option, the Parameter file should be set up as follows:

- 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 10.3-2 below:

TABLE 10.3-2. Example of Event Driven Test Data Set

```
100,12.1,10.,32.6
101,92.7,67.7                    Data from arrays
102,56.1,48.7,98.,220.1          output during the
200                              first test.
100,12.5,9.89,30.1
102,56.2,50.,100.5,210.6         Second test.
200
100,13.1,10.1,33.1
101,94.1,69
200                              Third test.
```

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 10.3-3. 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 10.3-3. Each line of data represents one test. Notice that blanks were inserted if the data set (conditional array) did not exist.

10.3.3.2 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 10.3.5.2) 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$_i$[val$_i$].

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 10.3-2. The Input File template would look like that shown in Table 10.3-4.

TABLE 10.3-4. 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)

10.3.4 Copy

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 (Section 10.3.1.4). 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 10.3.2) can be used when specifying the Copy condition.

For example, referring to Table 10.3-1, 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 10.3-1, conform to the above Copy conditions.

To use the time synchronize function, time element(s) must be specified in the Start Condition (Section 10.3.1.1). The user specifies 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 hrmn 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 hrmn 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 10.3.1. 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.

10.3.5 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. Up to 255 characters can be entered on one Select line. If this is a limitation, open the Input Data file twice (use the EDIT | ADD DATA FILE menu) and use the Select line in the second Input File template to define the additional operations.

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 10.3-5 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 10.3-5. 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 (See Section 10.3.6.1)	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.

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.

Processing is accomplished through arithmetic operators, math functions, spatial functions, and time series functions. The following is a list of operators, math functions, and Time Series functions supported by Split.

TABLE 10.3-6. Split Operators, Math Functions, and Time Series 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^x)
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

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 (see Table 10.3-7 Special Functions, this Section)

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.

The greatest number that can be output is determined by the field width (Table 10.3-3, "Column Widths" option). If the width is eight or greater, the maximum number is 99999; for widths less than eight the maximum is 9999. Larger numbers will be stored as 99999 or 9999.

TABLE 10.3-7. Split SPECIAL FUNCTIONS

Crlf	=	Insert carriage return line feed in Output File.
Date("format"S;H;D;Y)	=	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. See Section 10.3.5.3.
Edate("format"S;H;D;Y)	=	The same as the Date function except that the output text is not quoted. EDate can be used to create monthly time series summaries. See Section 10.3.5.3.
"Label"	=	Insert Comment in Output file. (Label is anything within the quote marks.)
Line	=	Number of lines written to Output file.
.PA	=	Page break to printer or to the .RPT file.

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 as shown in the table. For example AVG(1;Date(23;2002))

10.3.5.1 Mathematical Functions, Details, and Examples

The following array of PRINTABLE 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
Arctan(x)	returns the arc tangent of element x in degrees.
	Examples:
	Arctan(7) = 89.788
	Arctan(7/6) = 87.365
Cos(x)	returns the cosine of element x in degrees.
	Examples:
	Cos(5) = .57215
	Cos(5-6) = .73551
Exp(x)	returns the exponential base e to the power of element x.
	Example:
	Exp(4) = .00037
Frac(x)	returns the fractional value of the element x.
	Examples:
	Frac(4) = -.89
	Frac(6+7) = .95
Int(x)	returns the integer portion of the element x.
	Examples:
	Int(7) = 270
	Int(5∗6) = 685
Ln(x)	returns the natural log of element x.
	Examples:
	Ln(6) = 2.5217
	Ln(7/6∗5/1) = 2.4337
Sin(x)	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.

SpaAvg(x..y)   returns the spatial average of elements x through y.
Examples:
SpaAvg(1..7) = 258.74
SpaAvg(1,4,7) = 122.54

SpaMax(x..y)   returns the maximum value of elements x through y.
Examples:
SpaMax(1..7) = 1200
SpaMax(1,2,5) = 176

SpaMin(x..y)   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

10.3.5.2 Time Series Functions, Details, and Examples

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 10.3.3.2 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.

Results which are output at the end of a specified Interval are Interval Summaries. A typical select line that would produce an Interval Summary is:

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 24th 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 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	Air temp deg F	RH	Mean Wind Speed mph	Mean Wind Vec Dir	Std Dev of Dir
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^{th} 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 10.3-8 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^{th} 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^{th} 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^{th} 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 nth value is specified, a running total will be output every n^{th} 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 nth 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^{th} 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^{th} value. The Sample function must always have an interval specified.
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 nth value is specified then it outputs the sample on a maximum every n^{th} value, otherwise it outputs the sample on a maximum at the EOF.
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 nth value is specified then it outputs the sample on a minimum every n^{th} value, otherwise it outputs the sample on a minimum at the EOF. 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^{th} 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)

10.3.5.3 Special Functions, Details, and Examples

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 array based 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:
Smpl("Max Temp";24),Max(3;24),
Smpl(Crlf;24),Smpl("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.

"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

.PA   Sends a page break to the printer or .RPT file.

Examples:

2, 3, Smpl (.PA;12) =

100	58.56
200	57.48
.	.
.	.
.	.
1100	61.34
1200	60.61

Page 2 ---------------------------

1300	61.01
1400	60.93
.	.
.	.
.	.
2300	55.48

0   55.22

Date("format"; S; H; D; Y)   Converts a datalogger's time stamp to a different format and encloses it in double-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). See example below.

If "serial" is entered for the "format" string, a serial date will be output.

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 output the Bad Data string is the time and date resulting from the conversion will not fit in the specified column width.

Date Format Examples

Assume that in an array-based data file, element 2 is Year, element 3 is Day of Year, element 4 is Hour/Minute, and element 5 is Seconds.

String Entered	Output
date("mm/dd/yy, h:nn";5;4;3;2)	"02/25/02, 4:10"
edate("mm/dd/yy, hh:nn";5;4;3;2)	02/25/02, 04:10
edate("dddd, mmmm d, yyyy";5;4;3;2)	Monday, February 25, 2002
edate("'Date:' mmm d, yyyy";5;4;3;2)	Date: Feb 25 02

If a time element is missing from an array-based 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:

date("mm/dd/yy, h:nn";1;1;1;1)	"02/25/02, 4:10"
edate("mm/dd/yy, hh:nn";1;1;1;1)	02/25/02, 04:10

NOTE	When processing a data file from an array-based 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.

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 as shown in the table. 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.

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 was programmed. This example does calculations based on temperature and wind speed to determine the wind chill.
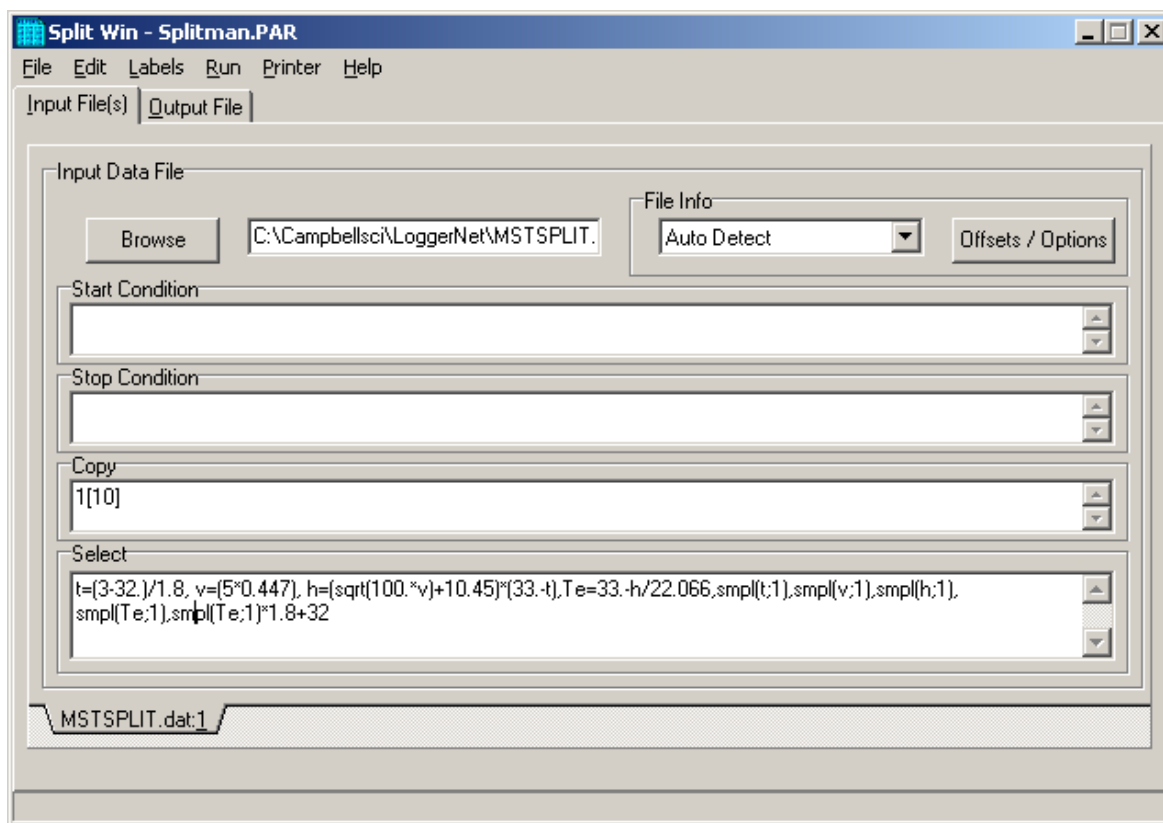
The following screen shows the output file setup including the column headings and the units.
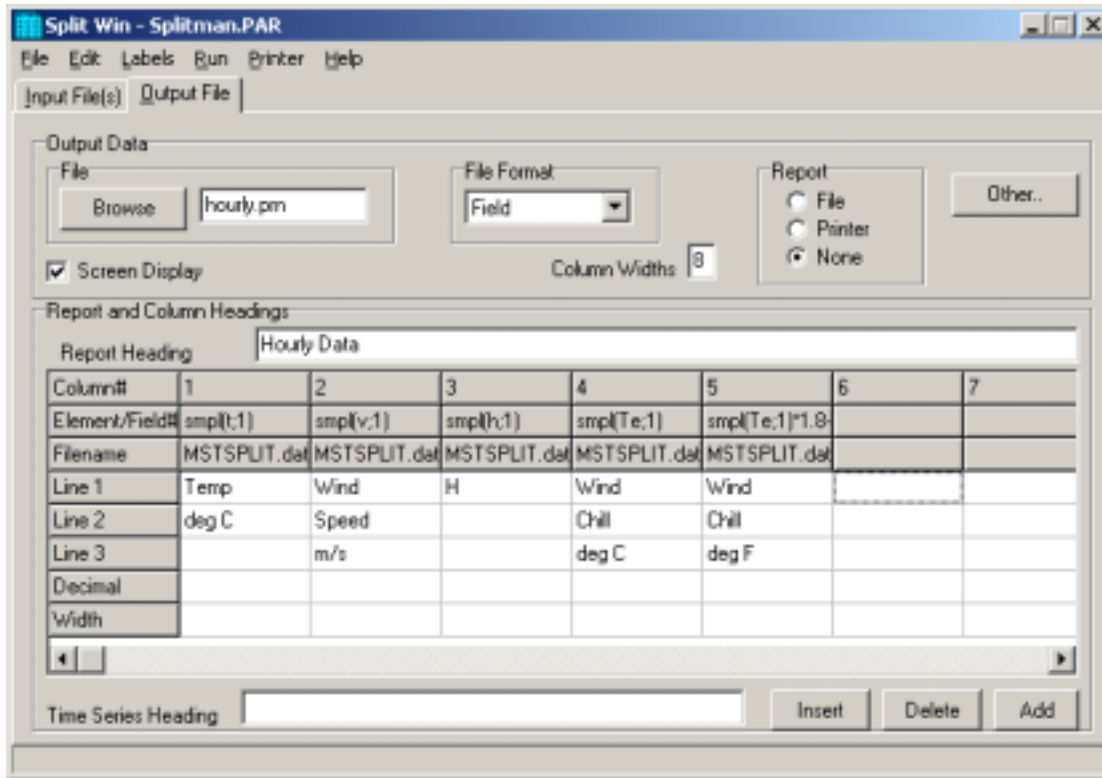
This .PAR file produces a wind chill summary of the Mt. Logan Peak data set. The formula for calculating wind chill is given as follows:

$$Te = 33-(h/22.066)$$

where

Te = Wind Chill equivalent temperature, degrees C
h = $((100V)^{0.5} + 10.45-V)(33-T)$

where

h = Kcal m^{-2} hr^{-1} wind chill index
v = wind speed in meters/second
T = 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).[1] 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
[1] "Wind Chill Errors", Edwin Kessler, Bulletin of the American Meteorology Society, Vol. 74, No. 9, September 1993, pp 1743-1744.

10.3.5.4 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 = 3×10^5).

- 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	INVALID EXPRESSION
Arctan (2/3)	Arctan ((2/3))
Arctan (2/3, 3/4, 4/5)	Arctan ((2/3, 3/4), 4/5)
Arctan (COS(2))	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 10.3-6 explains how values outside the specified range are treated.

- The count 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 count which determines the interval. Sample on maximum and sample on minimum require two elements or expressions also separated by a semicolon.

10.3.6 Output Files

To create an Output File, click on the **OUTPUT FILE** tab. The file is created on the default drive or directory unless the filename 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 Filename is entered, regardless of extension, an Output file is created only when the RUN | GO menu option is selected.

If this line 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 Filename cannot be the same as the Input Filename. 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 10.3-1. Output File Template

10.3.6.1 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 no output 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 10.3-1 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. Column widths of up to 37 characters can be set. 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, uncheck 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. Select the appropriate option in the Report field. 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. The default Output File is created. A basic HTML file is also created containing the formatted report data. The HTML file can be used as a display of the formatted data output in a web browser.
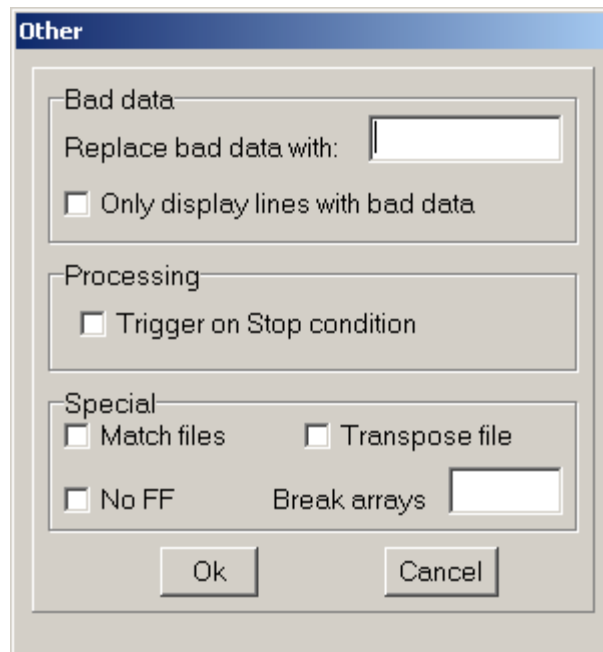
FIGURE 10.3-2. Other Output File Options

Other

The **Other** button allows the settings below to be modified:

    Replace bad data with - The text in the field, to the right of this option, is entered into the data set if data are blank, bad, or out of range. See Table 10.3-8 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 10.3-8. Definition of Blank or Bad Data for each Data File Format	
File Format	**Definition of Blank or Bad Data**
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. See Select line elements (Section 10.3.3.2).

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 Filename and Start Condition for each pass. Split may then be used to merge the multiple files.

No FF - Suppresses form feed and page breaks to printer or .RPT file when a report is generated.

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.

10.3.6.2 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 255 characters including backslashes and carriage returns. "\" characters break the report heading into multiple lines.

When Time Series functions are used in the Select field, 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.

"PCDATE" within the Report Heading inserts the computer's current date (Month-Day-Year). For the European format (Day-Month-Year), enter "PCEDATE".

10.3.6.3 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. The maximum Output field width is 37 characters.

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.

10.4 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 pops up.

10.5 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.

10.6 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 a Task in LoggerNet, from a batch file, or from a Windows 95/98/NT command line prompt.

10.6.1 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 filename. For example:

"Splitr LOGAN.PAR/R TEST.DAT TEST.PRN"

Replaces the Input and Output filenames 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 is limited to 255 characters for Windows 95 and 127 characters for Windows NT. The parameters must be in the following sequence: Input Filename, Output Filename, 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, space comma space (,) may be entered in the command line. For instance, if the parameter file LOGAN.PAR contained TEST1.DAT as an input filename, the following command line would leave the input file TEST1.DAT and change the output file to TEST.PRN.

"SPLITR LOGAN/R , TEST.PRN"

10.6.2 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.

10.6.3 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 as a Task in the LoggerNet Setup window.

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.

10.6.4 Command Line Switches

Splitr has three switches that can be used to control how files are processed.

10.6.4.1 Closing the Splitr.exe Program After Execution (/R 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

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."

10.6.4.2 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

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.

10.6.4.3 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 or LOGGERNET task 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, 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 11. View

The View screen provides a way to look at the data LoggerNet collected from the dataloggers. It will open data files (*.DAT, *.PRN, *.CSV), saved in a variety of formats including files from array-based and table-based dataloggers. View can also open other CSI file types (*.DLD, *.CSI, *.PTI, *.FSL, *.LOG, *.CR5, *.CR9).

Data can be viewed numerically, and up to two fields can be plotted on a graph. Both numeric data and graphs can be sent to a printer. Graphs can be saved to disk in BMP, WMF, or EMF format.

11.1 Overview

The View button on the LoggerNet Toolbar brings up the file view program. This program can be used to look at any data file collected by LoggerNet from dataloggers and storage devices, 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 ASCII, tabular, or hexadecimal format. However, since View is primarily a file viewing utility, a file cannot be edited or saved using this program.

11.2 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 11.6.2 Assigning Data Files to View for more information.

11.2.1 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: comma separated, printable ASCII, binary, or Table Oriented ASCII (TOA). The default file extension is DAT, but PRN and CSV files can also be opened in this mode. (Opening files of types other than these may cause unexpected results in View.)

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 an array-based 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. For more information, refer to Section 11.2.3 on FSL files.

NOTE

Table Oriented ASCII (TOA) files do not use FSL files. The data labels for each column of data is included as part of the data file.

View can be run as a stand-alone program by double-clicking the View.exe file in Windows Explorer. This will allow you to open larger data files without memory errors.

11.2.2 Opening Other Types of Files

To open a file that is not a data file (e.g., *.DLD, *.CSI, *.PTI, *.FSL, *.LOG, *.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.

11.2.3 Final Storage Label (FSL) Files

A final storage label (FSL) file is created by Edlog when an array-based 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 for column headers and to assign date and time values. To add or change an FSL file association, press the file open icon to the right of the FSL File field and select the desired FSL file.

View uses information from the *.FSL file to provide label names for the Data Panel column headings and Graph Panel traces. 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. 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.

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.

11.3 Data Panel

Data Panel is the text display for data files or other files opened by View. Data can be viewed as comma separated values, tab separated values in columns or in hexadecimal format.

Array Selection

Text View Options

Array Definitions

File Open Options

Printing Options

Graph View Options
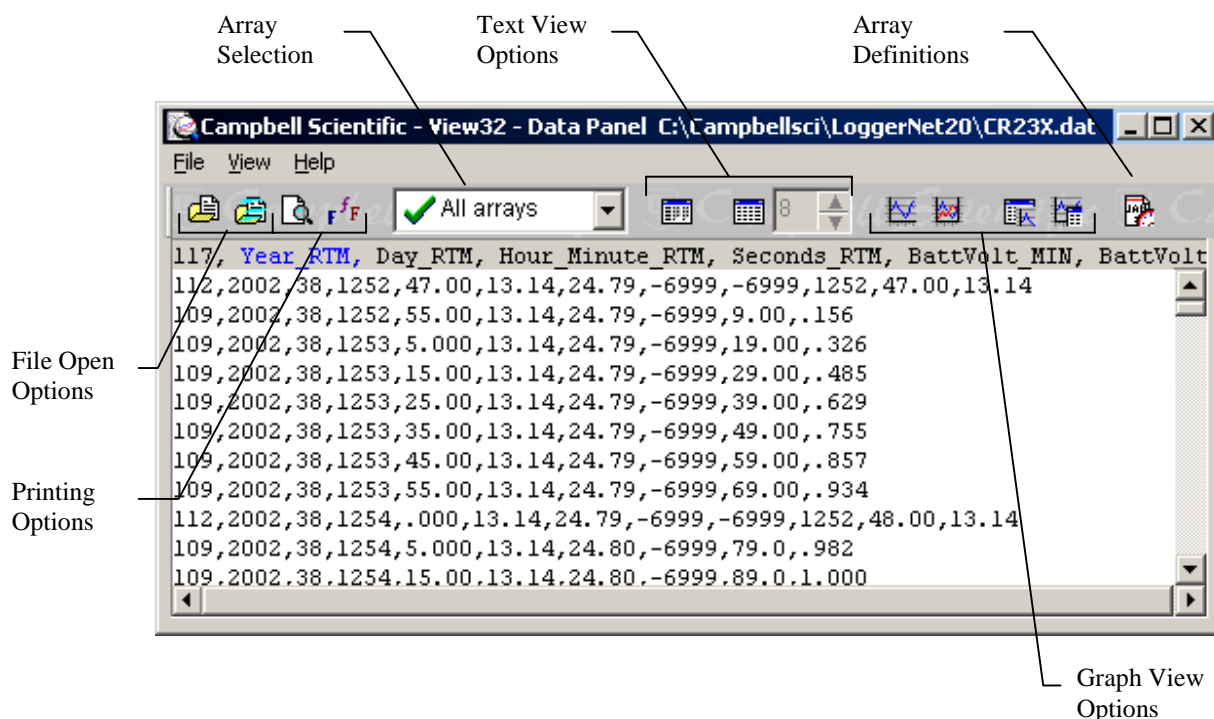
11.3.1 Array Selection

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 array-based dataloggers (CR510, CR10X, CR23X). Table-based dataloggers only store data for one table in each data file. CR5000, CR9000, and CRxxx-TD dataloggers store data in a table-based format.

11.3.2 Text View Options

By default, the Data Panel information 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 hexadecimal button ▦ 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 tab 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 8 to 27 character spaces.

NOTE

The tabular format is unavailable for files opened using Open As Text.

11.3.3 Changing the Font

The font used for the text display can be changed with the font selection dialog. Click on the Font button or select View | Font from the menu to change the font used for the printer and Data Panel 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.

11.4 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 Panel by clicking on one column and then holding the <ctrl> key down while clicking on 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 choose the type of graph by the button you click on the Data Panel 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.
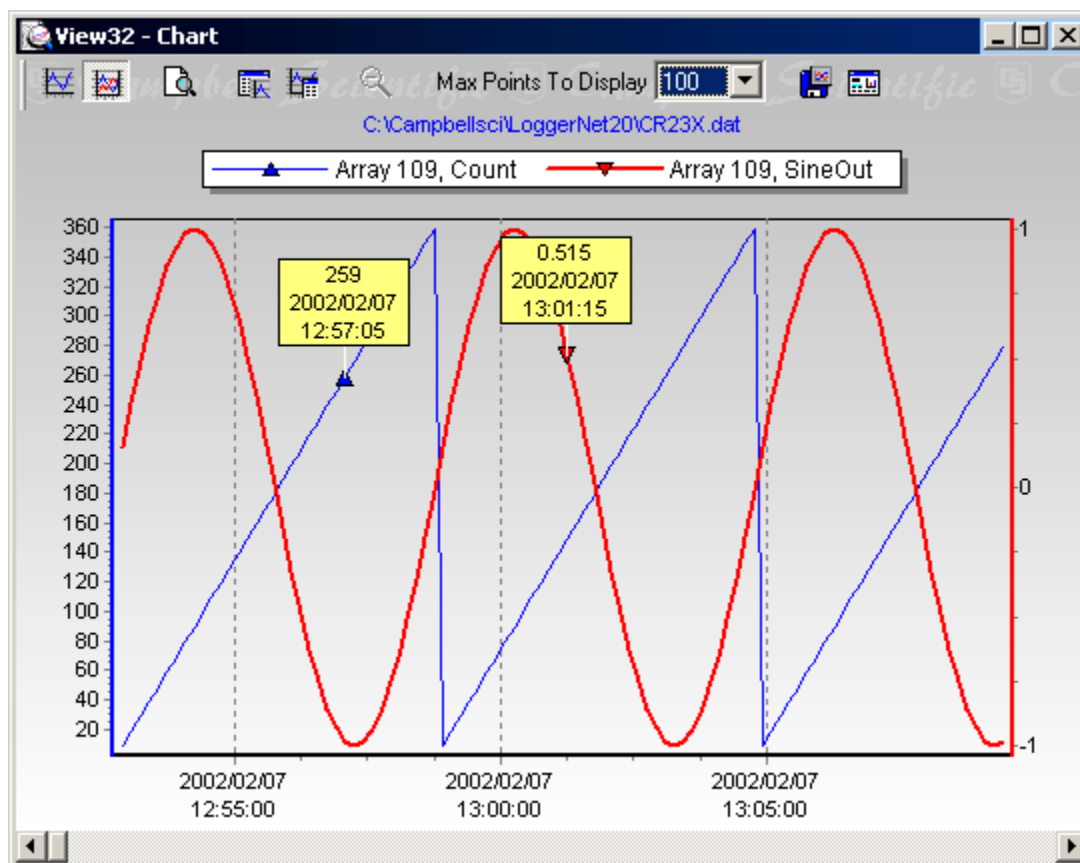
Graph Panel Options

The following options are available to set up how the Graph Panel displays the line graph. These options are available by clicking on 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.

Graphs two traces on the same Y-axis.

Graphs one trace on the left axis and the other on the right.

Invokes the Print Preview screen. Paper orientation and margins can be adjusted and the graph can be printed.

Forces the Chart Panel to be on top. This button is toggled on or off to keep the Chart Panel in front of other windows.

Forces the Data Panel to be on top. This button is toggled on or off to keep the Data Panel in front of other windows.

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.

Chart Axis Settings   ? ✕

☑ Show Axis   Scales

Axis:
 ⦿ Left
 ○ Right

☑ Automatic

☑ Auto   Change...   Maximum: 85.1

☑ Auto   Change...   Minimum: -6,999

Change...   Desired Increment: 0

☐ Logarithmic   ☐ Inverted

☑ Visible
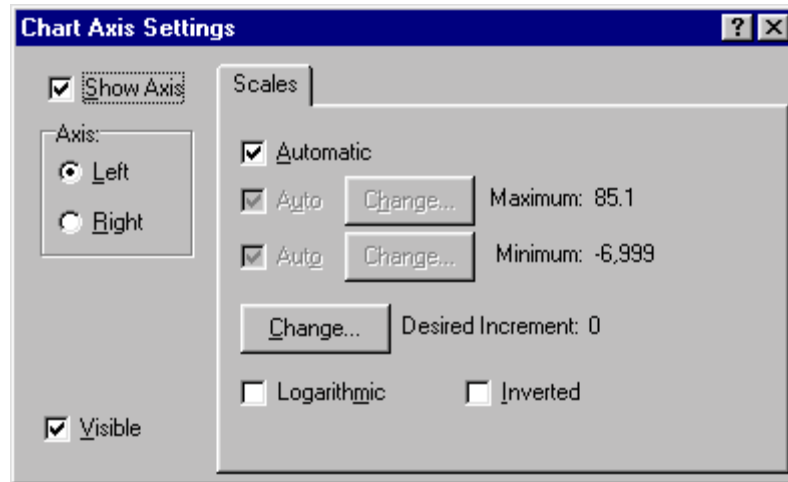
100 ▼   Allows you to set the number of points that are displayed across the width of the graph. Select the arrow to the right of this field to invoke a drop-down list box to select the number of points.

Data Labels Feature

199
2002/02/07
12:56:05
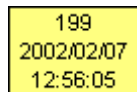
Data Labels allow you to get the exact numeric value and timestamp for a data point on the graph. Click on a data point to display a data label for that point. One data label is displayed for each trace on the graph.

Zoom Feature

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 zoomed. To return to normal view, click the Zoom Cancel button or hold the mouse button and drag the cursor from right to left.

Returns a zoomed graph to normal view.

You can use the scroll bar at the bottom of the graph to scroll forward or backward in time.

11.5 Printing Options

All printing is done through the Print Preview screen. Bring up the screen by clicking on 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.

11.5.1 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 on the page. To return to normal view, right click on 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.

11.5.2 Printing Graphs

With the Chart Panel window opened, click on the Print Preview to preview the printed page, and then select the Print button to print the graph. You can also right click on 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.

11.6 Advanced Topics

11.6.1 Assigning Data Files to View
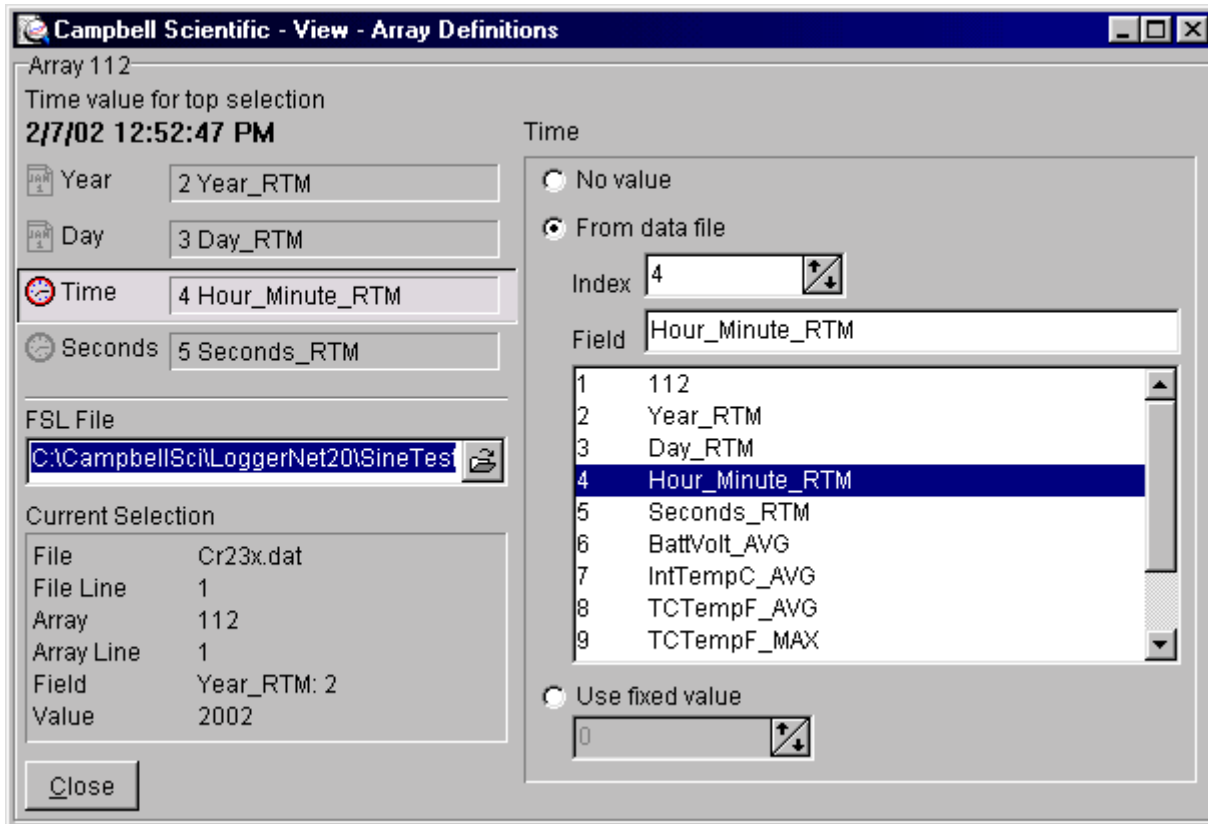
Windows will let you assign the program with which a particular file type will be opened based on that file's extension. When a file with an assigned extension is double-clicked, it will be opened with the associated program. You may want to associate *.DAT files with the View program for quick opening of data files. This association can be made by selecting **View | Options | File Types** from the 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.

11.6.2 Array Definitions (Array-based dataloggers only)

The Array Definitions button ![icon] on the toolbar brings up the Array Definitions window. You can use this window to manually assign correct time and date stamps to an array-based 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 on 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 Filename field to make this assignment. View will then use information from the *.FSL file to provide label names for the Data Panel column headings and Chart Panel traces. For more information, refer to FSL files section 11.2.3.

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.)

Section 12. Real-Time Monitor and Control

The Real-Time Monitor and Control (RTMC) software provides the ability to create and run graphical screens to display real-time data as LoggerNet collects it from the dataloggers. Controls are also provided to view and set datalogger ports and flags along with the data in settable variables. For graphical displays that can show historical data, such as line charts, RTMC will bring in and display the historical data available in LoggerNet's data storage. RTMC also provides expressions to condition the collected data for display (e.g., convert temperature in Celsius to Fahrenheit).

RTMC creates files that can be used with RTMC Run-Time on another computer that is connected to the LoggerNet computer by a local area network (LAN) or the Internet.

12.1 Overview

Real-Time Monitor and Control is an application that allows the user to quickly create graphical display screens that monitor real-time data, control set points, and toggle ports and flags. RTMC can combine data from multiple dataloggers on a single display. As LoggerNet collects data from the dataloggers, the displays in RTMC are automatically updated.

RTMC has two operating modes: Development and Run-Time. The Development mode allows you to create and edit a real-time graphic display screen to display the data collected from the dataloggers. Once the screen is built and saved as a file, the screen can be displayed using RTMC Run-Time. This allows graphic display screens to run on other computers with just the RTMC Run-Time program.

12.2 Development Mode

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

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

Project Component List - The panel on the left shows the hierarchy of the display components and how they are associated with each other. Every component of the display screen is shown in this list and it provides a shortcut to get to any graphical component.

Project Workspace - The middle panel is the display screen workspace. The graphic components are placed in the workspace, as they should appear on the final display.

Display Components List - The panel on the right contains the display screen components that can be placed in the workspace. Selecting a component and

clicking in the workspace places the component and brings up the Properties window for that component.

RTMC was designed to be easy and straightforward to use. Experiment with different combinations and options to get the display results you are looking for.
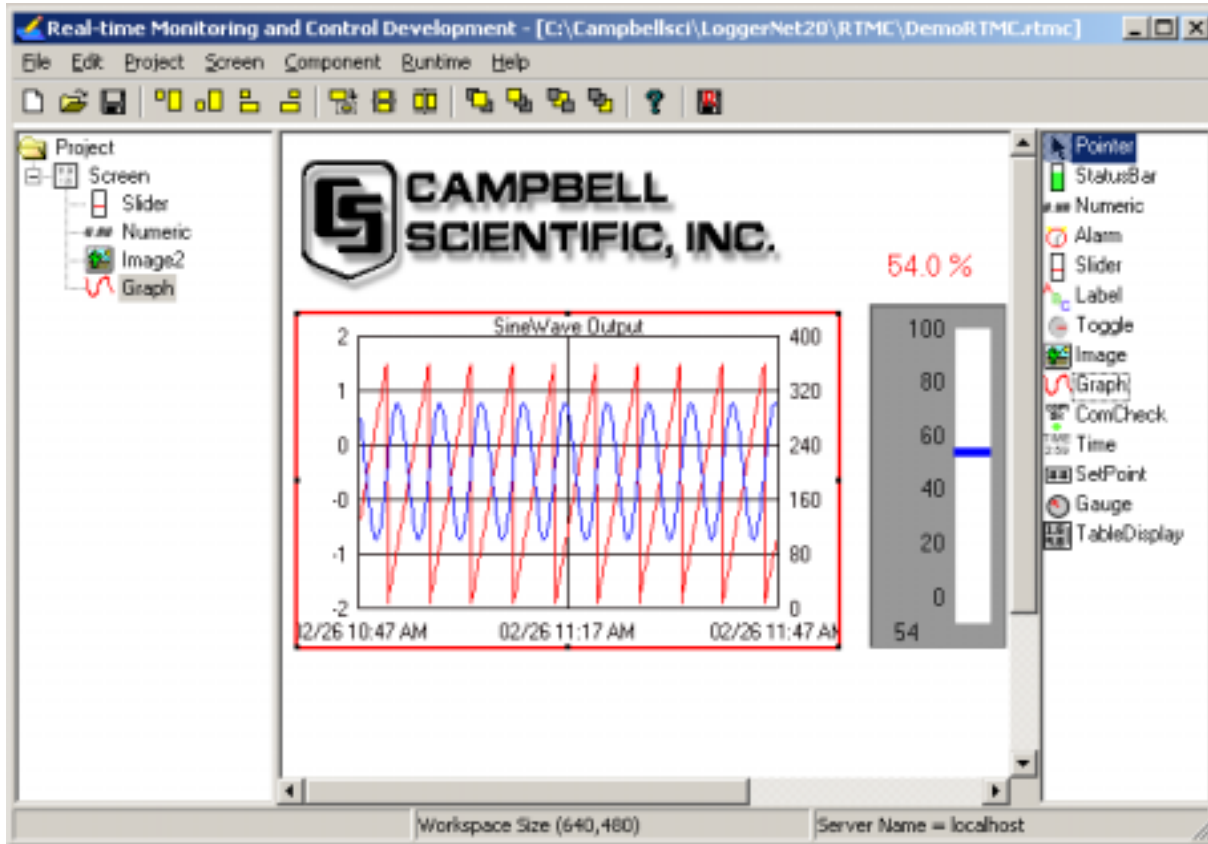
As seen in the example screen above, different types of graphic components can be combined to create an attractive real-time display. Company logos, maps, or any image stored in a standard graphic file format can be placed on the screen. Many images have been included with RTMC and others can be added as needed.

12.2.1 The RTMC Workspace

The RTMC workspace is a fixed size container for holding one or more display screens. As new display screens are added they appear as tabs in the project. Each display screen is a container for the various display components that make up each tab of the real-time display.

12.2.2 Display Components

Display components are the objects that are used to display data. To add a component to the workspace, click on an item in the Display Components List in the RTMC window and then click anywhere in the workspace. The

component's Properties window is automatically displayed when the object is first placed in the work area. The Properties window is used to set colors, scale values, text, etc., and to assign the data value to be displayed by the component.

NOTE

When a display component is linked to a data value, the value will be automatically updated on the display when data is collected by LoggerNet. If data collection is not set up in LoggerNet or the selected data value is excluded from collection, the values will not update. (Input locations, ports and flags for array based dataloggers do not have to be collected to be displayed in RTMC.)

After a component's properties have been set, select Apply to enable the changes, or Apply & Close to enable the changes and close the Properties window. Once the link to the data value has been applied, if there is data available from LoggerNet for the component, the value on the display will update.

To make changes in display component settings, the Properties window can be invoked by double clicking the component. If you make changes to a component's properties but then decide to reject those changes, press the Cancel button to return the properties to the last applied state. If Cancel is selected when a component is first placed in the work area (and Apply or Apply & Close has not been pressed), the display component will be removed from the screen.

Available Components

The following is an overview of the display components available. The online help has detailed information about each of the components and their properties.

Pointer 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.

Status Bar is used for displays such as a thermometer where the height of the colored bar is based on the data value. You can set high and low values, bar color and placement on a background graphic.

Numeric displays the selected data value as a numeric value. You can set the font size and type along with the number of decimal points and a units label.

Alarm is used to set up a display component that changes appearance when an alarm condition occurs. You can set the normal and alarm graphics, the alarm set point, and a sound file to play when the alarm occurs. In run-time mode,

right clicking the component with your mouse, and selecting Acknowledge Alarm will disable an audible alarm.

Slider is used to display a value as a point on a scale. It can also be used to set the value of an input location or Public variable. You can set the color, background graphic and the upper and lower limits.

Label is text that can be used to put titles or labels on the graphic components.

Toggle is used to display the value of flags, ports, or other boolean data that can take one of two values. In run-time mode, it can be used to set a port, flag or boolean value. Any non-zero number is On or True and zero (0) is Off or False. In run-time mode, right click a toggle to change its state. The option to change the state of a toggle with a double click can be enabled in the Properties window. You can set the two graphic images and separate locations to set and read the value.

Image is used to place graphic images on the display. Most types of graphic image file formats are supported including GIF, JPG, TIF, PNG, and BMP. You can set the angle of rotation if you want the graphic at an angle or turned sideways.

Graph 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 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 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.
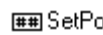
SetPoint is used like the Numeric component to display the selected data value as a number. When linked with an input location or Public variable in run-time mode, the data value can be set to a new value by double clicking the component and entering a new value in the resulting dialog box.

Gauge 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 array based 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.

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.

12.2.3 RTMC Operations

All of the RTMC operations are available from the menus at the top of the Development window. Many of the options are also available as buttons on the toolbar, or by right clicking on the components or other parts of the window. Operations that have toolbar buttons will have the button icon shown next to the description below.

File Menu

▯   **New Project** starts a new RTMC project. The currently opened project will be closed. If there are changes that have not been saved the user will be prompted to save changes.

📂   **Open** brings up the File Open dialog to open a previously saved project.

💾   **Save** will save the changes in the current project to the RTMC project file. If this is the first time the project has been saved, a Save As dialog will open to select the file name and directory for the project file.

       Save As brings up the Save As dialog to save the current project with another name or in a different directory.

       Exit closes RTMC. If there are unsaved changes, the user will be prompted to save changes before exiting.

Edit Menu

Undo cancels the last change made to the project.

Redo repeats the change that was just undone.

Cut/Copy/Paste are standard editing operations to take selected objects to an internal clipboard and paste back into RTMC.

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

Project Menu options work with the whole project or workspace.

Change Workspace Size allows you to specify the size in pixels of the run-time display screen. Default is 640 x 480.

Change Server Connection allows you to connect to a LoggerNet server on the same or a remote PC. The server name is the network name or IP address of the computer where LoggerNet is running. If you are connecting to a version of LoggerNet that supports server security, and security is enabled, you will need to enter the username and password. (LoggerNet 2.0 does not support security.) The port number should always be 6789.

Enable Connection Management tells LoggerNet to try and maintain an active communications link with the dataloggers for which data is being displayed by RTMC. This might be beneficial if you are trying to display data in RTMC with a fast update rate, over a communications link that takes time to establish (such as a phone modem).

Rename Project changes the name of the project as shown on the component tree.

Screen Menu options work with the tabbed screens in the project. The Screen Menu is also available by right clicking on any blank area of the workspace.

Screen Properties brings up the dialog to choose the background image for the current screen.

Add Screen adds a new screen to the project. Each screen appears as a tabbed page on the display. When the project is run the user can click on the tab to bring each screen to the front.

Delete Screen removes the current screen from the project. If there are components on the screen, they will also be removed.

Rename Screen brings up a dialog to change the name of the current screen. This is the name that appears on the screen tab in run-time mode.

The **Component Menu** is used to set the component properties, placement and alignment. The Component Menu is also available by right clicking on any of the components in the workspace.

Component Properties brings up the Properties window for the selected component.

Rename Component lets you change the name of the component in the list tree.

Delete Component Selection removes the selected component from the workspace.

Insert brings up a submenu allowing you select one of the components to insert on the screen. When the component is added to the screen the Properties window for the new component will come up.

Align provides some options for lining up a group of components with the first component selected. Select two or more components by using the cursor to click and drag a bounding box around the desired components. Components can also be selected by selecting the first component and then selecting the other components while holding down the <ctrl> key. With the components selected choose one of the alignment options. The components will be aligned based on the first component selected. The first component is identified by the dark red boundary. The other selected components have a light red boundary.

NOTE Be careful about the alignment you choose. Selecting Top Align for a group of components that are arranged vertically will cause all the components to end up on top of each other.

Top Align lines up the selected components to the top of the first component selected.

Bottom Align lines up the selected components to the bottom of the first component selected.

Left Align lines up the selected components with the left side of the first component selected.

Right Align lines up the selected components with the right side of the first component selected.

Size allows you to set two or more objects to the same overall size, width or height as the first object selected. Select one or more components by using the cursor to click and drag a bounding box around the desired components. The components can also be selected by selecting the first component and then selecting the other components while holding down the <ctrl> key. The first component is identified by the dark red boundary. The other selected components have a light red boundary.

Copy Size makes all the selected components the same size as the first component.

Copy Width makes the width of the selected components the same as the first component.

Copy Height makes the width of the selected components the same as the first component.

Ordering is used to manage the position of graphic objects on the workspace. Displays are often a combination of a background graphic and data display objects in front. Objects added to the workspace are, by default, placed on top of any existing objects. These operations are used to determine the order in which objects are displayed.

Bring to Front brings the selected component in front of all other display objects.

Send to Back places the selected component behind all of the other display objects.

Move Forward in areas where multiple objects are layered, brings the selected object forward ahead of display objects in the next layer up.

Move Backward in areas where multiple objects are layered, moves the selected object in back of display objects in the next layer down.

Run-Time Menu is used to Save and Run Current Project. This will save the project and start RTMC Run-Time to display the project.

Help Menu provides access to help for all of the features of RTMC. In Contents you can find an introduction and overview of RTMC as well as detail descriptions of all of the display properties and operations. The Index allows you to look up help topics based on a key word.

12.2.4 Expressions

RTMC has a built-in expression interpreter that allows the user to condition the data or create displays based on calculations on a data point.

The components that display data values (Status Bar, Numeric, Graph, and Gauge) can be processed using mathematical expressions. For instance, a temperature reading in degrees Celsius can be processed to display in degrees Fahrenheit using a mathematical expression.

An expression is entered for a component by first selecting the data value in the Select Data field, and then entering the mathematical expression after the defined data value. Using the above example, if the data value is defined as CR5000.TempData.Temp1 (datalogger.table.variable), you would enter

"CR5000.TempData.Temp1" *1.8+32

to convert the temperature reading from degrees Celsius to degrees Fahrenheit.

The data values from different dataloggers can be combined to form the expression. Data values are referenced as in the above example by datalogger name as it appears in the Setup screen, followed by the table name or array ID, followed by the data label. An array-based example 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.

12.2.4.1 Operators

Operator	Description
( )	Prioritizes parts of an expression within the larger expression.
*	Multiply by
/	Divide by
^	Raised to the power of
+	Add
-	Subtract
=	Equal
<>	Not equal
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to

12.2.4.2 Predefined Constants

Constant	Description
e	2.718282
PI	3.141593
True	-1
False	0

12.2.4.3 Functions

The following functions show the use and placement of the numbers the function operates on. The parentheses are not required unless there are two or more parameter values. (e.g., ATN2(y,x))

Function	Description
ABS(x)	Returns the absolute value of a number.
ACOS(x)	Returns the arc cosine of a number.
(x)AND(y)	Performs a logical conjunction on two numbers.

ASIN(x)	Returns the arc sine of a number.
ATN(x)	Returns the arc tangent of a number.
ATN2(y,x)	Returns the arctangent of y/x.
COS(x)	Returns the cosine of a number.
COSH(x)	Returns the hyperbolic cosine of a number.
CSGN(x)	Changes the sign of a number by multiplying by -1.0.
(x)EQV(y)	Performs a logical equivalence on two numbers.
EXP(x)	Returns e raised to the x power.
FIX(x)	Returns the integer portion of a number. If the number is a negative, the first negative integer greater than or equal to the number is returned.
FRAC(x)	Returns the fraction part of a number.
IIF(x,y,z)	Evaluates an expression (x) and returns one value if true (y), a different value if false (z).
(x)IMP(y)	Performs a logical implication on two numbers.
INT(x)	Returns the integer portion of a number. If the number is a negative, the first negative integer less than or equal to the number is returned.
LOG(x)	Returns the natural log of a number.
LOG10(x)	Returns the logarithm base 10 of a number.
(x)MOD(y)	Performs a modulo divide of two numbers.
NOT(x)	Performs a logical negation on a number.
(x)OR(y)	Performs a logical disjunction on two numbers.
PWR(x,y)	Raises constant x to the power of y.
RND	Generates a random number.
SGN(x)	Used to find the sign value of a number (-1, 0, or 1).
SIN(x)	Returns the sine of an angle.
SINH(x)	Returns the hyperbolic sine of a number.
SQR(x)	Returns the square root of a number.
TAN(x)	Returns the tangent of an angle.

TANH(x)   Returns the hyperbolic tangent of a number.

(x)XOR(y)   Performs a logical exclusion on two numbers.

12.2.4.4 Order of Precedence

Anything inside parentheses ( )

Exponentiation ^

Negation (unary) -

Multiplication *, division /

Modulo (remainder) MOD

Addition +, subtraction -

When consecutive operators have the same priority, the expression evaluates
from left to right. This means that an expression such as a-b-c is evaluated as
(a-b)-c.

12.2.5 Remote Connection

RTMC has the ability to connect to LoggerNet software running on another
computer. By default RTMC connects to the computer where it is running.
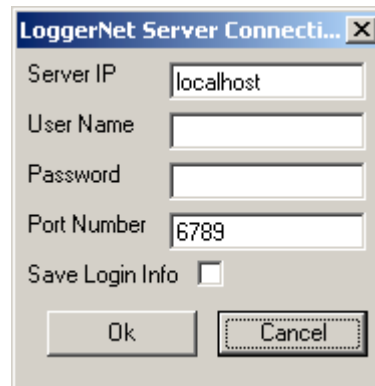
To set a connection to LoggerNet on another computer bring up the server
connection dialog from the Project | Change Server Connection menu. The
Server IP is the network computer name or IP address. The computer name is
defined by the network administrator. A connection can also be made over the
Internet or local area network using the 4 number Internet Protocol (IP)
address. This is a number that will have four digits between 0 and 255
separated by decimal points. An example would be 192.168.4.32. Do not put
leading zeros with the numbers.

The User Name and Password are only used if you are connecting to a
LoggerNet server that supports security (LoggerNet version 1.1) and the
network administrator has implemented security. The Port Number should
always be 6789.

Clicking the Save Login Info will save the computer address, user name and password as part of the RTMC file so the screen can be run without requiring the user to know the address.

12.3 Run-Time

The run-time operation allows you to run the real-time graphic display screen that was created in the developer mode. From the RTMC Development window you can test the operation of the display screen using the Runtime | Save and Run Current Project menu or clicking on the Run-Time icon on the toolbar. This will close the development window and start the project window with RTMC Run-Time as shown in the window below.

When the run-time display screen is started, the display components will be highlighted in reverse video until data is received from LoggerNet. If data is not displayed, check to see that the data is being collected by LoggerNet.

Once a project file has been created, the display screen can be run without starting the development mode window. From the Windows Start Menu under Programs | LoggerNet click on RTMC Run-Time. In the Run-Time window select File | Open Project to select the RTMC project screen to run.

RTMC Run-Time will print an image of the display screen by selecting File | Print Screen.
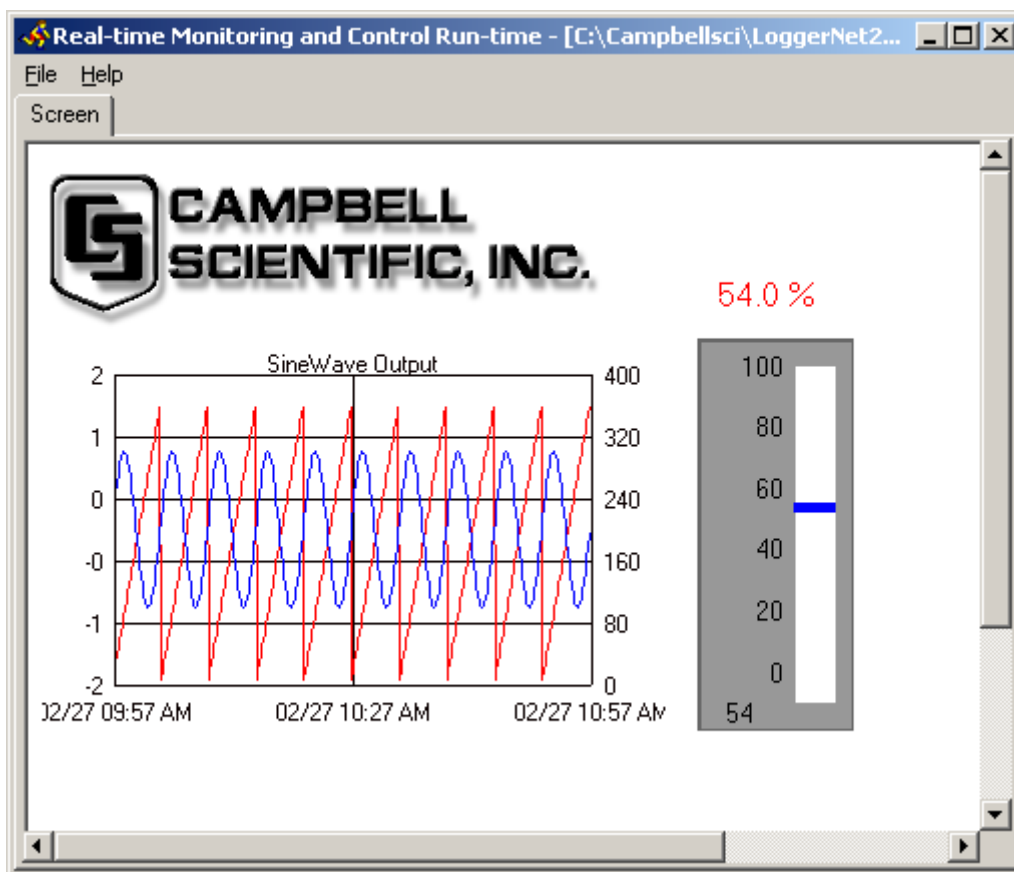
Section 13. Storage Module Software (SMS)

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 the Storage Modules through an SC532 asynchronous adapter or through a datalogger. 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.

SMS is used only with array-based dataloggers because storage modules are not supported in the table-based dataloggers. (CR5000, CR9000,CR510TD, CR10T, CR10X-TD, and CR23X-TD)

13.1 Overview

Clicking the Stg Module button on the LoggerNet Toolbar will bring up the Storage Module Software (SMS). Using a special interface, 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 reader.

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 formatted to Campbell Scientific's specifications.

Windows NT, Windows 2000, and Windows XP do not support 16-bit card services. 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.

13.2 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.
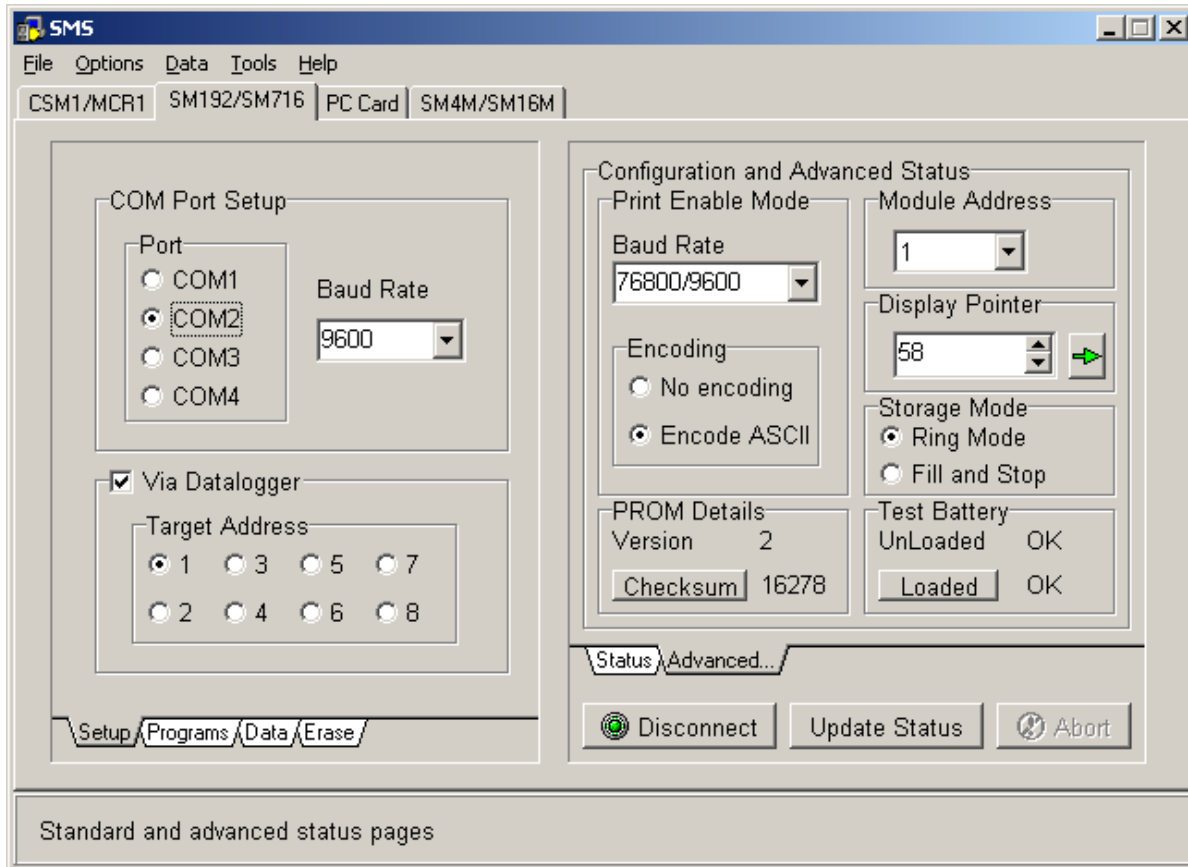
FIGURE 13.2-1. Typical SMS Screen

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 13.2-2 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.
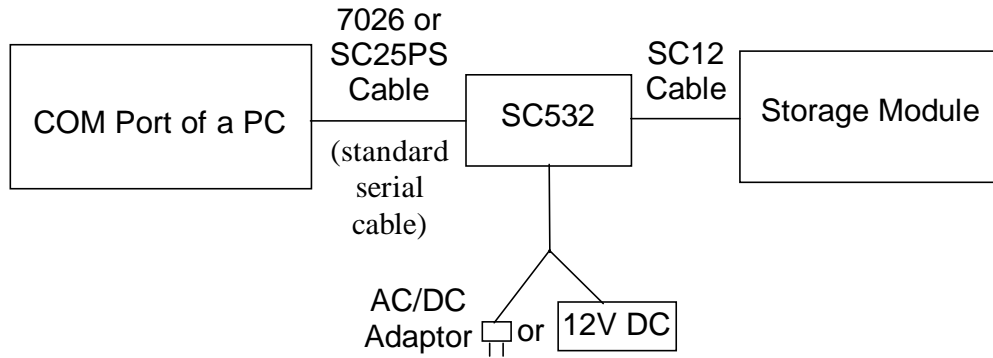
FIGURE 13.2-2. Connecting the Storage Module to a PC using an SC532 Interface

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 on 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.

13.2.1 The Setup Screen

The setup screen shown in Figure 13.2-3 is used to set the Communications (COM) Port and Baud Rate before communications between the module and computer are established.
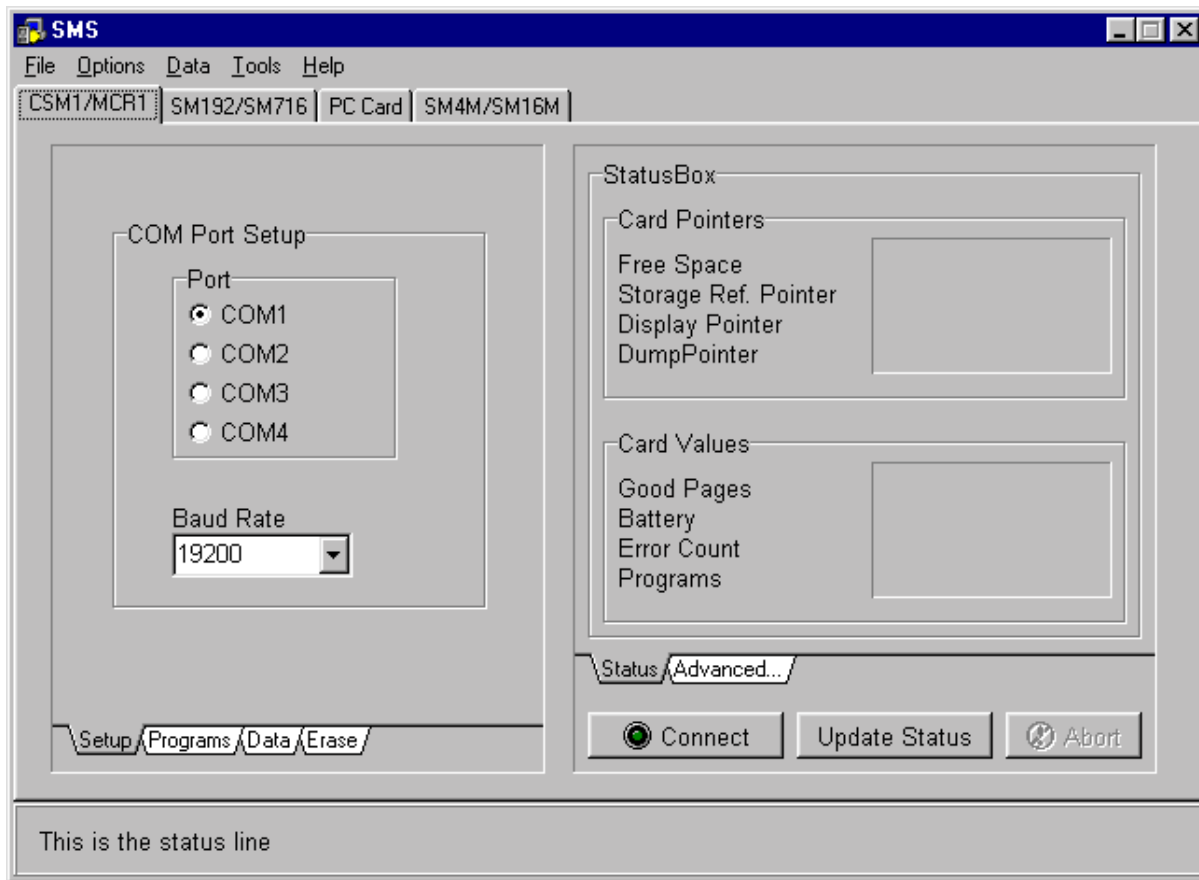
FIGURE 13.2-3. Typical SETUP Screen and StatusBox for the CSM1/MCR1 Module

13.2.2 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 reader you can set it up for use with SMS. This is accomplished by going to the **PC CARD SETUP** screen, and clicking the **Setup PC Card** button. When you click this button you will be guided through the correct setup procedure and the optimum software drivers will be used to establish communications.

NOTE	The PC Card drivers are NOT set up automatically, and should be set up as described above. Even if you are already using PC Cards successfully with your PC, it is recommended that you use the setup facility provided for use with SMS. This will not disrupt the normal use of other PCMCIA cards.

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 on the appropriate checkbox 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.

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.

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 13.2-4). Note, however, that the CR10(X), CR500, and CR510 dataloggers can only support a maximum baud rate of 9600.
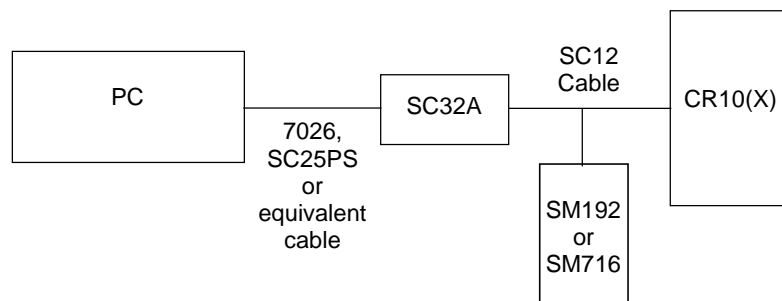
FIGURE 13.2-4. Connecting the SM192/716 to a PC and a CR10(X) Datalogger

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 checkbox can be used for connection. This sets the baud rate to 9600 and enables the Target Address checkbox. 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.

Connect/Disconnect

After connecting your storage module to your computer and selecting the required COM port and baud rate, click on 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.

13.3 Status Information

13.3.1 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 on 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.

13.3.2 Advanced Status Information

The **ADVANCED STATUS** screen is different for each type of module, and provides extra information and configuration facilities.

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.

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 checkboxes 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.

13-9

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.

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.

13.4 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 13.4-1 shows the program selection checkboxes 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.)
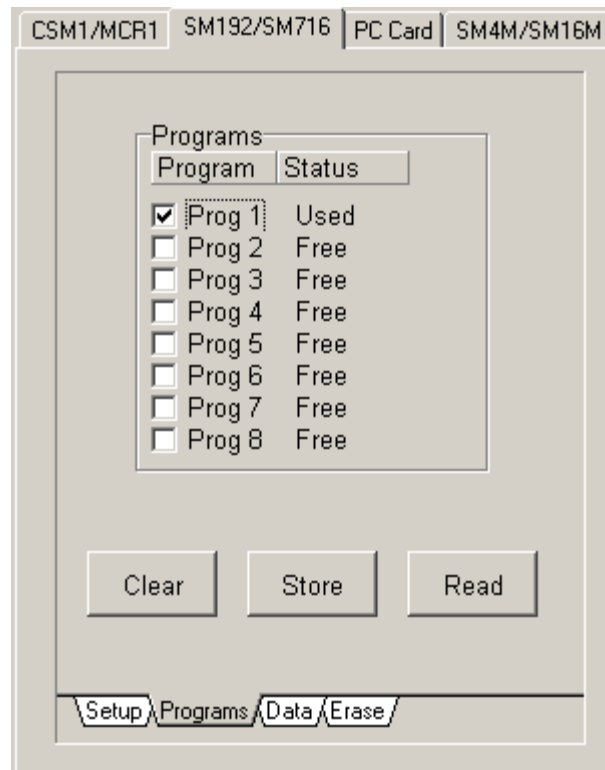
FIGURE 13.4-1. PROGRAMS Screen

13.4.1 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 on it to highlight its checkbox. Once you have selected the program location(s) you can carry out the desired action by clicking on one of the three buttons.

13.4.2 Clear

Clicking on 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.

13.4.3 Store

Clicking on 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.

13.4.4 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 on 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.

13.5 Data

Clicking on the **DATA** tab will bring up the **DATA CONTROL OPTION** screen as shown in Figure 13.5-1, below.
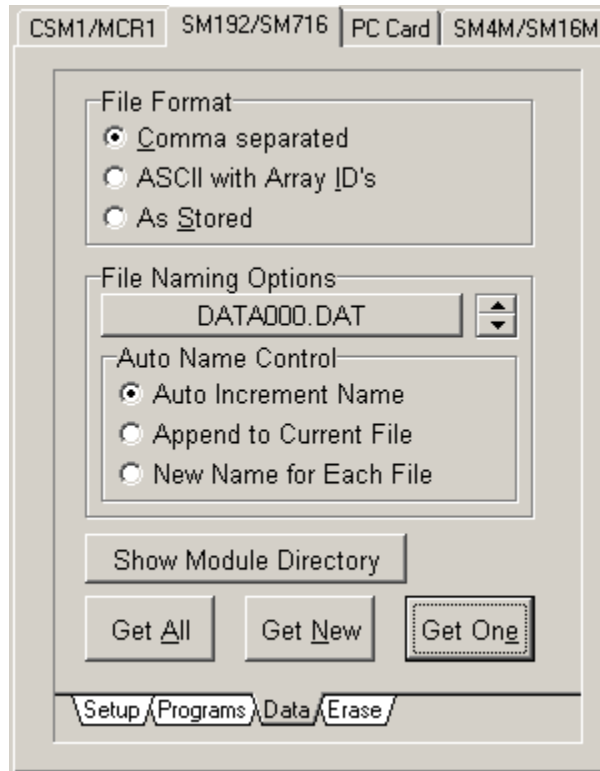
FIGURE 13-5.1. DATA CONTROL Screen

13.5.1 File Format

Before you attempt to "get" (collect) any data you should select your preferred file format.

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
```

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.

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 10) 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.

13.5.2 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.

Filename

Data files will normally be stored under a filename 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 on the button, a standard Windows Filename box appears that will allow you to type in the name you want.

Auto Name Control

You may choose to Auto Increment the filename, 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 filename and use automatic file incrementing. This means that the filename 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 filename will be incremented automatically as each file is collected. The program will scan the computer for files with the same base filename. 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" filename by clicking on the **Filename** button. The three-digit counter will be automatically added to the filename.

For example, if you chose a filename 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 **Filename** button). The new file will retain the current filename.

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 filename for each individual file.

13.5.3 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 on 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.
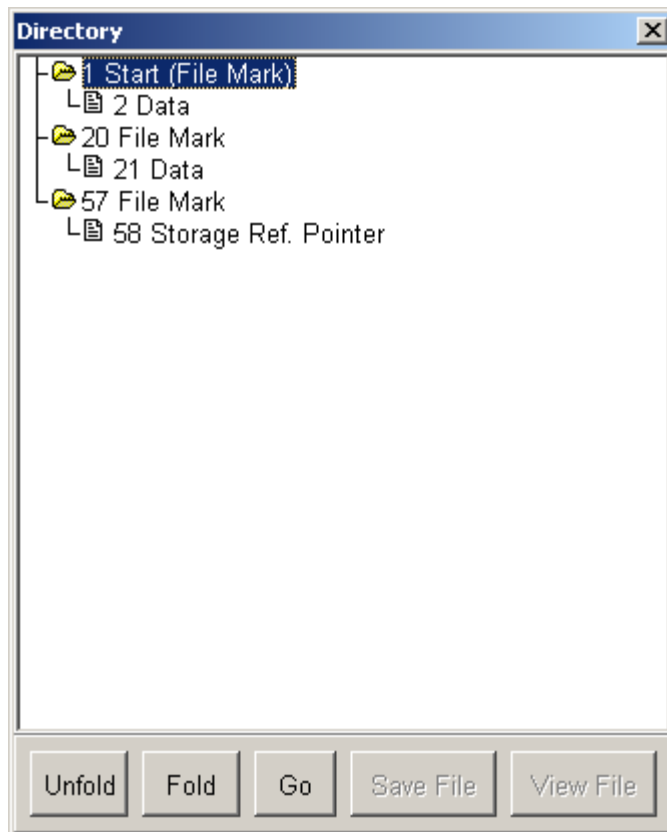
The Show Directory command is repeated on the menu bar. See the on-line help system for further details.

13.5.4 Get All

This option collects *all* valid data files in the card or module and stores them to the specified directory, starting with the filename shown in the Filename 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 filenames is displayed at the bottom right of the display.

13.5.5 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 filename 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.

13.5.6 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 13.2).

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 filename is displayed at the bottom right of the display.

13.6 Erase

Clicking on 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.

13.6.1 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.

13.6.2 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.

13.6.3 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.

13.7 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.

13.7.1 File

The **File** pull-down menu gives three additional options:

Terminal Emulator

Clicking on **Terminal** 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 13.5.3).

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 **Close** 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.

13.7.2 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 on 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.

13.7.3 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 13.5).
>
> 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 on the green arrow in the **ADVANCED STATUS** screen (see Section 13.3.2).

13.7.4 <u>T</u>ools

>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: <u>S</u>etup, <u>P</u>rograms, <u>D</u>ata, <u>E</u>rase, St<u>a</u>tus, and <u>A</u>dvanced. By selecting one of these options, you will be taken to that tab for the module to which you are connected.

13.7.5 <u>H</u>elp

>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.

13.8 Abort

>Most SMS operations can be aborted 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 14. 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.

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.

14.1 LoggerNet Server Problems

The following sections identify problems that have been observed with operation of the server. If you are experiencing problems with the server look through the following conditions to see if any of these match the problem you are having. If you find the problem listed, try the suggested remedies. If your problem is not listed or the remedies don't fix the problem, contact Campbell Scientific for technical assistance.

14.1.1 Starting LoggerNet and Connecting to the Server

Problem: Cannot start LoggerNet on a laptop computer.

> **Remedy 1**: If the computer is sometimes used on a computer network and no network card is installed, TCP-IP services may not be starting. Try starting the computer with the network adapter card installed.

> **Remedy 2**: If remedy 1 doesn't work, you will need to configure a non-existent dialup service with a fixed IP address. There is a description of this procedure in Section 2.2.

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.

14.1.2 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.

14.2 Application Screen Problems

Problem: Cannot set the clock from the Connect screen. Other communications with the datalogger succeed.

> **Remedy**: Check the Allowed Clock Deviation on the Clock tab for the datalogger in the Setup screen. If the difference in the datalogger and system clock is less than the deviation specified, the clock will not be set.

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.

14.3 General Communication Link Problems

Problem: Communications are not solid and difficulty is experienced sending programs to the datalogger.

> **Remedy 1**: If there are slow serial devices in the communication path, such as older modems, the server might be overrunning the buffers. Set the maximum packet size 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.

14.4 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 below click on 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.
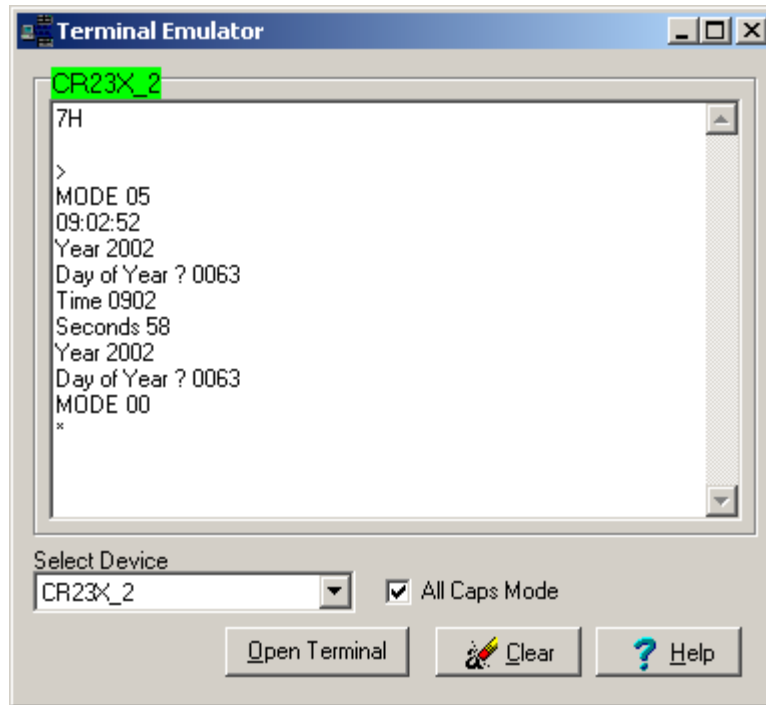
Dataloggers

The example above shows a terminal emulation session with a datalogger. Once you have selected the datalogger click on Open Terminal to start communications. Array-based dataloggers require you to type in the letters 7H (2178H for CR7 and 21X) and pressing 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 on 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 on Clear.

Phone Modems

Selecting a phone modem and clicking on 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	AT\J1 <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 trouble-shooting 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.

14.4 RF Communication Link Issues

There are two sets of problems that can degrade RF communications. The first is using combinations of RF components that do not work well together. The second is deterioration or failure of the RF components in the system. There are also situations where the equipment is performing as it should but marginal communications are due to 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.

14.4.1 Checking RF Components and Connections

Before testing RF signal strength, there are several things that should be done to verify that the right RF components are in place.

1. Check that the RF modem has the correct switch ID set on the DIP switches. (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 Ω. 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 Ω.

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.

14.4.2 RF Signal Strength Testing

Once you have verified that the right equipment is in place, make sure that all of the components have power. Then you are ready to proceed with performance testing.

To test a station's radio/cable/antenna transmission capabilities, a directional wattmeter is needed such as the Bird Electronic Corporation's Model 4304A Wattmeter. Proper connectors are also needed to place the wattmeter in series between the radio and antenna cable. A voltmeter is required to measure the battery voltage of the datalogger with and without radio transmission.

NOTE   If you are using a data radio that does not have a transmit button built in, you can easily build a push to transmit button from the documentation of the radio/RF modem interface connector. There will be one pin that when pulled high or pulled low will initiate radio communication. See the radio documentation to identify this pin. Connect a momentary pushbutton to either raise or ground that pin. **Always make sure that the antenna is connected to the radio before attempting to transmit.** Serious damage to the radio can occur if transmitting without an antenna.

Place the wattmeter in series between the radio and antenna cable. Set the wattmeter to the 15-Watt range, or the next highest wattmeter setting, and point the directional arrow first toward the antenna cable to measure forward power (Wf). Initiate radio communication, let the wattmeter stabilize, and record the wattmeter reading. Reverse the directional arrow so it is pointing back toward the radio, initiate radio communication, let the wattmeter stabilize, and record the wattmeter reading. This second reading is the reflected power (Wr). 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:

$$VSWR = [(1+R)/(1-R)]$$

$$Where, R = (Wr/Wf)^{\frac{1}{2}}$$

The impedance of the RF transmission cable (usually RG-8A/U) and antenna combination should match the impedance (50 Ω) 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 (Wf) is 5 Watts and the reflected power (Wr) 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.

14.4.3 Troubleshooting With Attenuation Pads

This test is used to measure the signal strength of the radio signal between two radios. There are situations where the signal from one radio can be heard by the other, but the signal is not strong enough to establish communications. In general a signal strength of greater than -95 dBm must be maintained for good communications.

There are many factors than can contribute to 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.
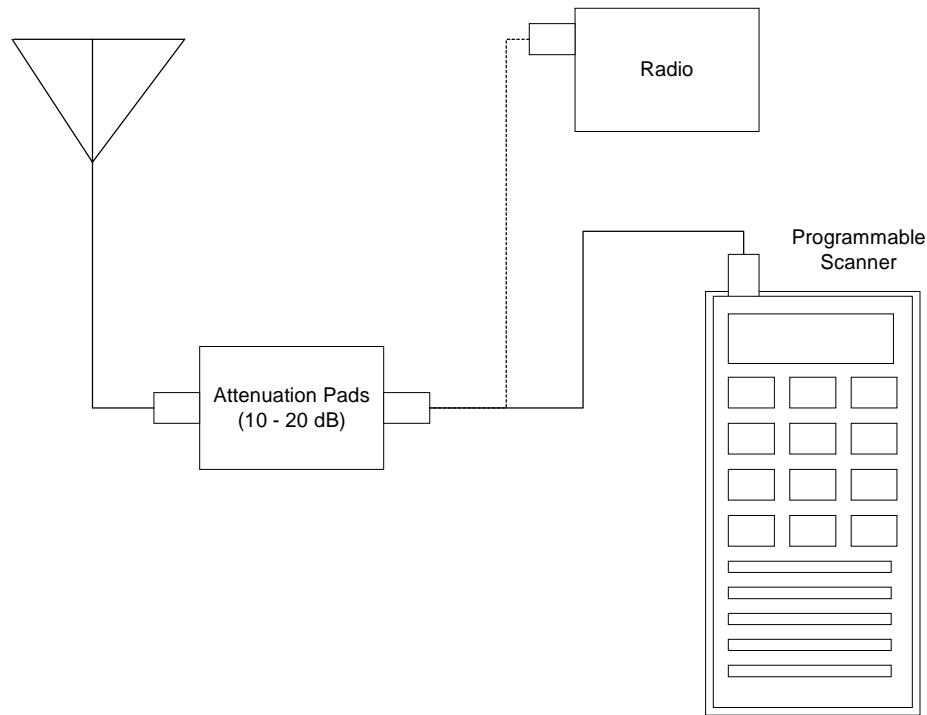
To test the power being received by a radio over an RF link, disconnect the radio from the antenna and insert the programmable scanner as shown in the figure above. Program the scanner to the radio frequency and adjust the squelch control until ambient RF noise is just cut out. This level will normally 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 pushbutton to either raise or ground that pin. **Always make sure that the antenna is connected to the radio before attempting to transmit.** Serious damage to the radio can occur if transmitting without an antenna.

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.

14.5 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 on the Start button. This will delete the old file and start a new one.

To start Data Table Monitor open Windows Explorer and got to the Program Files\CampbellSci\LoggerNet directory. Double click on the Tablemon2.exe file. The utility will start with a screen similar to the one shown below.
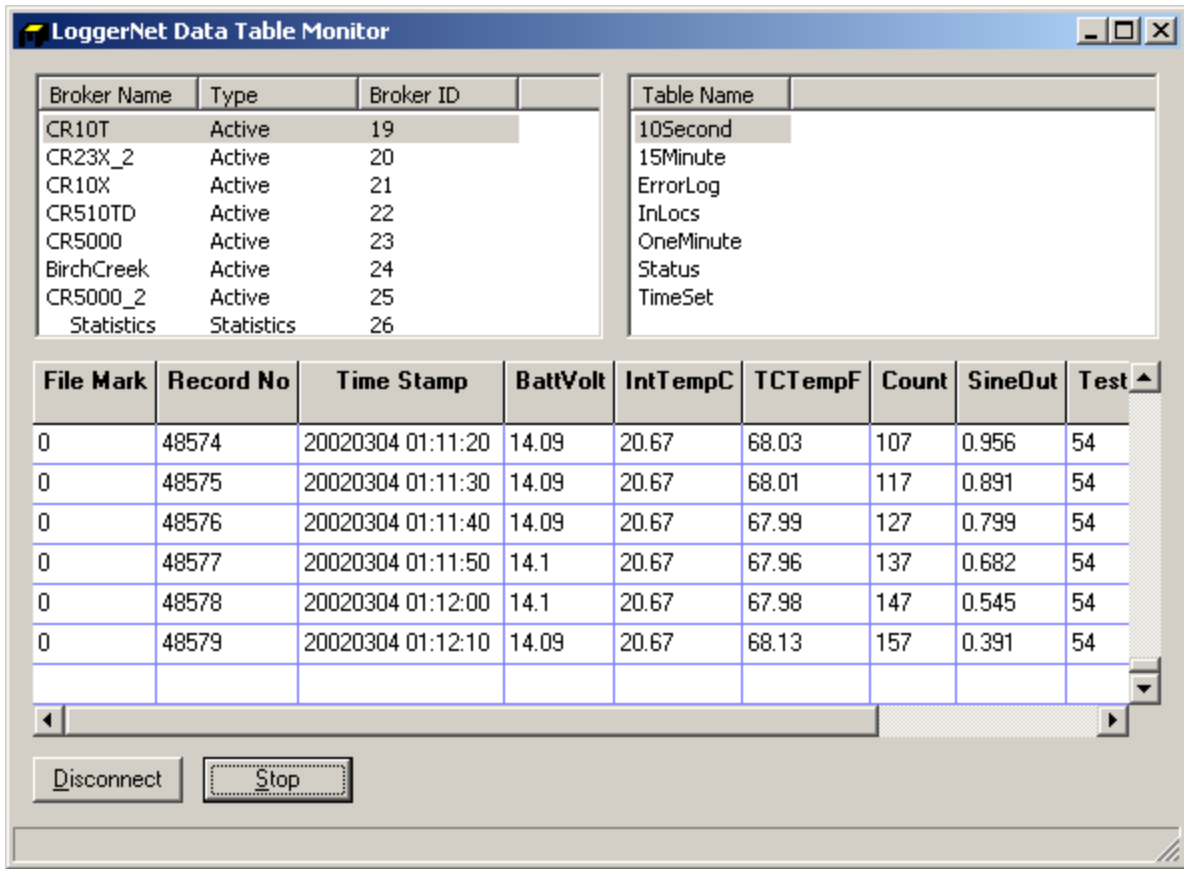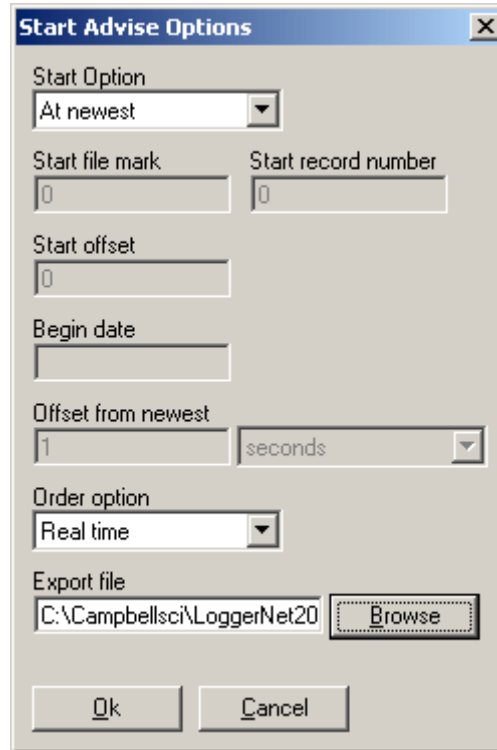
Click on the Connect button to connect to the LoggerNet server. The dialog box shown below will be displayed. If you are working on the same computer where LoggerNet is running leave the default Server Host Address as localhost. The Server Port number should also be 6789. The Logon Name and Logon Password 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 on OK a list of the dataloggers in the network will be shown in the upper left window.
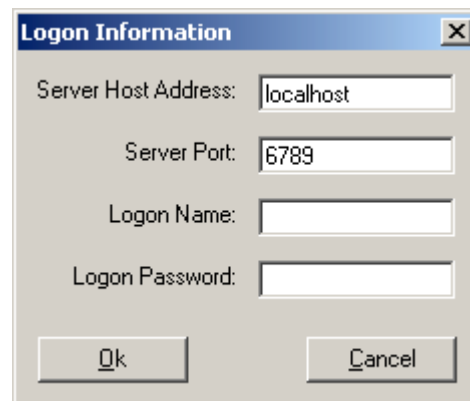
Selecting a datalogger will list the names of the data tables or array IDs in the datalogger. Note that if data collection has not been set up and enabled in the

14-13

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 on 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.

- 1 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.

- 2 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.

- 3 At Newest: This option will set the starting position to the last record stored in the data cache. This last record and any future records stored will be output.

- 4 After Newest: This option will set the starting position to be the next record stored in the data cache. Output begins with the next record stored in the data cache. No historical records will be output.

- 5 Relative to Newest: 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.

- 6 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 on the OK button to start writing to the file. The records are always displayed in the list box on the bottom of the screen. If you have set up an output file they are also sent to the output file.

Ordering Option:

- 1 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.

- 2 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.

- 3 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.

- 4 Real Time: the most recent data is always sent out starting with the last record stored. This will not provide a complete data set.

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 on OK to begin displaying the data.

Section 15. Implementing Advanced Communications Links

This section describes the configuration and operation of a variety of communications links. The communications links included here require special setup or configuration, or require special consideration in the implementation to work properly.

NOTE	Refer to Section 5.1 if you need general information on adding devices to the device map.

15.1 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.

15.1.1 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.

15.1.2 Operational Considerations

15.1.2.1 Extra Response Time

LoggerNet is pre-programmed to expect certain response times from devices depending on the type of intermediate devices and the communication rates chosen. If the datalogger network communications link is marginal, it may take longer to negotiate communication between the devices. Extra response time added to one or more of the devices may help to prevent the software from timing out. Note that the extra response times added for each device are cumulative for the entire communications link, and the total response time includes the default times plus any extra response time that has been added.

15.1.2.2 RF Address

The hardware settings for the address of the RF base must be 255 for phone to RF operation. Additionally, ensure that the addresses set for the RF remote hardware match the settings defined in Setup.

15.2 Phone to MD9

15.2.1 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.

15.2.2 Operational Considerations

15.2.2.1 MD9 Addresses

The address for an MD9 base device is set in the LoggerNet communications software at 255. The hardware configuration in the MD9 base modem must match for successful communications (refer to your MD9 users manual for information on setting the hardware switches within the device). In addition,

the address specified in Setup for the MD9 remote modem must match its hardware configuration.

15.2.2.2 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.

15.2.2.3 Grounding

Depending on the configuration and distance of the MD9 network, be sure to follow the grounding guidelines provided in the MD9 hardware manual. Grounding issues have been known to prevent reliable communications and data collection.

15.3 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.

15.3.1 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.
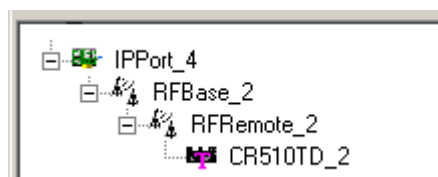
To begin, add an IPPort to the device map if one does not exist. Add an RF base modem to the IPPort, and to this, add the remote RF modem. To complete the network, add your datalogger to the remote RF modem. Review all of the settings for each device, and make any changes to customize the settings for your network configuration.

15.3.2 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.

15.3.3 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.

The RF base has a CSI I/O serial port instead of the RS-232 standard for communications. The NL-100 interface has a built-in CSI I/O port so no special cabling is required.

When connecting the RF Base to a third-party Internet Serial Server 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.
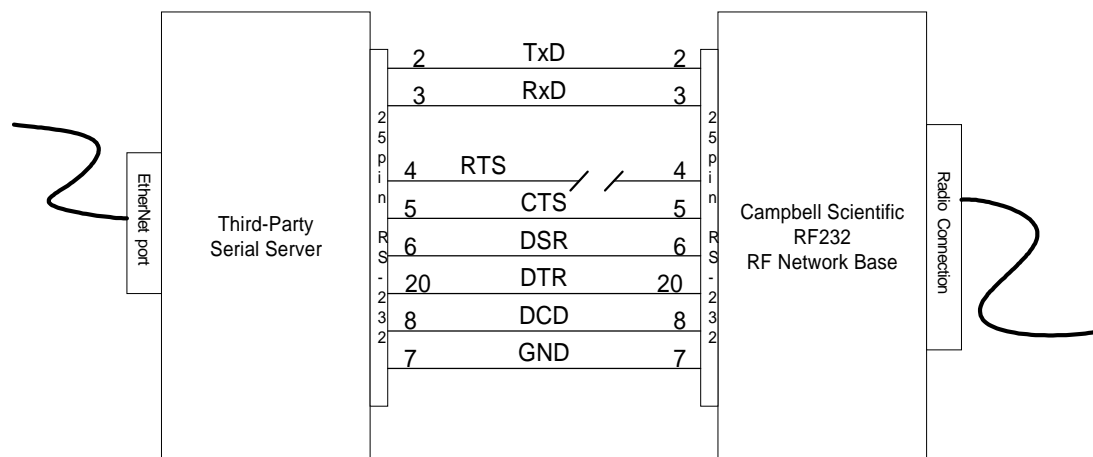
The serial server must be configured for this application before operation. The basic settings that must be configured are listed below. Depending on the specific serial server, there may be other settings that must be configured for proper operation.

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.

Appendix A. Glossary of Terms

A

Advise – See Data Advise

ASCII File - A computer file containing letters, numbers, and other characters using the ASCII character encoding.

Asynchronous - The transmission of data between a transmitting and a receiving device occurs as a series of zeros and ones. For the data to be "read" correctly, the receiving device must begin reading at the proper point in the series. In asynchronous communications, this coordination is accomplished by having each character surrounded by one or more start and stop bits that designate the beginning and ending points of the information (see Synchronous). The transfer of information is not otherwise coordinated between the sender and receiver.

Analog Channel - A terminal on the datalogger's wiring panel where leads for analog signals are connected. The analog channels are designated single-ended (SE) or differential (DIFF) on the wiring panel. Many sensors, such as thermistor temperature probes and wind vanes, output analog signals.

Array-based Datalogger - Array-based dataloggers save all output in the datalogger's final storage memory. When data is directed to final storage, a unique array ID number is stored, followed by other values as determined by the datalogger program. These are called "elements". Array-based dataloggers save all information that is directed to output storage to the same area of datalogger memory (as opposed to table-based dataloggers that store different output processing to separate tables in datalogger memory). Data retrieved by PC software is separated based on the array IDs.

B

Baling – The process by which the Baler produces data files containing the information collected by the LoggerNet Server during a user specified interval.

Batch Files - An ACSII text file that contains one or more DOS commands or executable file commands. When the batch file is run, the commands in the file are executed sequentially.

Battery - This entry in the status table returns the datalogger battery voltage.

Baud - The rate at which a communication signal travels between two devices.

Binary File - A file based on software defined formatting. A binary file can only be interpreted by the software programmed to decode the formatting. This format is used for more efficient data storage than is provided by ASCII.

BMP (Block Mode Protocol) – The communications protocol used by the server to communicate with table-based dataloggers and RF modems.

Broadcast – Part of the radio (RF) technique of polling remote radio modem datalogger sites. A single modem sends a message (broadcast) that all affected remotes hear and respond to.

C

Call-back - When a datalogger is programmed for Call-back, it will automatically call the host computer when a specified condition is met. The computer must be set up to look for such an incoming call.

Call-back ID Number - A three-digit number that is used to identify what datalogger has called the host computer. (Not available for Table-based dataloggers.)

Cancel - Choosing Cancel from a dialog box will ignore any changes made and close the box.

Carrier – An electrical signal used to convey data or other information. For example, radio and phone modems use carrier signals. Phone modems attempt to detect carrier when the call is placed. The red LED on the RF95T lights when the modem detects a carrier.

Child Node – See Node. A node that is accessed through another device (parent node). For example a remote radio (RF) site is accessed through the base RF232T. All nodes are child nodes of the PC.

Client – a software application designed to connect to a server. Usually provides some type of user interface or data acquisition.

Coaxial cable – Special type of cable with two conductors (center conductor and outer shield conductor). Classified by size, impedance, and loss characteristics. Used to connect MD9 modems and to connect radios to antennas.

Collection - (see Data Collection)

COM Port - A computer's serial communications port. Cables and other interface devices are connected between the computer's COM port and the datalogger.

Control Port - Dataloggers have digital output ports that can be used to switch power to sensors such as the HMP35C relative humidity circuit or to control relays. These digital outputs are called Control Ports and are labeled C1, C2, etc., on the wiring panel. Control ports on some dataloggers can also be used as inputs to sense the digital (high or low) state of a signal, or used as data input/output connections for SDI-12 sensors.

CoraScript – This is a command line interpreter that allows the user access to many of the capabilities of the LoggerNet server using direct commands or programmed script files.

CR10X-TD Family of Dataloggers - This includes the table-based dataloggers, specifically the CR10T, CR510-TD, CR10X-TD, and CR23X-TD.

CRBasic - The programming language used for CR5000/9000 dataloggers. The CRBasic editor is used to create the program files for these dataloggers.

D

Data Advise (Datalogger) – A mutual agreement between the server and the datalogger about which tables are to be collected every time the datalogger is contacted. Based on the dataloggers table definitions.

Data Advise (Server) – an agreement between a client application and the server to provide specified data as it is collected by the server.

Data Advise Notification – The packet of data sent by the datalogger based on the Data Advise agreement.

Data Cache – The storage for data collected from the datalogger by the LoggerNet server. This data is stored in binary files on the hard disk of the computer where the server is running.

Data Collection - Getting stored data from the datalogger and saving it in the communication server's data cache (compare to Data Retrieval).

Data Point - A data value which is sent to Final Storage as the result of an Output Instruction. A group of data points output at the same time make up a record in a data table.

Data Retrieval - Extracting data from the communication server's data cache to export to a file, network, or data display (compare to Data Collection).

Data Storage Table, Data Table - A portion of the datalogger's Final Storage allocated for a particular output. Each time output for a given data table occurs, a new record is written to the table. The size of the table (in number of records) and when records are written to the data table are determined by the datalogger's Data Table Instruction (P84). The fields (columns) of the table are determined by the Output Processing Instructions that follow the Data Table Instruction.

Data Table Instruction - Instruction 84. Used to create a Data Table and to cause records to be written to the Data Table.

DaysFull - This field in the status table shows the number of days remaining before any of the tables using automatic record allocation are filled.

Differential Analog Input - Some sensors have two signal wires and the measurement is reflected in the voltage difference between them. This type of sensor requires two analog connections. The channels marked DIFF on the datalogger wiring panel are used to connect differential sensors.

DLD File - The DLD file is an ASCII file that can be sent to program the datalogger. Dataloggers must be programmed to perform measurements, convert data to final units, and to save data for retrieval. Edlog is used to create these files that are saved to disk with a DLD filename extension. A program must be sent to the datalogger before the datalogger will begin to collect data.

E

Edlog - Campbell Scientific's editor application to create new or edit existing datalogger programs.

EEPROM - Electrically erasable read only memory. The memory CR10X-TD, CR510-TD, and CR23X-TD dataloggers use to store their operating system. A new operating system can be transferred to the datalogger using a special software package (see PROM).

Execution Interval - The periodic interval on which the datalogger program is run. The execution interval is sometimes referred to as the Scan Interval. For example, when an execution interval of 60 seconds is set, the datalogger will execute its program table every 60 seconds. Between executions the datalogger enters a sleep (quiescent) mode. This conserves battery power and creates predictable measurement intervals. The execution interval is synchronized with the datalogger's real-time clock.

Execution Time - The time required to execute an instruction or group of instructions. If the total execution time of a Program Table exceeds the table's Execution Interval, the Program Table will be executed less frequently than programmed. Each time this occurs, a Table Overrun occurs. Table Overruns are considered to be "errors" and are reported in the datalogger status information table.

Excitation Channel - Many sensors require a precise electrical voltage to be applied. The excitation channels, marked as E1, E2, etc., on the datalogger wiring panel, provide this required voltage.

F

Fault – Message relating to network activity where repeated problems or errors have occurred. Repeated faults usually indicate a failure of some kind.

F1 - In most instances, pressing the F1 key will provide context sensitive help for the highlighted object on the screen.

Final Storage - Final Storage is an area in the datalogger's memory where data is stored for collection to a PC. When you collect data from the datalogger you are collecting data from a Final Storage table.

Flag - Memory locations where the program can store a logical high or low value. These locations, called User Flags, are typically used to signal a state to another part of the program.

G

Ground Connection - Most sensors require one or more ground connections in addition to excitation or signal inputs. Ground connections may serve any of several purposes:

- a reference for a single-ended (SE) analog voltage

- a power return path

- a connection for cable shield wire to help reduce electrical noise

H

Highlight – Text or objects can be highlighted, by positioning the cursor where you want the highlight to begin, holding the left mouse button, and dragging it across the words or group of objects to be highlighted. A single

object can be highlighted, by clicking it once with the left mouse button. Highlighted items can then be edited or activated.

Holes – Because Data Advise always get the most recent data records, there can be sequences of older data available from the datalogger that have not yet been collected to the data cache. The server tracks and optionally collects these holes. This entry in the status table shows the number of data points in missed records for the data storage tables in that station.

Hole Collection – The process used by the server to collect from the datalogger, data records missing from the data cache. If Hole Collection is delayed or disabled, the memory in the datalogger can ring around and overwrite the missing data records resulting in an Uncollectable Hole.

Host Computer - The machine where the LoggerNet communication server software is running.

I

INI Files - Configuration files that are used to preserve the last known setups of a program or device.

Initialization String - A string of alphanumeric characters that are sent to a device, such as a modem, to prepare that device for communications.

InLocs - Abbreviation for "Input Locations". This entry in the status table shows the number of input locations allocated for the program.

Input Location Storage - Each time a measurement or calculation is performed the resultant value is stored in an Input (memory) Location, sometimes abbreviated as "InLoc."

Input/Output Instructions - Datalogger program instructions used to make measurements or send data automatically to other devices.

Intermediate Storage - Datalogger memory used to temporarily store values, typically to be used for output calculations. The datalogger uses Intermediate Storage to accumulate sensor readings until output.

L

Link – Communications route between two devices, for example the phone link between two phone modems.

Log Files - Log files are text files that are stored on the host computer's hard drive. They contain information about communications between the LoggerNet server and other devices in the datalogger network. Log files are typically used for troubleshooting purposes. LoggerNet has four types of log files: Communications Status, Object State, Transaction, and Low Level I/O. Refer to Appendix D for information on these log files.

M

MD9 - An MD9, or multi-drop modem, is a communications device that uses twisted pair cable for connection. Typically, the system consists of one MD9 base modem that is attached to the user's computer, with one or more remote modems at the datalogger field site. One remote modem is needed for each datalogger at the field site.

Measurements - Measurements are what the datalogger stores in an Input Location after reading an electronic signal from a sensor and converting the raw signal into meaningful units.

Modem - A device used to transmit and receive digital data over normally analog communications lines, usually as an audio signal on telephone circuits. A modem attached to a computer performs a digital-to-analog conversion of data and transmits them to another modem which performs an analog-to-digital conversion that permits its attached computer to use the data.

Monitor - A computer's CRT, or display, is referred to as a monitor.

N

NetAdmin – Network Administration program, used to create and edit network descriptions, and to display/modify the device settings.

Net Description – Description of dataloggers and communications devices that form the datalogger network. Entered using NetAdmin and used by the server to communicate with the various dataloggers.

Node – Part of the description of a datalogger network. Each node represents a device that the server will dial through or communicate with directly. Nodes are organized as a hierarchy with all nodes accessed by the same device (parent node) entered as child nodes. A node can be both a parent and a child node.

O

ObjSrlNo - This entry in the status table provides the revision number of the datalogger PROM.

Output Interval - The output interval is the interval in which the datalogger writes data to Final Storage. The output interval is defined by Instruction 84 in Edlog (for table-based dataloggers).

Output Processing - Writing to final storage memory a sample or summary statistic of data measurements. Output processing options include sending a sample, average, maximum, minimum, total, or wind vector of data to Final Storage. Each Output Processing data value is kept in a separate location within the datalogger. This allows multiple output processing for each measurement. For example, you can average air temperature over a 60-second interval, a one-hour interval, and a 24-hour interval.

Overrun Errors - Overrun errors occur when the actual program execution time exceeds the execution interval. This causes program executions to be skipped. When an overrun error occurs, the Table Overrun parameter in the datalogger's status table is incremented by 1.

Overruns - This entry in the status table provides the number of table overruns that have occurred. A table overrun occurs when the datalogger has insufficient time between execution intervals to complete one pass through the program. This counter is incremented with each table overrun.

P

Packet – a unit of information sent between two BMP devices that are communicating. Each packet can contain data, messages, programming, etc. Usually contains addressing and routing information.

Parameter - Numbers or codes which are entered to specify exactly what a given datalogger instruction is to do.

Path – The modems, or other devices that make up a link to communicate with a remote site datalogger.

Polling – Process where a datalogger or other communications device is periodically checked for any packets it needs to send. The server polls dataloggers for most communications links. With Radio (RF) the RF232T base or repeaters can poll datalogger sites.

Polling Interval – The user-specified interval that determines when to poll a given device.

PrgmFree - An entry in the status table that shows the amount of remaining program memory, in bytes.

PrgmSig - An entry in the status table that shows the signature of the datalogger program. The signature is a unique number derived from the size and format of the datalogger program

PromID - An entry in the status table that shows the version number of the datalogger PROM or OS.

PromSig - An entry in the status table that shows the signature of the datalogger PROM or OS. As with the PrgmSig, if this signature changes, the datalogger instruction set has somehow been changed.

Processing Instructions - Datalogger instructions that further process input location data values and return the result to Input Storage where it can be accessed for output processing. Arithmetic and transcendental functions are included in these instructions.

Program Control Instructions - Datalogger instructions that modify the sequence of execution of other instructions in the datalogger program; also used to set or clear user flags.

Program Signature - A program signature is a unique value calculated by the datalogger based on program structure. Record this signature in a daily output to document when the datalogger program is changed.

Program Table - The area where a datalogger program is stored. Programming can be separated into two tables, each having its own execution interval. A third table is available for programming subroutines which may be called by instructions in Tables 1 or 2. The length of the tables is constrained only by the total memory available for programming.

PROM - Programmable Read-Only Memory. PROM integrated circuit chips are used to store the datalogger Operating System (OS) in the CR10T datalogger. The PROM can be replaced to install a new operating system (see EEPROM).

Pulse Channel - Some sensors output voltage pulse signals. Such sensors can be connected to Pulse Channels for measurement (labeled as P1, P2, etc., on the datalogger's wiring panel).

Q

Quiescent Mode - Often referred to as "sleep mode". The datalogger is in a low power state between program execution intervals.

R

Real-Time Clock - All dataloggers have an internal clock. The date and time information from this clock are used in the time stamp for stored data. The datalogger's execution interval and timer are synchronized with the clock. The CR10X-TD, CR510-TD, and CR23X-TD have battery backups which maintain the clock even when 12 V power is not available.

Record - A group of data values output at the same time to the same data table. Records are written in response to the Data Table Instruction (84). The individual fields within each record are determined by the Output Processing instructions following the Data Table Instruction that created the data table.

RecNbr - An entry in the status table that shows the record number in the table.

Remote Site – Usually refers to the site where datalogger is located at the other end of a communications link. Also can refer to the site where a radio (RF) repeater is located.

Repeater – a radio (RF site that relays packets of information to a remote site. Used to extend the range of radio transmissions. Any remote datalogger site with radio can act as a repeater.

Retries – When a transaction or communication between two devices or programs fail, the transaction or communication is usually repeated until it succeeds.

Retrieval - (see Data Retrieval)

RF – Dealing with radio telemetry. Stands for Radio Frequency.

RTMS - Real Time Monitoring Software. A software application designed by Campbell Scientific for fast real time data acquisition. RTMS was designed for IBM's OS/2 PC operating system and replaced by LoggerNet.

S

Scan Interval - See Execution Interval.

SDI-12 - SDI-12 stands for Serial Digital Interface at 1200 baud. It is an electrical interface standard and communications protocol that was originally developed by Campbell Scientific and other manufacturers for the U.S. Geological Survey for hydrologic and environmental sensors. SDI-12 was designed to be a simple interface (ground, 12 volts, and signal) that improves compatibility between dataloggers and "smart" microprocessor-based sensors.

Other goals of the SDI-12 standard are:

- low power consumption for battery powered operation via the datalogger

- low system cost

- use of multiple sensors on one cable connected to one datalogger

- allow up to 200 feet of cable between a sensor and a datalogger

Security Code - A four-digit code entered into the datalogger to prevent unauthorized access to datalogger settings, programs, and data.

Server – Refers to a software application that accepts connections from client applications and provides data or other information as requested. The LoggerNet server manages all the communications and data collection for a network of dataloggers. The collected data is made available for client applications.

Signature – Number calculated to verify both sequence and validity of bytes within a packet.

Single-ended Analog Input - Some analog sensors have only one signal wire. (They will also have another wire that can be grounded and that is used as the reference for the signal wire.) With this type of sensor, only one analog connection is required. Hence, it needs a "single-ended" or SE analog input. The single ended channels are marked as SE on the datalogger wiring panel.

Socket Data Export – a software application that connects to the LoggerNet server and provides a TCP/IP socket for a user created application to receive data records from the server data cache.

Station - A datalogger site is often referred to as a station.

Station Number – The LoggerNet server assigns and uses station numbers for routing packets to the dataloggers. These numbers can be modified using CoraScript.

Storage - An entry in the status table that shows the number of final storage locations available.

Synchronous - The transmission of data between a transmitting and a receiving device occurs as a series of zeros and ones. For the data to be "read" correctly, the receiving device must begin reading at the proper point in the series. In synchronous communications, this coordination is accomplished by synchronizing the transmitting and receiving devices to a common clock signal (see Asynchronous).

T

Tab Windows - Some screens depict a series of related windows in a multi-tabbed notebook format. When you click on the file folder tab, the information on the tab you chose will be displayed.

Tables - An entry in the status table that shows the number of user-created data tables. (See also Data Table.)

Table-based Dataloggers - Table-based dataloggers store each record of data that follows an output instruction in a table. Each separate occurrence of an output instruction directs the datalogger to store the data in a separate table. Refer to Appendix B for additional information on table-based dataloggers.

Table Definitions – List of data available from a datalogger. The datalogger supplies this list on request. The tables are based on the datalogger programming. The LoggerNet server must have a current version of the table definitions to collect data from the datalogger.

Throughput - The rate at which a measurement can be made, scaled to engineering units, and the reading stored in Final Storage.

Time Stamp - The date and time when data are stored in the datalogger.

TMStamp - An entry in the status table that shows the date and time the status information was recorded.

Transaction - The exchange of data or information between two devices or programs. For example setting of the clock in a datalogger is a transaction between the server and the datalogger.

U

Uncollectable Hole – Occurs when a hole in the data cache cannot be collected from the datalogger before the data table wraps around and the records are overwritten.

V

Variable Name - Edlog uses variable names in expressions. Variables are another name for input location labels. For instance, in the equation TempF = (TempC*1.8) + 32, TempC is an input location label and TempF is a new location calculated from TempC.

W

Wiring Panel - The set of terminals and underlying circuits that enable connections of sensors, control and power supply wiring to the datalogger itself. Some dataloggers such as the CR23X-TD have built-in wiring panels. Others, such as the CR10X-TD, have removable wiring panels.

Watchdog - An entry in the status table that shows the number of watchdog errors that have occurred. The watchdog checks the processor state and resets it if necessary. If an error occurs, the watchdog error counter is incremented.

Appendix B. Table-Based Dataloggers

This section describes some of the characteristics and features of the CR10X-TD family and CRx000 family of table-based dataloggers. The dataloggers included in these families are CR510-TD, CR10T, CR10X-TD, CR23X-TD, CR5000 and CR9000.

B.1 Memory Allocation for Final Storage

The datalogger memory includes four important areas: the datalogger program storage, input storage, intermediate storage, and final storage. When a program is downloaded to the datalogger and compiled, datalogger memory is allocated for each of these areas.

The CR10X family of array-based and table-based dataloggers are identical in hardware and differ only in the operating system. The primary distinction between array-based and table-based dataloggers is how final storage is allocated and filled. CRx000 family of dataloggers are based on CRbasic programs and have a different memory allocation structure.

B.1.1 CR10X-TD Family Table-Based Dataloggers

CR510-TD, CR10T, CR10X-TD, and CR23X-TD table-based dataloggers store data from different intervals in different final storage tables. Final storage tables are made up of records and fields. Each row in a table represents a record and each column represents a field. The number of fields in a record is determined by the output processing instructions in the datalogger program that follow the Data Table output instruction (P84 Output Table; refer to your datalogger user's manual for more information). The total number of fields for each table will be the number of output processing instructions multiplied by the number of values stored by each of the output instructions.

The number of records to be kept in a table before the oldest data is overwritten can be fixed by the user, or left for the datalogger to determine automatically. With automatic allocation the datalogger tries to set the sizes of automatically allocated tables such that all of the tables will fill up at about the same time. Once the sizes of the tables are determined, the datalogger allocates available final storage to these tables.

Note that the tables are allocated by size with the smallest tables first. For dataloggers with extended flash memory, any tables that will not fit in SRAM memory are allocated to flash memory. Flash memory is allocated in 64 K blocks; therefore, even a very small table will take 64 K of flash memory. If the table sizes specified by the user exceed the amount of memory available for that purpose in the datalogger, an error will occur when the program is compiled by the datalogger.

NOTE	Event driven tables should have a fixed size rather than allowing them to be allocated automatically. Event driven tables in CR10X-TD type dataloggers that are automatically allocated are assumed to have one record stored per second in calculating the length. Since the datalogger tries to make the tables fill up at the same time, these event driven tables will take up most of the memory leaving very little for the other, longer interval, automatically allocated data tables.

B.1.2 CR5000/CR9000 Memory for Programs and Data Storage

The datalogger memory for the CR5000 and CR9000 is divided between Random Access Memory (RAM) and Electrically Erasable Programmable Read Only Memory (EEPROM). The EEPROM, or flash memory, is used to store the operating system and the user programs that have been saved in the datalogger. When the datalogger powers up, the program marked as "Run on Power-up" is transferred to RAM and executes from there. Additional storage is available using PCMCIA cards.

When a datalogger program is sent to the datalogger, it is divided into two tasks that run simultaneously. All of the program instructions that deal with measuring sensors, controlling outputs, or are time sensitive, are placed in the **measurement task**. The instructions that deal with data processing, including calculations, data storage, averaging, minimum and maximum tracking, and data I/O operations, are placed in the **processing task**.

The measurement task is executed at the precise specified scan rate and stores the raw data into a memory buffer. As soon as the measurement task has completed filling the buffer for the current scan, the processing task starts the data processing on the buffered data. There are at least two memory buffers, allowing the measurement task to fill one buffer while the processing task is working with the data in the other.

The data processing task stores data as records in final storage data tables. LoggerNet can collect the records from these data tables either manually or with scheduled data collection. The datalogger program can also make some or all of the variables used for measurement storage or calculations available to LoggerNet. These variables are found in the Public table, which is similar to the Input Location table in CR10X or CR10X-TD type dataloggers.

Final storage tables are made up of records and fields. Each row in a table represents a record and each column represents a field. The number of fields in a record is determined by the number and configuration of output processing instructions that are included as part of the Data Table definition.

The number of records to be kept in a table before the oldest data is overwritten can be limited by the user, or left for the datalogger to determine automatically. The datalogger tries to set the sizes of automatically allocated tables such that all of the tables will fill up at about the same time. Once the sizes of the tables are determined, the datalogger allocates the available memory to these tables.

If the amount of memory requested for the data tables exceeds the available memory, the program will not run.

NOTE	Event driven tables should have a fixed size rather than allowing them to be allocated automatically. Event driven tables in CR5000/CR9000 dataloggers that are automatically allocated are assumed to have one record stored per execution interval in calculating the length. Since the datalogger tries to make the tables fill up at the same time, these event driven tables will take up most of the memory leaving very little for other, longer interval, automatically allocated data tables.

B.2 Converting an Array-Based Program to a CR10X-TD Table-Based Program using Edlog

The following information is provided for those users familiar with writing programs for array-based dataloggers or users who have existing array-based datalogger programs that need to be changed to table-based programs.

B.2.1 Steps for Program Conversion

If you are converting a program for the same series of datalogger (e.g., a CR10X program to a CR10X-TD program) you can edit the existing program in Edlog. If you are converting a program from one datalogger series to another (e.g., CR10X to CR23X-TD), you may need to start the program from scratch.

To convert a program for the same series of datalogger:

1. Open the CSI file in Edlog.

2. The first line of the file will read

    ;{CR10X}

    Change this line to

    ;{CR10X-TD}

3. **Review all of the instructions provided in the section below**. If any of these are included in your program, format them as a comment or delete them from the program.

4. Save the file to a new file name, but **do not compile the file when prompted**.

5. Open the newly created file in Edlog. It will be opened using the CR10X-TD datalogger template instead of the CR10X. Make any changes necessary to replace the commented or deleted instructions.

6. Save and compile the program, correcting any errors that may be found by
    the compiler.

B.2.2 Program Instruction Changes

Several programming instructions have changed or are not used in table-based
datalogger programs. Make sure you "comment out" any of these instructions
before you try to convert the array-based program. These are listed below:

- Check any instructions that may set the Output Flag (Flag 0) high or low
  by using the Command Code Options. The output flag is not used in table-
  based programming. Instructions that may include reference to the output
  flag are: P83, If Case; P86, Do; P88, If (X< = > Y); P89, If (X< = > F);
  P91, If Port/Flag; and P92, If Time.

  If any of these instructions set the output flag high, the instruction can be
  replaced with Instruction 84, Table Data. Instruction 84 is used to define a
  table of final storage data. New records of data are stored in the table-
  based on time (interval data) or when a user flag is set (event data). Time
  based output intervals are specified in seconds.

- **Instruction 18, Time** - Instruction 18 is used to store the current time into
  an input location. Parameter 1 designates what format will be used when
  storing the time. There are differences in this instruction's Parameter 1 for
  the two datalogger types.

- **Instructions 73 and 74, Maximum and Minimum** - These instructions
  are used to store the maximum or minimum for a value over a period of
  time. Parameter 2 in these instructions is used to designate a time option.
  There are differences in the instructions' Parameter 2 for the two
  datalogger types.

- **Instruction 77, Real Time** - Instruction 77 is used to store the current
  time in final storage for array-based dataloggers. This instruction is not
  included in table-based dataloggers, since the time is assigned to records
  automatically when data is retrieved (see Section 5.3 for information on
  how data is time stamped).

- **Instruction 80, Set Active Storage Area** - Instruction 80 is used to direct
  output processing to final storage area 1, final storage area 2, or an input
  location. This instruction is not included in the table-based programming
  instructions. Output processing can be redirected to input locations in a
  table-based datalogger using Instruction P84, Table Data (see Edlog's
  help).

- **Instruction 92, If Time** - Instruction 92 is used to perform one or more
  actions based on time. The interval for table-based dataloggers is in
  seconds only; array-based dataloggers offer the options of seconds or
  minutes. The instruction for array-based dataloggers defaults to minutes,
  so if you are using this instruction it may need to be changed.

  Also, check any Instruction 92s for Command Codes that may affect the
  output flag (see discussion above on output flag instructions).

- **Instruction 96, Serial Output** - Instruction 96 is used to send data in the active Final Storage area to a storage module, computer, printer, or alternate final storage area. This instruction is not included in the table-based programming instructions.

- **Instruction 97, Initiate Telecommunication** ("Call-back") - Instruction 97 is used to program the datalogger to call a remote computer, voice modem, or other datalogger when a user flag is set low. This instruction is not included in the CR10T datalogger. Newer operating systems for the other dataloggers do support instruction 97.

- **Instruction 98, Send Printer Character** - Instruction 98 is used to send characters to either an addressed or pin-enabled printer. This instruction is not included in the table-based programming instructions.

- **Conditional Data Output** – check to make sure that the output data is not being output conditionally. Table-based dataloggers require that the size of the output record is constant. Any instructions that dynamically change the number of data values in a record or the size of the record need to be removed. (e.g., don't change data resolution from low to high based on a conditional.)

B.3 Table Data Overview

In the datalogger all data is organized into tables with fixed data records. Each of these tables has a definite number of records that is either fixed by the datalogger program or allocated when the program is compiled by the datalogger. Once the maximum number of records for a table have been stored, the next record stored will overwrite the oldest record in the table.

Tables that are automatically allocated in the datalogger program are allocated a number of records based on the time interval for the records. The datalogger attempts to allocate these tables so that all of the automatically allocated tables fill up at the same time. For example a tables with records stored every 30 minutes and 60 minutes would have twice as many records allocated for the 30-minute table.

NOTE

Event driven tables should have a fixed size rather than allowing them to be allocated automatically. Event driven tables in the CR10X-TD type dataloggers that are automatically allocated are assumed to have one record stored per second in calculating the length.

In CR5000/CR9000 dataloggers event tables are assumed to have one record stored per execution interval.

Since the datalogger tries to make the tables fill up at the same time, these event driven tables will take up most of the memory leaving very little for the other, longer interval, automatically allocated data tables.

Within a data table, data is organized in records and fields. Each row in a table represents a record and each column represents a field. To understand the concept of records it may be helpful to consider an example.

Example:
A CR10X-TD is to be used to monitor three thermocouples (TC). Each hour a temperature for each of the three thermocouples is to be stored. The table has five fields: DATE_TIME, RECORD #, TEMP1, TEMP2, TEMP3.

The program is written so that each hour an Instruction 84, Table Data, generates a new "record" in the data table. This hourly table would then be organized as follows:

DATE_TIME	RECORD #	TEMP1	TEMP2	TEMP3
2002-01-27 10:00:00	14	23.5	24.6	28.2
2002-01-27 11:00:00	15	24.2	22.4	23.4

Only the hourly data triggered by the Instruction 84 above would be written to this table. If other table data instructions existed, the output for these tables would be written to their own tables.

Data tables can also be event driven rather than interval driven. That is, a new record is stored when a specified event occurs rather than based on time.

Each table is completely independent of any other tables and all records in a given table have the same number of fields.

B.4 Default Tables

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

- **Timeset Table** - The Timeset table contains a history of clock sets for the datalogger. It includes three fields: TimeStamp, RecordNumber, and OldTime. TimeStamp is the time and date the clock was set. RecordNumber is incremented each time the clock is set. When the datalogger is reset or a new program is loaded, RecordNumber is reset to 1. OldTime is the datalogger's clock value before the time was set (CR10X-TD family dataloggers only).

- **Errorlog Table** - The Errorlog table contains any errors that occur in the datalogger. It includes three fields: TimeStamp, RecordNumber, and ErrorCode. TimeStamp is the time and date the error occurred. RecordNumber is incremented each time an error occurs. When the datalogger is reset or a new program is loaded, RecordNumber is reset to 1. ErrorCode is the code returned by the datalogger when an error occurs. Refer to the datalogger user's manual for a list of all error codes (CR10X-TD family dataloggers only).

- **Inlocs Table** (CR10X-TD family dataloggers) or **Public Table** (CR-x000 dataloggers) - When a datalogger measures a sensor, the sensor reading is

stored in a temporary register called an input location or variable. With each new measurement, the old value is overwritten by the new value. The Inlocs or Public table contains a time stamp, record number, flag status, port status, and the reading from each sensor scanned or user created input locations.

- **Status Table** - The Status table contains information on the datalogger. Data is written to the table with each datalogger program execution. Note that the actual fields contained in the table are datalogger-specific. Table B-1 below describes each of the fields in the table for a CR10T. CR10X-TD, CR510TD and CR23X-TD status tables are similar. See the datalogger operator's manual for specifics. Table B-2 describes the fields for the CR5000 datalogger. CR9000 is similar.

Table B-1. Example of Status Table Entries (CR10T)	
TMStamp	Date and time the status information was recorded.
RecNBR	The record number in the table.
Battery	Datalogger battery voltage.
Watchdog	The watchdog checks the processor state, software timers, and program related counters. If an error occurs, the watchdog counter is incremented.
Overruns	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.
InLocs	Number of input locations allocated for the program.
PrgmFree	Amount of remaining program memory, in bytes.
Storage	Number of final storage locations available.
Tables	Number of user-created data tables.
DaysFull	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.)
Holes	Number of missed records in all data storage tables.
PrgmSig	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.
PromSig	Signature of the datalogger PROM. As with the PrgmSig, if this signature changes, the datalogger instruction set has somehow been changed.
PromID	Version number of the datalogger PROM.
ObjSrlNo	Revision number of the datalogger PROM.

Table B-2. CR5000 Status Table Entries	
ROMVersion	Version of the ROM code. This value is stored in the ROM and read by the OS at compile time.
OSVersion	Current version of the operating system.
OSItem	The CSI item number for the operating system.
OSDate	Date that the Operating System was compiled.
StationName	String stored as the Station Name of the CR5000.
ProgName	The Name of the currently running program.
StartTime	Time that the program began running.
Battery	Current value of the battery voltage. This measurement is made in the background calibration.
PanelTemp	Current Panel temperature measurement.
LithiumBattery	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.
CPUSignature	The Operating System signature. The value should match the value obtained by running the CSI sig program on the *name*.obj operating system file.
DLDSignature	Signature of the current running program file.
ProgSignature	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.
PC-CardBytesFree	Gives the number of bytes free on the PC-Card.
MemoryFree	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.
DLDBytesFree	Amount of free space in the CPU RAM disk that is used to store program files.
ProcessTime	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.
MaxProcTime	The maximum time required to run through processing for the current scan. This value is reset when the scan exits.
MeasureTime	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.

Table B-2. CR5000 Status Table Entries	
SkippedScan	Number of skipped scans that have occurred while running the current program.
SlowProcTime	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.
MaxSlowProcTime	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.
LastSlowScan	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.
SkippedSlowScan	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.
MeasureOps	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.
WatchdogErrors	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.
Low12VCount	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.
StartUpCode	A code variable that allows the user to know how the system woke up from poweroff.
CommActive	A variable signaling whether or not communications is currently active (increments each time the autobaud detect code is executed).
ProgErrors	The number of compile (or runtime) errors for the current program.
ErrorCalib	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.
VarOutOfBound	Flags whether a variable array was accessed out of bounds.
SkippedRecord	Variable that tells how many records have been skipped for a given table. Each table has its own entry in this array.
SecsPerRecord	Output interval for a given table. Each table has its own entry in this array.
SrlNbr	Machine specific serial number. Stored in FLASH memory.
Rev	Hardware revision number. Stored in FLASH memory.
CalVolts	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.

Table B-2. CR5000 Status Table Entries	
CalGain	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.
CalSeOffset	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.
CalDiffOffset	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.
CardStatus	Contains a string with the most recent card status information.
CompileResults	Contains any error messages that were generated by compilation or during run time.

Appendix C. Software Organization

C.1 LoggerNet/Client Architecture

The LoggerNet communication server provides the interface to all of the dataloggers and the support for the different communications mediums. It runs in the background and provides an attachment for the clients that provide the user interface. The server handles all communications with the dataloggers.

The LoggerNet server 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 some applications such as RTMC 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.

C.2 LoggerNet Server Data Cache

The LoggerNet server data cache is a set of files kept on the hard disk of the computer where the server is running. These data files are in binary format and can only be used or interpreted by the LoggerNet server. The data cache files are stored in addition to the output data files.

C.2.1 Organization

The data cache is set up to emulate the way data is stored in the datalogger. When a new datalogger station is defined for the network and communication is established with the station, the server requests the table definitions from 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.

C.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.

C.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. A server data advise can also be set up that will automatically send the selected data out to the client applications as it is collected from the dataloggers to the data cache. This improves efficiency since the client doesn't have to keep checking to see if new data has been collected.

C.2.4 Updating Table Definitions

When the table definitions are obtained from the datalogger they are kept in the server and used to identify the data available in the data cache. Every time new data is collected from a datalogger, a table definition signature is sent that should match the signature stored in the server. If this table definition signature doesn't match it indicates that the table definitions in the datalogger have changed.

There are a number of things that could cause datalogger table definitions to change. A new program may have been downloaded to the datalogger, or the keyboard display may have been used to manually make changes to the datalogger program.

NOTE	If the datalogger program is re-compiled without changing table definitions, the record numbers will reset to zero causing the server to assume the datalogger record numbers have wrapped around. This will result in the re-collection of all of the data in the datalogger.

When a change in table definitions is detected, the server stops data collection and indicates in 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.

C.3 Directory Organization
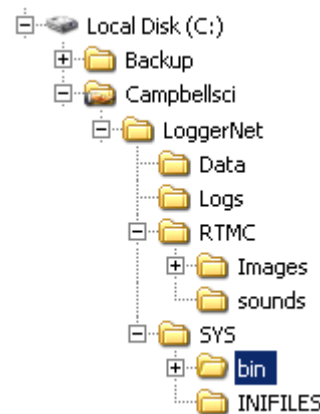
The default installation of the LoggerNet software installs two directories: the C:\CampbellSci\LoggerNet working directory and C:\Program Files\CampbellSci\LoggerNet program directory.

C.3.1 C:\CampbellSci\LoggerNet Directory (Working Directory)

The C:\CampbellSci\LoggerNet working directory includes 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 Inifiles 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.

C.3.2 C:\Program Files\CampbellSci\LoggerNet Directory (Program File Directory)
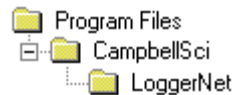
All files necessary for running the LoggerNet programs are stored in the C:\Program Files\CampbellSci\LoggerNet subdirectory.

No other files are saved to these program file subdirectories in order to protect the integrity of the LoggerNet communication server and the clients.

C.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 D. Log Files

D.1 Event Logging

As LoggerNet performs its work, it will create records of various kinds of events. The log files are very useful for troubleshooting problems and monitoring the operation of the datalogger network. See Section 7.2.10 for information on saving the logs to disk.

D.1.1 Log Categories

The LoggerNet server logs 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.

D.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.

D.1.3 Log File Message Formats

D.1.3.1 General File Format Information

The communications status, transaction, and object state logs all share the same basic file format. Each record in a log file ends with a carriage return and line feed. A single record will consist of two or more fields where each field is surrounded by quotation marks and separated by commas.

The two fields that will be present in all records are:

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.

D.1.3.2 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.

Message Text	Message Meaning	User Response to Message
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.

D.1.3.3 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)	

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
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.
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.

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
11	Clock check started		A clock check has been initiated. This clock check is not sent out to the station until the transaction is sent.	
12	Clock set	Device time before set; Server time;	The device clock has been set.	
13	Clock checked	Datalogger time	The datalogger clock has been checked.	
14	Clock check failed	Reason code: 3. Communication failure 4. Invalid datalogger security clearance 5. Invalid transaction number specified (already in use) 6. Communications are disabled for this device 7. The transaction was aborted by client request 8. The device is busy with another transaction	The clock check/set failed for the reason specified in the reason code.	Check the connections of the communication path to the datalogger, make sure the datalogger is connected and has power, check the security setting in the datalogger and in NetAdmin, check that communications are enabled in NetAdmin for all the devices in the path.
15	Starting BMP data advise transaction		A start data advise operation has been initiated. Data advise is not in place until the datalogger responds.	
16	Stopping BMP data advise transaction		A stop data advise operation has been initiated.	
17	BMP data advise transaction started		The message from the datalogger confirming the start of data advise has been received.	
18	BMP data advise transaction stopped		The message from the datalogger confirming the suspension of data advise has been received.	
19	BMP data advise transaction failed		The attempt to start or stop a data advise with the datalogger has failed or the operation has timed out waiting for a response.	Check communications with the datalogger by trying to check the clock. If that fails follow the steps for message 14.

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
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.	
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.	

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
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.	
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.

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
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.	
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.

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
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.
54	Program file receive start		The server is requesting the datalogger program. The actual program segments will appears 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.	

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
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.	
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.	

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
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. 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. See section 7.4.3

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
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.	
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.	

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
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.
85	Data restore failed		On server startup the data broker data storage area could not be created.	This is a computer problem. The files are either not present or are corrupted. See notes for message 83.
86	Manual poll transaction started	Client logon name	The listed client is starting a manual poll operation according to the scheduled collection settings. A manual poll is initiated from the "Update Data Cache" button on the Control Panel data retrieval tab.	
87	Manual poll transaction complete		The manual poll operation has received the data from the datalogger.	

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
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.
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.	

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
98	Scheduled polling failed		Scheduled data collection failed.	Check communications with the datalogger by trying to check the clock. If that fails, follow the steps for message 14.
99	Collect area first poll		This message is posted either the first time data is collected for a collect area, or holes were lost for the datalogger.	If this is not the first poll for the collect area, this message indicates that data that had been stored in the datalogger was lost before it could be collected.
100	Table mount failed	Table name; Operating system information regarding the failure	The server was not able to create a data collection area from the stored table configuration file or new table definitions. This could be the result of trying to create table files that are too large for the computer system.	Check the computer operating system integrity. Verify that the LoggerNet system configuration files exist and the directory has not been corrupted.
101	Add record failed	Table name; Beginning record number; End record number; A reason for the failure	The server was not able to write data records to the data storage area.	This indicates a problem writing to files on the computer hard disk. Verify write permissions are set and that there is sufficient space left on the disk.
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

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
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.	

Code	Message Text	Message Parameters	Message Meaning	User Response to Message
108	Table reset	The name of the table that was reset; The account name of the logged in client	The name of a table was changed at the request of a client. On CR5000 and CR9000 loggers this is a reset for the table in the datalogger and on the PC.	
109	Collect schedule reset	The account name of the logged in client	The collection schedule was reset by the indicated client.	
110	Collect area setting changed	The name of the collection area; The setting identifier for the setting that was changed; The new value of the setting; The account name of the logged in client.	One of the settings for the specified collect area was changed. The identifiers for the setting can be found in Cora Script help.	
111	PakBus route added		A new PakBus route has been added to the routing table.	
112	PakBus route lost		A PakBus route has been lost and will be removed from the routing table.	
113	PakBus station added		A new PakBus station was added to the network.	
114	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"
```

D.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 E. Using RTDM with LoggerNet

E.1 RTDM

Real Time Data Monitoring (RTDM) software provides graphical displays of data from data files and automatically updates those displays as the data file is updated. RTDM requires file formats similar to those generated by Campbell Scientific's PC208W software. LoggerNet users can generate files in those formats using Data Table Monitor, a utility accessed from Start | Programs | LoggerNet | Tools | Data Table Monitor. RTDM expects files in an array based format, an example of which appears below:

101,1999,120,1724,19.000,0,0,75.46,75.02,532.5,12,0.668
101,1999,120,1724,20.000,0,1,75.45,75.03,531.7,12,1.336
101,1999,120,1724,21.000,0,2,75.48,75.03,531.7,12,1.002

The fields in this format represent:

> Array ID, Year, Day of Year, HourMinutes, Seconds, Filemark #, Record #, Data values…

RTDM checks a data file periodically and updates its display with new data if any has been added. Display options include graphs, charts, digital values, and other graphic components to present the data.

Data Table Monitor can only output the data from a single table on one datalogger to an output file. If you need to monitor the data from more than one table you can run multiple copies of the Data Table Monitor application. Each will create its own file. RTDM can read multiple files for the same display. See the section 14.2.3 for directions to start and set up Data Table Monitor.

CAUTION

One caution about the data file created by Data Table Monitor—there are no limits to size or longevity. If you plan to use this export feature on a regular basis, make sure to either restart Data Table Monitor or delete the files periodically. The data file can be easily deleted and restarted by clicking on the Start Export button.

Data Table Monitor should be running on the same computer where the RTDM Application is running to avoid trying to keep the output file open across the network connection.

E.2 Setting up RTDM

For detailed descriptions of the setup and configuration of RTDM refer to the Getting Started section in RTDM's Help and documentation included with RTDM.

This section is included to help with setup considerations that are different than the normal RTDM operation.

On the Data Source Tab in the Data Source Editor on the RTDM form, enter the filename and path for the datafile being created by TableMon. Select an update rate that is appropriate for the data being monitored.

Table data files include final storage labels in the table definition kept in the datalogger. The FSL file created by Edlog for table-based dataloggers does not contain array ID information and cannot be used by RTDM.

To see the sample data on the bottom you need to add an array and change the selected array ID to 101. The only array ID provided by Data Table Monitor is 101. You should now be able to see a sample of the data if the selected file has data in it. This will be very helpful in the next step.

On the Data Values tab you will need to create a new Item for each data value you are monitoring. When you click on the New Item button an item with the name FIELD is created. Change the name of the item to a meaningful label and change the selected array index to point to the data. As the Index is changed the related position in the record is highlighted on the display at the bottom.

Select the Output Series to link the data to the graph object.

To get the timestamp information from the record go to the Date/Time Definitions tab. Set up the year, day, hours-minutes, and seconds to be read from the array, in each case setting the Index value to point to the appropriate field.

Appendix F. CoraScript

F.1 CoraScript Fundamentals

CoraScript is a command line interpreter that reads its commands as text from its standard input device and writes the results of those commands as text to its standard output device. This style of input and output makes it possible to externally control the LoggerNet server operation using input and output redirection. It also makes it possible to string together commands in scripts that can be executed from the command line.

The CoraScript command interpreter is started by executing the program file Cora_Cmd.exe in a command prompt environment. CoraScript is also available through the LoggerNet program files start menu under "Tools". When the script processor starts up it will output a response:

CoraScript 1,1,1,30

The numbers indicate the version number of the current CoraScript.

CoraScript is a batch processing interpreter. It treats its input (from the standard input device) as a sequence of commands that are processed serially from the first to the last. As a command is processed, the results are written to the standard output device. A command is defined as the text up to a semicolon (;). The semicolon tells Cora Script that the command is complete and ready to execute.

The flexibility of the commands available within CoraScript and the independence from user interface considerations make CoraScript a valuable tool for testing, troubleshooting, and automating LoggerNet server operations.

NOTE

Because the commands available in CoraScript operate directly on the LoggerNet server and not through a user interface, there are no confirmation prompts for critical operations. Care should be exercised in using the commands to avoid interrupting normal server operations.

There is an extensive on-line help file available for CoraScript. To bring up the help file, type "help;" on the command line. (Make sure and include the semicolon ';' at the end and leave off the quotes.) Read through the directions and try some examples.

F.2 Useful CoraScript Operations

The following sections provide an overview of some common and very useful commands available with CoraScript. Some rules about formatting input and interpreting the responses:

- Always end the command with the semicolon (;) character. CoraScript uses the semicolon to mark the end of the command input and will not process anything until it is detected.

- Command parameters are often set using a combination of the parameter name and the value in this format: --name=login. Be sure to read the help for the command you are using.

- A response preceded by a plus sign (+) indicates that the command was successfully processed.

- A response preceded by a minus sign (-) indicates that the command failed and will usually be accompanied by a reason for the failure.

F.2.1 Connecting to the LoggerNet Server

Before you can execute any commands a connection to the LoggerNet server must be established. The connect command sets up the server context in which subsequent commands will operate until the end of the script is reached or another connect command is processed. The following segment shows a sample use of the connect command:

```
connect              # the name of the command
 localhost           # specifies the server's host address
  --name="bilbo"     # specifies the logon name (optional)
  --password={baggins}  # specifies the password (optional)
  ;                  # marks the end of the command
```

This command would normally appear on one line as follows:

```
 connect localhost  --name="bilbo"  --password={baggins} ;
```

For a more detailed explanation of the interpretation of the symbols and syntax refer to the CoraScript help.

F.2.2 Checking and Setting Device Settings

The current value of any of the configuration settings for a device is available using the *get-device-setting* command. Devices are referred to by the name as shown in the NetAdmin network map and settings are referenced by number. The setting numbers and their meanings are described in the CoraScript help.

To set the configuration setting for any device, use the *set-device-setting* command. As with *get-device-setting* the device is referred to by name and the setting by number.

F.2.3 Creating and using a Network Backup Script

One of the most useful commands for network maintenance is *create-backup-script*. Executing this command will create a script file of CoraScript commands that can be used to rebuild the existing datalogger network and configure all of the device settings. It will not save or restore any data that has been collected but can serve as a means to restore the network map and settings.

To create the script enter the command along with a filename and path for the script file:

create-backup-script c:\temp\mynetwork.script;

This will write all of the commands necessary to build and configure the network map into the file *mynetwork.script.* To restore the network map, remove all of the devices in the current map and then from a command prompt enter:

Cora_cmd.exe < c:\temp\mynetwork.script

The less than symbol (<) is redirection that tells the CoraScript executable to read commands from the specified file. For more information about this command see the CoraScript help.

F.2.4 Hole Management

There are several commands available with CoraScript to manage the hole collection process. These functions are not available through the standard user interface applications. See the CoraScript help for details about these commands.

List-holes

Purge-holes

Delete-holes

F.2.5 Scripting CoraScript Commands

To automate network processes, scripts can be created with other scripting language tools that would call the CoraScript interpreter, and send commands to the LoggerNet server. This provides an alternate means of controlling data collection, hole collection and maintenance functions such as clock check and set.

Appendix G. Importing Files into Excel

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.

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 you are using a table based datalogger and 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.
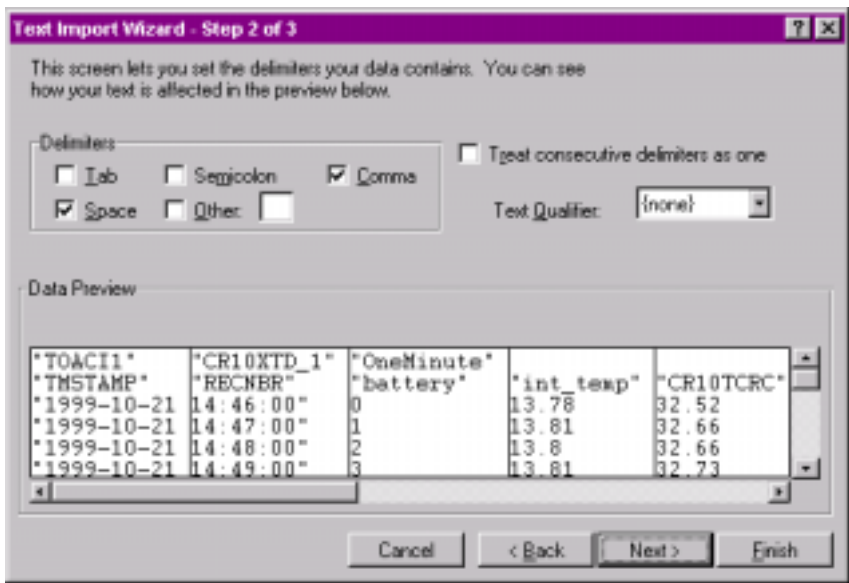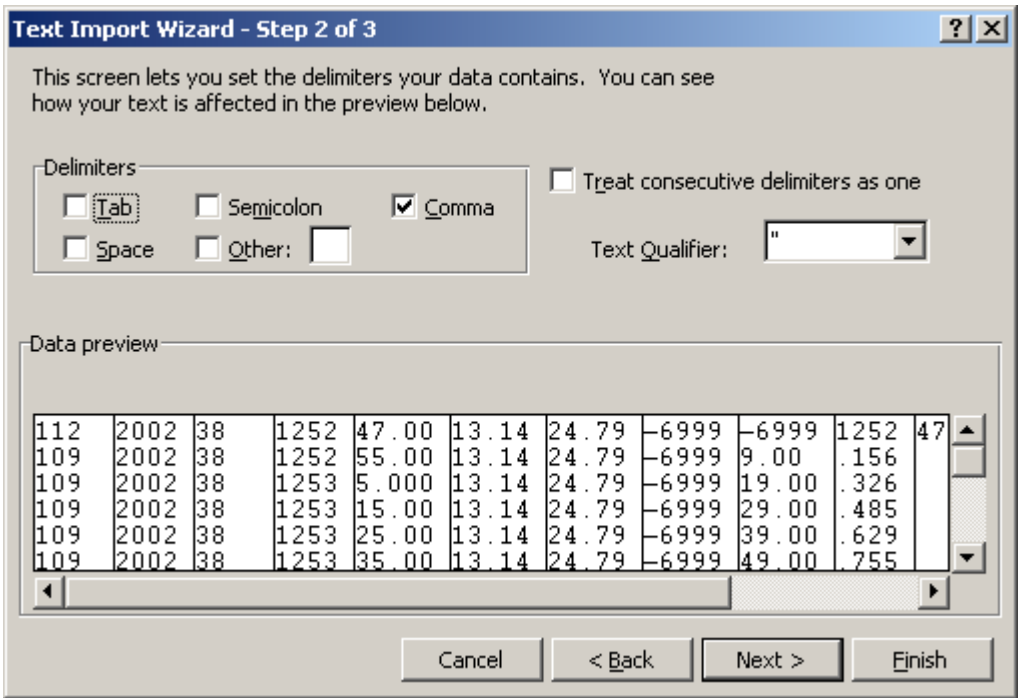
Step 3 of 3

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.
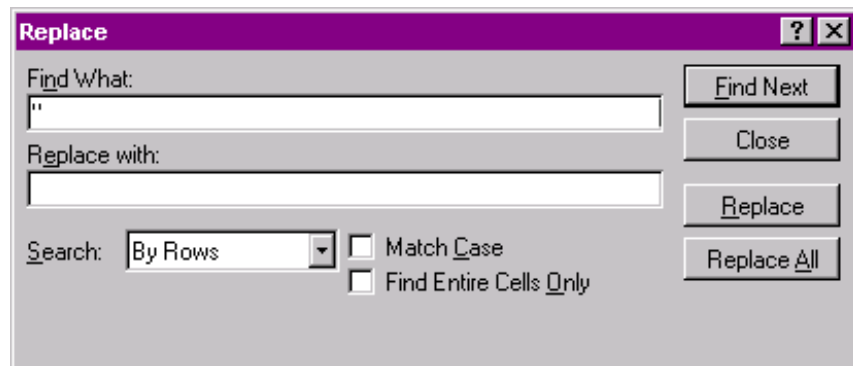
To complete the import, select the Finish button.
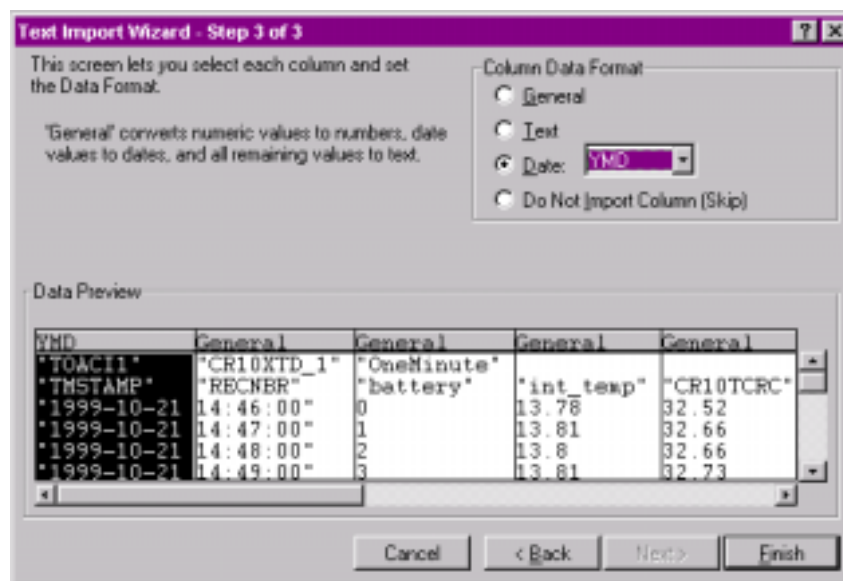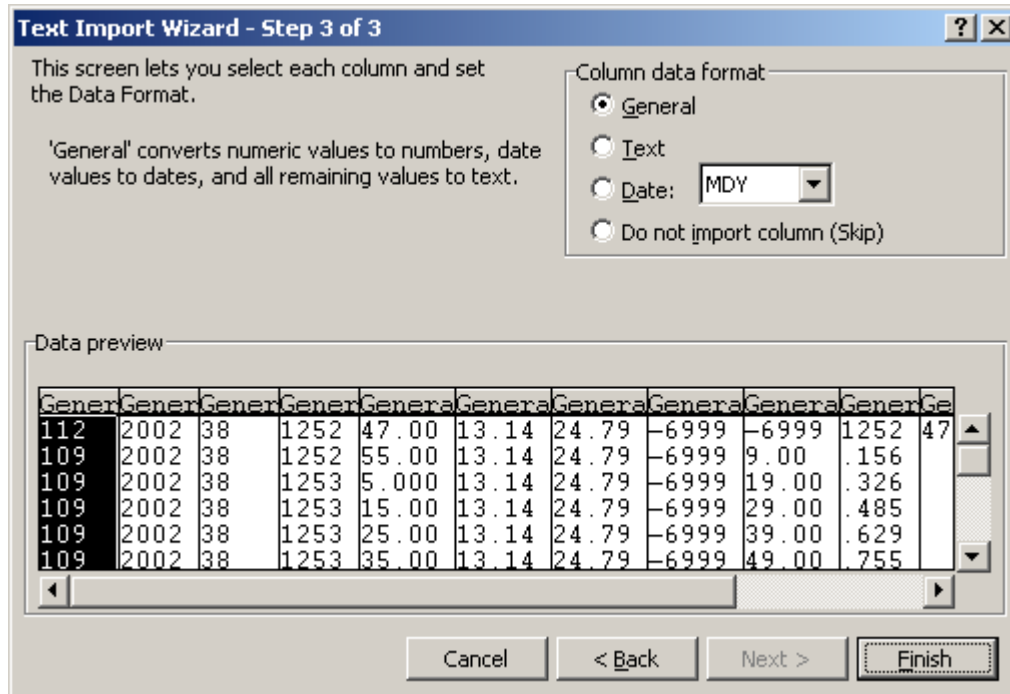
As imported, the Date and Time fields have a quotation mark in the field.

"TOACI1"	"CR10XTD	"OneMinute"				
"TMSTAMP"	"RECNBR'	"battery"	"int_temp"	"CR10TCR	"TC_TEMF	"HMPTem|
"1999-10-21	14:46:00"	0	13.78	32.52	31.26	28.75
"1999-10-21	14:47:00"	1	13.81	32.66	31.27	28.84
"1999-10-21	14:48:00"	2	13.8	32.66	31.3	28.78
"1999-10-21	14:49:00"	3	13.81	32.73	31.33	28.78
"1999-10-21	14:50:00"	4	13.8	32.77	31.38	28.68
"1999-10-21	14:51:00"	5	13.81	32.8	31.4	28.75
"1999-10-21	14:52:00"	6	13.82	32.84	31.44	28.83
"1999-10-21	14:53:00"	7	13.82	32.87	31.47	28.79

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.