

CRESTRON

e-control™ Mail

(SW-MAIL)

version 1.4

Contents

| | |
|--|-----------|
| How to Use This Manual | 4 |
| A Note on Printing This Document | 4 |
| Quick Start | 4 |
| Section Summary | 4 |
| Quick Start Guide 1: Sending e-Mail (demo3) | 6 |
| Quick Start Guide 2: Receiving e-Mail (demo4) | 7 |
| Introduction | 8 |
| What is Crestron e-control Mail? | 8 |
| System Terminology and Topology | 8 |
| Leading Specifications | 11 |
| Requirements | 11 |
| Installation | 12 |
| Licensing | 12 |
| Basic Server Setup | 14 |
| Communications Setup | 15 |
| Test Communications | 19 |
| Additional Server Side Setup | 19 |
| Server Configuration In Depth | 20 |
| Specifying a Configuration File | 20 |
| Password Access | 21 |
| e-Mail Options | 22 |
| COM Settings Definition | 24 |
| Signal Block Definition | 26 |
| Server Windows and Menus | 37 |
| The e-Mail Database Tables | 42 |
| Table design | 42 |
| Editing the Existing Database | 46 |
| Composing e-Mail | 46 |
| Recipient Lookup | 47 |
| Message Lookup | 47 |
| Implicit Recipient Name & Address | 47 |
| Manual Selection of Recipient and/or Message | 48 |
| Text Substitution and File Inclusion | 48 |
| Text Substitution Directives | 48 |
| Text File Inclusion | 49 |
| Rescanning | 49 |
| Control Messages | 49 |
| Identification | 50 |
| Message types | 50 |
| Syntax | 50 |
| Example | 51 |
| Demos | 52 |
| Demo 1: The Shortcut Signals | 52 |
| Demo 2: The LookupMsg and LookupRcpt Signals | 53 |
| Demo 3: Interactive Scrollers | 55 |
| Demo 4: Receiving e-Mail | 56 |

| | |
|---|------------|
| Appendix A: Theory of Operation | 59 |
| Server Protocol | 59 |
| Signal Block Definition / Activation | 59 |
| Signal Block Enable / Disable | 59 |
| Signal Block Error Reporting | 59 |
| Appendix B: Intersystem Communications and Signal Space Considerations | 61 |
| System Connections | 61 |
| Appendix C: Signal Reference | 65 |
| Definition of Terms | 65 |
| String Proxies | 65 |
| Bit Patterns | 65 |
| Standard e-mail Address Format | 66 |
| Error Reporting | 67 |
| Signal Summary | 67 |
| Signal Reference..... | 68 |
| Appendix D: System limitations | 107 |
| Serial Transmissions..... | 107 |
| Signal Definitions..... | 107 |
| Appendix E: Standard Scroller / Custom Scroller Feature Comparison | 108 |
| Appendix F: Dial-up TCP/IP configuration | 109 |

Crestron e-control Mail

How to Use This Manual

Adobe and Acrobat are trademarks of Adobe Systems Incorporated.

A Note on Printing This Document

This Portable Document File (PDF) can be printed with Adobe® Acrobat® Reader. Printing from a Windows 95 platform, version 4.0 or later, is strongly recommended because the figures print poorly with earlier versions. The latest version is freely available from Adobe at <http://www.adobe.com/acrobat/>.

Quick Start

To see an “out-of-the-box” demonstration of Crestron *e-control Mail* as quickly as possible, turn to the Quick Start Guide(s) beginning on the next page.

You will need:

- A Crestron CNMSX-PRO control system
- A touchscreen (LC-3000, CT-3000, CT-3500, or VT-3500); and
- A null-modem cable

Follow the instructions in the guides precisely in the order given and you should be up and running in a matter of minutes.

NOTE: The demos included with this package are all compiled to two versions, a **COM** version for use with an RS-232 serial connection, and a **TCP** version for use with an Ethernet connection (the latter case requiring the CNX Gateway). The Quick Start Guides refer only to the **COM** versions of these programs because setting up a serial connection is far simpler. We strongly recommend getting at least one demo to work first using a serial connection. Once that works, try the **TCP** versions. Instructions for setting up TCP/IP communications are provided under “Communications Setup, Control System Side, TCP/IP,” on page 17).

You do not need to license the software to try the demos provided you are still within the 15-day free trial period.

Section Summary

For more information, start with “Introduction” on page 8.

Detailed **setup and configuration instructions** follow “Introduction”.

Then comes information about **the e-Mail database** file, including information about text substitution and file inclusion directives.

After that, the **examples (demos)** are described and illustrated in detail.

Appendices include **Theory of Operation**, along with a complete **Signal Reference**.

Quick Start Guide 1: Sending e-Mail (demo3)

1 Install this package on your PC

Presumably, since you are reading this PDF file, you have already done this.

2 Connect a CNMSX-PRO (with touchscreen)

Connect a programming cable (a standard modem cable) from any **COM** port on your PC to the **COMPUTER** port on the front or back of the CNMSX-PRO control system. Connect a touchscreen to the control system set up for CRESNET ID 03.

3 Upload all control system software

The demo files can be found in the demos folder (also accessible through the Start Menu shortcut e-control Mail Demos)

Open the Crestron *Viewport* and establish communications with your control system. Use the **FileTransfer | Send Touchpanel...** command to upload **demomail.hex** to the touchscreen at ID 03. Use the **FileTransfer | Send Program...** command to upload the compiled SIMPL windows file **demo3COM.bin** to the CNMSX-PRO.

4 Connect the null modem cable

Make sure pins 4, 5, and 6 are not connected.

Connect a null-modem cable from **COM1** on the PC to **COM A** on the CNMSX-PRO.

5 Run the “server”

The installer sets the server to use the Configuration Settings file demomail.ini.

Select shortcut **e-control Mail Server** from the Crestron folder in the Windows *Start* Menu. If the title bar of the window does not read “e-control Mail Demos,” use the **File | Configuration file...** command to select the file **demomail.ini**.

6 Specify an SMTP (outgoing mail) server address

The initial configuration password is: crestron2

Give command **Server | Configure...**, select the *e-Mail* tab, and enter an SMTP server’s IP address or domain name. Close the window.

7 Enter your e-mail address

Depress the down-arrow key to leave the row, and click Yes to the “Commit Changes?” dialog.

For testing purposes, you must provide an e-mail address which you can check for incoming mail. Give command **e-Mail | Tables...** to bring up the *e-Mail Tables* window. Enter your address in the *addr* field (column) of the first record (row) (GEORGE WASHINGTON) of the *eMail_Addr* table. Save the change by either repositioning the cursor to a different record or by closing the window.

8 Start server protocol

Give the command **Server | Start Server w/Signal Analyzer**. (The Signal Analyzer is good for demos because it shows you the various signals going back and forth.)

9 Start the demo

On the touchscreen, navigate through the setup instructions to the demo screen. This final page-flip to the demo screen starts the demo.

10 Send an e-Mail message!

From the **SILLY MESSAGES** list, select the message called, **INVITE**. Notice that the default recipient for this message is “George Washington” (you). Or, select another message, then select **GEORGE WASHINGTON** from the **U.S. PRESIDENTS** list to override the default. Press the **Send Now** button. When the button feedback goes low, the message has been sent. (You should now check your incoming mail for a message from the control system.)

Quick Start Guide 2: Receiving e-Mail (demo4)

1 Install this package on your PC

Presumably, since you are reading this PDF file, you have already done this.

2 Connect a CNMSX-PRO (with touchscreen)

Connect a programming cable (a standard modem cable) from any **COM** port on your PC to the **COMPUTER** port on the front or back of the CNMSX-PRO control system. Connect a touchscreen to the control system set up for CRESNET ID 03.

3 Upload all control system software

The demo files can be found in the demos folder (also accessible through the Start Menu shortcut e-control Mail Demos)

Open the Crestron *Viewport* and establish communications with your control system. If you have not already done so, use the **FileTransfer | Send Touchpanel...** command to upload **demomail.hex** to the touchscreen at ID 03. Use the **FileTransfer | Send Program...** command to upload the compiled SIMPL windows file **demo4COM.bin** to the CNMSX-PRO. You may now close the *Viewport*.

4 Connect the null modem cable

Make sure pins 4, 5, and 6 are not connected.

Connect a null-modem cable from **COM1** on the PC to **COM A** on the CNMSX-PRO.

5 Run the “server”

The installer sets the server to use the Configuration Settings file demomail.ini.

Select shortcut **e-control Mail Server** from the Crestron folder in the Windows *Start* Menu. If the title bar of the window does not read “e-control Mail Demos,” use the **File | Configuration file...** command to select the file **demomail.ini**.

6 Specify a POP3 (incoming mail) server address

The initial configuration password is: crestron2

Give command **Server | Configure...** In the resulting *Configuration Options* window, select the *e-Mail* tab and enter a POP3 server’s IP address or domain name.

7 Enter your e-mail address

Select the *Signal Blocks* tab. Select the signal block called “DEMO4” and click the **Modify...** button. In the resulting *e-Mailbox Signal Block Definition* Window, provide an e-mail account name and password. In each window, click **OK** to close.

8 Start the “server protocol”

Give the command **Server | Start Server w/Signal Analyzer**. (The Signal Analyzer is good for demos because it shows you the various signals going back and forth.)

9 Start the demo

On the touchscreen, navigate through the setup instructions to the demo screen. This final page-flip to the demo screen starts the demo.

10 Download new mail

Touch the button **Touch here to check manually for NEW MAIL**. The button feedback stays active while the server contacts the e-mail host. All new messages are displayed. Scroll the listing to see all the messages.

11 Read a message!

Touch the listing to select a message to read.

Introduction

What is Crestron e-control Mail?

Crestron *e-control Mail* (SW-MAIL) empowers any Crestron control system with the ability to construct and transmit, and receive and view e-mail messages.

Simply by asserting specific signals, your control systems can send arbitrary text, whole text files, canned messages, alerts, status updates, *etc.*, to any e-mail address. Messages can be sent to a control system for display and to assert specific signals.

*The term “server” does not imply a need for specialized hardware. Any PC meeting the minimum requirements on page 11 will suffice to run **swserver.exe**.*

The actual e-mailing is not carried out by the control systems themselves, but by a remote e-e-mail host. The control system relies on intermediaries to translate communications protocols and supply other services. One of these intermediaries is the freely distributed Crestron *e-control Software Server*. SW-MAIL is a licensable component of this “server” application (**swserver.exe**) which is hosted on a standard PC running Windows® 95/98/NT and provides the following core technologies:

- Signal-level communications with the control system
- Access to database tables
- Access to external services (such as e-e-mail hosts) through the PC’s network connection.

The server is connected to the control system via either a serial cable through an RS-232 port or an Ethernet network through a LAN port. To effect the latter type of connection, the control system relies on an intermediary, the Crestron *CNX Gateway*, to translate communications protocols.

To aid in making all this clear, the following illustrated discussion of system terminology and topology should prove useful at this point.

System Terminology and Topology

This manual simultaneously discusses several different inter-connected computer systems. To reduce confusion, throughout the manual, these systems are referred to using the terms in the following table. (Also refer to the diagrams on the next page.)

| Term | Explanation |
|--|---|
| The system <i>or the control system</i> | One of a number of Crestron <i>control system(s)</i> , which may include any combination of the following models: CNMS, CNRACK, CNMSX-PRO, CNMSX-AV, and CNRACKX. |
| The server <i>or the software server</i> | The Crestron Software Server, swserver.exe , which runs on a PC under Microsoft® Windows® 95 or Windows NT®. |
| The gateway <i>or the CNX Gateway</i> | A communications conduit that sits between the <i>server</i> and the control system(s). |
| The host <i>or the e-mail host</i> | An e-mail host used for exchanging mail with the <i>system</i> and other e-mail clients. |

The *control system(s)* are connected to the *server* via direct RS-232 serial connection or via TCP/IP to the *gateway* and thence via TCP/IP to the *server*.

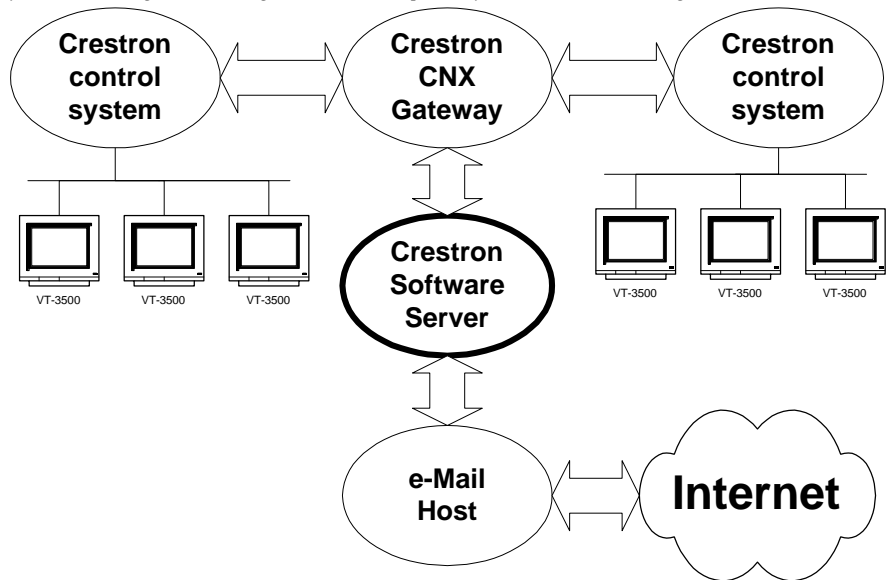
NOTE: “Connected via TCP/IP” means any node (computer) visible on the Local Area Network (LAN). If the LAN is connected to the Internet, this could include any node visible anywhere on the Internet. Since a node can also see itself, this implies that multiple services can run on the same machine. For example, the *gateway* and the *server* can be “self-hosted” in this way.

The *server* is also connected to the *e-mail host* via TCP/IP (which can also be self-hosted on the same machine, although this is not normally the case).

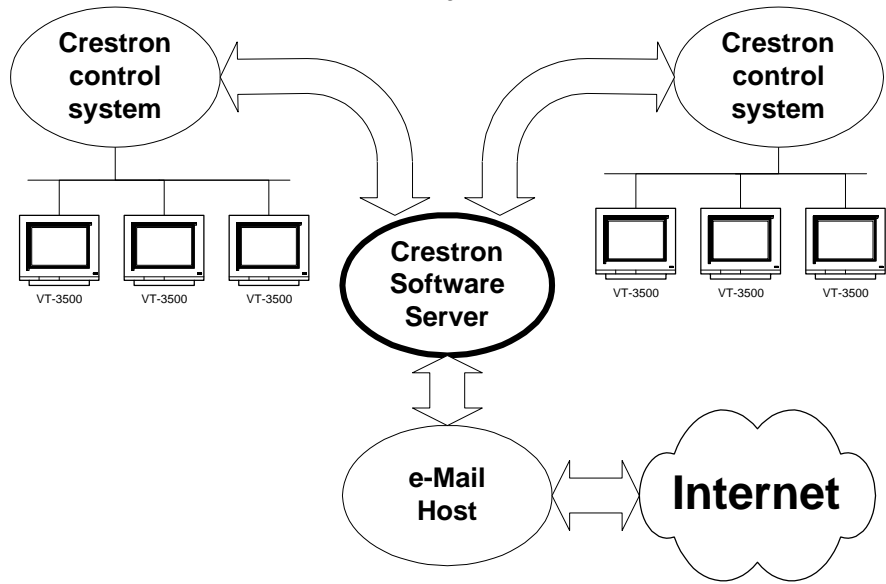
The e-mail host is actually comprised of two separate pieces of software, one for incoming mail (a POP3 server) and one for outgoing mail (an SMTP server). While these software generally reside on the same physical computer (host), this is not always the case. Whatever the case, we generally use the term *host* herein to refer to both of them as a single entity. (Nevertheless, it should be pointed out that if your application requires only sending mail or only receiving mail, you only need to be connected to one server or the other.)

In the illustrations that follow, the communication pathways are represented by the arrows. The physical network is not represented, however.

System block diagram, showing communication pathways (all connections using TCP/IP)



System block diagram, showing communication pathways
(curved arrows are RS-232 serial connections; straight arrows are TCP/IP)



NOTE: The CNX Gateway is not necessary when using RS-232 serial communications.

TCP/IP connections between the server and the control systems require that each side of the connection be provided with the IP address of the other. This kind of connection also requires the use of the CNX Gateway which is separately licensed software that facilitates communication between the server and the control system. The CNX Gateway is typically installed on the server (when sufficient TCP/IP sockets are available) or it can be installed on any computer visible (*i.e.*, pingable) on the TCP/IP network. There only needs to be one Gateway running on one computer to service the needs of all the computers and CNX control systems on the network. However, multiple Gateways are perfectly permissible as long as they are run on different computers.

Leading Specifications

Specifications Table

| SPECIFICATIONS | DETAILS |
|---|--|
| SWSERVER.EXE <i>(included with this package)</i> | Version 1.4 |
| CNMS/RACK Operating System | Version 3.18.12 or later |
| CNMSX/RACKX Operating System | Version 5.09.25 or later |
| CNMSX/RACKX Monitor | Version 5.09.25 or later |
| CNMSX/RACKX TCP/IP Stack | Version 5.09.10 or later |
| CNX Gateway | Version 2.08.04 or later |
| SIMPL™ Windows® | Version 1.4 or later; or Version 1.3 with Symbol Library Update 067 or later |
| VisionTools™ Pro (VT Pro-e) | Version 2.0.8.2 or later |

Requirements

The server should meet these minimum system requirements.

Windows 95/98/NT Operating System hardware requirements

32 MB RAM

100 MB hard drive space

133 MHz or faster Pentium processor

A faster processor is recommended for serving multiple connections simultaneously

800 x 600 or higher screen resolution

COM ports

Required to make serial (RS-232) connections to control systems (one port per control system). (See Cable requirements below.)

Network Interface Card

Required to make EtherNet connections to control systems. Also required in order to contact an e-mail host server via EtherNet connection.

Dial-up connection

Alternate means of contacting an e-mail host server via TCP/IP. (See Appendix F for configuration information.)

TCP/IP sockets

(These are software constructs provided by your operating system. The maximum number of sockets is operating system dependent.)

Server requires one socket per server-control system connection

Required for EtherNet control system connections only.

CNX Gateway (see below) requires one socket + one additional socket per server-control system connection

Cables

Precise CNSP-532 specs are available in the Crestron Cable Database.

Null modem cable, Crestron model CNSP-532 or equivalent
*Required for serial control system connections only.
Warning: Do not use a generic null modem cable.*

Auxiliary software

CNX Gateway

Required for TCP/IP (EtherNet) connections between the server and the control systems. Not required for serial connections.

E-mail hosts

An SMTP server for sending mail

To send e-mail, this server must be running on a host visible to your system, either locally (on your LAN) or remotely (via an Internet connection).

A POP3 server for receiving mail

To receive e-mail, this server must be running on a host visible to your system, either locally (on your LAN) or remotely (via an Internet connection).

Installation

As of this writing, the Crestron Downloads page can be found at:

<http://ftp.crestron.com/library/>

To install the Software Server, first download the installer package from the Crestron FTP site. To do this, first go to the Crestron website and select the **Downloads** page. New users must register. Proceed to the **ECONTROL** Library. Simply click on **SW-MAIL.EXE** to start the download.

Once the install package arrives on your PC, double-click the icon to initiate the install. Directions for the install are provided. The package is typically installed in C:\Crestron\control. During the install, the package reminds the user that a CNX Gateway is required. (This is actually only true for TCP/IP connections. Direct RS-232 connections do not require the CNX Gateway.)

Licensing

A 15-day free trial follows initial installation. If you are still within the 15-day period, you have the option to postpone licensing and skip to the next section.

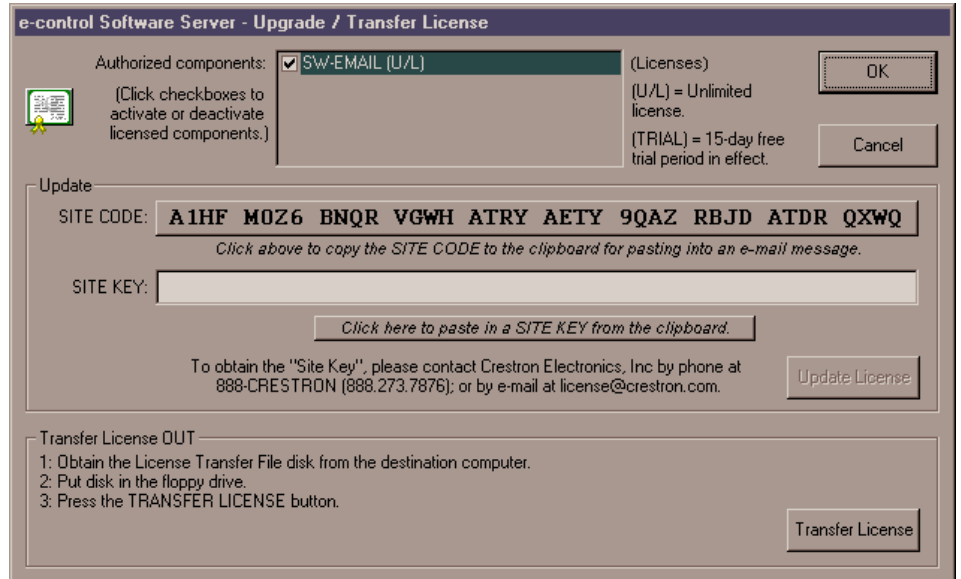
Server components are separately licensed. A license for the e-mail component must be obtained from Crestron even if other components are already in use.

Both the Software Server and the CNX Gateway are a licensed products, which means that although both software packages may be freely downloaded from the Crestron FTP site, use of the software requires purchase of licenses from Crestron. Each server running the Software Server must be individually licensed. In addition, to use Ethernet, you must acquire a CNX Gateway license with sufficient connections to accommodate all servers and control systems on your network.

Each package, once installed, generates a “Site Code” specific to the server on which it is running. Upon being provided with these Site Codes, Crestron can issue the appropriate “Site Keys,” which, once entered into each package’s licensing window, enables the full functionality of the software.

Obtaining a License

e-control Software Server – Upgrade/Transfer License window showing an “unlimited” e-mail license — shown activated (checked)



*You must use the **Copy** button to copy the SITE CODE to the clipboard. (Copying with Ctrl+C does not work from this field.)*

Open the server application. Select **Server | License** to open the *e-control Software Server – Upgrade/Transfer License* window, shown above. The license can be obtained over the phone or via e-mail. Call Crestron Customer Support with the “Site Code” shown in the *Site Code* field. However, it is easier and far more reliable to copy the “Site Code” into an e-mail message addressed to license@crestron.com. Once received, Crestron Customer Support issues a “Site Key” which must either be typed or pasted into the *Site Key* field of the window. Once entered, click on the **Update License** button. If the key is valid, the licensed components appear in the list above. Before closing the License Window, be sure to activate the components you plan to use. In the above example, the user has checked the box next to SW-MAIL.

It is permissible to exit the program while waiting for a “Site Key” to be issued. The application can be restarted and the “Site Key” entered at a later time. The “Site Key” issued is only valid on the same computer. It does not work on a different computer.

The License Window of the CNX Gateway is almost identical to the above. See the documentation that comes with the Gateway package for specific instructions.

Transferring an Existing License to Another Computer

As mentioned, a license is only valid on the computer for which it was obtained. However, a license can be transferred from one computer to another without the need to contact Crestron first. There are several reasons to transfer a license. The application developer may set up the system off-site, then transfer the license to the actual computer on-site when ready. Alternatively, if the hardware or operating system on the computer where the server is licensed is upgraded, the license may cease to be valid, but could be transferred to another computer before the upgrade and then back to the original machine after the upgrade.

On both the source computer (where the license is currently valid) and the destination computer (where the license is to be transferred), open the server application. Select **Server | License** to open the *e-control Software Server –*

Upgrade/Transfer License window (shown above). Make sure this window is active on both computers.

Step 1. On the destination computer, create a preparation file on a diskette in the A: drive by inserting a blank, formatted diskette and selecting **Prepare Diskette**. This creates a file on the diskette which indicates who is receiving the license. A second, backup copy of the file is also created. Alternatively, these files can be created on another portable media (e.g. Zip disc) or a network drive by simply browsing for a new file location in the save file window. *If you plan to transfer via a network drive, first make sure that both computers have the appropriate read/write access to the drive and folder being used.*

Step 2. After the above step has completed, remove the diskette from the drive and insert it into the source computer's floppy drive. *Do not flip the write-protect tab; the diskette must remain write-enabled.* Click on the **Transfer License** button. The source computer reads the preparation file to see which computer wants the license. It encodes the license for the destination and writes it back to the same file on the floppy diskette (or network drive). The source computer has now passed the license to the file. Only the designated computer can use the license, so the server is no longer licensed on the source computer.

NOTE: At this point in the transfer procedure the server license resides on a file on the diskette or network drive, and not on the computer. If this file should become lost or damaged, the license is lost as well. Because of this, please use the utmost care while performing this transfer.

Step 3. Bring the diskette back to the destination computer. Click on the **Transfer License** button. The computer reads the license information off the diskette and transfers the license to itself. The server is now licensed on this machine.

Basic Server Setup

This product requires a proper physical connection between both “sides” of the system — the server and the control system. Furthermore, the software on both sides must be properly configured. As previously discussed, the connection can be either serial via RS-232 cable or Ethernet via Local Area Network (LAN). Choose your mode of communication and refer to the following sections to make the proper physical connections and to configure the software.





The following sections include specific notes *in italics* for setting up the server and the control system to run the included demo programs. Although the focus is therefore on the demos, the same basic procedures would be followed to ready the system for any other programming as well.

NOTE: If you are setting up the system for the demos, note that the contents of the following sections are also contained in brief in the touchpanel pages for demos 2 and 3. If you are already familiar with system setup procedures, and you are willing to start with demo 2 or 3, you might want to load of the touchpanel as a first step, and follow the on-screen instructions instead. Choose setup instructions for Serial or TCP/IP. Note that both sets of instructions lead to the same demo screen.

The files for all four demos are in a folder called `demos` which can be located through the following *Start* Menu shortcut:

```
Start Menu
  | Programs
  | Crestron
  |   e-control Mail
  |   e-control Mail Demos
```

Inside this folder you will find the four demos, with support files:

| | | |
|---|---------------------|---|
|  | demo1 | |
|  | demo2 | |
|  | demo3 | |
|  | demo4 | |
| | demomail.vtp | <i>VisionTools touchscreen project file with pages for all four demos (compiled version of above)</i> |
| | demomail.hex | <i>Server's Configuration Settings file which accommodates all four demos</i> |
| | demomail.ini | <i>Sample database file for use with all four demos</i> |
| | Maildemo.mdb | |

The installer registers **demomail.ini** as the currently selected Configuration Settings file. (If the server's title bar does not read "e-control Mail Demos," use the **File | Configuration file...** command to reset it.) This file configures the server for all four demos.

Each of the four demo folders contain the following files:

| | |
|---------------------|---|
| Demo?COM.smw | <i>SIMPL Windows project file (RS-232 version)</i> |
| Demo?TCP.smw | <i>SIMPL Windows project file (TCP/IP version)</i> |
| demo?COM.bin | <i>compiled SIMPL program code (RS-232 version)</i> |
| demo?TCP.bin | <i>compiled SIMPL program code (TCP/IP version)</i> |

RS-232 is featured in the Quick Setup Guide because it is easy to set up. Because we anticipate strong interest in TCP/IP, we have pre-built both versions for your convenience.

In the above, ? stands for the demo number. The two versions of the SIMPL program for each demo, (COM and TCP) are almost identical, both being configured for a CNMSX-PRO, using the front panel device and a touchpanel with CRESNET ID = 03. Both versions have ports defined for both serial (RS-232) communications via the CNMSX-PRO's built-in **COM A** port (slot 4, port A), and Ethernet (TCP/IP) communications via the **LAN** port on a CNXENET card installed in the CNMSX-PRO's DPA slot. In the COM versions, the TCP/IP port is commented off while in the TCP versions, the RS-232 port is commented off. *This is the only difference between the two versions.*

The following sections separately describe the setup procedures for connecting multiple control systems via either RS-232 or TCP/IP connections. Actually, a mixture of connections is permitted. For example, two control system might be connected via RS-232 (using the **COM1** and **COM2** ports) while two more might be simultaneously connected via the TCP/IP network connection.

In the following, the indented, italicized paragraphs contain advice on setting up the server and a control system specifically to run the supplied demo files. You will find that most of the steps have already been accomplished because they are specified by the supplied demo configurations.

Communications Setup

Server Side

1. Run server application by selecting Database Manager from the Crestron folder of your Start menu.
2. Select config file. Specify a Configuration Settings file (.ini file) by selecting **File | Configuration File....** Refer to "Specifying a Configuration File," page 20.

*The server is installed with a **demomail.ini** pre-selected as the default configuration file. (This is intended to simplify the Quick Start Guides.)*

3. Set communications mode. Select **Server | Configure** and enter a password to open the *Configuration Options* window. (Refer to "Password Access" on page 21). Select the *COM Settings* tab. The settings for each connection to a control system must match those on the other end (the control system side) of the actual connections. Click on each connection in turn, click the **Modify...**

button, and choose either RS-232 (and select the port and speed) or TCP/IP (and set the IP address and IP ID). Click **OK** to make the changes for each connection.

The demos are pre-configured to use RS-232.

Control System Side, RS-232

Serial communication requires wiring the server directly to the control system.

NOTE: Serial communications requires neither the CNX Gateway software nor the use of an Ethernet network.

1. **Connect PC for programming purposes.** For each control system to be connected to the server, temporarily connect the PC containing the control system and touchscreen project files to the control system via a serial cable between any available **COM** port of the server and the **COMPUTER** port of the CNX control system. (This could be — but need not be — the same physical machine that runs the Software Server.) Refer to the CNMSX manual (latest revision of Doc. 8118) for instructions. This connection can be removed once the control system is programmed.
2. **Install control system program.** Upload the compiled SIMPL Windows program file (.bin file) to each control system.

As supplied, the demo programs are configured for a CNMSX-PRO control system. For other models, using SIMPL Windows, convert the program as described below and recompile.

3. **Install touchpanel pages.** Upload the compiled VT Pro-e project file (.hex file) to each control system.

As supplied, the demo touchpanel file, which contains pages for all the demos, is configured for a CT-3000 touchpanel; and the accompanying .hex file is compiled for same. This file however also works fine with an LC-3000, CT-3500, and a VT-3500. If you have one of these models, go ahead and upload the .HEX file as is. If you are working with another panel, convert the file to your target panel and recompile.
4. **Connect to server.** Connect null-modem cables (Crestron model CNSP-532) from each control system to the server. Each connection requires its own COM port on the server side. The port to use on the control system depends on the specific model:

CNMSX-PRO. Use one of the built-in COM ports.

The demo files are all configured for a CNMSX-PRO using COM A (slot 4, port A).

CNMSX-AV. Use one of the built-in COM ports.

Use SIMPL Windows to convert the demo files. In the Configuration Manager, drag & drop a CNMSX-AV system onto the CNMSX-PRO. The converted system does not have a front panel, so compile “notices” appear — which can be ignored.

CNRACKX. Install a CNXCOM-2.

Use SIMPL Windows to convert the demo files. In the Configuration Manager, drag & drop a CNRACKX system onto the CNMSX-PRO. The converted system has a CNXCOM-2 card in slot 4; use Port A. The converted system does not have a front panel, so compile “notices” appear — which can be ignored.

CNMS. Install a CNCOMH-2 card. Use of the built-in COM ports for the present purpose is not recommended.

Use SIMPL Windows to convert the demo files. In the Configuration Manager, drag & drop a CNMS system onto the CNMSX-PRO. The converted system has a CNCOMH-2 card in slot 5; use Port A. The converted system does not have a front panel, so compile “notices” appear — which can be ignored.

CNRACK. Install a CNCOMH-2.

Use SIMPL Windows to convert the demo files. In the Configuration Manager, drag & drop a CNRACK system onto the CNMSX-PRO. The converted system has a CNCOMH-2 card in slot 4; use Port A. The converted system does not have a front panel, so compile “notices” appear — which can be ignored.

Control System Side, TCP/IP

For more information on control system TCP/IP setup, consult the e-control Overview document, **overview.pdf**, installed with the CNX Gateway software; or the SIMPL Windows release notes, installed with SIMPL Windows.

TCP/IP communications requires a control system with a LAN/Internet port. Therefore, a CNX generation control system is required (CNMSX-AV, CNMSX-PRO, CNRACKX, or CNRACKX-DP). The CNX control system and the server are both connected to the same network. This connection, once properly configured, can then be used both for system communications (uploading, Test Manager support, Viewport functions) and run-time server/client (server/control system) communications as well. (The latter function however requires the addition of the CNX Gateway software.)

1. **Install Ethernet card.** Install the CNXENET card into the Direct Processor Access (DPA) slot of each CNMSX. Refer to the CNXENET manual (latest revision of Doc. 8129) for instructions.
2. **Connect server.** Connect the CNX control system(s) to the server using one of the following two methods:
 - Connect the control system into the same LAN as the server. Use a commercially available Ethernet hub to expand the number of connections available by plugging in the LAN, the server, and the control system into the same hub.
 - Alternatively, make a two-device private network by connecting an Ethernet “crossover” cable between the Ethernet port of the server’s Network Interface Card and the LAN port of the CNX control system’s CNXENET card. **Do not attempt this with a regular Ethernet cable.**
3. **Connect PC for programming purposes.** For each control system to be connected to the server, temporarily connect the PC containing the control system and touchscreen project files to the control system via a serial cable between any available **COM** port of the server and the **COMPUTER** port of the CNX control system. (This need not be the same machine that runs the software Server.) Refer to the CNMSX manual (latest revision of Doc. 8118) for instructions. This connection can be removed once the control system is programmed. Open the Viewport and issue the **Setup | Communications Settings...** command to reconfigure communications for RS-232.
4. **Check firmware versions.** Before proceeding, however, verify that the CNX control system has been loaded with the proper versions of firmware. Still in the Viewport, select **File Transfer | Update Control System** to bring up a window box containing the current versions of monitor, operating system, and TCP/IP stack. Verify the versions per the “Leading Specifications” (page 11).
 To upgrade any of these files, retrieve a copy of the latest upgrade package from the Crestron website (OPSYS Library). These files have an extension of **.upz** which contains all three system components in one compacted file. Once downloaded, browse for the appropriate file in the *Update Control System* window. Click **Send** to upload the files to the control system. (When upgrading the system in this manner, always send all three components to avoid incompatibilities.)
5. **Define control system IP address.** Still in the Viewport, select **Functions | Set Control System IP Information**. The *Set Control System IP Address* window opens. Assign an IP address for the CNX control system. The address should be obtained from the MIS department. The IP address has four fields separated by periods (e.g. 192.168.2.3) and must be unique. Click **OK**.

In early versions of the CNMSX, it may be necessary to upgrade to an intermediate version of the monitor first and then to the required version of the monitor. (The Viewport issues a notice if this is necessary.)

6. Enter gateway address. Still in the Viewport, select **Functions | Setup IP Table** to open the *IP Table* window. Click on the **Retrieve Current IP Table from Control System** button to display the current listing. Verify that the IP address for the PC running the CNX Gateway (often but not necessarily the server itself) appears with an IP ID of 03. If it does not appear, use the **Add...** button to add an entry for IP ID 03. Then click the **Send IP Table to Control System** button.
7. Switch to TCP/IP. Now that TCP/IP is properly configured, the Ethernet connection can be used for all subsequent system communications (from SIMPL Windows, Test Manager, Vision Tools Pro-e, and all Viewport functions). See the section below titled “Test Communications” on page 19. Open the Viewport and issue the **Setup | Communications Settings...** command to reconfigure communications for TCP/IP. The serial cable can now be removed.
8. Install control system program. Upload the compiled SIMPL Windows program file (.bin file) to each control system.

As supplied, the demo programs are configured for a single CNMSX-PRO control system. For other models, use SIMPL Windows to convert the program as follows and recompile:

CNMSX-AV.
In the Configuration Manager, drag & drop a CNMSX-AV system onto the CNMSX-PRO. The converted system does not have a front panel, so compile “notices” appear — which can be ignored.

CNRACKX. Install a CNXCOM-2 card in slot 4 and use Port A.
In the Configuration Manager, drag & drop a CNRACKX system onto the CNMSX-PRO. The converted system has a CNXCOM-2 card in slot 4; use Port A. The converted system does not have a front panel, so compile “notices” appear — which can be ignored.
9. Install touchpanel pages. Upload the compiled VT Pro project file (.hex file) to each control system.

As supplied, the demo touchpanel file, demomail.vtp (which contains pages for all four demos), is configured for a LC-3000 touchpanel; and the accompanying .hex file is compiled for same. This file however also works fine with an CT-3000, CT-3500, and a VT-3500. If you have one of these models, go ahead and upload the .hex file as is. If you are working with another panel, convert the file to your target panel and recompile.

Test Communications

At this point, test your connections.

RS-232 Control Systems

Use the Viewport to verify communications between the server and the CNX control system. Select **Diagnostics | Establish Communications**. If properly connected, the PC responds with the COM port and baud rate.

TCP/IP Control Systems

First test the IP address of the CNX control system by “pinging” it. From a networked PC bring up an MS-DOS prompt (Windows 95/98) or “Command Prompt” (Windows NT) and type “ping <IP ADDRESS>”, as shown below. The control system responds with several lines “Reply from address < IP ADDRESS >...”. If no response is received from the “ping” to the IP address of the CNX control system, repeat the procedure in “Control System Side, TCP/IP,” page 17.

```
C:\WINDOWS>ping 132.149.2.2

Pinging 132.149.2.2 with 32 bytes of data:

Reply from 132.149.2.2: bytes=32 time=8ms TTL=60
Reply from 132.149.2.2: bytes=32 time=5ms TTL=60
Reply from 132.149.2.2: bytes=32 time=5ms TTL=60
Reply from 132.149.2.2: bytes=32 time=5ms TTL=60
```

Once a reliable connection is established, test that the CNX control system is listening and responding properly. Reconfigure Viewport communications to use TCP/IP by selecting **Setup | Communications Settings**. Once the *Port Settings* window opens, select **TCP/IP** as the *Connection Type*. For *IP Address*, Click on **Fixed** and enter the CNX control system IP address in the active field. Test the new connection by issuing the **Diagnostics | Check Operating System Version** command.

Additional Server Side Setup

In addition to properly setting up and testing communications with each connected system, the following steps are also required to make the server operational:

1. Select database file. Supply the full pathname to the database under the *COM Settings* tab. This file is the sole source of all database tables accessed by all signal blocks. See “The e-Mail Database Tables,” page 42, for more information.
*The demos are pre-configured to point to the file **maildemo.mdb** in the **demos** folder.*
2. Set e-mail host IP addresses. Select the *e-Mail* tab. The SMTP server’s domain name or IP address must be supplied for sending e-mail (demo1, demo2, and demo3); and the POP3 server’s domain name or IP address must be supplied for receiving e-mail (demo4). These are often the same. Refer to “Options for Sending e-mail” on page 23 and “Options for Receiving e-mail” on page 24.
3. Set sender e-mail address. Optional. Still in the *e-Mail* tab, fill in the *Static From: Addr* field. Any address is usually acceptable; it does not have to be a

real address. If you leave this field blank, all mail sent will appear to be from `anonymous@unknown.net`.

4. **Set recipient e-mail address.** The database file must contain a valid e-mail address for testing purposes.

The demo database file, `maildemo.mdb`, does not ship with this data. You must enter at least one such address in order to confirm that the demos 1, 2, and 3 are actually sending mail.

*To do this, close the Configuration Options window and select **Tables | e-Mail** to open the e-Mail Tables window. Enter an e-mail address for some record in the `eMail_Addr` table (such as record ID=1, "George Washington"). (Use an address to which you have access so that you can confirm that the system sent the mail successfully.)*

5. **Set mailbox e-mail address.** Each e-mailbox signal block must contain a valid e-mail account name and password. See "Mailbox Name and Password" on page 34.

The demo configuration does not ship with this data; you must fill in the blank fields in the DEMO4 signal block to run demo4 and test receiving e-mail.

6. **Indicate control system connection.** Point each active signal block to a COM Settings definition. (If you have not yet defined the connection through which this signal block will communicate, you can leave this blank for the now. However, the signal block cannot be activated until it references a COM Settings definition.) See "COM Settings" on page 29 for a description of how to point a signal block to a COM Settings definition.

All the signal blocks in the demo configuration already point to a COM Settings definition.

7. **Start server protocol.** Select **Server | Start** to start the server protocol, or **Server | Start w/ Signal Analyzer** to start the server protocol with the *Signal Analyzer* window opened. This window can be freely opened and closed while the server is active. While opened, the Signal Analyzer monitors signals going back and forth between the server and the control systems. Incoming and outgoing signals can also be simulated from here; and all signal block signal lists can be printed from here.

The Signal Analyzer is intended for debugging and demo purposes only. It should not normally be left opened because it does place additional processing demands on the server, creating a perceptible delay on slower systems.

Server Configuration In Depth

This section is a reference to all the options available in the *Configuration Options* window. Changes to options in this window are saved to the current Configuration Settings file when the **OK** or the **Apply** buttons are actuated. Therefore, it is important to make sure you are operating on the appropriate Configuration Settings file before opening the window.

Specifying a Configuration File

The installer registers the file `demos\demomail.ini` as the current Configuration Settings file. This file pre-configures the server for all four demos; and particularly for use with the Quick Start Guides — which instruct you to load demo3 and demo4.

You can use the **File | Configuration file...** command to select a Configuration Settings file of your choice. The file pathname so specified is stored in the Windows registry on your machine. In addition to specifying the configuration filename, this command also instantly reconfigures the server based on the named file. This is a very useful feature for the developer working on multiple projects.

NOTE: If the server cannot open a specified configuration file, the server uses default values for all options. If any changes are made, a new config file is created (using the specified pathname).

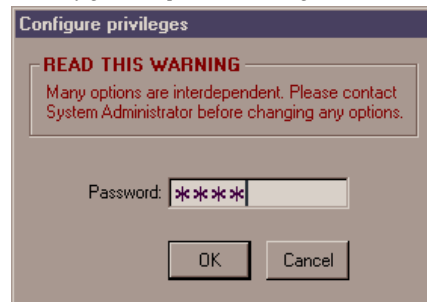
The installer registers the file `demos\demomail.ini` as the current Configuration Settings file. This file pre-configures the server for all four demos. The Quick Start Guides — which instruct you to upload demo3 or demo4 — depend on this file being set as the current config file. If you have changed the config file and wish to run the demos again, you must first change it back.

Password Access

Access to the server’s *Configuration Options* window is password-protected. This is to prevent end-user meddling with the configuration options, which can very possibly disable the server’s proper operation.

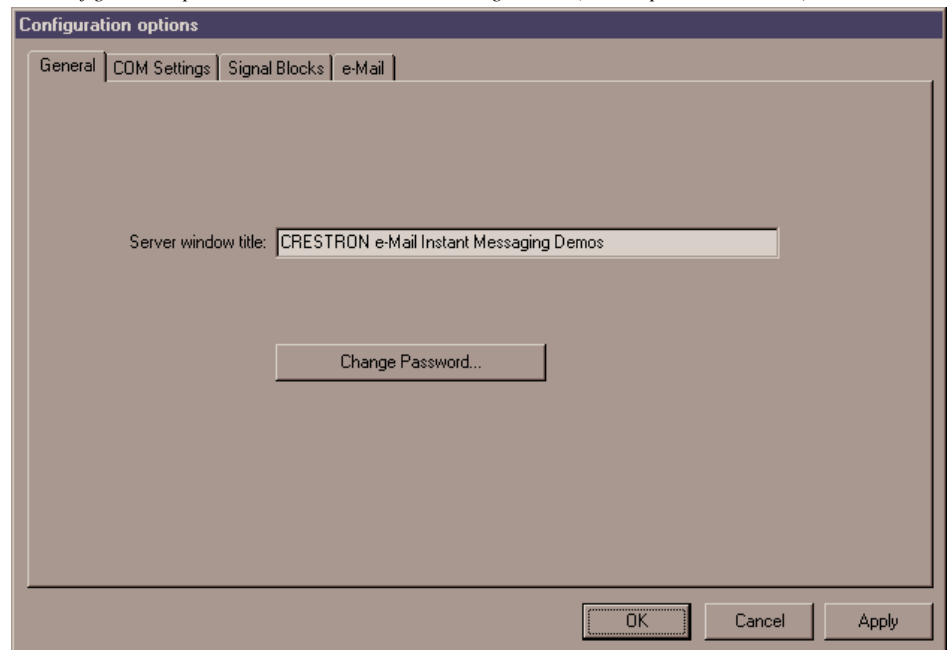
Select **Server | Configure**. The server prompts the user for a password.

The configuration password dialog — controls access to the Configuration Options window



Enter `crestron2` which displays all tabs. Entering anything else displays the *General* tab only.

The Configuration options window, General tab, showing all tabs (correct password entered).



Level 1 and 2 passwords may be changed from the *General* tab. Click on the **Change Password** button to open the *Change Password* window. Enter the old password and the new password twice. Click **OK** to complete the change.

Resetting the Configuration Password

In the event the password is misplaced, be aware that it is not stored in readable form. Rather, values derived from the password are stored in the configuration files. The password can effectively be reset by locating the configuration file and then either deleting or editing it.

Use the **File | Configuration file...** command to note the pathname of the currently selected configuration file. Exit the server.

Deleting the file means that all configuration variables revert to their default values the next time the server is run. The problem with this approach, of course, is that you lose any settings already made.

To reset the password only (without affecting the rest of the configuration), edit the **.ini** file using the Notepad application (**Start | Programs | Accessories | Notepad**). Locate and delete the following key in the [GENERAL] section (the value may differ):

```
privilegeLevel_2=180350152
```

Exit the Notepad application, saving the file.

The password is now reset to its default — which is “crestron2.”

Run the server again. Issue the **Server | Options...** command. Enter the default password. You can now change the password to whatever you want by clicking the **Change Password** button.

e-Mail Options

The information under this tab in the “Options” window is applied to every message sent by all e-mailer signal blocks and received by all e-mailbox signal blocks.

The Configuration Options window showing the e-Mail tab with fictitious mail host URLs.

Configuration options

General | CDM Settings | Signal Blocks | e-Mail

Sending e-mail

The options below are common to all e-mailer servlets (defined under the Servlets tab).

SMTP Server URL

cc: all msgs to

Above may contain an e-Mail address or comma-separated list of addresses only. That is, do not include recipient's real names.

Lookup substitution defaults Queue messages (Requires SMTP Express from www.quiksoft.com.)

Receiving e-mail

The options below are common to all e-mailbox servlets (defined under the Servlets tab).

POP3 Server URL

Check mail every minutes (0 = no automatic check)

OK Cancel Apply

Options for Sending e-mail

These options affect the behavior of all e-mailer signal blocks.

SMTP Server URL Field

Supply an SMTP server domain name. Typically, this is the organization's domain name with the sub-domain *smtp* prefixed, as in *smtp.organization.com*, but quite often it could be something else. It may or may not be the same as the POP3 server URL. If in doubt, check how the e-mail client is set up at any PC on site or check with the organization's MIS department or Internet Service Provider.

NOTES:

- 1 This field must be completed before the server protocol can be started.
 - 2 Avoid "raw" IP addresses; they can change. Use fully qualified domain names instead.
 - 3 Avoid fictitious IP addresses or domain names; they cause excessively long DNS lookups with each attempted mailing. This is a real problem when not using SMTP Express, because the server cannot process any other signals while waiting for the SMTP server to "come back." *(This is only a problem for the SMTP server name; it is not a problem for fictitious domain names in the addresses of outgoing e-mail.)*
-

cc: Field and Checkbox

This field accepts a comma-separated list of e-mail addresses (without real names). The purpose of this field is to assist system managers who are monitoring outgoing e-mail. If a high volume of mail is expected, we recommend setting up a separate e-mail account name for this purpose.

A check in the checkbox indicates that this information is sent with every e-mail.

Lookup Substitution Defaults Checkbox

Text substitution *directives* are normally replaced with the contents of text substitution *registers*. The contents of the registers are defined at run-time. If a directive invokes an undefined register, it is replaced with a null string (essentially removing the directive — and substituting it with nothing).

Checking *Substitution defaults*, however, directs the server to use the directive's parameter number to lookup a record with matching ID field in the **eMail_Subst** table and to use substitution text found therein. *If the register in question has a defined value, that value is used, as usual; only when a register is undefined does this default action take place.*

For example, if register 3 is undefined and *Substitution defaults* is unchecked, then a directive that invokes this register (e.g., "Room C{3}" in a message's subject header or body text), is removed (replaced by nothing), yielding the result "Room C." If on the other hand, *Substitution defaults* is checked, the directive is replaced by the value of the *substitution* field in the record in the **eMail_Subst** table whose *ID* field contains the value 3. If that record exists and its *substitution* field contains "525" then the resulting string would read "Room C525" — and this is what would be sent in the e-mail.

Queue Messages Checkbox

Since the server cannot respond to incoming signals while waiting for an e-mail message to finish sending, it is preferred to send messages in the background. A check in this checkbox permits such an option by queuing messages to the *SMTP Express* application which does the actual sending.

For more information about text substitution, see "Text Substitution Directives" on page 48.

SMTP Express is a third-party product which can be obtained from the following web page:
<http://www.quiksoft.com/ea/symail/smtpepress/>

Options for Receiving e-mail

These options affect the behavior of all e-mailbox signal blocks.

POP3 Server URL Field

Supply a POP3 server domain URL. Typically, this is the organization's domain name with the sub-domain *pop*, *mailbox* or *postoffice* prefixed, as in *mailbox.organization.com*, but quite often it could be something else. It may or may not be the same as the SMTP server URL. If in doubt, check how the e-mail client is set up at any PC on site or check with the organization's MIS department or Internet Service Provider.

NOTE: The same notes under "SMTP Server URL Field" on page 23 apply to the POP3 server URL as well.

Check mail every minutes

A value in this field causes all enabled e-mailbox signal blocks to periodically check for new mail. The initial value for this field is 0 (no automatic checks).

COM Settings Definition

A data structure called a "system" must be created for each connection you intend to make to your control system.

All active signal blocks (*Signal Blocks* tab) must reference such a structure. See "Interface Definition" on page 29 for instructions on defining such a reference for your signal blocks.

The COM Settings tab

The *COM Settings* tab of the Configuration Options window contains a list of data structures called "COM Settings definitions" which represent connections to control systems. From this tab, you can activate and deactivate such definitions, and define additional ones.

NOTE: Connections may be defined before or after signal blocks are defined. However, signal blocks cannot be activated until they reference a defined connection.

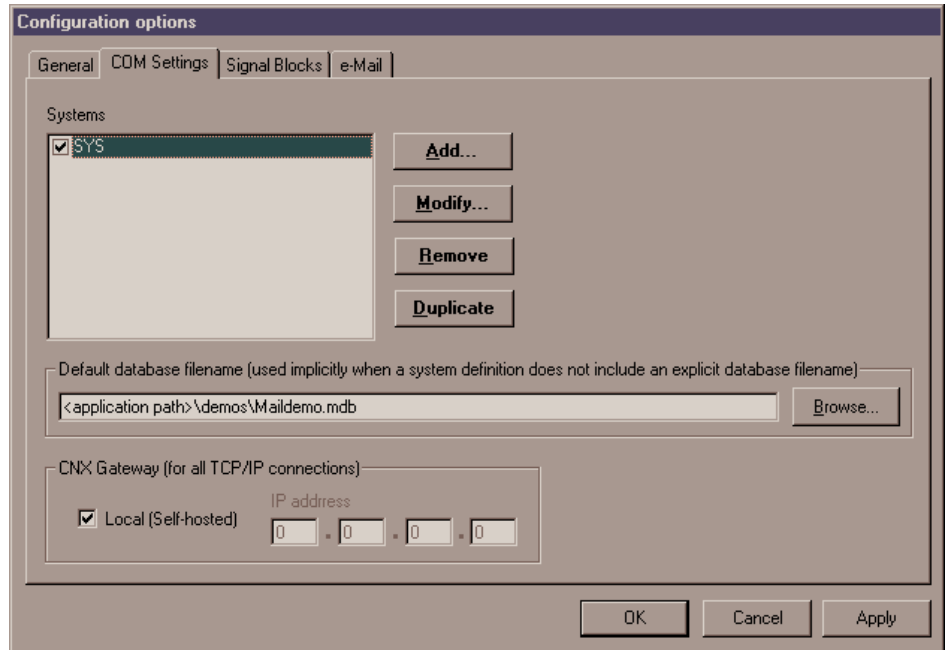
Refer to the figure below.

To remove a COM Settings definition, select it and click the **Remove** button.

To duplicate an existing definition, select it and click the **Duplicate** button. The new definition differs from the original in that it is given a unique name which is derived from the name of the original, incremented by one. (If the original did not end in a number, the name of the duplicate is the name of the original with a "1" suffixed to it.)

Click the **Add...** button to define a new connection; or select one of the definitions already listed and click **Modify...** to modify it. The *COM Settings* window opens:

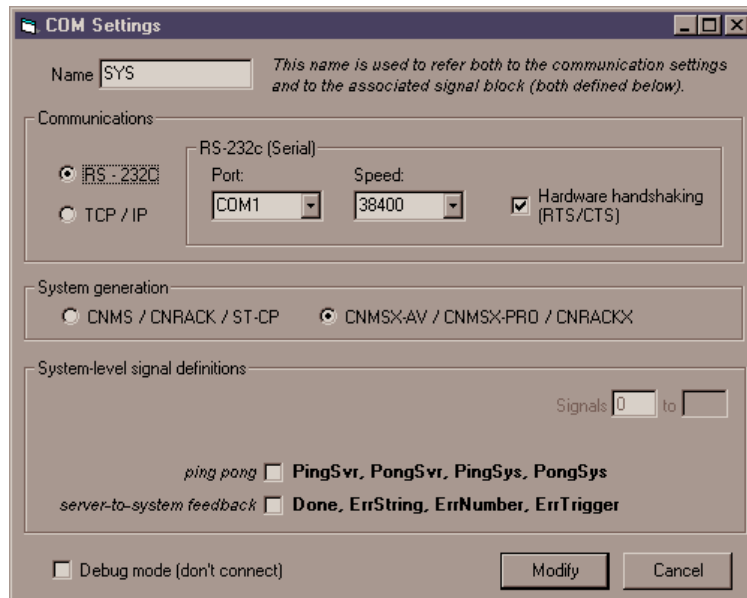
The Configuration Options window, COM Settings tab, showing the only connection defined in the demo configuration (selected).



COM Settings definitions (connections) can be active or inactive. A check in the box next to the definition name indicates that the connection is activated. If not activated, it is ignored when the server protocol is started.

The COM Settings window

The COM Settings window for the connection defined in the demo configuration showing RS-232 communications selected ...



... and if TCP/IP communications were selected, it would look like this (fictitious IP address shown):

Definition name

Each COM Settings definition requires a unique name. A field for this data can be found at the top left of the *COM Settings* window. We recommend choosing a name that reflects either the location of the control system (such as SUITE3) or its function (such as PHONEBOOK).

This name is used in the server's user interface to identify the system data structure. It is also sent along with error messages to the actual control system to identify the source of an error resulting from processing one of the system-level signals defined herein.

Control system generation

Here you specify the type of control system. The server uses this information to take into account minor differences in the way the older generation of Crestron control systems functioned in terms of timing and data capacity.

Communications mode

In this frame you choose RS-232 or TCP/IP connections. The details are described in the Server Side configuration sections for RS-232 (page 16) and TCP/IP (page 17).

System-level signal definitions

In this window you can also define optional system-level signals by checking the appropriate boxes. Doing so defines a special signal block which communicates with its own **Intersystem Communications** symbol in your SIMPL Windows program. In this case, you should also fill in the *Signals* field, as follows:

Signals

This is the *offset* of the **Intersystem Communications** symbol in your SIMPL Windows program. The connection's signal block must not overlap any other signal block (channel 1 of) these COM Settings or else the server protocol will not be able to be started.

Refer to the "Appendix C:" on page 65 for more information on each of the signals listed in the window.

Signal Block Definition

Data structures called a "signal blocks" are created on the server, each communicating with its own **Intersystem Communications** symbol on a control system. Each active signal block must reference a "COM Settings" data structure which defines a connection to a control system. See "COM Settings Definition," above.

A “signal block” is a software construct defined in the server which communicates with special symbols in the SIMPL program running in your control system.

e-Mailer signal blocks communicate with **Send e-Mail** symbols.

e-Mailer signal blocks communicate with plain **Intersystem Communications (XSIG)** symbol.

Each **Standard Scroller** signal block communicates with a particular **DBMScroller** macro.

The *Signal Blocks* tab (see below) displays a list of defined signal blocks. Three types of signal blocks are available with an SW-MAIL license:

| | |
|--------------------------|---|
| e-Mailer | prepare and send outgoing e-mail messages |
| e-Mailbox | download, select, and read incoming e-mail messages <i>or</i> download and scan for control signal syntax and send such signals if found. |
| Standard Scroller | for interactive display of database tables in support of the above. |

The **Standard Scroller** signal block is a constrained form of the more robust **Custom Scroller** signal block — which is only available if you are also licensed for *e-control Database Manager* (SW-DBM). The **Standard Scroller** signal block has a static configuration designed to interface with the included **DBMScroller** SIMPL Windows macro. (If you use a **Custom Scroller**, you cannot use the macro.)

Without an SW-DBM license, **Standard Scroller** signal blocks cannot be directly enabled via a signal from a control system. In that case, they are only useful when attached to another type of signal block designed to control scrollers, in this case **e-Mailer** and **e-Mailbox** signal blocks — which use them in support of their primary functions, as follows:

- An **e-Mailer** signal block can use scrollers to display an address book and prepared messages.
- An **e-Mailbox** signal block can use scrollers to display an IN box and a message.

When attached to a controlling signal block, a scroller (either type) is enabled automatically when the controller is enabled. When not attached to a controlling signal block, **Standard Scrollers** can only be enabled directly with an SW-DBM license.

Specific differences between the two types of scroller signal blocks are summarized in “Appendix E: Standard Scroller / Custom Scroller Feature Comparison” on page 108.

The Signal Blocks tab

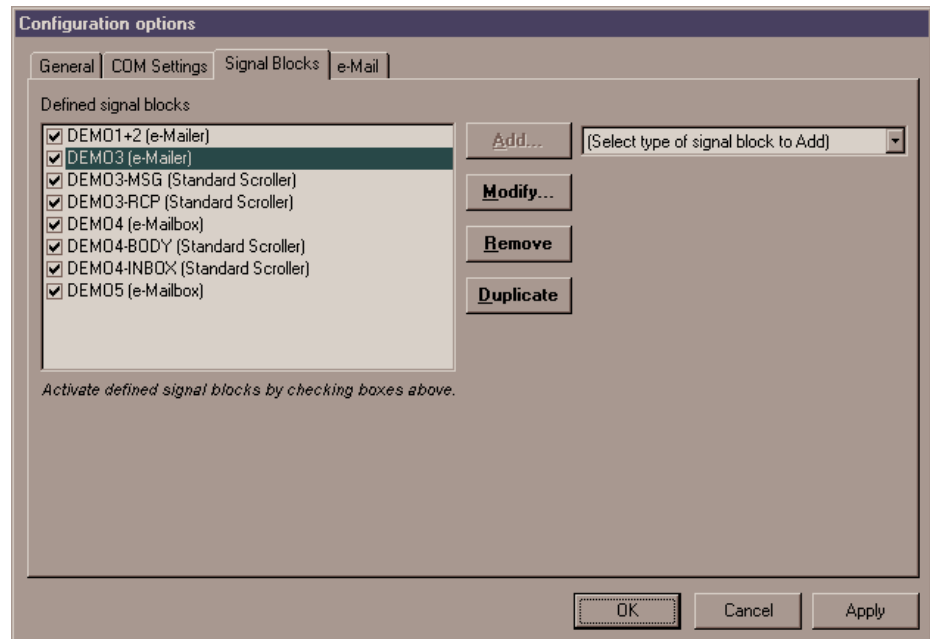
The signal blocks tab contains a list of the currently defined signal blocks. Refer to the next figure.

To remove a signal block definition, select it and click the **Remove** button.

To duplicate an existing definition, select it and click the **Duplicate** button. The new definition differs from the original in that it is given a unique name which is derived from the name of the original, incremented by one. (If the original did not end in a number, the name of the duplicate is the name of the original with a “1” suffixed to it.)

New signal blocks can be added by selecting a signal block type from the *New signal block type* list box and clicking the **Add...** button. Existing signal blocks can be modified by highlighting the signal block in the *Defined signal blocks* list and clicking the **Modify...** button.

The Configuration Options window, Signal Blocks tab, showing all the signal blocks defined in the demo configuration. As shown, all seven signal blocks are active (checked); and the e-mailer signal block for demo3 is selected.



Signal Blocks, once defined, can be active or inactive. A check in the box next to the signal block name indicates that the signal block is activated. If a signal block definition is not checked, it is ignored when the server protocol is started, neither accepting nor responding to incoming signals in its range. Inactive signal blocks are not considered for signal space conflicts with other signal blocks when the server protocol is started.

NOTE: Signal Blocks may be defined before or after the COM Settings to which they need to refer are defined. If the signal block is defined first, you will not be able to specify the COM Settings yet. This is permitted. However, such signal blocks cannot be activated until they reference defined COM Settings.

The server can have any number of signal blocks defined and active simultaneously.

Selecting a signal block from the list in the *Signal Blocks* tab and clicking the **Modify...** button — or defining a new signal block with the **Add...** button — opens a signal block definition window.

Such a window shows a particular signal block definition. The definition includes:

- **Interface definition.** The options across the top of the *Signal Block Definition* window are common to all types of signal blocks and include the signal block's name, system connection, and signal offset. (The term *interface* refers to the server-signal block interface; *i.e.*, information that all signal blocks must have to be handled by the server as signal blocks.)
- **Optional signal definitions.** Words shown in the *Signal Block Definition* window in **bold** case are names of optional signals implemented by the signal block. These are included in the signal block definition (they are "defined") either by checking the adjacent checkbox, or (in the case of a set of enumerated signals) by supplying a non-zero number in the adjacent text field. Undefined signals do not appear in the signal list and must not appear in the matching **Intersystem Communications** symbol on the control system side. Be aware that there may also be a number of non-optional signals which are not shown in the window.

- **Behavior options.** These have specific effects on signal block behavior when the server protocol is running.

The highest numbered signal in the signal block's input or output signal lists is shown in the box in the upper-right corner. This is based on the signal offset entered in the adjacent box and the current signal block definition. This value is updated synchronously as the user interacts with the window. This box turns red when the highest analog or serial signal number on either the input or the output lists exceed 1023; or should the highest digital signal number exceed 4095.

Interface Definition

All signal blocks require the following basic information. Fields for these data are shown across the top of all *Signal Block Definition* windows.

Name

A unique signal block name is required here. This name is used in the server's user interface to identify the signal block. It is also sent along with error messages to the control system to identify the source of the error. We recommend choosing a name that reflects either the location of the control system (such as BOOTH3) or its function (such as PHONEBOOK).

COM Settings

The *COM Settings* list box contains the names of all the COM Settings definitions from the *COM Settings* tab. Point the signal block to a particular COM Settings definition by selecting it from the list.

Each signal block must be associated with a control system. Control systems are defined separately under the *COM Settings* tab. You may define the signal blocks first if you like, then define the systems, and come back and make the associations later. Note, however, that signal blocks cannot be activated without first associating a system, through its COM Settings connection, with the signal block.

Channel

For systems with multiple Virtual COM Port channels defined, select a channel here.

Signal Offset

When a signal block shares a signal space (a channel) with another signal block, they both cannot begin numbering their signals at 0. In such a case, supply values space the signal blocks' signals properly — such that they do not overlap with each other. If any signal blocks' signal spaces overlap, attempting to start the server protocol results in an error.

Auto-enable

If this box is checked, the signal block is automatically enabled when the server protocol is started.

e-Mailer Signal Block Options

Selecting DEMO3 and clicking the *Modify...* button opens the *e-Mailer Signal Block Definition* window.

The e-Mailer Signal Block Definition window, showing the definition of the DEMO3 e-mailer. A fictitious e-mail address has been entered into the Static “From:” Addr field.

This window allows the user to assign a default “To:” name and a default “To:” address (for all recipients) as well as a default “From:” name and a static “From:” address (for the sender). These options (and others) are set separately for each e-mailer signal block.

Sender Header

The sender header in an e-mail message is the line beginning with “From:” at the top of the message; it contains the name (optional) and e-mail address (required) of the entity sending the message.

NOTE: The sender address while required does not necessarily have to be an address known to your E-mail host. Different servers have different rules regarding this point. Some require just the top-level domain (e.g., “.com”) to be known, while others only send mail for known URLs. Still others may require that the entire address represent a known e-mail account.

Two fields appear in the *Sender Header* frame of the *e-Mailer Signal Block Definition* window.

The Default “From” Name field contains a default sender name. As a default, this information is only used when not otherwise specified (via control system signals). The should describe the sender. We recommend using a name that describes the location and/or function of the system originating the mail. For example, a specific system could be located in a room (e.g., “Video Suite 3”) or in a building (e.g., “Bldg 4 HVAC”). If the control system does not specify a name and this field is left blank, the originating e-mail address appears without an accompanying sender name.

The Static “From:” Addr field contains a typical e-mail address, such as:

accountname@domainname

Unlike the name field, the value in this field is not a default; the control system cannot override this value; it is always used on all outgoing mail originating from this e-mailer signal block. We recommend using an address that describes the originating control system such as:

system5@myclient.com

If the control system does not specify an address and this field has been left blank, the specific address

anonymous@unknown.net

is sent instead (subject to the following rules):

1. If neither the name nor the address is specified, both these values are used.
2. If a name is provided, but no address, the default address is used along with the provided name.
3. If an address is provided, but no name, then neither of these values are used, and the resulting "From:" header contains only the provided address.

Although, as noted, the "From:" address cannot be specified by the control system at run-time, the "From:" name *can* be, so that a control system with devices in several rooms can send e-mails with "From:" headers such as:

```
From: Ballroom <system3@thehotel.com>
(etc.)
```

or:

```
From: Control Booth B <system3@thehotel.com>
(etc.)
```

Recipient Header

The recipient header in an e-mail message is the line beginning with "To:" at the top of the message; it contains the name (optional) and e-mail address (required) of the entity to whom the message is being sent.

Two fields appear in the *Recipient Header* frame of the *e-Mailer Signal Block Definition* window. As defaults, the information provided therein is only used when not otherwise specified (via control system signals).

The Default "To" Name field contains a default recipient name. If the control system does not specify a name and this field has been left blank, the recipient header appears without a name.

The Default "To:" Addr field contains a default recipient e-mail address. Use this field to specify a default destination for unaddressed mail. If the control system does not specify an address and this field has been left blank, the e-mail message cannot be sent.

The following rules apply to forming the recipient name and address:

1. If neither the name nor the address is specified, both these defaults are used.
2. If a name is provided, but no address, the default address is used along with the provided name.
3. If an address is provided, but no name, then neither default is used, and the resulting "To:" header contains only the provided address.

Bound scrollers

These are references to scroller signal blocks. For interactive preparation of e-mail messages, scrollers are typically configured to be bound to the canned e-mail

messages table, **eMail_Msg**, and the canned recipients table, **eMail_Addr**. Binding these signal blocks in turn to the e-mailer signal block permits the signal block to:

- Automatically enable/disable the scrollers when the signal block is enabled/disabled
- Inform the signal block upon receiving a valid scroller pick

Once the signal block “knows about” the scrollers, simply picking items from the scrollers has the following effects:

- The e-Mail parameters are automatically set.
- Such parameters are echoed to the control system for display.

Picking an item from the “message” scroller sets the subject and body. In addition, each message has a reference to an ID of a record in the recipient table. If this reference is non-null and if a name and address have not already been given, the information from that record are echoed back to the control system and are used to set the name and address.

Picking an item from the “recipient” scroller sets just the name and address. Normally, these values overwrite any previously set name and address. However, see the *Allow Multiple Recipients* option, described below.

It is permissible to have either scroller without the other; or neither. If neither scroller is used, mail can still be composed from the tables via the two Lookup signals; or verbatim using the various Set signals (see “Appendix C: ” on page 65).

Run-time properties

Three checkboxes appear in the *Properties* portion of the *e-Mailer Signal Block Definition* window.

Allow Multiple Recipients

This option controls the number of recipients. When checked, a list is developed until the mail is sent or the list is cleared. When this box is not checked (default), each recipient specified replaces the last, such that an e-mail message is sent exclusively to the most recently specified recipient.

Echo Lookups

This option controls feedback behavior for each e-mailer signal block. If the box is checked, data from table lookups is sent back to the control system via the various **Echo** serial signals.

A subordinate option, *Show Indirection*, displays indirection for recipient name and address by prefixing one or two right angle brackets (>) to the echoed recipient name and/or address data, as follows:

- When the recipient name and address is set from a record in the **eMail_Addr** table and a field is null (blank), the default is used and an angle bracket (>) is prefixed to the echoed data.
- When the data come indirectly from the **eMail_Addr** table (through a reference in **eMail_Msg** table), an angle bracket is prefixed to the echoed data.

Log SMTP sessions

This option is meant as a debugging aid when the e-mail host fails to respond. The entire “conversation” with the server is logged in the main server window. Each line of the conversation is preceded by the following string of characters:

SMTP.signal block>

In the above, *signal block* is replaced with the actual name of the e-mailer signal block carrying on the conversation. This option should normally remain off (unchecked).

Optional signal definitions

This frame lists optional signals which can be excluded from the signal block (and from the congruent **Intersystem Communications** symbol on the control system side). For further information on each of the optional signals listed in this frame, refer to the “Signal Reference” on page 68. To exclude a (group of) signal(s) from the signal block, either uncheck its checkbox or enter a zero (or blank) into its textbox.

Using “Send e-Mail” SIMPL symbol

This option is a convenience feature. Checking it forces the selection of optional signals to conform to the **Send e-Mail SIMPL** symbol (used in place of an **Intersystem Communications** symbol). This option is checked in the figure above, which means that the optional signals are forced to the following definitions:

| Optional Signal | | Send e-Mail symbol value |
|---|------------------------|--------------------------|
| SetParm_n | where <i>n</i> = 1 to: | 4 |
| LookupParm_n | where <i>n</i> = 1 to: | 4 |
| Shortcut_n | where <i>n</i> = 1 to: | 16 |
| Status | | <i>undefined</i> |
| ErrString, ErrNumber, and ErrTrigger | | <i>defined</i> |

None of these signals definitions can be altered so long as the *Using “Send e-Mail” SIMPL symbol* option is checked.

e-Mailbox Signal Block Options

Selecting DEMO4 and clicking the *Modify...* button opens the *e-Mailbox Signal Block Definition* window.

The e-Mailbox Signal Block Definition window, showing the definition of DEMO4, the e-mailbox signal block used in demo4, with the e-Mail account information frame filled in, and the Process control messages option checked.

e-Mailbox Signal Block Definition

Intersystem Communications Interface
 Name: DEMO4 COM Settings: SYS Vcom Channel: 1 Signals: 200 to 228 Auto-enable

e-Mail account information
 Mailbox name: youraccount @ yourmail.yourcompany.com
 Password: *****

Optional signal definitions

open message PageFlip
download status Status
echo headers EchoDate,
 EchoSubj,
 EchoFrom,
 EchoRcpt
folder info NewCount,
 OldCount
local feedback ErrString,
 ErrNumber,
 ErrTrigger

CTRL msgs

Analog signals (a)
 SignalA1..a

Serial signals (s)
 SignalS1..s

Digital signals (d)
 SignalD1..d

Referenced signal blocks

IN Box scroller

Message scroller

Reply e-mailer

Process control messages

Behavior

Delete from Mail Server as downloaded Mark for deletion from Mail Server when deleted from IN Box Check mail automatically (see e-Mail tab)

Using "Receive e-Mail" SIMPL symbol

NOTE: The domain name part of the *Mailbox name* is set for all e-mailbox signal blocks under the *e-Mail* tab in the *Configuration Options* window (see "POP3 Server URL Field" on page 24).

Mailbox Name and Password

The *Mailbox name* is an account name established on the e-mail host. The *password* is the password for that account. It is required of all active e-mailbox signal blocks. If missing from any such signal block, the server protocol cannot be started.

Optional signal definitions

This frame lists optional signals which can be excluded from the signal block (and from the congruent **Intersystem Communications** symbol on the control system side). For further information on each of the optional signals listed in this frame, refer to the "Signal Reference" on page 68. To exclude a (group of) signal(s) from the signal block, either uncheck its checkbox or enter a zero (or blank) into its textbox.

Control Messages

Check the *Process control messages* option if you want the server to check for control messages and issue these signals. If checked, also specify here the number of control signals to define in each signal category (Analog, Serial, and Digital). See "Control Messages" on page 49 for more information about control messages.

Using "Receive e-Mail" SIMPL symbol

This option is a convenience feature. Checking it forces the selection of optional signals to conform to the **Receive e-Mail** SIMPL symbol (used in place of an

Intersystem Communications symbol). This option is checked in the figure above, which means that the optional signals are forced to the following definitions:

| Optional Signal | Send e-Mail symbol value |
|---|--------------------------|
| SignalA_n where <i>n</i> = 1 to: | 3 |
| SignalS_n where <i>n</i> = 1 to: | 3 |
| SignalD_n where <i>n</i> = 1 to: | 10 |
| PageFlip | <i>defined</i> |
| Status | <i>defined</i> |
| EchoDate, EchoSubj, EchoFrom, and EchoRcpt | <i>defined</i> |
| NewCount, OldCount | <i>defined</i> |
| ErrString, ErrNumber, and ErrTrigger | <i>defined</i> |

None of these signals definitions can be altered so long as the *Using “Receive e-Mail” SIMPL* symbol option is checked.

Referenced Signal Blocks

References to other signal blocks are optional. When given, they enhance the functionality of the e-mailbox signal block, as described below.

IN Box scroller

Supplying a reference to a scroller here causes the signal block to download new mail into the database table associated with the named scroller. This table serves as an “IN box,” a repository for incoming e-mail.

If the *Process control messages* option is checked, only non-control messages will end up in the IN box.

If an IN box scroller is not specified, non-control messages are ignored (and possibly deleted from the server; see “Delete from Mail Server as downloaded,” below).

If the *Process control messages* option is not checked *and* an IN box scroller is not referenced, the signal block configuration is considered invalid and attempting to start the server protocol results in an error.

Message scroller

This table is required in order to display a message chosen from the IN box.

If it is not specified, the IN box scroller is still usable, but picking a message from the scroller does not produce any visible result. Note, however, that picking a record from the IN box scroller still responds with the **PageFlip** signal (if defined) and the various **Echo** signals; and doing so still has utility for the **Delete**, **KeepAsNew**, and **Reply** signals.

Reply e-mailer

An e-mailer signal block is specified here to be used for replying to messages. If it is not specified, the **Reply** signal results in a run-time error message. When an active e-mailer signal block is properly specified here, the **Reply** signal prepares a new outgoing e-mail message using the specified e-mailer signal block, as follows:

- The recipient name and address of the new message is set to the sender’s name (when available) and address from the present message.

- The subject field of the new message is set to the subject field of the present message. "Re : " is suffixed to this string if it does not already begin with same.

Use the **SetBody** and **SendNow** signals to complete the reply.

Behavior

The following options affect behavior of the e-mailbox signal block while it is running.

Delete from Mail Server as downloaded

Check this option to delete each message from the e-mail host after downloading. This is a good idea because as the number of old messages retained by the host builds up, checking for new mail slows down, inasmuch as each message needs to be checked against the contents of the IN box to see if it needs to be downloaded. Because this is such a good idea, this option is checked in a newly created e-mailbox signal block.

The option is provided for the case where this e-mailbox signal block does not have exclusive access to the e-mail account in question. If the account is also routinely checked from another mail client, it is not desirable to delete new mail until the other client(s) have had ample opportunity to download the new mail themselves. In a corporate setting, where a new e-mail account can easily be set up for exclusive use by this e-mailbox, this would likely not be an issue. However, in a residential system, the end user may wish to have his Crestron system display messages from his existing e-mail account, which he presumably routinely checks when he is on-line as well.

NOTE: Control messages are always deleted as soon as they are processed — regardless of the setting of this option. Therefore, this option only applies to messages not accepted as control messages.

Mark for deletion from Mail Server when deleted from IN box

If this option is checked, mail deleted from the local database is automatically deleted from the e-mail host the next time the host is checked for new mail.

NOTES:

1. If the *Delete from Mail Server as downloaded* option is checked, this option would not normally be checked as well. The only time checking both options would make sense is if *Delete from Mail Server as downloaded* although checked now, was not checked previously, and a large amount of mail has built up on the mail host (which, as discussed in the preceding section, is an undesirable situation).
 2. When the *Mark for deletion from Mail Server when deleted from IN box* option is checked, there is a slight overhead (a database lookup) for each marked message when mail is checked. We recommend that when the amount of old mail residing on the mail host is or becomes nil, that this option be turned off (unchecked).
 3. As noted in the preceding section, Control messages are always deleted as soon as they are processed. Therefore, setting this option when you are expecting only control messages is pointless.
-

Check mail automatically

If this option is checked, this signal block performs an automatic check mail operation every *n* minutes — where *n* is the value given in the *e-Mail* tab of the *Configuration Options* window.

NOTE: Regardless of this setting, mail will not be checked if $n = 0$. See “Check Mail Every Minutes” on page 23 for more information.

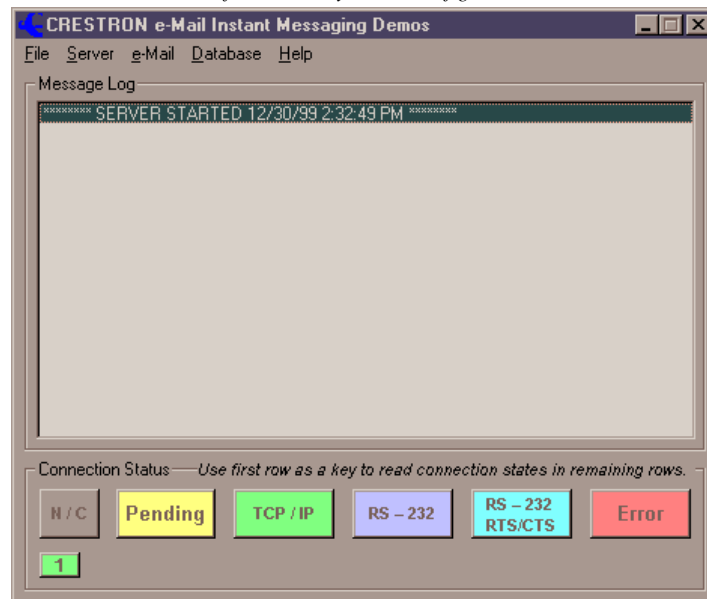
Server Windows and Menus

This section contains descriptions of the server's two main windows, the *Server Monitor* window and the *Signal Analyzer* window.

The Server Monitor Window

The *Server Monitor* window is the server's main window. It appears upon starting up the server application, and remains open until the server application terminates.

The Server Monitor window. The server protocol has been started with a single system connected via TCP/IP. Note the name of the currently loaded configuration in the title bar.



The Message Log Frame

When the server protocol is running, the *Message Log* frame shows status and error messages; for example, it lists each time the protocol is started and halted.

The System Connection Status Frame

This frame contains a colorful legend above a series of small numbered rectangles, representing each of the defined systems. The color of a control system's rectangle indicates its connection status, according to the legend. When the server protocol is not running, all systems show a status of "Not Connected" (gray). In figure above, the server protocol has been started. There is only one system defined and its status is "[connected via] TCP/IP" (green) meaning that a successful TCP/IP connection has been made to that system. The other possible states are "Waiting [for connection or disconnection]" (yellow), "[connected via] RS-232" (blue), and "Fault" (red). If a system cannot connect, it turns red and stays that way until the next connection attempt. The protocol runs if at least one system connects successfully.

The File Menu

The following command is only available when the server protocol is halted:

- **Configuration file....** This command can be used to instantly reconfigure the server by indicating an alternate configuration settings file. Any configuration changes made henceforth are saved to this new file. The name of this file is stored in the Windows registry and becomes the default configuration. *Use*

this command to select the appropriate configuration file for each demo before running it.

The following command is always available:

- **Exit** terminates the server application. If the server protocol is running, a warning message appears.

The Server Menu

Before the server protocol is started, the following commands are available:

- **Server | License...** opens the *e-control Software Server – Upgrade/Transfer License* window for licensing and activating the various server components.
- **Server | Configure...** opens the *Configuration Options* window described beginning on page 20.

To start the server protocol, use one of the following commands:

- **Server | Start** connects to the control systems and starts the server protocol. If no successful connections are made, the protocol remains halted.
- **Server | Start w/Signal Analyzer** connects to the control systems, starts the server protocol, and opens the *Signal Analyzer* window (see below).
- **Server | Start without connecting** opens the *Signal Analyzer* window and starts the server protocol but without connecting to the control systems. This is useful for testing server behavior simulating incoming signals and **watching** the signals generated in response (which are not actually sent).

The above commands all become disabled (dimmed) when the server protocol starts, whereupon the following signals, normally disabled, become enabled:

- **Server | Stop** halts the server protocol.
- **Server | Signal Analyzer** opens or closes the *Signal Analyzer* window. When this item is checked, the window is opened. When it is unchecked, the window is closed.

The remaining commands are always available:

- **Server | Log | Timestamps.** Selecting this option puts a checkmark next to it and henceforth all log items will contain a timestamp of the form `hh:mm:ss` (24-hour clock) at the beginning of each line. Selecting the command again removes the checkmark and timestamps will no longer be included in the log.

NOTES:

1. This option affects the server log and the signal log in the *Signal Analyzer* window as well.
 2. This option is “sticky” — meaning that its most recently set state is saved in the Windows Registry and is automatically applied to the option the next time the window is opened.
-

- **Server | Log | Clear** clears the message log.

The e-Mail Menu

This menu contains a single command, **e-Mail | Tables**, which opens the *e-Mail Tables* window. This window provides display and edit access to three essential tables in the database file named in *Configuration Options* window. These tables are used for composing outgoing e-mail messages. See “Tables for sending e-mail” beginning on page 43 for more information.

The Database Menu

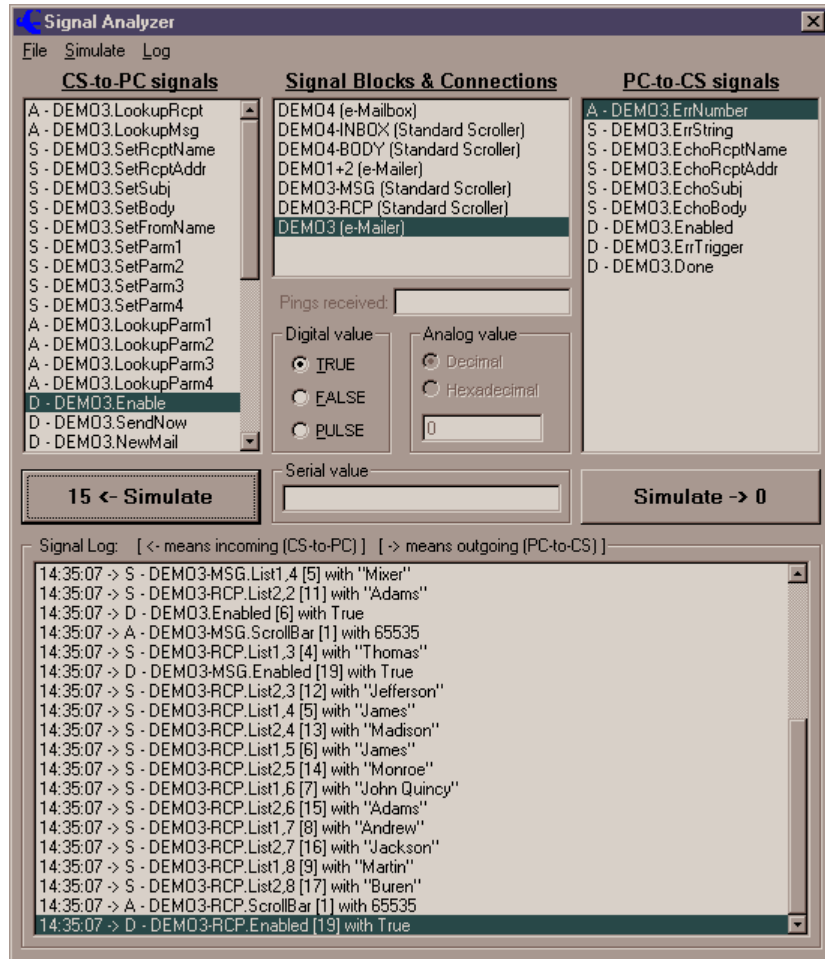
This menu contains a single command, **Database | Queries Table**, which opens the *Queries Table* window. This window provides display and edit access to this essential table in the database file named in *Configuration Options* window. See

“Editing the Queries table” in the SW-DBM manual (Doc. 5823) for more information.

The Signal Analyzer Window

While the server protocol is running, the **Server | Start w/Signal Analyzer** command from the *Server Monitor* window opens the *Signal Analyzer* window. (If the server is already running, toggling **Server | Signal Analyzer** does the same thing.)

The *Signal Analyzer* window, showing the all the active signal blocks defined in the demo configuration. The *DEMO3* signal block is selected. Therefore the *DEMO3* signals are displayed in the lists to the left and right. The **Timestamps** option is on; the **Debug Info** option off.



Signal Simulator

The top part of the window is for simulating receipt of incoming signals and transmission of outgoing signals.

Signal Blocks & Connections

This list contains all active signal blocks as well as all active connections that have signals defined (and hence can behave as signal blocks too).

To simulate an incoming or outgoing signal, you must first select an item from this list.

Signal lists

There are two lists which are displayed for all signal blocks (including system signal blocks for systems with signals defined), an incoming list on the left labeled *CS to PC signals* which contains all of the signals that go from the control system (CS) to the server (PC); and an outgoing list on the right labeled *PC to CS signals* which contains all the signals that go from the server to the control system.

The letters A for “analog”, S for “serial” (or “string”), or D for “decimal” preceding each signal in the lists indicate the type of signal expected (incoming) or to be sent (outgoing).

NOTE: A special feature of the server converts an analog signal to a string when that signal is received with a signal number that expects a serial signal.

To simulate a signal, after selecting your signal block, select an item from one of the signal lists. You are now ready to send the signal. To do so, give the **Simulate | Incoming** or **Simulate | Outgoing** command (*see below*).

The value frames

Values for simulated signals are entered here. (*See Incoming and Outgoing commands, below.*)

Signal Log

The bottom part of the window logs all signals going back and forth from all signal blocks to and from all control systems.

Each signal logged consists of the following:

- An optional timestamp (see below); followed by
- an incoming signifier (<-) or an outgoing signifier (->); followed by
- one of the letters A (“analog”), S (“serial” or “string”), or D (“decimal”); followed by
- the name of the signal block the signal is a part of (based on its signal number and the connection through which it has come or will go); followed by
- the name of the signal; followed by
- the signal number (relative to the start of the signal block) ; and, finally,
- the value of the signal.

There are two special signal values, “[Blank]” which indicates a null string and “Pulse” which indicates a true/false sequence for outgoing digital signals. (Pulse is never shown for incoming signals.)

NOTES:

1. The capacity of the log is limited to the 32,767 youngest (most recent) signals.
 2. The present release does not dump the log to a disk file.
 3. Signals are only logged when the Signal Analyzer window is opened. However, in general, do not keep the window opened unnecessarily as the logging routines can cause a noticeable degradation of server responsiveness when the server is running on a slower PC.
-

The File Menu

The only currently implemented commands in this menu print the input and output signal lists for the currently selected signal block (**File | Print Signal List | Selected**), or for all active signal blocks (**File | Print Signal List | All**). This printout can be used to create matching **Intersystem Communication** symbols in

SIMPL Windows. To this end, the lists contain signal labels identical to the labels used in that symbol.

The Simulate Menu

This menu contains the following commands for simulating signals. Simulated signals are added to the Signal Log on the bottom portion of the window, just like real signals.

- **Simulate | Incoming** simulates receipt of the signal currently selected in the incoming (“CS-to-PC”) signal list. Before issuing the command, set the value to be “received” with the signal by entering data into one of the three value frames. The frame to use is based on the signal selected. This function is also available by clicking the **rx** button under the incoming signal list.
- **Simulate | Outgoing** simulates receipt of the signal currently selected in the outgoing (“PC-to-CS”) signal list. Before issuing the command, set the value to be transmitted with the signal by entering data into one of the three value frames. The frame to use is based on the signal selected. Note that simulating an outgoing signal when the system associated with the signal block is not connected has no practical effect. This function is also available by clicking the **tx** button under the incoming signal list.

The Log Menu

This menu contains the following commands that affect the Signal Log display:

- **Log | PINGs and PONGs.** Uncheck this item to suppress logging of the **Ping** and **Pong** signals available in the system signal blocks. The intent would be to keep the log uncluttered when a control system has been programmed to ping to server on a periodic basis.
- **Log | Clear** clears the signal log.
- **Log | Timestamps.** Selecting this option puts a checkmark next to it and henceforth all log items will contain a timestamp of the form `hh:mm:ss` (24-hour clock) at the beginning of each line. Selecting the command again removes the checkmark and timestamps will no longer be included in the log.
- **Log | Debug Info.** Selecting this option puts a checkmark next to it and henceforth all logged signals as well as the **rx** and **tx** buttons will contain additional signal information.

The log items normally contain an element of the form `[n]` where `n` is the signal number relative to the start of the signal block. When **Debug Info** is checked, this element takes on the form `[com(ch):m=n]` where `com` is the COM Settings name; `(ch)` is the Virtual COM Port channel number (included only when `> 1`); `m` is the absolute signal number (`n + the signal block's offset`) (included only when different from `n` — *i.e.*, only when the offset is non-zero); and `n` is the relative signal number (as above);

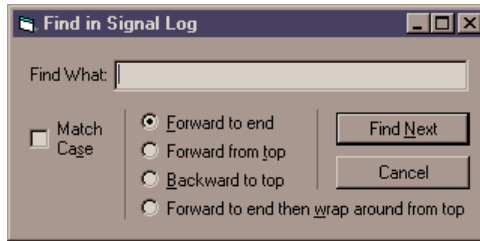
The **rx** and **tx** buttons each normally contain a number, `n`, the relative signal number of the currently selected signal from the respective signal list. When **Debug Info** is checked, the buttons each contain additional information of the form `m=n` where `m` is the absolute signal number (included only when different from `n`).

Selecting the command again removes the checkmark and the additional information is excluded.

This option is also “sticky” like the **Timestamps** command.

- **Log | Find...** brings up the following modal window which helps locate a specific signal.

Signal log search window.



Searches can be performed with and without case sensitivity by checking the *Match Case* option. When the window opens this option is unchecked — meaning that the search algorithm disregards the upper- and lower-case status of the characters in the search key.

Also when the window opens, the search direction is set to *Forward to end* meaning that the search will begin with the signal on the highlighted line and will continue to the end of the log. The other options are *Forward from top* which searches the entire log; *Backward to top* which searches beginning with the signal on the highlighted line and continuing back to the beginning of the log; and *Forward to end then wrap around from top* which also searches the entire log starting with the signal on the highlighted line.

The e-Mail Database Tables

The supplied sample e-Mail Database file (**maildemo.mdb**) contains 22 tables:

| Table | Viewable from Server | Description |
|---|----------------------|--|
| Queries | ✓ | Table used by scrollers to make queries into eMail_XXX tables. |
| eMail_Msg eMail_Addr eMail_Subst | ✓ ✓ ✓ | Tables of messages, addresses, and substitution text used by all <u>e-mailer</u> signal blocks. |
| Inbox1 to Inbox5 msg1 to msg5 (rcpt1 to rcpt5) | | Tables for five IN boxes and five message displays used by scroller signal blocks referenced by up to five separate e-mailbox signal blocks. (See below regarding recipient tables.) |

The design of these tables have certain minimum requirements, particularly their names and the names of the required fields therein. *[A future release may remove this constraint in order to accommodate pre-existing databases.]*

Table design

All tables consist of a set of congruent records (rows), each containing an identical set of fields (columns). The tables retain their particular structure even when empty.

The Queries table

The `Queries` table is used by scroller signal blocks to form queries which provide access to the data in the other tables. Using the **Database | Queries Table...** command, you can view this table's data. As supplied, the `Queries` table in the sample database file is already set up for accessing the other tables. For more information on the structure and use of the `Queries` table, refer to the SW-DBM manual, Doc. 5823.

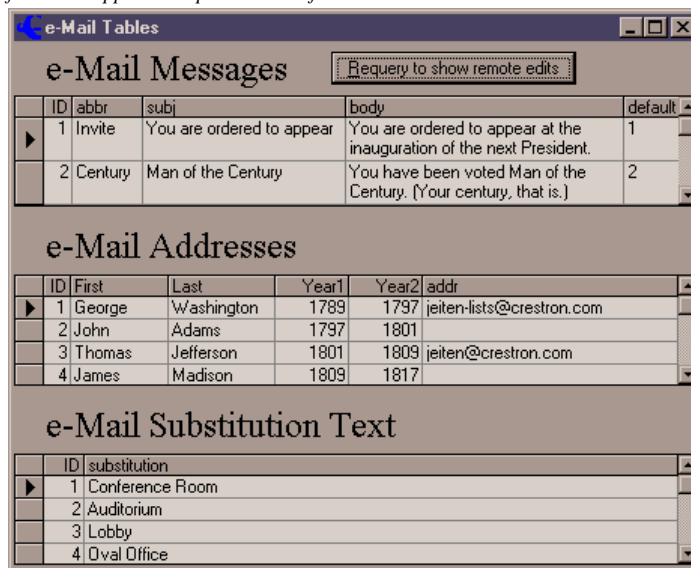
Tables for sending e-mail

The *e-Mail Tables* window (shown on the next page) provides access to all three tables used by the e-mailer signal blocks for sending e-mail. However, in the figure, which shows the contents of the supplied sample database file, you can see that a number of records have already been added to each table. You may edit the fields right in this window (see “Editing the Existing Database” on page 46).

All the fields visible in this window represent required fields which must not be eliminated. Using other software, such as Microsoft® Access, you may insert additional fields into the tables for your own purposes. Such fields will be accessible in the *e-Mail Tables* window. Although not used by the e-mailer signal block, these fields can appear indirectly, as columns in touchscreen “scrollers” bound to the e-Mailer signal block.

One thing to note about these three tables is the one field they share in common. The *ID* field is used in all three tables a “primary key” for looking up data. As a primary key, the value of this field must be unique for each record in a given table.

The “e-Mail Tables” window, showing contents of tables from the supplied sample database file



The following subsections discuss the interesting features of each table shown above. Note that all defined e-mailers will use the same three tables.

The `eMail_Msg` Table

Each record in the `eMail_Msg` table must contain at least the following fields: *ID*, *abbr*, *subj*, *body*, and *defaultAddr*. The *defaultAddr* field can either be blank or may contain a reference to a single default recipient, represented by the ID number of a record in the `eMail_Addr` table.

The eMail_Addr Table

Each record in the **eMail_Addr** table must contain at least the following fields: *ID*, *first*, *last*, and *addr*. An entry with null *addr* field contents is still useful as a way of labeling an outgoing message which will be sent to the default recipient as described by the e-mailer signal block defaults.

The eMail_Subst Table

Each record in the **eMail_Subst** table must contain at least the following fields: *ID* and *substitution*. This table supports text substitution. In brief, the purpose of this table is as follows:

Signals from the control system define the contents of text substitution “registers.” The register is defined either directly by the string data of a serial signal or indirectly by the analog datum of a lookup signal. In the latter case, the datum is used to lookup a record in the **eMail_Subst** table, based on the *ID* field. The *substitution* field of the matching record contains the new value of the register.

For more information about text substitution, see “Text Substitution Directives,” 48.

NOTE: Normally, there is no direct relationship between the text substitution directive’s parameter value and the *ID* field of a particular record in the **eMail_Subst** table. However, for directives referencing undefined text substitution registers, the *Substitution Defaults* checkbox (see “Lookup Substitution Defaults Checkbox,” page 23) can reverse this statement.

Tables for receiving e-mail

The following subsections discuss the tables available to an e-mailbox signal block. There are two such tables, *InBoxn* and *msgn*. They are maintained dynamically by the activity of the signal block, and are not accessible from the *e-Mail Tables* window.

These tables are for displaying an IN box and displaying a selected message. Both tables are optional. The tables are hooked to the signal block by referencing scroller signal blocks in the *e-Mailbox Signal Block Definition* window. The referenced scroller’s query numbers refer to entries in the *Queries* table. Each entry in turn refers to a specific IN box or message table. The names of these tables are not “hard-wired”; they are taken from the *Table* field of the *Queries* table.

IN Box Table

The IN box table is completely maintained by the e-mailbox signal block. It is populated when the signal block downloads new mail; it is de-populated when the control system issues the Delete signal. There should never be a need to edit these tables directly. For your edification, the fields in these tables are summarized below. Note that the field names are “hard-wired” in the server and must not be changed.

| Field | Description |
|-------------|---|
| ID | e-Mail host’s unique message ID. Used by the server to positively identify messages already downloaded to avoid downloading them again. |
| msgDate | Verbatim text from messages date header. |
| msgFromName | Name of sender (if available). |
| msgFromAddr | Address of sender (always available). |
| msgSubj | Verbatim text from messages subject header. |
| msgRecp | Recipient list consisting of either the name (when |

| Field | Description |
|---------|---|
| | available) <i>or</i> the address (but not both) for each recipient, separated by carriage-returns. |
| msgBody | Verbatim text of message. |
| msgNew | Indicates if message has been read yet or not. Used by server to produce the new and old “mail views.” |
| msgDel | Indicates if the message has been marked for deletion (<i>i.e.</i> , the Delete signal was received following a Pick_n signal that opened this message). |

Message Display Table

Like the IN box table, the message display table is completely maintained by the e-mailbox signal block. The structure of this table is simply a series of 80-character lines with an ID field for proper ordering. The contents of this table are transient; it emptied and repopulated every time the server opens a mail message.

Recipients Table

(Recipient tables are not used by the server at this time. A future server upgrade may support a scroller for recipients as well. The table is included here so that today’s database file will be compatible with tomorrow’s upgrade.)

Supplied Empty Tables

Creating new IN box and message tables is not possible from the server. To create a new IN box table, use Microsoft® Access to copy one from the supplied sample database file. (You only need to copy the structure of the table, not the data.)

As a courtesy to those users who do not have a copy of Microsoft® Access handy, the supplied sample database file already contains five (5) sets of empty IN box and message tables ready for use by up to five distinct e-mailbox signal blocks. Looking at the *Queries* table (using the **Database | Queries Table...** command), you can see five pairs of entries that refer to these five sets of tables. The essential information is summarized below:

| Query Number (ID) | IN box table name (Table) | Query Number (ID) | Message table name (Table) |
|-------------------|---------------------------|-------------------|----------------------------|
| 10 | Inbox1 | 11 | msg1 |
| 12 | Inbox2 | 13 | msg2 |
| 14 | Inbox3 | 15 | msg3 |
| 16 | Inbox4 | 17 | msg4 |
| 18 | Inbox5 | 19 | msg5 |

The first set, queries 10 and 11, are used in Demo4.

Feel free to change the query numbers if you like (being sure to change them as well in your scroller signal block definitions!), but without MS Access, you cannot change the names of the tables to which they refer, so do not change the table names in the *Table* field.

Editing the Existing Database

It is not necessary to have Microsoft Access to view or edit the e-mail tables; they can be edited directly from within the server application. Open the server and select **e-Mail | Tables**. The *e-Mail Tables* window opens to display the contents of the `eMail_Msg`, `eMail_Addr`, and `eMail_Subst` tables of the database file named in the *COM Settings* tab of the *Configuration Options* window. You can also use the **Database | Queries Table...** command to open (and edit) the **Queries** table.

NOTE: The IN box and message tables are not viewable (nor editable) from the server. Normally these tables would not require any maintenance. The tables can only be viewed using Microsoft® Access (available separately).

Each “data grid” in this window features the following interactivity for editing its respective table:

- For viewing purposes (only), tables can be sorted by clicking on the head of each column (field).
- Resize the window by clicking and dragging on the lower right corner. Note that column widths adjust proportionately.
- A column’s width may be adjusted manually by clicking on the far right of its header and dragging left (to reduce its width) or right (to augment its width).
- Row height can also be adjusted (for all rows at once) by similarly clicking between rows in the row’s left margin and dragging up or down. (This is useful to know when a cell’s contents wraps onto additional lines.)
- Records can be deleted by selecting the entire row (by clicking in the row margin), and depressing the **DEL** key.
- The information in each cell can be replaced by simply highlighting the cell and typing over its contents; or you may click once to highlight the cell and again to precisely place the cursor for editing purposes.
- Records so changed (as evidenced by the little pencil icon in the row margin) can be changed back by selecting a cell or an entire row (by clicking in the row margin) and depressing the **ESC** key.
- Additional records can be entered by scrolling down to the row containing an asterisk (*) in the left-most column and typing the information.
- Use **ENTER** to advance to the next cell in a row. The arrow keys also navigate between cells in a rows, and between rows as well.
- Use the “Requery to show remote edits” button to display changes made elsewhere since you opened the window. For example, if the database file has been placed on a file server and has recently been edited from another computer, you can display those edits by clicking this button.

NOTE: Changes are not actually made until you leave the row (or close the window). When leaving a row in which you have made a change, you will be asked whether or not to keep (“commit”) the changes. To accept the changes, click **YES**; to reject the changes, click **NO** and then use the **ESC** key as described above.

Composing e-Mail

The “Set” serial signals (**SetRcptName**, **SetAddr**, **SetSubj**, and **SetBody**) are used to send verbatim serial data to set the To: header, Subj: header, and message body text.

The contents of the subject or the body (so set) is replaced by subsequent **SetSubj** and/or **SetBody** signals, and/or the **LookupMsg** signal (see below), or picking a

record from the interactive message scroller (see below). These values are replaced continually until the **SendNow** signal is issued.

Note that the name and address are actually added to the list only when the **SetAddr** signal is issued. At that moment, the name most recently sent via **SetRcptName** is paired with the address and added to the recipient list.

When the *Allow multiple recipients* option is in effect (checkbox in e-mailer signal block definition window is checked), the recipient list can be any length. It is constructed by setting the name and address repeatedly, either with (**SetRcptName** and) **SetAddr**, or with **LookupRcpt** (see below); or picking a record from the interactive recipient scroller (see below).

When the *Allow multiple recipients* option is not in effect, only a single recipient is replaced by any subsequent use of the **SetAddr** or **LookupRcpt** signals, or picking a record from the interactive recipient scroller, until the **SendNow** signal is issued.

Recipient Lookup

The term recipient as used here refers to both the name and the e-mail address of the recipient.

A practical method of setting both the name and address is to look it up in an “address book” database table. The **LookupRcpt** analog signal looks up and adds to the recipient list the name and address found in record $ID = n$ of table **eMail_Addr** (where n is the signal value). As described above, these values either add to (build) a recipient list, or are replaced by subsequent sets/lookups/picks until the **SendNow** signal is issued, depending on the state of the *Allow multiple recipients* option.

Message Lookup

The term message as used here refers to both the subject and the body of the e-mail message.

A similar method of setting both the subject and body is to look it up in a message table in the database. The **LookupMsg** analog signal looks up the subject and body found in record $ID = n$ of table **eMail_Msg** (where n is the signal value). As described above, these values are replaced by subsequent sets/lookups/picks until the **SendNow** signal is issued.

When a message is selected with a **LookupMsg** signal (or when a user manually picks a message from a scroller — see below), the e-mailer signal block records the subject and body from the selected record. If the *Echo Lookups* option is in effect, the subject and body are echoed back to the control system (using the **EchoSubj** and **EchoBody** signals).

Implicit Recipient Name & Address

If a recipient has not yet been explicitly specified, an implicit (default) name and address is echoed. Where these defaults come from depends on the value of the *defaultAddr* field and the contents of the fields in the record in the **eMail_Addr** table referenced by this value.

If a positive, non-zero value is present in the *defaultAddr* field, the name and address from the record in the **eMail_Addr** table referenced by this value is recorded. If either (or both) fields are null, then one (or both) e-mailer signal block defaults are recorded instead.

*The contents of the first and last fields are concatenated together when echoed by the **EchoRecp** signal.*

If the *Echo Lookups* option is in effect, these data are echoed back to the control system as well (using the **EchoRecp** and **EchoRcptAddr** signals).

If the *Show Indirection* option is also checked, a > character is prefixed to the echoed strings to indicate the indirection. If e-mailer signal block default data is being echoed, a second > character is prefixed (double default). (These > characters are not recorded; they are echoed for display purposes only.)

If the *defaultAddr* field is zero (or less) (or blank), the e-mailer signal block default name and address is recorded and echoed (if *Echo Lookups* is on) (with double > characters, if *Show Indirection* is on).

All of the above only occurs when no name has been explicitly selected. If an explicit selection is subsequently made, it overrides any defaults already recorded.

The usefulness of the behavior described above can be better appreciated by playing with demo 2 (using record ID numbers), or demo 3 (using the interactive scrollers).

Manual Selection of Recipient and/or Message

e-Mail scroller signal blocks — or custom scroller signal blocks if you are licensed for the e-control Database Manager product (SW-DBM) — can be defined and “bound” to an e-mailer signal block. First create the scroller signal block, then bind it to an e-mailer signal block in the latter’s “Signal Block Definition” window. Add the **DBMScroller** macro to your SIMPL program to display on the touchscreen the database tables as interactive scrolling lists. Picking (touching) elements in these lists, depending on the list, either add a recipient to the recipient list or set up a message — in exactly the same way as if the “lookup” signals had been used.

Text Substitution and File Inclusion

There are two text substitution schemes which are applied consecutively and repetitively to both the subject header text and the body text, [text substitution directives](#), and [textfile inclusion directives](#). Consecutively means that text substitution occurs before textfile inclusion. Iteratively means that after both schemes have been applied to the text, if any substitutions were made, both schemes are reapplied. This process continues until no further substitutions occur.

Text Substitution Directives

*The number of **SetParm** and **LookupParm** signals defined in a typically e-Mailer signal block are four (4) signals (each).*

[Text substitution directives](#) reference a set of [text substitution registers](#). Think of text substitution registers as numbered cubbyholes, each containing a text string. Text substitution registers are referred to by number, starting from 1 and continuing up through *n*, where *n* is defined as the greater of the number of **SetParm** signals and **LookupParm** signals defined for a particular e-Mailer signal block.

The text substitution routine scans the text for directives, represented by constructs with the following syntax: ~{*m*} where *m* is a base-10 number from 1 to *n*. Wherever such a construct is found in the text, it is replaced with the current contents of the text substitution register.

Note that directives do not require any further delimitation. That is, a directive does not need to be surrounded by spaces or any other punctuation. This makes it possible for a directive to appear in the middle of a word, which is often useful.

*An e-mailer signal block is enabled by assertion of its **Enable** signal from the control system.*

All registers are undefined when the e-Mailer signal block is enabled. A register is defined by receipt of a **SetParm** and **LookupParm** signal. Register definitions are "sticky" — which means that they persist and can be referenced in successive e-Mail messages. The registers remain defined until the server protocol is halted.

If a directive references an undefined register, or the number in the directive is out of range (less than 1 or greater than *n*), the string "..." is inserted into the text in place of the erroneous invocation.

If, however, the *Lookup substitution defaults* check-box is checked under the *e-Mail* tab, an undefined or out of range directive is replaced by a value from the **eMail_Subst** table. This would be the value in the substitution field of the record with an ID value equal to the bogus register number (for more information on the **eMail_Subst** table, see “The eMail_Subst Table” on page 44). If there is no such record, or if its substitution field is null, the string "..." is inserted into the text in place of the erroneous invocation. (You might want to arrange your eMail_Subst table such that the first *n* entries are true defaults for the undefined in-range directives; and all remaining entries can be thought of as defining the out-of-range directives.)

Be aware that because text substitution does not take place until the mail is actually sent (upon receipt of the SendNow signal, or a Shortcut signal), the registers invoked by the text only need to be defined prior to that time — which could be subsequent to specifying the subject and/or body text.

Text File Inclusion

A future release will permit an alternate folder to be named in the e-Mailer signal block Definition window, a pathname either relative to the application folder, or absolute (fully qualified).

The text substitution routine scans for file inclusion directives, represented by a constructs with the following syntax: `~[abc]` where `abc.TXT` is used as the name of a file in the application folder. This file is opened and its entire contents is inserted into the text in place of the construct. If the file cannot be opened or read from, the string "..." is inserted instead.

Rescanning

Inasmuch as such text substitution operations can obviously create new substitution directives, the entire text is now rescanned.

The above schemes allow for tremendous flexibility. For example:

- Substitution parameters and text files can contain further substitution directives.
- Text filenames can reside in the original text, or the text can contain parameter substitution directives and the substitution strings can contain file names.
- Two adjacent substitutions can resolve to the left and right halves of a new substitution directive.

Control Messages

Control messages are e-mail messages with embedded signals. When the server reads one of these messages, it interprets the text and sends the embedded signals immediately. It then deletes the message from the server. Such messages are never placed in the IN box (if there is one).

Identification

For the server to recognize control messages, the *Process control messages* option must be checked in the *e-Mailbox Signal Block Definition* window. Otherwise, none of the following applies, and messages which might have been interpreted as control messages are instead interpreted as normal messages.

NOTE: e-Mailbox signal blocks must either have an IN box (*i.e.*, a scroller bound to the signal block to act as an IN box), or have the *Process control messages* option checked. If neither of these options are selected, there is nowhere for the mail to go. This is an error condition and if it exists, the server protocol will not start. ¶If there is an IN box but *Process control messages* is not checked, all messages are downloaded to the IN box. ¶If there is no IN box and *Process control messages* is checked, only control messages are considered. Other messages are ignored (and possibly deleted if the *Delete messages as downloaded* option is checked — which is strongly recommended to avoid having to download the headers of all such messages every time mail is checked).

Spaces and case are ignored. For example, a control message could have a subject header starting with `ctrl >` [with a space before the angle bracket].

Secondly, for the server to recognize a control message as such, it must conform to the definition of a control message which is any message with the string `CTRL>` at the start of the subject header. Once the server decides that a message is a control message, the message must conform to certain syntax rules. Otherwise, error messages appear in the server log.

Message types

There are two basic types of control message:

- A single-signal control message where the signal is in the subject line itself. In this case, the server never downloads the body text from the e-mail host, which is clearly a more efficient way to go when there is only one signal to transmit to the control system.
- A multi-signal control message where the subject line contains the simple command `SCRIPT`. When the server encounters such a message, the body text is downloaded and any signals found therein are transmitted to the control system.

Syntax

Whether a single control signal appears in the subject line, or multiple signals are dispersed throughout the body text, the syntax is the same.

There is one signal per line. All such lines begin with `CTRL>` followed by a signal equate, as follows:

`CTRL> type number = value`

where:

| | |
|---------------|--|
| <i>type</i> | is A , D , or S , for analog, digital, or serial, respectively. |
| <i>number</i> | is the number, n , in the SignalA_n , SignalD_n , or SignalS_n signal. |
| <i>value</i> | is arbitrary, subject to the syntax for the signal type. |

The syntax for signal values is as follows:

| | |
|--------|--|
| Analog | An unsigned 16-bit value represented either as: A decimal (base 10) integer, in the range is 0 to 65,535. An hexadecimal (base 16) integer, in the range \$0000 to |
|--------|--|

| | |
|---------|---|
| | <p>\$FFFF. See “Base 16 used for notational purposes” on page 66 for more information about hexadecimal.</p> <p>Embedded spaces are ignored.</p> |
| Digital | <p>To assert the signal, specify one of: <i>Assert, True, On, 1, Yes, Set, Ok, or Okay</i></p> <p>To de-assert the signal, specify one of: <i>Deassert, Deassert, False, Off, 0, No, Reset, or Cancel</i></p> <p>Case is not significant and embedded spaces are ignored.</p> |
| Serial | <p>Up to 83 characters. Note that the string is “trimmed” which means that leading and trailing spaces are ignored. Case and embedded spaces are retained verbatim.</p> |

Example

The following example of a multi-signal control message can be sent to the control system’s mailbox for reading by Demo4. Using your usual mail client software, create a new e-mail message with

```
CTRL> SCRIPT
```

in the subject header. Paste the following script into the body of the message. (Be sure to neutralize any formatting that may come along with the paste action!)

```
Test script. Any lines that do not begin with CTRL> are
ignored and can be used for comments (such as this line). (1)
Turn on first and last digitals:

CTRL>d1=yes
CTRL>d3=on

Notice in the above that 'yes' and 'on' are equivalent; see
documentation for additional synonyms. (2) Set the three
analog gauges to 1/3, 2/3, and 3/3:

CTRL>a1=21845
CTRL > a2 = 43 691
CTRL>a3=65535

Notice in the above that spaces are generally ignored. (3) Set
some indirect text:

CTRL>s1="Cogito, ergo, sum."
CTRL>s2=[I think; therefore I am.]
CTRL> s3 = - René Descartes

Spaces before the equals sign are ignored as usual. Although
the serial signal strings are "trimmed" (which means that
leading and trailing spaces are removed), spaces embedded
within the strings are sent verbatim. Therefore, the spaces
within the above quote are properly retained. However, the
three spaces before the dashes in the second string are trimmed
off.

CTRL>end

The above END command is optional. When present, it terminates
scanning of the text at that point. Therefore, the following
command is ineffectual:

CTRL>d2=1
```

```
-- end of file --
```

Demos

Four demonstrations on the use of *e-control Mail* are included with the package. Each demo is described along with an accompanying “bird’s eye view” diagram of its SIMPL program. All demos use the following three files:

- A VT Pro-e source file (**demomail.vtp** file), containing pages for all four demos, ready to be compiled LC-3000 touchpanel.
- A compiled touchscreen file (**demomail.hex** file); derived from the above; ready to upload to an LC-3000, CT-3000, CT-3500, or VT-3500.
- A Configuration file (**demomail.ini**) configures the server for all four demos.

In addition, each individual demo has a folder containing the following files:

- A SIMPL Windows source file (**demo?.smw** file), ready to be compiled for a CNMSX-PRO control system (“PRO” = front panel with LCD display) with a CNXENET (Ethernet) card in the DPA slot.
- A compiled control system program file which uses serial RS-232 communications (**demo?COM.bin** file); derived from the above by commenting off the Virtual COM port; ready to upload to such a control system.
- A compiled control system program file which uses EtherNet communications (**demo?TCP.bin** file); derived from the above by commenting off the serial COM port; ready to upload to such a control system.

The supplied SIMPL and VT Pro-e files may require conversion prior to compiling and uploading if your target touchpanel is not one of those mentioned above. Conversion is a simple matter using VT Pro-e.

Before attempting to run the demos, use the **File | Configuration File...** command to make sure the server is using the configuration file named above.

Demo 1: The Shortcut Signals

This example demonstrates use of the digital “shortcut” signals. A single pulse of one of these signals sends a predetermined message to a predetermined recipient.

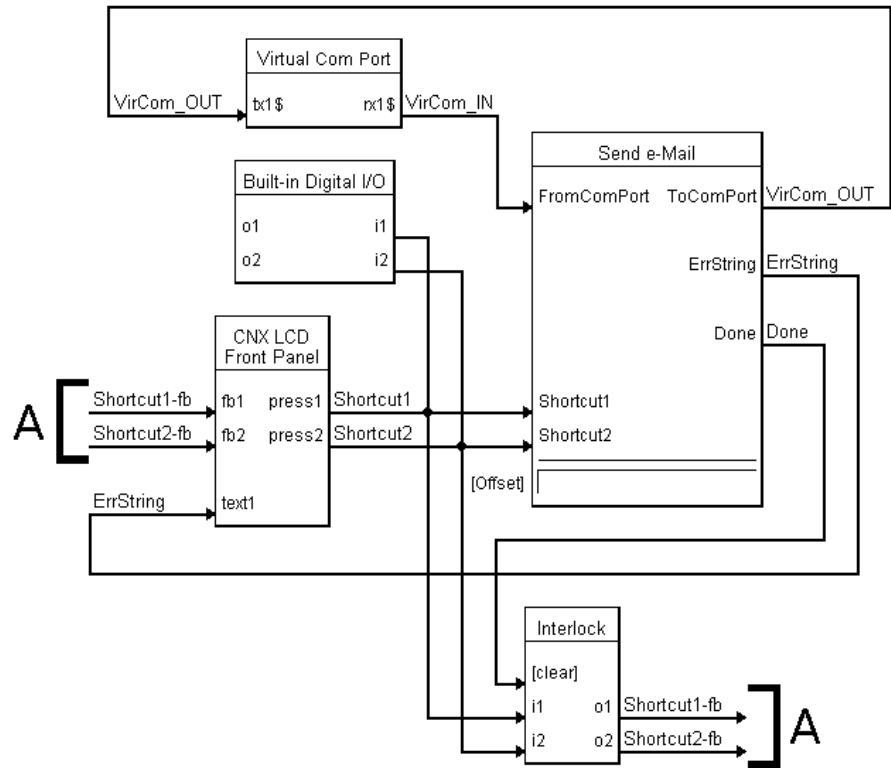
In this example, the control system senses a problem with one of the devices it controls. A typical problem might be a projector bulb failure. The control system reacts to the problem (in this case, a closure on pins 1 or 2 of the digital I/O device) by producing a digital pulse which is routed to one of the shortcut signals on the **Send e-Mail** symbol. Refer to the block diagram on the next page for an overall graphical representation of this first example:

Notice that the “bulb failure” can be also be reported by depressing one of the function buttons on the CNMSX-PRO front panel. The digital signal assertion triggers the control system to send a *Shortcut_n* signal via a **Send e-Mail** or an **Intersystem Communication** symbol to the *tx* stream and out the COM port (or LAN port, if using TCP/IP) to the server.

The server receives the signal and responds by sending a predetermined (constant) e-mail message to a predetermined (constant) recipient. The origin of the message and recipient are provided from a lookup performed on the **eMail_Msg** table (using the first record with *ID* = the shortcut number), and, indirectly, the **eMail_Addr**

table (using the first record with *ID* = the value of the message record's *defaultAddr* field).

Demo 1 Block Diagram



NOTE: This block diagram shows the use of a virtual COM port for TCP/IP communications with the server. For serial communications using RS-232, the virtual COM port is simply replaced with a real COM port.

Demo 1 SIMPL Windows Program

Building this simplest example is trivial. Only two items need to be added to the control system in the Configuration Manager workspace in SIMPL Windows. From the *Control Systems* folder in the *Device Library* select CNMSX-PRO; and also add a two-way serial driver to a COM port on the CNXCOM-6 board from the *Serial Drivers (General)* folder (or add a Virtual Communication Port from the *Ethernet Control Modules* folder and a driver as above).

In the Programming Manager, the other two blocks are added to the system from the *Logic Symbols* folder in the *Symbol Library*: **Interlock** is selected from the *Memory* sub-folder and **Send e-Mail** is selected from the *System Control* sub-folder.

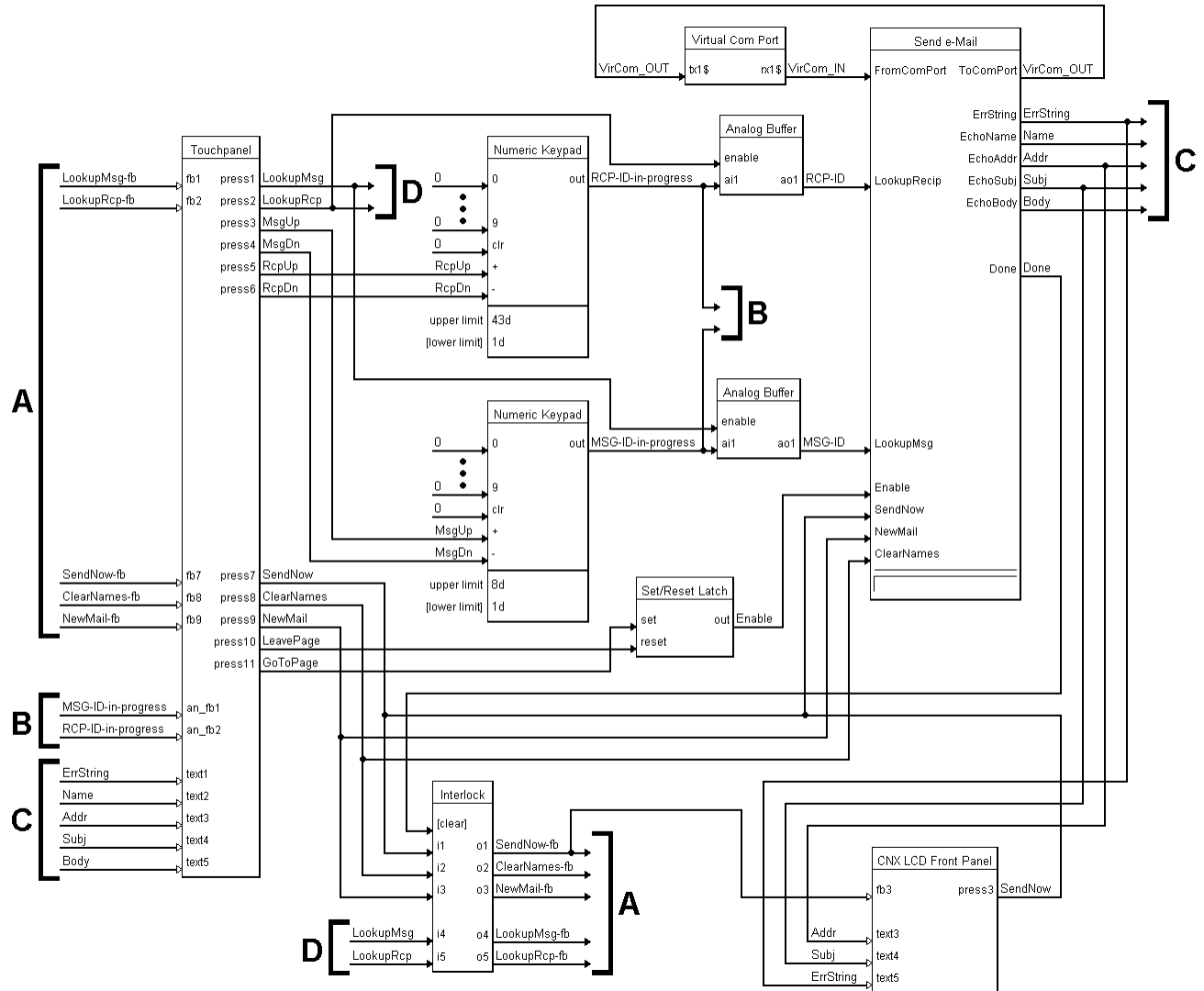
Demo 2: The LookupMsg and LookupRcpt Signals

This example demonstrates use of analog “lookup” signals. These signals choose messages and recipients based on the signal values. An additional signal, **SendNow**, is required to actually send the constructed message.

A “lookup” requires the program to search for various information in a database table based on a search key (a record ID number). Demo 2 illustrates the use of the lookup signals available as inputs to the supplied **Send e-Mail** symbol. Refer to the block diagram after this paragraph for an overall graphical representation of this

example. Note that the **Send e-Mail** symbol's *rx* and *tx* streams are connected to the (virtual) COM port symbol.

Demo 2 Block Diagram



NOTE: This block diagram shows the use of a virtual COM port for TCP/IP communications with the server. For serial communications using RS-232, the virtual COM port is simply replaced with a real COM port.

LookupRcpt is an analog signal whose value represents a “key” (a record ID number) which is used to “look up” a name and address from an address book table. **LookupMsg** is used similarly to “look up” a subject and a message body from a table of messages.

Notice how **LookupMsg** also supplies a default recipient name and address through the message table's *defaultAddr* field, or, if no name and address are discovered that way, using the e-mailer signal block's defaults. This behavior is laid out in detail in the section titled “Implicit Recipient Name & Address,” page 47. However, it is much easier to appreciate when you see it in action here (and in demo 3 as well), by performing the following actions:

- Choosing the “Invite” message (ID=1 in the **eMail_Msg** table) provides the default address for Thomas Jefferson (ID=2 in the **eMail_Addr** table).
- Choosing the “Century” message (ID=2 in the **eMail_Msg** table) provides no default address.
- Choosing any other message from the **eMail_Msg** table provides the default address for the system manager (ID=0 in the **eMail_Addr** table).

Demo 2 SIMPL Windows Program

Two items need to be added to the control system in the Configuration Manager in SIMPL Windows. From the *Control Systems* folder in the *Device Library* select CNMSX-PRO; and also add a two-way serial driver to a COM port on the CNXCOM-6 board from the *Serial Drivers (General)* folder (or add a Virtual Communication Port from the *Ethernet Control Modules* folder and a driver as above).

In the Programming Manager, the other blocks are added to the system from the *Logic Symbols* folder in the *Symbol Library*: **Interlock** and **Set/Reset Latch** are selected from the *Memory* sub-folder. The two **Numeric Keypads** and two **Analog Buffers** are selected from the *Analog Operations* sub-folder. **Send e-Mail** is selected from the *System Control* sub-folder.

Demo 2 VT Pro-e Program

Page 2: Sending e-mail via database lookup: The first page of demo 2 allows the user to select either TCP/IP or serial setup instructions. Additional pages provide brief setup instructions for the control system and server for either TCP/IP or serial (depending on choice from first page). (These instructions are detailed in “Communications Setup,” beginning on page 15, herein.) Either path leads to the same demo page.

NOTE: Before proceeding to the actual demo page, start the server protocol by issuing the **Server | Start** command.

Page 2-1: Lookup gauges and message display: The final page of the VT Pro-e program is the actual demo screen. The arrow buttons in areas 1 and 2 are used to increase or decrease the message and recipient ID numbers. Use the **Lookup** button to update the *To:*, *Subj:*, and text (unmarked) fields with the information in the database tables. When the desired data appears in the fields, use the **Send Now** button to send the e-mail. Use the **Clear Names** button to eliminate any names from the *To:* field. Use the **Clear All** button to empty all fields.

Demo 3: Interactive Scrollers

This example demonstrates use of two “scrollers,” signal blocks which service interactive scrolling lists on the touchscreen, representing views into message and recipient database tables. Picking (touching) an item on one of these lists triggers a database lookup (similar to the lookup signals used directly in Demo 2). An additional signal, **SendNow**, is required to actually send the message so constructed.

Demo 3 illustrates the setup of the **DBMScoller** macros, with their many inputs and outputs connected to appropriate buttons and indirect text fields on the **Touchpanel** symbol; and the setup of the **Send e-Mail** symbol as well. Refer to the block diagram on the next page for an overall graphical representation of this example. Note that both the **DBMScoller** macros and the **Send e-Mail** symbol have *rx* and *tx* streams that must be connected to the (virtual) COM port symbol.

It is also worth noting that there is no direct connection between the **DBMScroller** macro and the **Send e-Mail** symbol. Interpretation of scroller picks as e-mail composition actions occurs within the server as the scroller signal blocks make calls on the e-mailer signal block.

Demo 3 Block Diagram

[Figure unavailable]

NOTE: This block diagram shows the use of a virtual COM port for TCP/IP communications with the server. For serial communications using RS-232, the virtual COM port is simply replaced with a real COM port.

Demo 3 SIMPL Windows Program

Two items need to be added to the control system in the Configuration Manager in SIMPL Windows. From the *Control Systems* folder in the *Device Library* select CNMSX-PRO; and also add a two-way serial driver to a COM port on the CNXCOM-6 board from the *Serial Drivers (General)* folder (or add a Virtual Communication Port from the *Ethernet Control Modules* folder and a driver as above).

In the Programming Manager, the other blocks are added to the system from the *Logic Symbols* folder in the *Symbol Library*: **Interlock** and **Set/Reset Latch** are selected from the *Memory* sub-folder; the two **Numeric Keypads** and two **Analog Buffers** are selected from the *Analog Operations* sub-folder; and **Send e-Mail** is selected from the *e-Mail/Database* sub-folder. The **DBMScroller** macros can be found in the *Crestron Modules* folder, *e-Mail/Database* sub-folder.

Demo 3 VT Pro-e Program

The structure of the demo 3 touchscreen pages echoes that of the demo 2 touchscreen pages (see above).

NOTE: Before proceeding to the actual demo page, start the server protocol by issuing the **Server | Start** command.

Page 3-1: Scrollers and message display: This final page of the VT Pro-e program is the actual demo screen. Pick items from the scrolling lists to compose the e-mail message. When the desired data appears in the fields, use the **Send Now** button to send the e-mail. Use the **Clear Names** button to eliminate any names from the *To:* field. Use the **Clear All** button to empty all fields.

Demo 4: Receiving e-Mail

This example demonstrates receiving e-mail into a scrolling “IN box” (list of messages) which has both a “new mail” and an “old mail” *view*; displaying any particular e-mail message in a scroller; and processing incoming “control messages.”

The scrollers are of course implemented as Standard Scroller signal blocks which are referenced in the e-mailbox signal blocks configuration. Demo 4 again illustrates the setup of two **DBMScroller** macros to service these scrollers; and the setup of the **Receive e-Mail** symbol as well. Refer to the block diagram on the next page for an overall graphical representation of this example. Note that both the **DBMScroller** macros and the **Receive e-Mail** symbol have *rx* and *tx* streams that must be connected to the (virtual) COM port symbol.

It is also worth noting that there is no direct connection between the **DBMScroller** macros and the **Receive e-Mail** symbol. Interpretation of scroller picks as IN box message selection occurs within the server as the IN box scroller signal block makes calls on the e-mailbox signal block.

Demo 4 Block Diagram

[Figure unavailable]

NOTE: This block diagram shows the use of a virtual COM port for TCP/IP communications with the server. For serial communications using RS-232, the virtual COM port is simply replaced with a real COM port.

Demo 4 SIMPL Windows Program

Two items need to be added to the control system in the Configuration Manager in SIMPL Windows. From the *Control Systems* folder in the *Device Library* select CNMSX-PRO; and also add a two-way serial driver to a COM port on the CNXCOM-6 board from the *Serial Drivers (General)* folder (or add a Virtual Communication Port from the *Ethernet Control Modules* folder and a driver as above).

In the Programming Manager, the other blocks are added to the system from the *Logic Symbols* folder in the *Symbol Library*: **Interlock** and **Set/Reset Latch** are selected from the *Memory* sub-folder; and **Receive e-Mail** is selected from the *e-Mail/Database* sub-folder. The **DBMScroller** macros can be found in the *Crestron Modules* folder, *e-Mail/Database* sub-folder.

Demo 4 VT Pro-e Program

Page 4: Receiving e-mail into an IN box: The first page of the VT Pro-e program has some introductory information about the demo and a button labeled “Go to demo screen.”

NOTE: Before proceeding to the actual demo page, start the server protocol by issuing the **Server | Start** command.

Page 4-1: New mail view: The next page is the actual demo screen, containing an area for “Control Signal Interpretation”; the DEMO4-INBOX scroller; and “check mail” button along with a progress bar. Above the IN box scroller is a pair of buttons, *New Mail* and *Old Mail*, disguised as folder tabs. Each tab also contains a digital gauge, used to indicate the number of rows in the table. *New Mail* is a dummy on this page (join=NONE) and *Old Mail* leads to Page 3 via local page flip. Pick items from the scrolling lists to display an e-mail message; the server responds with the **PageFlip** signal, used here to flip to Page 4.

Page 4-2: Old mail view: This page looks very similar to the first, containing the same scroller object group, all using the same join and indirect text numbers. Here, however, the *Old Mail* button is the dummy and *New Mail* leads back to the previous page. In addition to the local page flip to this page, the *Old Mail* button also latches the **ViewOldMail** signal which causes the scroller to requery for all mail which has already been read at least once. Similarly, the *New Mail* button resets the **ViewOldMail** signal which causes the scroller to requery for all mail which has not already been read

Page 4-3: Message view: Upon receipt of a **Pick** signal from the IN box scroller, the contents of this page are set by the server using the **Echo**___ signals to display the message headers; and the optional DEMO4-BODY scroller to display the actual message. Note that the **Pick** signals of this second scroller are all unconnected

(join=NONE). Only after the desired data has been sent, the server issues the **PageFlip** signal.

Appendix A: Theory of Operation

This section describes the operation of the e-mailer signal block as it processes signals from the control system. Signal names are shown as conjoined words with initial caps set in **bold** type, such as **SendNow**. Refer to the “Signal Reference” on page 68 for in-depth information on these signals.

Server Protocol

Run the Crestron Software Server application, **swserver.exe**, to license (**Server | License...**) and configure (**Server | Configure...**) server operation. Although the server *application* is now running, the server *protocol* must be manually started (**Server | Start**) to establish the communications link. Once started, licensing and configuration options are off limits. To change a configuration option, the server protocol must be halted (**Server | Stop**).

Signal Block Definition / Activation

Definition / activation of signal blocks is performed in the Signal Blocks tab of the Configuration Options window — which can only be accessed while the server protocol is halted.

Signal Blocks (in general) are “defined” (created) in the “Configure options” window, *Signal Blocks* tab. Once defined, they are saved in the Configuration Settings file.

Signal Blocks are not “active” (listening for signals) unless activated in the list under the *Signal Blocks* tab by checking the box next to the signal block name. The number of active signal blocks of each type cannot exceed the respective limits of your license. Should this happen, a warning appears on the *Signal Blocks* tab and the server protocol does not start.

There is no theoretical limit to the total number of signal blocks which may be defined in a given configuration. (A practical limit has yet to be determined.)

Signal Block Enable / Disable

*Whereas most digital signals are pulsed, the **Enable** signal is unusual in that it is state-sensitive.*

Once the signal block has been defined and activated, and the server protocol is running, the signal block still has very limited functionality until it is enabled with an assertion of the **Enable** signal by the control system. Most other signals require the signal block to be enabled first; otherwise they produce an “Object not loaded” error. Once enabled, system resources are loaded and the signal block to respond to signals from the control system to send or receive e-mail.

De-asserting the **Enable** signal causes the signal block to unload those system resources, making it dormant once again.

The following signals are exceptions to this rule. They can be processed without first enabling the signal block:

- **Enable** (assert) of course
- All the e-mailbox’s **Set**____ signals
- The e-mailer’s **Shortcut** signals. When received by a disabled e-mailer, the e-mailer is “auto-enabled” — but only for the duration of the signal.

Signal Block Error Reporting

All signal blocks have a *Local error signals* option to define a set of signals for the purpose of reporting of errors “locally.” Otherwise, errors are reported “globally” to the control system through the optional signal block that can be associated with a

COM Settings definition which contains a similar set of error reporting signals.. These signals include **ErrNumber**, **ErrString**, and **ErrTrigger**. See the “Signal Reference” on page 68 for details.

Appendix B: Intersystem Communications and Signal Space Considerations

The **Send e-Mail SIMPL Windows** symbol is available from the Crestron FTP site. Under **SIMPLWIN**, look for “Library update” (version equal to or greater than that specified in “Leading Specifications” on page 11).

The **DBMScroller SIMPL macro** is installed with **SIMPL Windows 1.4**. It is also installed into the **Modules** folder. For use with **1.3**, move the file to your currently set user macros folder.

The **Intersystem Communications** symbol is commonly known by its speedkey name, **XSIG**.

The following discussion applies in general to all Crestron Software Server components. Keep in mind while reading this section that use of the **Intersystem Communications SIMPL** symbol is the most general approach for setting up communications with the signal blocks defined in the server. To simplify the control system side programming, be aware of the following alternatives to the **Intersystem Communications** symbol:

- For e-mailer signal blocks, we recommend the more specific **Send e-Mail** symbol. Checking *Using “Send e-Mail” SIMPL symbol* in the signal block definition constrains the definition to ensure compatibility with this symbol.
- Likewise, Standard Scroller signal blocks can use the **DBMScroller SIMPL Windows** macro.

Each active signal block exchanges data with a particular running in a control system connected to the server. Typically, several signal blocks communicate with the same control system, using several Intersystem Communications symbols. This section discusses how to properly connect signal blocks to their target **Intersystem Communications** symbols.

System Connections

Signal Blocks are connected to **XSIGs** through a physical connection (a hardware communications port). The control system accesses the port through a driver. There are different drivers for each kind of port:

| System | Protocol | Hardware | Port | Serial Driver |
|------------------------|----------|---|-------------------------|--------------------------------------|
| CNRACK or CNMS | RS-232 | CNCOMH-2 plug in control card | A or B | CNCOMH-2 Two-way serial driver |
| | TCP/IP | <i>N o t a v a i l a b l e</i> | | |
| CNRACKX or CNMSX | RS-232 | Built-in serial port | A through F | CNXCOM Two-way serial driver |
| | TCP/IP | CNXENET direct processor access (DPA) card | NET | Virtual Communication Port |
| PC | RS-232 | Built-in COM ports or 3 rd -party serial cards | COM1 through COM8 | Manufacturer- specific |
| | TCP/IP | Network Interface (NIC) card | NET | |

In all cases, the control system receives data as a serial data stream from the driver and transmits data by sending a serial data stream to the driver. This is precisely the kind of data the **XSIG** symbols use.

Encoding and Decoding the Serial Data Stream

Analog, serial, and digital signals to be sent from the control to the server are fed into the input (left) side of an Intersystem Communications symbol which *encodes* the signals into a serial data stream, available as an output labeled $\tau x\$$ (for *transmitter*). This data stream is connected to the input side of the serial driver symbol, also labeled $\tau x\$$, and is sent out the COM port to the server.

Convenience feature: You can connect an analog signal to a serial signal input of a signal block's symbol. Upon receipt, the server automatically converts the analog value to a decimal (base 10) numeric string.

The server receives the data on a certain connection, via a certain channel number (if TCP/IP). It searches through its active signal blocks for one with a signal range that can accommodate the incoming signal. It also knows the signal type (analog, serial, or digital) and sends the signal to the appropriate method implemented by the signal block. A useful exception to this general paradigm is that when the server receives an analog signal when it expected a serial signal (*i.e.*, the signal number matched that of a serial signal, the server as a courtesy, automatically converts the analog value to a decimal (base 10) numeric string.

Data from the server received at the COM port is available on the output (right) side of the serial driver symbol, labeled $r x\$$ (for *receiver*). This data must then be *decoded* by the control system into system signals it can use. To do this, the $r x\$$ stream is connected to the input side of an Intersystem Communications symbol, also labeled $r x\$$. The outputs of this symbol are analog, serial, and digital signals.

The Intersystem Communications Symbol Signal Space

The term “signal space” refers to the number of signals available to the **Intersystem Communications** symbol connected to a given i/o stream. Specifically, there are a total of 4096 signals available. However, while the first 1024 signals may be of any type (Analog, serial, or digital), the remaining 3072 signals can only be used for digital signals. An additional constraint imposed by the SIMPL Windows compiler is that each symbol (and therefore each signal block on the server side) must list all its digital signals after all its analog/serial signals (which may be intermixed).

Since each signal block's signals are numbered consecutively within this space, and since the non-digital signals of all signal blocks must be positioned below 1024, only that portion of the space is normally of practical use.

NOTE: The various signal block definition windows all provide a dynamic display of the highest signal number currently defined. This value changes as the user selects options and enters values for the size of enumerated signal sets. This value is also affected when the user changes the value of the symbol's offset. If any of these actions would place the highest digital signal above 4095 or the highest non-digital signal above 1023, the text box containing the highest signal number turns red to alert the user. The user must bring the signals within range before attempting to activate the signal block or an error is added to the server log. Also note that when attempting to start the server protocol, an error is reported if any **Intersystem Communication** symbols sharing the same connection have signal ranges that overlap.

Large configurations are often not able to fit all their signal blocks within this space. TCP/IP connections use a Virtual COM port which offers 128 discrete channels. For such connections, each channel supports a separate i/o stream, and hence a separate signal space. The solution is to apportion your signal blocks among a number of channels. For RS-232 connections, there is only the one channel with which to work. In such cases, a second physical connection is needed.

The following sections discuss different connection models in detail.

One Connection, Many Signal Blocks

Signal Blocks configured in the server for a particular control system can “talk” to their respective **Intersystem Communications** symbols through a single connection

to the control system. All the symbols' rx\$ and tx\$ streams are tied to the same serial driver symbol. The set of signals intended for a particular **Intersystem Communications** symbol are distinguished from the other sets by their *offset* and/or their *channel number*.

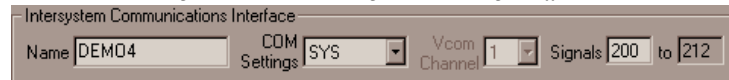
Normally, the signals in an **Intersystem Communication** symbol's input list and its output list are internally enumerated starting at the top of each list with zero (0). The next signal down the input list and the next down the output list are numbered 1; the next pair, 2; and so forth. These "signal numbers" are transmitted along with the data and are used at the receiving end to determine what to do with the data.

NOTE: The signal numbers under discussion here are for communications purposes only and are not utilized in any way outside the **Intersystem Communications** symbol context. The numbering scheme displayed on the Intersystem Communication symbols in SIMPL Windows has nothing to do with the actual signal numbering.

Additional **Intersystem Communications** symbols using the same connection must specify a value to offset their signal numbers by a constant amount. The top signals in such an **Intersystem Communications** symbol do not begin with zero, but rather with the supplied offset. This offset must be specified on both sides of the connection, as follows.

On the server side, the offset is entered into the textbox in the upper right corner of the signal block definition window. Note in the figure below that the textbox in question is not labeled *Offset* as one might expect, but *Signals*. The adjacent box labeled *to* shows the signal number of the last signal in the input or output lists (whichever is higher). This information helps to determine the offset of the next signal block.

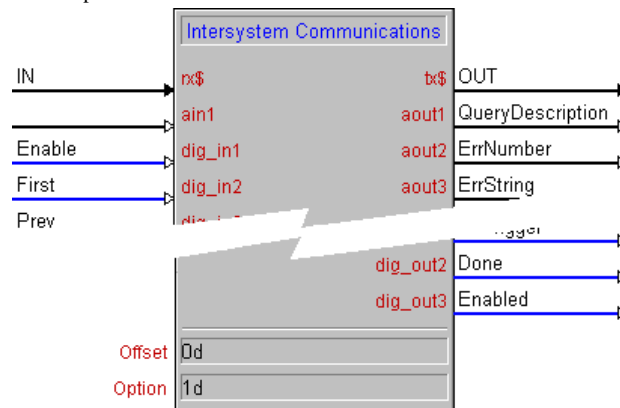
The Signal Block Interface frame from a typical signal block definition window, showing the signal block's COM Settings, Vcom channel assignment, and signal offset.



NOTE: In the above example, *Vcom channel* is dimmed because "SYS" specifies a serial connection. (Only TCP/IP connections have channels.)

On the control system side, in SIMPL Windows the offset is entered into the *offset* textbox at the bottom of the **Intersystem Communications** symbol.

A portion of an Intersystem Communications symbol, showing the Offset and Option textboxes.



NOTES:

1. Always suffix a **d** (for decimal) to values typed into the *Offset* textbox.
 2. Always enter **1d** into the *Option* textbox for all **Intersystem Communications** symbols.
-

Multiple channels

Multiple channels apply to connections made through Virtual COM Ports only (*i.e.*, TCP/IP connections only). Each Virtual COM Port can have up to 128 channels, where each channel can be thought of as a separate COM port. If a separate channel is used for each **Intersystem Communications** symbol, then all **Intersystem Communications** symbols can use an offset of zero (*i.e.*, no offset). The advantage is simplicity; each channel has its own signal space. In this case, the programmer never needs to worry about offsets on either side.

NOTE: Only systems connected to a Virtual COM Port (via TCP/IP) have multiple channels. The channel list is disabled (dimmed) in the signal block definition window for systems connected via RS-232. RS-232 connections are single-channel connections. Do not use multiple channels if there is any possibility of needing to revert to an RS-232 connection.

Multiple Connections

Usually all **Intersystem Communications** symbols from a single control system “talk” to the server through one physical connection, although it is possible to install multiple connections, such as any combination of RS-232 connections (each with its own cable) and/or or a TCP/IP connections (all referencing the same IP address, but each with its own IP ID).

NOTE: Multiple connections between a single control system and a single server delivers better response because of the additional buffers involved. However, keep in mind that multiple TCP/IP connections do need to be individually licensed.

The COM Settings Signal Block

An **Intersystem Communications** symbol to service the COM Settings signal block need only be present if at least one signal is defined in the *System Definition* window. However, it need not be at signal offset 0 but could be positioned anywhere within the signal space subject to the constraints discussed above. If present, the COM Settings signal block’s **Intersystem Communications** symbol is always connected to channel #1 in a Virtual COM port.

Appendix C: Signal Reference

Definition of Terms

| | |
|---------------------|--|
| <i>Address</i> | The e-mail address, for example, “ jdoe@company.com ,” or the account name part thereof (in this case, “jdoe”). |
| <i>Connection</i> | A connection to a system which can be either serial (RS-232) or EtherNet (TCP/IP). |
| <i>Data fields</i> | The indirect text fields which receive the data that is echoed when a record is opened (“picked”). |
| <i>List fields</i> | The indirect text fields in scrollers. There can be more than one field (column) per row in the scroller. |
| <i>Name</i> | The real, human-readable (“friendly”) name, such as “John Doe” as opposed to the e-mail address. |
| <i>Recipient</i> | The e-mail account referred to by the address that appears in the e-mail “To:” header. |
| <i>Scroller</i> | (1) Scrolling lists as displayed on touchscreens, made up of (a) a column of buttons containing indirect text fields, along with (b) transport buttons, and (c) an optional analog gauge object serving as a scrollbar. Also (2) the signal block which services such a construct. |
| <i>Sender</i> | The e-mail account referred to by the address that appears in the “From:” header. |
| <i>Server</i> | The Crestron Software Server. That is, swserver.exe running on a Windows PC. |
| <i>Signal Block</i> | Active component within the Server which listens for and responds to signals from connected control systems. For example, each scroller requires its own signal block. If not specified explicitly as some other type of signal block, this term refers to a custom scroller signal block. |
| <i>System</i> | A Crestron control processor along with appropriate programming. |

String Proxies

The various **Echo** signals keep a server-side “proxy” of the data last sent to the system. Before sending, the new value is compared with the proxy and is only actually sent if it differs. The proxy is then updated to match the new value. In this way, identical string data are not continually resent.

Bit Patterns

For those unfamiliar with bit patterns, here is a primer.

Base 2 Basics

A quantity is represented in computer memory as a binary (base 2) number (*i.e.*, a string of 1s and 0s). As you know, in base 10, the least significant (low-order) “digits” are on the right, and the most significant (high-order) are on the left. This is

also true of base 2: The lowest order “bit,” (binary digit), on the far right, and are numbered starting from 0. Thus, when sixteen bits are available, they are numbered 0 to 15 from right to left — because each bit on its own represents the quantities 2^0 through 2^{15} .

Bit patterns as used here do not represent quantities. Rather, the values of the individual bits (0 or 1) turn features off and on (respectively). Thus, when the documentation refers to Bit 5 as controlling feature X, this means that feature X is “enabled” when Bit 5 is set to 1.

Base 16 used for notational purposes

Straight base 2 notation (a long string of 0s and 1s) is considered to be too unwieldy to be useful to the human eye as it is too easily prone to misrepresentation and misinterpretation. Hexadecimal (base 16) notation is used to conveniently specify the bit patterns for the signals that use them (*i.e.*, the **Config** and **SignalA_n** signals). Hexadecimal groups all sixteen bits into four sets of four bits each, assigning the base 10 digits 0 to 9 plus the first six letters of the alphabet A to F to the sixteen possible combinations of the four bits.

For example, the sixteen-bit (base 2) pattern

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

can be broken into the four four-bit sets

| | | | |
|------|------|------|------|
| 0000 | 0001 | 1101 | 0101 |
|------|------|------|------|

which get assigned to them the four hexadecimal “hex digits”

| | | | |
|---|---|---|---|
| 0 | 1 | D | 5 |
|---|---|---|---|

Note that **D** stands for the quantity we normally express in base 10 as thirteen (13). Therefore, the above bit pattern is referred to conveniently as **\$01D5**, or since leading zeros are optional in any base, **\$1D5**. The missing (high-order) bits are assumed to be 0000. Although \$1D5 represents a quantity — which happens to be four hundred sixty-nine (469) in base 10 — again, we are only interested in bit *patterns* here, not the quantities they may represent.

Standard e-mail Address Format

The format is the standard:

<account name> @ <IP address / domain name>

(All three parts are required.)

Sender's e-mail Address

There is no signal for setting the sender's e-mail address, so this part of the “From:” header is discussed here.

The actual sender address is specified once, in the e-Mailer signal block definition, and that value is used for all mail generated from that signal block. An address must be supplied to log in to the SMTP server in order to send mail. Therefore, this value may not be null (blank). The initial value of this setting (for a newly defined e-mailer signal block) is null. However, at run-time, a null value is replaced with the value:

anonymous@unknown

The sender's "real" name is specified by the **SetFromName** signal, or by the default specified in the e-Mailer signal block definition.

While most mail (SMTP) servers accept any account name (such as "anonymous") for outgoing mail, some servers only accept outgoing mail from a known account. In these rare cases, the sender's real name may be ignored by the SMTP server and replaced with the name on record for that account. Or mail may be rejected by a picky server if the specified name does not match the name on record exactly.

Error Reporting

At this time error handling is a little haphazard, with some errors being tallied back to the control system, while others are logged in the *Server Monitor* window.

Tallied Errors

A descriptive string is sent with all errors (via **ErrString**). However, while some are accompanied with error numbers (via **ErrNumber** + **ErrTrigger**), some are not.

Log Items

The log is intended for after-the-fact analysis of important events. The log is an in-memory FIFO list (first in, first out). When the log approaches capacity, log items are removed, one by one, starting with the oldest items first, until there is enough room for the new item.

All logged items are preceded with either a per-cent sign (%) or a dollar sign (\$). Items preceded with % are informational. Items preceded with \$ are errors, reflecting unanticipated situations.

Signal Summary

The "Signal Reference," below, is an alphabetical list of all signals from COM, e-Mailer, and e-Mailbox signal blocks. (Signals from Scroller signal blocks are not listed in this document. See the SW-DBM document, Doc. 5823.)

Certain signal names are used in all types of signal blocks. However, only one entry for each signal type appears in the reference. Among these are the **Done** signal and the three error signals (**ErrNumber**, **ErrString**, and **ErrTrigger**). This identical naming reflects the fact that these signals function similarly regardless of where they appear.

When these signals are defined in a signal block, the server uses them. When not defined, however, the server sends the signal via the "owning" signal block. If the signal is not defined there either, or the scroller has no owner, the signal is sent to the COM Settings signal block. If the signal is also undefined in the COM Settings signal block, the signal is lost (not sent).

| Signal Block(s) | Signal Name | Direction | Type |
|-----------------|-------------------|------------------|----------|
| e-mailbox | CheckMail | server-to-system | D |
| e-mailer | ClearNames | server-to-system | D |
| e-mailbox | Delete | server-to-system | D |
| <i>All</i> | Done | system-to-server | D |
| e-mailer | EchoBody | server-to-system | S |
| e-mailbox | EchoDate | system-to-server | S |
| e-mailbox | EchoFrom | system-to-server | S |
| e-mailbox | EchoRcpt | system-to-server | S |

| Signal Block(s) | Signal Name | Direction | Type |
|------------------------|-------------------------------|------------------|----------|
| e-mailer | EchoRcptAddr | system-to-server | S |
| e-mailer | EchoRcptName | server-to-system | S |
| e-mailer, e-mailbox | EchoSubj | server-to-system | S |
| e-mailer, e-mailbox | Enable | system-to-server | D |
| e-mailer, e-mailbox | Enabled | server-to-system | D |
| <i>All</i> | ErrNumber | server-to-system | A |
| <i>All</i> | ErrString | server-to-system | S |
| <i>All</i> | ErrTrigger | server-to-system | D |
| e-mailbox | KeepAsNew | system-to-server | D |
| e-mailer | LookupMsg | system-to-server | A |
| e-mailer | LookupParm_n | system-to-server | A |
| e-mailer | LookupRcpt | system-to-server | A |
| e-mailbox | NewCount | server-to-system | A |
| e-mailer | NewMail | system-to-server | D |
| e-mailbox | OldCount | server-to-system | A |
| e-mailbox | PageFlip | server-to-system | D |
| COM Settings | PingSvr | system-to-server | D |
| COM Settings | PingSys | server-to-system | D |
| COM Settings | PongSvr | system-to-server | D |
| COM Settings | PongSys | server-to-system | D |
| e-mailbox | Reply | system-to-server | D |
| e-mailer | SendNow | system-to-server | D |
| e-mailer | SetAddr | system-to-server | S |
| e-mailer | SetBody | system-to-server | S |
| e-mailer | SetFromName | system-to-server | S |
| e-mailer | SetRcptName | system-to-server | S |
| e-mailer | SetParm_n | system-to-server | S |
| e-mailer | SetSubj | system-to-server | S |
| e-mailer | ShortCut_n | system-to-server | D |
| e-mailbox | SignalA_n | server-to-system | A |
| e-mailbox | SignalD_n | server-to-system | D |
| e-mailbox | SignalS_n | server-to-system | S |
| e-mailbox | Status | server-to-system | A |
| e-mailbox | ViewOldMail | system-to-server | D |

Signal Reference

The alphabetical reference proper begins on the next page.

CheckMail

| | |
|----------------|---|
| Applies to | e-Mailbox signal blocks |
| Description | Checks the e-mail host for waiting mail. |
| Direction | system-to-server |
| Type | Digital |
| Value | Pulse (actually, leading edge is the trigger; trailing edge is ignored) |
| Expected Reply | Done pulse |
| Comments | <p>The server logs onto the POP3 (incoming e-mail) host and retrieves waiting mail. Precise behavior depends on whether or not there is an attached IN Box table in which to save the mail, as follows:</p> <p><u>Case 1: Signal Block configured with "IN box" scroller</u> (see "IN Box scroller" on page 35).</p> <p>The message is skipped (not downloaded) if it is already in the IN box (already downloaded).</p> <p>Furthermore, when the message is already in the IN box, <u>and</u> it is found to have been previously marked for deletion, it is deleted from the host at this time.</p> <p>If the message is new, it is downloaded in full (headers plus body text).</p> <p>If the signal block is configured to Process control messages <u>and</u> the message is a control message, control equates in the subject header or body text are processed and the message is deleted; it is not saved in the IN box. (See "Control Messages" on page 34.)</p> <p>When <i>Delete messages as downloaded</i> is checked in the <i>e-Mailbox Signal Block Configuration</i> window, messages are always deleted from the host immediately after downloading.</p> <p><u>Case 2: Signal Block <i>not</i> configured with "IN box" scroller</u> (<u>must</u> be configured to Process control messages):</p> <p><u>All</u> messages resident on the host are considered, as follows:</p> <p>Each message's headers (only) are downloaded; if the subject header was not received (depends on the host), the full message is downloaded.</p> <p>The subject header is checked to see if this is a control message. If there is a control equate in the subject header, it is processed. If the control equate is null (just CTRL> with nothing else), the body text is downloaded from the host (if not already downloaded) and all CTRL> lines therein are processed.</p> <p>A control message is always deleted from the host immediately after it is processed; regardless of whether there is an IN box attached or not.</p> <p>Note that if the message is not a control message, the headers are discarded and the body text may never be downloaded.</p> <p>The above process can also be set up to occur implicitly on a regular schedule using an option in the <i>e-Mailbox Signal Block Configuration</i> window. Note that explicit CheckMail signals are ignored if the an implicit check is already in progress..</p> |
| See Also | NewMail signal |

| <h1>ClearNames</h1> | |
|-----------------------|---|
| Applies to | e-Mailer signal blocks |
| Description | Clears the recipient list. |
| Direction | system-to-server |
| Type | Digital |
| Value | Pulse (actually, leading edge is the trigger; trailing edge is ignored) |
| Expected Reply | Done pulse |
| Comments | <p>Also clears any To: name which may have been previously sent with SetRcptName but has not yet actually been added to the recipient list with SetAddr.</p> <p>The Done pulse should be used to de-highlight the touchscreen button that originated the ClearNames signal. Keeping the screen button highlighted while the server is busy keeps the user better informed and encourages his patience.</p> |
| See Also | NewMail signal |

| <h1>Delete</h1> | |
|-----------------------|---|
| Applies to | e-Mailbox signal blocks |
| Description | Signals server to delete the message most recently picked from the IN box. |
| Direction | server-to-system |
| Type | Digital |
| Value | Pulse |
| Expected Reply | None |
| Comments | <p>The message is deleted from both the e-mail host and the IN box database table — but not until the next time the host is contacted. Also deleting the message from the host ensures the same message is not downloaded again.</p> <p>If the message is no longer on the e-mail host (presumably because it was deleted by the host in the meantime), it is deleted from the IN box anyway.</p> <p>It is possible that the server can be reconfigured to contact a different host henceforth. If this is the case, the message marked for deletion is not found. As stated above, it is deleted from the IN box. If this happens, and the configuration is changed back to point to the host containing the message, it may never be deleted.</p> |
| See Also | ErrDescription , ErrNumber , and ErrTrigger signals |

| <h1>Done</h1> | |
|-----------------------|---|
| Applies to | System signal blocks e-Mailer signal blocks e-Mailbox signal blocks scroller signal blocks ("standard" or "custom") |
| Description | Indicates requested operation completed successfully |
| Direction | server-to-system |
| Type | Digital |
| Value | Pulse |
| Expected Reply | None |
| Comments | <p>Sent to control system in response to most system-to-server signals, indicating that the requested operation has been successfully completed. Done is only sent if the operation was successful. If it was unsuccessful, the Err signals are sent instead.</p> <p>Typically, there is a 0.2-sec. delay between the leading and trailing edge of the pulse.</p> <p>Signals which respond this way list Done in their "expected reply" field in this reference.</p> <p>This signal is optional in systems, but always present in all other signal blocks.</p> <p>The Done signal is typically used to provide feedback to the user, typically by de-asserting a button's feedback. Therefore, buttons which function together should be grouped together in an Interlock symbol with the Done signal being connected to the <i>clear</i> line of that symbol.</p> |
| See Also | ErrDescription , ErrNumber , and ErrTrigger signals |

EchoBody

| | |
|-----------------------|--|
| Applies to | e-mailer signal blocks |
| Description | String sent as feedback in response to setting a new subject header |
| Direction | Server-to-system |
| Type | Serial |
| Value | New body text |
| Expected Reply | None |
| Comments | <p>Definition of this signal is optional.</p> <p>This signal is generated in response to lookups only. Signals that might precipitate this are LookupMsg, ClearNames, NewMail, and manually picking a message from a bound message scroller. Specifically excluded are Shortcut signals, and the SetSubj signal.</p> <p>Serial data is limited to 83 characters. If the body text exceeds this length, the transmitted string is truncated.</p> <p>For efficiency, strings are only echoed when they change. <i>E.g.</i>, if two lookups both produced the same result, the string would only be echoed the first time.</p> |
| See Also | |

| <h1>EchoDate</h1> | |
|-----------------------|---|
| Applies to | e-mailbox signal blocks |
| Description | String sent as feedback in response to opening an e-mail message |
| Direction | Server-to-system |
| Type | Serial |
| Value | Verbatim text from the message's date header |
| Expected Reply | None |
| Comments | <p>Definition of this signal is optional.</p> <p>If the date header is somehow null, the following string is substituted:</p> <p style="text-align: center;">[rec'd mm/dd/yy hh:mm:ss AM]</p> <p>The actual format of the date and time is determined by your system settings; see the <i>Regional Settings</i> control panel. (The format is generally as shown.)</p> <p>This signal is generated in response to opening an e-mail message. The server "opens" e-Mail messages by sending a Pick signal (see) through the referenced IN box scroller.</p> <p>For efficiency, strings are only echoed when they change.</p> |
| See Also | EchoFrom , EchoDate , and EchoSubj signals |

| <h1>EchoFrom</h1> | |
|-----------------------|--|
| Applies to | e-mailbox signal blocks |
| Description | String sent as feedback in response to opening an e-mail message |
| Direction | Server-to-system |
| Type | Serial |
| Value | Name <u>or</u> address from the message's sender ("From:") header |
| Expected Reply | None |
| Comments | <p>Definition of this signal is optional.</p> <p>This signal is generated in response to opening an e-mail message. The server "opens" e-Mail messages by sending a Pick signal through the referenced IN box scroller.</p> <p>Only the name <u>or</u> address of the sender is sent; the name is sent when available; otherwise the address is sent.</p> <p>For efficiency, strings are only echoed when they change. When the sender is the same for messages opened consecutively, the string would only be sent when the first message is opened.</p> |
| See Also | <p>EchoDate, EchoRcpt, and EchoSubj signals</p> <p>See <i>SW-DBM documentation (Doc. 5823)</i> for more information about the Pick signal.</p> |

| <h1>EchoRept</h1> | |
|-----------------------|--|
| Applies to | e-Mailbox signal blocks |
| Description | String sent as feedback in response to opening an e-mail message |
| Direction | Server-to-system |
| Type | Serial |
| Value | List of recipient names <u>or</u> addresses from the message's recipient ("To:") header |
| Expected Reply | None |
| Comments | <p>Definition of this signal is optional.</p> <p>This signal is generated in response to opening an e-mail message. The server "opens" e-Mail messages by sending a Pick signal through the referenced IN box scroller.</p> <p>The whole recipient list is sent. Serial data transmission is limited to 83 characters. If the recipient list exceeds this length, the transmitted string is truncated and an ellipsis is substituted for the remaining characters. To make the most of the 83 characters, only the name <u>or</u> address of each recipient is put into the list; the name is sent when available; otherwise the address is sent. Recipient name or address is sent on a line by itself (<i>i.e.</i>, a carriage-return character is appended to each).</p> <p>For efficiency, strings are only echoed when they change. Typically, the recipient list would consist only of the signal block's e-mail account name. If that is the case for messages opened consecutively, the string would only be sent when the first message is opened.</p> |
| See Also | <p>EchoDate, EchoFrom, and EchoSubj signals</p> <p>See <i>SW-DBM documentation (Doc. 5823)</i> for more information about the Pick signal.</p> |

EchoRcptAddr

| | |
|-----------------------|--|
| Applies to | e-Mailer signal blocks |
| Description | String sent as feedback in response to setting a new recipient address |
| Direction | Server-to-system |
| Type | Serial |
| Value | New recipient address |
| Expected Reply | None |
| Comments | <p>Definition of this signal is optional.</p> <p>This signal is generated in response to lookups only. Signals that might precipitate this are LookupRcpt, LookupMsg, ClearNames, NewMail, and manually picking an item from a bound data scroller. Specifically excluded are Shortcut signals, and the SetSubj signal.</p> <p>Echoing the recipient address does <u>not</u> necessarily imply that the address has already been added to the recipient list because default addresses are also echoed.</p> <p>Serial data is limited to 83 characters. If the recipient address exceeds this length, the transmitted string is truncated.</p> <p>For efficiency, strings are only echoed when they change (e.g., if two lookups both produced the same result, the string would only be echoed the first time).</p> |
| See Also | For more information on how default addresses are resolved, refer to "Implicit Recipient Name & Address" on page 47. |

| <h1>EchoRcptName</h1> | |
|-----------------------|---|
| Applies to | e-Mailer signal blocks |
| Description | String sent as feedback in response to setting a new recipient name |
| Direction | Server-to-system |
| Type | Serial |
| Value | New recipient name |
| Expected Reply | None |
| Comments | <p>Definition of this signal is optional.</p> <p>This signal is generated in response to lookups only. Signals that might precipitate this are LookupRcpt, LookupMsg, ClearNames, NewMail, and manually picking an item from a bound data scroller. Specifically excluded are Shortcut signals, and the SetRcpt signal.</p> <p>Default names are also echoed.</p> <p>Serial data is limited to 83 characters. If the recipient name exceeds this length, the transmitted string is truncated.</p> <p>For efficiency, strings are only echoed when they change (e.g., if two lookups both produced the same result, the string would only be echoed the first time).</p> |
| See Also | For more information on how default names are resolved, refer to "Implicit Recipient Name & Address" on page 47. |

| <h1>EchoSubj</h1> | |
|-----------------------|--|
| Applies to | e-Mailer signal blocks e-Mailbox signal blocks |
| Description | String sent as feedback in response to setting a new subject header |
| Direction | Server-to-system |
| Type | Serial |
| Value | Text of subject header from selected message |
| Expected Reply | None |
| Comments | <p>Definition of this signal is optional.</p> <p><u>e-Mailer signal block:</u> This signal is generated in response to lookups only. Signals that might precipitate this are LookupMsg, ClearNames, NewMail, and manually picking a message from a bound message scroller. Specifically excluded are Shortcut signals, and the SetSubj signal.</p> <p><u>e-Mailbox signal block:</u> This signal is generated in response to opening an e-mail message. The server "opens" e-Mail messages by sending a Pick signal through the referenced IN box scroller.</p> <p>Serial data is limited to 83 characters. If the subject header exceeds this length, the transmitted string is truncated and an ellipsis is substituted for the remaining characters.</p> <p>For efficiency, strings are only echoed when they change.</p> <p><i>See SW-DBM documentation (Doc. 5823) for more information about the Pick signal.</i></p> |

Enable

| | |
|-----------------------|--|
| Applies to | e-Mailer signal blocks e-Mailbox signal blocks scroller signal blocks (“standard” or “custom”) |
| Description | This signal enables/disables the signal block. |
| Direction | system-to-server |
| Type | Digital |
| Value | <u>Assert</u> to enable the signal block <u>De-assert</u> to disable the signal block |
| Expected Reply | None |
| Comments | <p><u>Assert</u>: Allocates server resources so server can respond to signals from the signal block.</p> <p><u>Deassert</u>: Deallocates resources associated with the signal block.</p> <p>The signal block must be enabled prior to use. If not, all other signals will produce the following error:</p> <p style="text-align: center;">Object not loaded.</p> <p>The only exception to this rule is the Shortcut signal which “auto-enables” the e-mailer signal block for the duration of the signal. (If it was already enabled when the Shortcut signal was issued, it remains enabled after the signal is processed; otherwise it is disabled.) <i>Demo 1 depends on this feature.</i></p> <p>Two special situations are noted below:</p> <p>(1) <u>Referenced scrollers are auto-enabled</u>: When a scroller signal block is referenced* by another signal block, it is automatically enabled when the referencing signal block is enabled. Therefore, the Enable signal of such a scroller signal block should not be connected to anything.</p> <p>(2) <u>Standard Scrollers’ Enable signal is non-functional</u>: In fact, although the Enable signal of a Standard Scroller signal block is defined, it is ignored by the server.† The reason the signal is defined even though it cannot be used is that it can however be used from the <i>Signal Analyzer</i> window (for debugging purposes).</p> <p>* Examples of “referenced scrollers” are the optional message and recipient scrollers of an e-Mailer signal block; or the optional IN box and body text scrollers of an e-Mailbox signal block.</p> <p>† This prevents the scroller from being used for general purposes — for which a database license is required.</p> |
| See Also | “Signal Block Enable / Disable” on page 59. |

| <h1>Enabled</h1> | |
|-----------------------|---|
| Applies to | e-Mailer signal blocks e-Mailbox signal blocks scroller signal blocks ("standard" or "custom") |
| Description | "Handshake" response to the Enable signal. |
| Direction | server-to-system |
| Type | Digital |
| Value | <u>Asserted</u> in response to assert of Enable signal. <u>De-asserted</u> in response to de-assertion of Enable signal. |
| Expected Reply | None |
| Comments | <p>Although this signal is often ignored, in a proper implementation, the control system should:</p> <ol style="list-style-type: none"> (1) Wait for assertion of Enabled before issuing any other e-mail signals. (2) Indicate an error condition if assertion of Enabled does not occur after a certain amount of time (typically 30 seconds). <p>Although the server ignores the Standard Scroller signal block's Enable signal, it does maintain the Enabled signal as usual.</p> <p><i>Demo1 does not use the Enable signal (containing as it does only Shortcut signals).</i></p> <p><i>Demo 2 ignores this signal.</i></p> <p><i>Demos 3, 4, and 5 respond to this signal by implementing (1), above, but not (2).</i></p> |
| See Also | Enable signal |

| <h1>ErrNumber</h1> | |
|-----------------------|---|
| Applies to | System signal blocks e-Mailer signal blocks e-Mailbox signal blocks scroller signal blocks ("standard" or "custom") |
| Description | When the server encounters an error, it uses this signal to send an error number to the control system. |
| Direction | Server-to-system |
| Type | Analog |
| Value | New error number |
| Expected Reply | None |
| Comments | <p>This signal works in conjunction with the ErrString and ErrTrigger signals which always follow immediately.</p> <p>These signals are optional. For systems, they are defined by checking the <i>Error Processing</i> checkbox in the <i>System Definition</i> window. For signal blocks, they are defined for a custom scroller by the <i>Local Errors</i> checkbox in the <i>Signal Block Definition</i> window. (The Err signals cannot be defined in a Standard Scroller signal block.)</p> <p>In the case of a scroller signal block associated with an e-mailer or e-mailbox signal block, if that scroller signal block does not define its Err signals (as is always the case for Standard Scroller signal blocks), the server uses the Err signals of the e-mailer or e-mailbox signal block in question.</p> <p>However: When an e-mailer or e-mailbox signal block does not define its Err signals, the server attempts to use the Err signals of the system associated with the signal block, unless those signals too are not defined.</p> <p>These signals can be safely ignored.</p> |
| See Also | ErrTrigger and ErrString signals |

| <h1>ErrString</h1> | |
|-----------------------|---|
| Applies to | System signal blocks e-Mailer signal blocks e-Mailbox signal blocks scroller signal blocks ("standard" or "custom") |
| Description | Description of error |
| Direction | Server-to-system |
| Type | Serial |
| Value | Error message for display |
| Expected Reply | None |
| Comments | <p>Although this signal can be safely ignored, it is easily hooked to indirect text fields on a touchscreen and/or on the CNMSX-PRO front panel, <i>etc.</i></p> <p>This signal is used by the server in two ways:</p> <ol style="list-style-type: none"> (1) Used in conjunction with the ErrNumber and ErrTrigger signals, the latter following immediately. (2) Used alone for informational messages. Sometimes these messages represent error conditions and sometimes they are purely informational. <p><i>See additional notes under ErrNumber.</i></p> |
| See Also | ErrNumber and ErrTrigger signals |

| <h1>ErrTrigger</h1> | |
|-----------------------|--|
| Applies to | System signal blocks e-Mailer signal blocks e-Mailbox signal blocks scroller signal blocks ("standard" or "custom") |
| Description | Trigger for ErrNumber and ErrString |
| Direction | Server-to-system |
| Type | Digital |
| Value | Pulse |
| Expected Reply | None |
| Comments | This signal is sent after the ErrNumber and ErrString to indicate that an error condition has occurred. <i>See additional notes under ErrNumber.</i> |
| See Also | ErrNumber and ErrString signals |

| <h1>KeepAsNew</h1> | |
|-----------------------|--|
| Applies to | e-Mailbox signal blocks |
| Description | Marks the opened e-mail message as new |
| Direction | system-to-server |
| Type | Digital |
| Value | Pulse |
| Expected Reply | Pulse of Done signal |
| Comments | Mail is immediately transferred to the IN box's "old mail view" the first time it is opened. This signal moves the message back to the "new mail view." This is useful when more than one person is using the same mailbox, such as several members of a household. After reading a message, the reader might decide to keep it as new to increase the likelihood of his fellows seeing the new message. |
| See Also | Other signals that operate on "opened" e-mail messages include Delete . |

LookupMsg

| | |
|-----------------------|--|
| Applies to | e-Mailer signal blocks |
| Description | Performs a lookup on a message in the database, and sets <i>subj</i> and <i>body</i> accordingly |
| Direction | system-to-server |
| Type | Analog |
| Value | A number matching the contents of the <i>ID</i> field of a record in the eMail_Msg table of the database file. Both the <i>subj</i> and <i>body</i> fields from the matching record are added to the recipient list. |
| Expected Reply | None |
| Comments | <p>Use of this signal is optional.</p> <p>If no matching record is found whose <i>ID</i> field matches this signal value, the following error is displayed:</p> <p style="text-align: center;">No such record.</p> <p>If the record is found but contains a null <i>subj</i> field, the e-Mail message is considered incomplete and is not sent. However, the following error message is sent back to the control system:</p> <p style="text-align: center;">No message specified - mail cannot be sent.</p> <p>(Note that a null <i>body</i> field is acceptable.)</p> <p>The message record also contains a reference to a record in the recipient name and address table. That record is used to provide an implicit recipient name and address — but only when no recipient is explicitly specified.</p> <p>The eMail_Msg table must contain at least these three fields (<i>ID</i>, <i>subj</i>, <i>body</i>, and <i>defaultAddr</i>). However, it is perfectly permissible for additional fields to be included in the table (for list display purposes, for example). An example of such a field is <i>abbr</i> which is often used in the list display of a bound scroller. The <i>abbr</i> field is used only if it is listed in the Queries table. No direct reference to such a field is made in the application code. For more information on the Queries table, refer to the latest revision of documentation for e-control Database Manager (SW-DBM), Doc. 5823.</p> |
| See Also | <p>For more information on how default names and addresses are resolved, refer to "Implicit Recipient Name & Address" on page 47.</p> <p>Other methods of setting the name and address are: The SetSubj signal, the SetBody signal, and manually picking a message from the bound message scroller.</p> |

| <h1 style="margin: 0;">LookupParm_n</h1> | |
|--|---|
| Applies to | e-Mailer signal blocks |
| Description | Sets a particular text substitution register |
| Direction | system-to-server |
| Type | Analog |
| Value | A number matching the contents of the <i>ID</i> field of a record in the eMail_Msg table of the database file specified in the e-Mailer signal block definition. The <i>substitution</i> field from the matching record is set as the value for substitution parameter <i>n</i> . |
| Expected Reply | None |
| Comments | <p>Definition of these signals is optional.</p> <p>These signals provide new values for the same substitution parameters affected by the SetParm signals.</p> <p><i>See additional comments under SetParm.</i></p> |
| See Also | For more information on how substitution parameters are utilized, refer to "Text Substitution and File Inclusion" on page 48. |

LookupRcpt

| | |
|-----------------------|--|
| Applies to | e-Mailer signal blocks |
| Description | Performs a lookup on a recipient in the database, and adds it to the recipient list |
| Direction | system-to-server |
| Type | Analog |
| Value | A number matching the contents of the <i>ID</i> field of a record in the <code>eMail_Addr</code> table of the database file. The name and address from the matching record are added to the recipient list. |
| Expected Reply | None |
| Comments | <p>Use of this signal is optional.</p> <p>If the <i>Allow multiple recipients</i> option is checked in the e-Mailer signal block definition, LookupRcpt (or SetAddr) may be used repeatedly to add (names and) addresses to the recipient list. Otherwise, when <i>Allow multiple recipients</i> is <u>unchecked</u>, mail can only be sent to a single name and address. That is, each additional LookupRcpt (or SetAddr) signal simply replaces the previously specified (name and) address with the most recently specified address (and name).</p> <p>If no matching record is found whose <i>ID</i> field matches this signal value, nothing is added to the recipient list.</p> <p>If the SendNow signal is issued but no address has been set (by any method), the following error message is sent back to the control system:</p> <pre style="text-align: center;">No recipient specified -- mail cannot be sent.</pre> <p>If the record is found but contains a null <i>addr</i> field, the default recipient address specified in the e-Mailer signal block definition will be used instead. In this case, the name is also checked, and if it too is null, the default recipient name is utilized.</p> <p>If the default is also null (and all other addresses in the recipient list are also null), attempts to send the mail are rejected by the SMTP server (a message is posted to the log).</p> <p>Note that the default's initial value (for a newly-defined e-mailer signal block) is null. Fill the default to send all (unaddressed) mail to a single address.</p> <p>The recipient list is reset to the default by the ClearNames, NewMail, and SendNow signals.</p> <p>If a recipient name had previously been specified with a SetRcptName signal, this signal overwrites it with the contents of the <i>first</i> [name] and <i>last</i> [name] fields of the matching record.</p> <p>The <code>eMail_Addr</code> table must contain at least four fields (<i>ID</i>, <i>first</i>, <i>last</i>, and <i>addr</i>). However, it is perfectly permissible for additional fields to be included in the table (for list display purposes, for example).</p> |
| See Also | Other methods of setting the name and address are: The SetRcptName signal, the SetAddr signal, and manually picking a recipient from the bound recipient scroller. |

| <h1>NewCount</h1> | |
|-----------------------|--|
| Applies to | e-Mailbox signal blocks |
| Description | Number of messages in the “new mail view” of the IN box. |
| Direction | system-to-server |
| Type | Analog |
| Value | Quantity intended for a digital gauge object |
| Expected Reply | Pulse of Done signal |
| Comments | <p>Definition of this signal is optional.</p> <p>Updated whenever it changes:</p> <ul style="list-style-type: none"> When the signal block is enabled via Enable As new mail is received into the IN box via CheckMail or automatically as scheduled When an “opened” message is moved back to the new mail view via KeepAsNew; When a message is opened from the new mail view — and hence moved to the old mail view — via one of the IN box scroller’s Pick signals. |
| See Also | OldCount signal |

| <h1>NewMail</h1> | |
|-----------------------|---|
| Applies to | e-Mailer signal blocks |
| Description | Resets all e-Mail data |
| Direction | System-to-server |
| Type | Digital |
| Value | Pulse (actually, leading edge is the trigger; trailing edge is ignored) |
| Expected Reply | Done pulse |
| Comments | <p>Specifically:</p> <p>Clears the From: name back to the default specified in the e-Mailer signal block definition.</p> <p>Clears Subj: header and body text.</p> <p>Clears recipient list (see ClearNames).</p> <p>The Done pulse should be used to deselect the touchscreen button that originated the NewMail signal. Keeping the screen button highlighted while the server is busy keeps the user better informed and encourages his patience.</p> <p>In the demos, the button that sends this signal is labeled Clear All.</p> |
| See Also | ClearNames signal |

| <h1>OldCount</h1> | |
|-----------------------|---|
| Applies to | e-Mailbox signal blocks |
| Description | Number of messages in the “old mail view” of the IN box. |
| Direction | system-to-server |
| Type | Analog |
| Value | Quantity intended for a digital gauge object |
| Expected Reply | Pulse of Done signal |
| Comments | <p>Definition of this signal is optional.</p> <p>Sent:</p> <ul style="list-style-type: none"> When the signal block is enabled via Enable When an “opened” message is deleted via Delete When an “opened” message is moved back to the new view via KeepAsNew |
| See Also | NewCount signal |

| PageFlip | |
|-----------------------|---|
| Applies to | e-Mailbox signal blocks |
| Description | A message has been "opened" |
| Direction | server-to-system |
| Type | Digital |
| Value | Pulse |
| Expected Reply | <i>None</i> |
| Comments | Definition of this signal is optional. A message is opened by a valid pick from the IN box scroller, <i>i.e.</i> , a Pick signal on a non-null row of the scroller. |
| See Also | |

| <h1>Reply</h1> | |
|-----------------------|---|
| Applies to | e-Mailbox signal blocks |
| Description | Prepares a reply message based on the currently opened message |
| Direction | system-to-server |
| Type | Digital |
| Value | Pulse |
| Expected Reply | Pulse of Done signal |
| Comments | <p>If there is a bound e-mailer signal block, issues the following signals to same:</p> <ul style="list-style-type: none"> NewMail In case a message is in progress (it is discarded) SetRcptName Using the from name of the opened message (but only when known) SetRcptAddr Using the from name of the opened message (which is always known) SetSubj Using the subject header of the opened message — prefixed with the string "Re:" (but only when not already present) <p>It is left to the user to complete the message (with either a LookupMsg signal or a SetBody signal) and send the mail (issue the SendNow signal). Note that when using the LookupMsg signal, the subject line is normally replaced with the subject from the canned message. However, when replying to a message, the subject line is not replaced.</p> |
| See Also | |

| SendNow | |
|----------------|---|
| Applies to | e-Mailer signal blocks |
| Description | Sends the e-Mail message. |
| Direction | system-to-server |
| Type | Digital |
| Value | Pulse (actually, leading edge is the trigger; trailing edge is ignored) |
| Expected Reply | Done pulse |
| Comments | <p>You must select a name and a message before sending. If no value was specified for the subject header, or no signal was issued to set a recipient address, the mail is not sent, and an error code along with the following message is sent to the control system (via the ErrNumber, ErrString, and ErrTrigger signals), as described in SetAddr, SetSubj, and LookupRcpt, and LookupMsg.</p> <p>SendNow performs an implied NewMail <u>after</u> the message is successfully sent. If the message fails to send, the NewMail is <u>not</u> performed. Therefore, if you are not checking for an error code, you must send your own NewMail signal. For this reason, in practice it is best to send an explicit NewMail before starting a new message.</p> <p>Messages can be sent in either of two ways, depending on the state of the <i>Queue messages</i> checkbox in the e-Mailer signal block definition:</p> <p><u>Immediate mode.</u> The <i>Database Manager</i> itself contacts the SMTP (outgoing mail) server and carries on the dialog. This method unfortunately ties up the <i>Database Manager</i> until the dialog is complete. Although this typically only takes a few seconds, it could take longer — especially when the domain name of the SMTP server is unknown to the local DNS server. In such a case, it is possible that the <i>Database Manager</i> app's input buffers could overflow and incoming signal data from control systems could be lost. Even if all incoming data is buffered intact, there will a noticeable delay in touchscreen response if service is requested while the server is tied up talking to the SMTP server. This is more an issue when the server is serving multiple touchscreens. In a single touchscreen environment, while the "Send" button is highlighted, the user is aware of the process (sending his mail) and is usually content to wait.</p> <p><u>Queued mode.</u> The <i>Database Manager</i> submits the mail to the <i>SMTP Express</i> application which carries on the SMTP dialog independently of the server. This solves all of the problems with the other method. However, this method requires the downloading and installation of <i>SMTP Express</i> and the separate purchase of a license for same from Quiksoft. Queued mode is invoked by checking <i>Queue messages</i> in the e-Mailer signal block definition. If this option is checked but messages cannot be queued (usually due to <i>SMTP Express</i> not being properly installed), the <i>Database Manager</i> then tries to send it using the immediate mode method (above).</p> <p>The Done pulse should be used to de-highlight the touchscreen button that originated the SendNow signal. Keeping the screen button highlighted while the server is busy keeps the user better informed and encourages his patience.</p> |
| See Also | http://www.quiksoft.com/easymail/smtexpress/ |

| <h1>SetBody</h1> | |
|-----------------------|--|
| Applies to | e-Mailer signal blocks |
| Description | Sets the body text of the message. |
| Direction | System-to-server |
| Type | Serial (string) |
| Value | New value for the actual body of the e-Mail message. |
| Expected Reply | None |
| Comments | <p>Optional.</p> <p>The string specified for the body text is subject to text substitution at the time the message is actually sent. For more information, refer to "Text Substitution and File Inclusion" on page 48.</p> <p>Note that SetBody does not send any mail. Subsequent use of the SendNow signal is required to actually send a message.</p> |
| See Also | <p>The other "Set" signals (SetFromName, SetRcptName, SetAddr, and SetSubj).</p> <p>Other methods of setting the subject are: the LookupMsg signal, and manually picking a message from the bound message scroller.</p> |

SetFromName

| | |
|-----------------------|---|
| Applies to | e-Mailer signal blocks |
| Description | Sets the sender's real name. This name is included in quotes in the "From:" header of the e-Mail message, in front of the sender's e-Mail address. It is not considered part of the address and may be set to anything — or excluded entirely. |
| Direction | system-to-server |
| Type | Serial (string) |
| Value | New sender name. May contain spaces and punctuation. |
| Expected Reply | None |
| Comments | <p>This signal is not required to send e-mail.</p> <p>The sender name may be any string. It normally does not have to match the name on record for the sender's e-Mail account and can vary from message to message. It may be set to null (blank) by sending a null string.</p> <p>The sender name is reset to the default by the NewMail signal, as well as the SendNow signal which does an implicit NewMail. The default sender name is specified in the e-Mailer signal block definition. The default may be null (blank). The initial value of the default (for a newly defined e-mailer signal block) is:</p> <p style="text-align: center;">Control System</p> <p>Note that SetFromName does not actually send any mail. Subsequent use of the SendNow signal is required to actually send a message.</p> |
| See Also | The other "Set" signals (SetRcptName , SetAddr , SetSubj , and SetBody). |

| SetParm _n | |
|-----------------------|--|
| Applies to | e-Mailer signal blocks |
| Description | Sets a particular text substitution register |
| Direction | system-to-server |
| Type | Serial (string) |
| Value | New value for register <i>n</i> . Can contain additional text substitution directives, or parts thereof. |
| Expected Reply | None |
| Comments | <p>Definition of these signals is optional.</p> <p>These signals provide new values for the same substitution parameters affected by the LookupParm signals (which see).</p> <p><i>The remaining comments apply equally well to SetParm and LookupParm:</i></p> <p>Use of these signals are completely optional.</p> <p>All register values specified are retained by the server for use in subsequent messages. However, these values are lost when the server protocol is halted. It is not necessary to set them up for every message (unless there is a change, obviously).</p> <p>The specified text substitution register can contain additional text substitution directives, or parts thereof.</p> <p>The number of these signals actually defined in the XSIG is specified in the e-Mailer signal block definition. The number of signals which may be defined can vary from 0 (no text substitution signals at all) to any value — limited only by practical considerations. (See “Appendix D: System limitations” on page 107, for a discussion of these limits.) The number of SetParm and LookupParm signals defined <u>must</u> be set to four each in order to use the signal block with the Send e-Mail SIMPL symbol (as opposed to building your own XSIG to work with your signal block definition).</p> |
| See Also | For more information on how substitution parameters are utilized, refer to “Text Substitution and File Inclusion” on page 48. |

| SetRcptAddr | |
|-----------------------|---|
| Applies to | e-Mailer signal blocks |
| Description | Sets the recipient's e-mail address. |
| Direction | system-to-server |
| Type | Serial (string) |
| Value | New recipient address |
| Expected Reply | None |
| Comments | <p>This signal is not required to send e-mail. This signal is one of several methods of setting the recipient address.</p> <p>If the <i>Allow multiple recipients</i> option is checked in the e-Mailer signal block definition, SetAddr (or LookupRcpt) may be used repeatedly to add addresses to the recipient list (along with names, <i>see below</i>). Otherwise, when <i>Allow multiple recipients</i> is <u>unchecked</u>, mail can only be sent to a single name and address. That is, each additional SetAddr (or LookupRcpt) signal simply replaces the previously specified address (and name) with the most recently specified address (and name).</p> <p>Besides setting the recipient address, the SetAddr signal acts as a "trigger," taking the most recently specified recipient name (value set with SetRcptName), along with the present value for recipient address, and adding both to the recipient list. Because of this aspect of the SetAddr signal, it is imperative that the SetRcptName signal is sent <u>before</u> the SetAddr signal with which it is to be associated.</p> <p>This last point is particularly important when specifying multiple recipients. If the signals are sent in reverse order, the results will be completely erroneous: The first address has no SetRcptName signal associated with it (and uses the default name); each subsequent address is associated with the name intended for the previous address; and the last SetRcptName signal is not in the list at all because there is no subsequent SetAddr signal before the SendNow signal is issued.</p> <p>If the SendNow signal is issued but no address has been set (by any method), the following error message is sent back to the control system:</p> <p style="text-align: center;">No recipient specified -- mail cannot be sent.</p> <p>If, however, a null (blank) address is specified, the default recipient address specified in the e-Mailer signal block definition is used instead. In this case, the name is also checked, and if it too is null, the default recipient name is utilized.</p> <p>If the default is also null (and all other addresses in the recipient list are also null), attempts to send the mail are rejected by the SMTP server (a message is posted to the log).</p> <p>Note that the default's initial value (for a newly-defined e-mailer signal block) is null. Fill the default to send all (unaddressed) mail to a single address.</p> <p>The recipient list is reset to the default by the ClearNames, NewMail, and SendNow signals.</p> <p>Note that this signal does not actually send any mail. Subsequent use of the SendNow signal is required to actually send a message.</p> |
| See Also | <p>The other "Set" signals (SetFromName, SetRcptName, SetSubj, and SetBody).</p> <p>Other methods of setting the recipient address are: The LookupRcpt signal, and manually picking a recipient from the bound recipient scroller.</p> |

| SetRcptName | |
|-----------------------|--|
| Applies to | e-Mailer signal blocks |
| Description | Sets the recipient's real name. This is included in quotes in the "To:" header of the e-Mail message, in front of the recipient's e-Mail address. It is not considered part of the address and may be set to anything — or excluded entirely. |
| Direction | system-to-server |
| Type | Serial (string) |
| Value | New recipient name. May contain spaces and punctuation. |
| Expected Reply | None |
| Comments | <p>This signal is not required to send e-mail. This signal is one of several methods of setting the recipient name.</p> <p>The recipient name may be any string. It normally does not have to represent the real name of the recipient and in fact may be set to null (blank). For example, when the recipient's real name is not known, it could simply be specified as "Instructor."</p> <p>The recipient name as specified is recorded but is not actually added to the recipient list until the SetAddr signal is issued, that signal acting as a "trigger." Therefore, this signal may be sent and then resent repeatedly with different values; the most recently sent value is eventually used. Specifying the recipient name <u>after</u> the actual address does not work as expected.</p> <p>If no name has been specified when the SetAddr signal is issued <u>and</u> if that address is null (blank), <u>then and only then</u> if the name is also null, the default recipient name is used. The default recipient name is specified in the e-Mailer signal block definition and may itself be null. The initial value of the default (for a newly defined e-mailer signal block) is:</p> <p style="text-align: center;">System Manager</p> <p>The recipient list is reset to the default by the ClearNames, NewMail, and SendNow signals.</p> <p>Note that SetRcptName does not actually send any mail. Subsequent use of the SendNow signal is required to actually send a message.</p> |
| See Also | <p>The other "Set" signals (SetFromName, SetAddr, SetSubj, and SetBody).</p> <p>The comment under SetAddr for information involving multiple recipients.</p> <p>Other methods of setting the recipient name are: the LookupRcpt signal, and manually picking a recipient from the bound recipient scroller.</p> |

| <h1>SetSubj</h1> | |
|-----------------------|---|
| Applies to | e-Mailer signal blocks |
| Description | Sets the e-Mail message's Subj: header. |
| Direction | system-to-server |
| Type | Serial (string) |
| Value | New value for the Subj: header of the next e-Mail message to be sent. |
| Expected Reply | None |
| Comments | <p>Required. The e-Mail message is considered incomplete if no subject is provided. (This is the only piece of information required to send a message. The body is not required; and all other headers have default values.) If the SendNow signal is issued but the subject header has not been set (by any method), the following error message is sent back to the control system:</p> <p style="text-align: center;">No message specified -- mail cannot be sent.</p> <p>The string specified for the subject header is subject to text substitution at the time the message is actually sent. For more information, refer to "Text Substitution and File Inclusion" on page 48.</p> <p>Note that this signal does not actually send any mail. Subsequent use of the SendNow signal is required to actually send a message.</p> |
| See Also | <p>The other "Set" signals (SetFromName, SetRcptName, SetAddr, and SetBody).</p> <p>Other methods of setting the subject are: the LookupMsg signal, and manually picking a message from the bound message scroller.</p> |

| <h1>Shortcut_n</h1> | |
|-------------------------------|--|
| Applies to | e-Mailer signal blocks |
| Description | Sends message <i>n</i> to its default recipient |
| Direction | system-to-server |
| Type | Digital |
| Value | Pulse (actually, leading edge is the trigger; trailing edge is ignored) |
| Expected Reply | Done pulse |
| Comments | <p>Definition of these signals is optional. When the number of shortcut signals given in the <i>e-Mailer Signal Block Definition</i> window is 0, no such signals are defined.</p> <p>Accomplishes exactly the same as if the following signals had been sent:</p> <p style="margin-left: 40px;">NewMail LookupMsg <i>Using n, the shortcut number, as the lookup parameter</i> SendNow</p> <p>The number of Shortcut signals defined in the XSIG is specified in the e-Mailer signal block definition. The number of signals which may be defined can vary from 0 (no text substitution signals at all) to any value — limited only by practical considerations. (See “Appendix D: System limitations” on page 107 for a discussion of these limits.)</p> <p>The number of Shortcut signals defined should be set to 16 to accommodate all sixteen possible signals which may be issued from the Send e-Mail SIMPL symbol. If necessary, the list may be extended by increasing the setting on the server side and adding an External Communications w/offset (XSIG2) symbol to accommodate the additional signals. The offset should be calculated directly follow Shortcut16 in the XSIG signal space. Alternatively, replace the Send Mail symbol with a custom built XSIG.</p> <p>A Shortcut signal has the special ability to “auto-enable” the e-mailer signal block for the duration of the signal. (If it was already enabled when the Shortcut signal was issued, it remains enabled after the signal is processed; otherwise it is disabled.)</p> <p>The Done pulse should be used to de-highlight the touchscreen button that originated the Shortcut signal. Keeping the screen button highlighted while the server is busy keeps the user better informed and encourages his patience.</p> |
| See Also | <p>Enabled signal</p> <p>Refer to “Signal Block Enable / Disable” on page 59.</p> |

| SignalA_n | |
|----------------------------|---|
| Applies to | e-mailbox signal block |
| Description | Analog signal received in an e-mail "control" message |
| Direction | Server to System |
| Type | Analog |
| Value | Arbitrary 16-bit value |
| Expected Reply | None |
| Comments | <p>Definition of these signals is optional. The number of signals defined is controlled by the value in the <i>A</i> textbox, <i>Control messages</i> frame, <i>e-Mailbox Signal Block Definition</i> window. A zero (0) in this box means that no SignalA signals are defined at all.</p> <p>A message with the following "control equate" syntactical construct in the subject header <u>or</u> in any line of the body text causes the server to send an analog signal with the value <i>a</i> through the signal <i>i</i> immediately upon receipt of the message:</p> <p style="text-align: center;">CTRL>A<i>i</i>=<i>a</i></p> <p>For example, the control equate</p> <p style="text-align: center;">CTRL>A3=32768</p> <p>would send the value 32768 through signal SignalA3 as soon as the message is downloaded from the host.</p> |
| See Also | <p>SignalD_n and SignalS_n signals</p> <p>For more information on control messages, see page 34.</p> |

| <h1>SignalD_n</h1> | | | | | | | | | | | | | | | | | | | | | | |
|------------------------------|--|---------------------|----------|---------------------|------|-------|--|----|-----|--|---|---|--|-----|----|--|-----|-------|--|----|----------------|--------|
| Applies to | e-mailbox signal block | | | | | | | | | | | | | | | | | | | | | |
| Description | Analog signal received in an e-mail "control" message | | | | | | | | | | | | | | | | | | | | | |
| Direction | Server to System | | | | | | | | | | | | | | | | | | | | | |
| Type | Digital | | | | | | | | | | | | | | | | | | | | | |
| Value | Arbitrary digital state | | | | | | | | | | | | | | | | | | | | | |
| Expected Reply | None | | | | | | | | | | | | | | | | | | | | | |
| Comments | <p>Definition of these signals is optional. The number of signals defined is controlled by the value in the <i>A</i> textbox in the <i>e-Mail control message signals</i> frame of the <i>e-Mailbox Signal Block Definition</i> window. A zero (0) in this box means that no SignalA signals are defined at all.</p> <p>A message with the following "control equate" syntactical construct in the subject header <u>or</u> in any line of the body text causes the server to send an analog signal with the state <i>d</i> through the signal <i>i</i> immediately upon receipt of the message:</p> <p style="text-align: center;">CTRL>Di=<i>d</i></p> <p>For example, the control equate</p> <p style="text-align: center;">CTRL>D2=true</p> <p>would send an assert through signal SignalD2 as soon as the message is downloaded from the host.</p> <p>Note the syntax for <i>d</i> is case-insensitive and the following values are permitted:</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <td>Assert</td> <td>Deassert</td> <td><i>or</i> de-assert</td> </tr> <tr> <td>True</td> <td>False</td> <td></td> </tr> <tr> <td>On</td> <td>Off</td> <td></td> </tr> <tr> <td>1</td> <td>0</td> <td></td> </tr> <tr> <td>Yes</td> <td>No</td> <td></td> </tr> <tr> <td>Set</td> <td>Reset</td> <td></td> </tr> <tr> <td>Ok</td> <td><i>or</i> Okay</td> <td>Cancel</td> </tr> </table> | Assert | Deassert | <i>or</i> de-assert | True | False | | On | Off | | 1 | 0 | | Yes | No | | Set | Reset | | Ok | <i>or</i> Okay | Cancel |
| Assert | Deassert | <i>or</i> de-assert | | | | | | | | | | | | | | | | | | | | |
| True | False | | | | | | | | | | | | | | | | | | | | | |
| On | Off | | | | | | | | | | | | | | | | | | | | | |
| 1 | 0 | | | | | | | | | | | | | | | | | | | | | |
| Yes | No | | | | | | | | | | | | | | | | | | | | | |
| Set | Reset | | | | | | | | | | | | | | | | | | | | | |
| Ok | <i>or</i> Okay | Cancel | | | | | | | | | | | | | | | | | | | | |
| See Also | <p>SignalA_n and SignalS_n signals</p> <p>For more information on control messages, see page 34.</p> | | | | | | | | | | | | | | | | | | | | | |

| SignalS _n | |
|-----------------------|---|
| Applies to | e-mailbox signal block |
| Description | Analog signal received in an e-mail "control" message |
| Direction | Server to System |
| Type | Serial |
| Value | Arbitrary serial data |
| Expected Reply | None |
| Comments | <p>Definition of these signals is optional. The number of signals defined is controlled by the value in the <i>A</i> textbox in the <i>e-Mail control message signals</i> frame of the <i>e-Mailbox Signal Block Definition</i> window. A zero (0) in this box means that no SignalA signals will be defined at all.</p> <p>A message with the following "control equate" syntactical construct in the subject header or in any line of the body text causes the server to send a serial signal with the string <i>s</i>, trimmed, through the signal <i>i</i> immediately upon receipt of the message:</p> <p style="text-align: center;"><code>CTRL>Si=s</code></p> <p>For example, the control equate</p> <p style="text-align: center;"><code>CTRL>S4= Wag the dog.</code></p> <p>would send the serial string "Wag the dog." through signal SignalS4 as soon as the message is downloaded from the host.</p> <p>Note that the string <i>s</i> is "trimmed" (leading and trailing spaces are removed) before sending.</p> |
| See Also | <p>SignalA_n and SignalD_n signals</p> <p>For more information on control messages, see page 34.</p> |

| <h1>Status</h1> | |
|-----------------------|--|
| Applies to | e-mailbox signal block |
| Description | Analog signal received in an e-mail "control" message |
| Direction | Server to System |
| Type | Analog |
| Value | Progression of values representing check mail progress |
| Expected Reply | None |
| Comments | <p>Definition of this signal is optional.</p> <p>The Status signal is sent multiple times during the check mail process. The series of unsigned 16-bit values are intended as input to an analog gauge display — which might be on a touchscreen page or on a control system's front panel.</p> <p>The signal is sent first with a value of 0 (empty gauge); followed by a number of increments, all equal in size. The final increment is the maximum value ($2^{16} - 1 = 65,535$) which fills the gauge. The number of increments equals the number of messages resident on the host. The server checks the IN box database table for each message ID and downloads that message when missing from the table.</p> |
| See Also | CheckMail signal |

| <h1>ViewOldMail</h1> | |
|-----------------------|--|
| Applies to | e-mailbox signal block |
| Description | Switches IN box scroller between the “new mail view” and the “old mail view.” |
| Direction | Server to System |
| Type | Digital |
| Value | <u>Assert:</u> Requery IN box scroller recordset and display only old mail <u>De-assert:</u> Requery IN box scroller recordset and display only new mail |
| Expected Reply | Possible new data to IN box scroller’s List and Scrollbar signals |
| Comments | “New mail” means mail marked as new in the database table. Mail is so marked when it is first downloaded. The server automatically unmarks a message when it is “opened.” Messages can be marked as new again using the KeepAsNew signal. |
| See Also | KeepAsNew , NewCount , and OldCount signals |

Appendix D: System limitations

Serial Transmissions

The length of the value of all serial signals (all the **Set**— signals) is limited to 83 characters. Furthermore, the total length of all signals (including header bytes) to be transmitted in a single logic “wave” must not exceed 255 characters. If the total length of all signals required to specify the strings for an e-Mail message exceeds 255 characters, you must ensure that transmission occurs in several waves (by installing sufficient delays between signals).

Signal Definitions

The XSIG format limits the number of signals to 4096, only the first 1024 of which may be analog or serial signals. Furthermore, the older generation of Crestron control systems has a limit of 512 analog/serial signals. (This is not an issue with the newer CNMSX and CNRACKX systems which allow a full 4096 analog/serial signals.)

Appendix E: Standard Scroller / Custom Scroller Feature Comparison

| Standard scroller options <i>no license required</i> | Custom scroller options <i>SW-DBM license required</i> |
|--|--|
| Enable signal non-functional except in simulation (from <i>Signal Analyzer</i> window). Enabled signal sent by server as usual (but not available through <code>DBMScroller</code> macro). | Fully functional Enable and Enabled signals |
| Maximum of 8 rows x 2 columns | Any number of rows/columns in displayed list |
| e-Mail data echoed through e-Mailer signal block only | Echo selection of fields from “picked” records through Data signals |
| <i>Same</i> | Specify list fields, data fields, and SQL queries |
| <i>Same</i> | Sort by list fields or ID field |
| N/A | Modify any field(s) through Write signals |
| N/A | Add new records |
| N/A | Delete “picked” record |
| N/A | Local error reporting signals |
| N/A | Successive query signals |
| N/A | Auto-pick feature |

Appendix F: Dial-up TCP/IP configuration

You should install the Windows *Dial-up Networking* software. Configure it to connect using Point-to-Point Protocol (PPP) without user confirmation and to disconnect after idle for 1 minute.

Our experience is that the initial settings upon a fresh installation of *Dial-Up Networking* are usually already set to allow for dialing without user confirmation. If, however, you find this is not the case, you may be able to reconfigure the software by following these steps:

1. Locate and start *Dial-Up Networking*. A shortcut to this application can usually be invoked from the *Start* menu, by selecting **Start | Programs | Accessories | Dial-Up Networking** (or possibly **Start | Programs | Accessories | Communications | Dial-Up Networking**).
2. In the resulting window, click *More* to reveal a pop-up menu from which select **User Preferences**.
3. In the resulting window, select the *Appearance* tab
4. Make sure the checkbox labeled *Always prompt before dialing* is unchecked.
5. Click **OK**.

One way of configuring the connection to auto-disconnect after 1 minute is as follows:

1. Bring up *Dial-Up Networking's User Preferences* window again.
2. Select the *Dialing* tab.
3. Enter 60 into the textbox labeled *Idle seconds before hanging up*.
4. Click **OK**.

Note that different versions of *Dial-Up Networking* may have different windows than those described above. Also note that *Dial-Up Networking* is not the only software available for establishing a PPP connection over a dial-up line. If your computer uses other software, you will have to hunt around for options similar to those described above.