CTI 5250-BAS ACCESS MODULE
INSTALLATION AND OPERATION GUIDE
Autojectors Model - SPQ#150

Version 1.1
CTI Part #062-00145

This Installation and Operation Guide is published by Control Technology Inc., 5734 Middlebrook Pike, P.O. Box 59003, Knoxville, TN 37950-9003, telephone 1-800-537-8398.

DOCUMENT DISCLAIMER STATEMENT

Every effort has been made to ensure the accuracy of this document; however, errors do occasionally occur. CTI provides this document on an "as is" basis and assumes no responsibility for direct or consequential damages resulting from the use of this document. This document is provided without express or implied warranty of any kind, including but not limited to the warranties of merchantiblility or fitness for a particular purpose. This document and the products it references are subject to change without notice. If you have a comment or discover an error, please call us toll-free at 1-800-537-8398.

PREFACE

This ***Installation and Operation Guide*** provides installation and operation instructions for the Autojector's Model of the CTI 5250-BAS Operator Panel for SIMATIC TI Series 305, 405, 500 and 505 programmable controllers. We assume you are familiar with the operation of SIMATIC TI Series 305, 405, 500 and 505 programmable controllers. Refer to the appropriate SIMATIC TI user documentation for specific information on the SIMATIC TI Series 305, 405, 500, and 505 programmable controllers and I/O modules.

The 5250-BAS Operator Panel is intended for use with the SIMATIC TI Series 305, 405, 500 and 505 programmable controllers. The rugged compact package is suitable for panel mounting in industrial environments.

The 5250-BAS allows the development of a custom operator interface using the BASIC programming language. Source code for application programs described in Appendix A are contained on the diskette which is shipped with the 5250-BAS Operator Panel.

Figure 1 The 5250-BAS (Autojector's Model) Operator Panel

USAGE CONVENTIONS

NOTE:
Notes alert the user to special features or procedures.

CAUTION:
Cautions alert the user to procedures which could damage equipment.

WARNING:
Warnings alert the user to procedures which could damage equipment and endanger the user.

TABLE OF CONTENTS

TABLE OF FIGURES

SECTION I. INSTALLATION AND OPERATION

CHAPTER 1. DESCRIPTION

The 5250-BAS Operator Panel is intended for use with the SIMATIC TI Series 305, 405, 500 and 505 programmable controllers. The rugged compact package is suitable for panel mounting in industrial environments.

The 5250-BAS Operator Panel implements a subset of the industry standard Intel MCS BASIC-52. Additional features have been added to support the 4 line by 20 character LCD display, to control the operator keypad, to communicate with the SIMATIC TI programmable controllers through the PROGRAM Port, and to support communication through the AUX Port.

This manual should be used in conjunction with the Intel MCS BASIC-52 USERS MANUAL provided with this product. The 5250-BAS Operator Panel includes the functionality described in the Intel documentation except for the EPROM programming functions. In addition, statements such as ONEX1, PWM, UI1, UI0, UO1, UO0, and IDLE, which depend on the specific hardware implementation, should not be used.

TCTALK supports the BASIC interpreter provided with the 5250-BAS Operator Panel. TCTALK runs on a personal computer attached to the 5250-BAS Operator Panel through the AUX port and provides the programming environment to build and debug BASIC programs executing on the 5250-BAS Operator Panel. TCTALK also provides a line editor for developing programs and controls for uploading and downloading programs to the Operator Panel.

TCTALK is fully described in Section II.

1.1 Features

- Subroutine Calls to access PLC memory for the SIMATIC TI 305, 405, 500 and 505 Series programmable controllers
- Program control of 20 keypad inputs and 80 character LCD display
- Intel MCS BASIC-52 Interpreter
- TCTALK terminal emulation program with line editor and HELP screens
- Installed application program that supports "out of the box" operation
- Auxiliary serial port
- Program port for communication to PLC or serial device
- Custom front panel option available

1.2 Applications

Three applications are included on the diskette shipped with each 5250-BAS Operator Panel:

Memory Access for the SIMATIC TI 305/405/500/505 PLCs
This program provides read access to each memory type (Timer, Counter, Stage, V memory, etc.) supported by the selected PLC. The program allows operator selection of the attached PLC so that all PLC types are supported.

Write access to each memory type is supported by the 5250-BAS Operator Panel, but the code to implement write access is not included in the program.

Loop Accesss Module
This program provides a LOOP Access Module for the SIMATIC TI 545 PLC. Loop parameters for up to 64 PID loops may be monitored and modified with this program.

Template Program
This program provides an organized approach to building customized operator interface applications. The program is data-driven in that the operator scripts are built from BASIC data statements without modifying the program source code.

Source code for these applications are provided on a diskette shipped with each 5250-BAS Operator Panel.

1.3 Theory of Operation

The BASIC interpreter program and commands which allow the Operator Panel to interact with the SIMATIC TI controller are permanently stored in internal ROM (Read Only Memory). The terminal emulation program TCTALK is provided on diskette for installation and execution on a personal computer attached to the Operator Panel through the Auxiliary port. HELP screens are available with TCTALK to assist the user in building and executing programs.

BASIC program statements are described in the Intel MCS BASIC-52 documentation.

1.4 Front Panel Description

A brief functional description is given here.

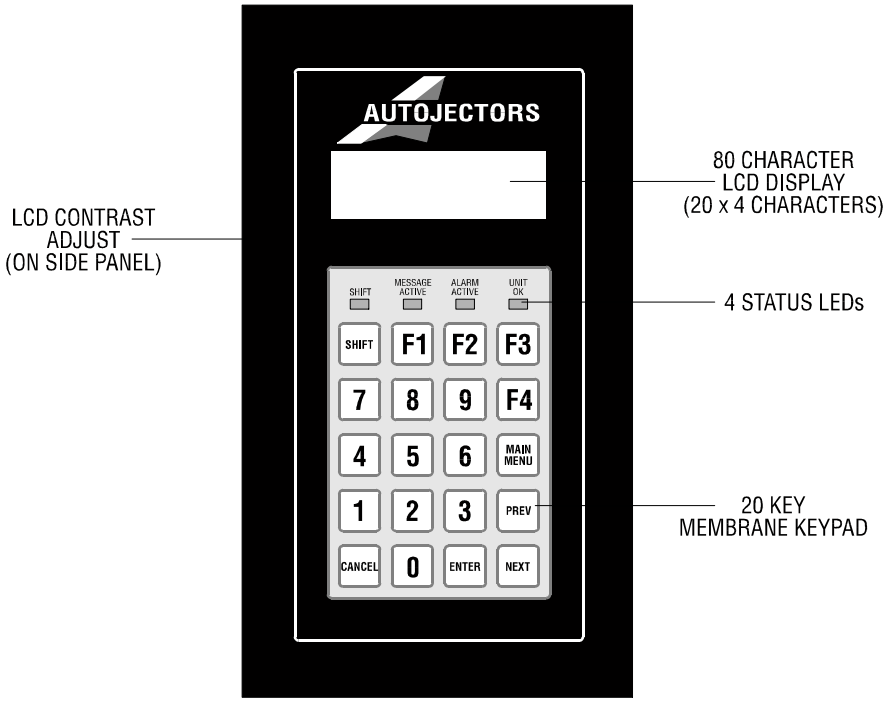


Figure 2 Front Panel Description

The **Keypad** allows the user to enter auxiliary commands or to read or write data to the PLC, which will be displayed to the LCD.

LCD Brightness Control - On the left side of the 5250-BAS there is an adjustment control for the brightness of the LCD display. A small screwdriver may be used to adjust the control (the diameter of the control is 1/8 inch).

The range of the adjustment control is slightly less than one full turn. DO NOT PUSH on the adjustment control with the screwdriver.

The **Shift LED** is illuminated when the Shift function is activated.

The **L1, L2 and L3 LEDs** are illuminated under the control of the executing BASIC program.

1.5 Communications

1.5.1 Operator Panel to Programmable Logic Controller

The Operator Panel interfaces with the SIMATIC TI Series PLC via the appropriate communication cable. See Chapter 2 for cable descriptions.

The Operator Panel Program Port is configured as DTE (Data Terminal Equipment). Program Port communication settings are controlled by the executing BASIC program. Default settings are 9600 baud, 7 data bits, odd parity and one stop bit.

1.5.2 Operator Panel to Auxiliary Port

The auxiliary output serial port on the 5250-BAS Operator Panel is used for communicating with TCTALK terminal emulation program or for printing or displaying message information. The Auxiliary Port is configured as DTE (Data Terminal Equipment) and may be connected to any RS-232 device. Auxiliary Port settings are 9600 baud, 8 data bits and no parity.

1.6 Battery Replacement

The 5250-BAS is equipped with an internal 3.6V lithium battery that should provide memory backup for approximately 8 years when operated in normal environments. Battery life can diminish if the unit is subjected to extreme temperatures or frequent temperature cycles.

It is recommended that the unit be returned to CTI for service when battery replacement is required. However, a qualified technician can access the battery by removing the rear panel. A replacement battery can be ordered from CTI (Part #075-00009).

CHAPTER 2. INSTALLATION

2.1 Mounting the 5250-BAS Operator Panel

To comply with NEMA 4 specifications, the 5250-BAS must be installed in a NEMA 4 enclosure according to the following procedure:

Step 1. Locate the position in the NEMA 4 enclosure where the module is to be mounted. Ensure that there is adequate panel and depth clearance for mounting. Refer to the following figure for overall clearances.

Step 2. Cut an opening and drill 10 mounting holes in the enclosure panel as shown in the figure below.

Step 3. Line up the studs on the module with the holes in the enclosure panel. Do not remove the gasket from the studs.

Now insert the module into the cut-out through the front of the panel. The gasket should be between the module and the front of the panel.

Step 4. Remove the nuts from the packing envelope and install them onto the studs. Tighten the nuts in the sequence and to torque specifications shown in the figure below.

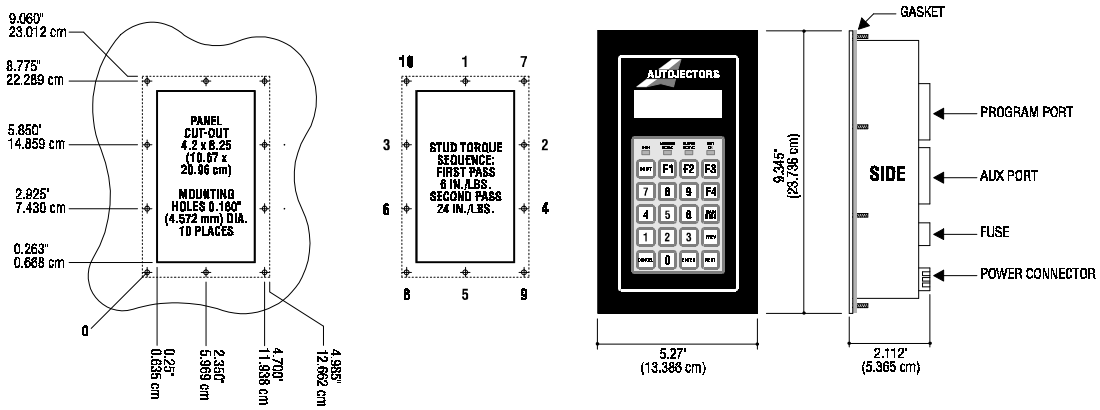


Figure 3 Mounting Dimensions

2.2 Connecting Power to the 5250-BAS Operator Panel

110VAC power is supplied to the Operator Panel by connecting the appropriate lines to the screw terminals of the power connector located on the rear panel.

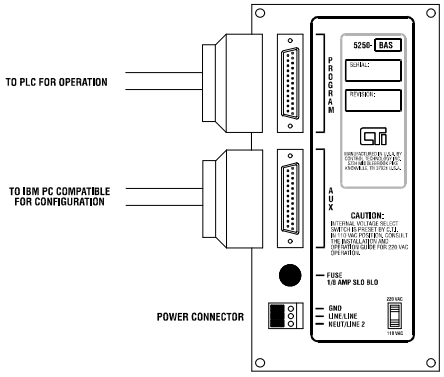


Figure 4 Module Wiring

For 220VAC operation, remove the 4 screws from the back panel, change the voltage select switch (see figure) from 110V to 220V position. Reinstall the back panel before connecting power to the unit. The supply line wires should be 14-18 AWG and rated at 600V.

The Model 5250-BAS is protected by a 125 mA 250V fuse located directly above the line voltage selector. To remove the fuse, insert a flat-blade screwdriver into the fuse hold cap and turn counter-clockwise. The replacement fuse should be Littlefuse #218.125 or equivalent. To replace the fuse, first insert a new fuse in the holder cap. Next insert the new fuse and cap into the fuse holder barrel. Turn the fuse holder cap clockwise using a flat-blade screwdriver.

WARNING:

Ensure that the voltage select switch is in the correct position before connecting 220 VAC. Equipment damage can occur which will void the product warranty. The unit is factory configured for 110 VAC.

2.3 Connecting the 5250-BAS Operator Panel to a Series 500 and 505 PLC

The 5250-BAS Operator Panel is connected to the Series 500 and 505 PLC Programming Port using a single 25-pin subminiature Type "D" male to 25-pin subminiature Type "D" male cable. The Program Port is configured as DTE (Data Terminal Equipment). Communications port settings on the PLC are: 9600 baud, 7 data bits, odd parity, one stop bit.

If the need arises for a cable longer than the standard cables available from CTI (see figures 5 and 6 for standard CTI numbers), one can be constructed as shown in the following figures.

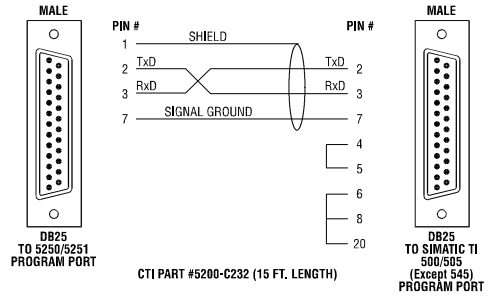


Figure 5 RS-232 to a Series 500 PLC

Cable length for RS-232 should be limited to 50 feet or less. The RS-232 should be constructed using Belden 8771 (or equivalent) cable.

Cable length for RS-422 should be limited to 4000 feet or less. The RS-422 cable should be constructed of Belden 8723 (or equivalent). The cable shield (drain) should be attached to pin 1 on the Operator Side of the cable only.

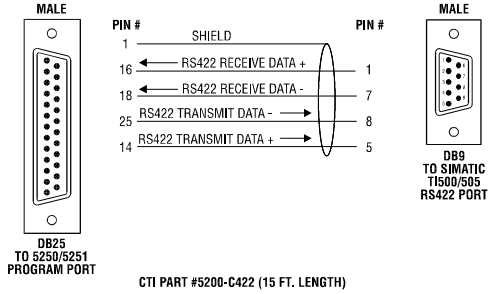


Figure 6 RS-422 to a Series 500 PLC

2.4 Connecting the 5250-BAS Operator Panel to a Series 405 PLC

The 5250-BAS Operator Panel is connected to the Series 405 PLC Programming Port using a single 15-pin subminiature Type "D" male to 25-pin subminiature Type "D" male cable. The Program Port is configured as DTE (Data Terminal Equipment). Communications port settings on the PLC are: 9600 baud, 7 data bits, odd parity, one stop bit.

If the need arises for a cable longer than the standard cable available from CTI (see Figure 7 for the standard CTI number), one can be constructed as shown in the following figure.

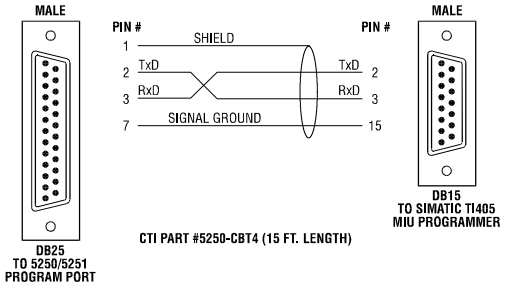


Figure 7 RS-232 to a Series 405 PLC

Cable length for RS-232 should be limited to 50 feet or less. The RS-232 should be constructed using Belden 8771 (or equivalent) cable.

2.5 Connecting the 5250-BAS Operator Panel to a Series 305 PLC

The 5250-BAS Operator Panel connects directly to the SIMATIC TI 335 PLC through the RS-232C serial port and connects through the SIMATIC Data Communications Unit (DCU305-03DM) to the SIMATIC TI 315 and 330 PLCs.

SIMATIC 315 and 330 PLCs:

The CTI cable 5200-C232 connects the 5250-BAS Operator Panel to the serial port on the DCU. The 5200-C232 cable diagram follows:

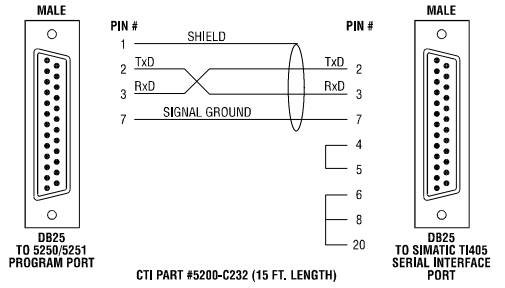


Figure 8 5250-BAS to Serial Port on DCU

Cable length should be limited to 50 feet or less. The RS-232 should be constructed using Belden 8771 or equivalent cable.

Switch settings for the DCU are as follows:

1. Set the online/offline switch to ONLINE

2. Set DIP Switch 1 to 9600 baud (Position 2 on, all other positions off)

3. Set DIP Switch 2 to Station Address 1. (Position 1 on, all other positions off)

The programming port is configured as DTE (Data Terminal Equipment). Communication settings are 9600 baud, 8 data bits and no parity.

Refer to TISOFT2 SIMATIC TI305 User Manual for more details.

SIMATIC 335 PLC:

The cable diagram for connecting the 5250-BAS Operator Panel to the SIMATIC TI 335 PLC follows:

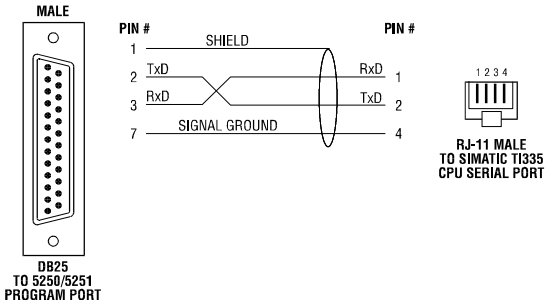


Figure 9 RS-232 to a Series 335 PLC

The RS-232C serial port for the SIMATIC TI 335 PLC requires an RJ Handset jack. The baud rate for the port is set using switch 2, positions 3 and 4. 9600 baud is selected when positions 3 and 4 are in the OFF position.

Refer to the SIMATIC TI 335 Central Processing Unit manual for additional details.

2.6 Connecting the 5250-BAS Auxiliary Port to a Programming Device

The additional auxiliary output serial port is used for programming the 5250-BAS. Additionally, this port can be used for printing or displaying messages to an external serial device. The Auxiliary Port may be connected to any RS-232 display device or printer. The Auxiliary Port is configured as DTE (Data Terminal Equipment). Auxiliary Port communication settings are: 9600 baud, 8 bits, no parity.

Cable diagrams for connecting to an IBM compatible PC or AT computer are outlined below. Custom cables can be fabricated by using the pin-outs and signals shown in the following diagram:

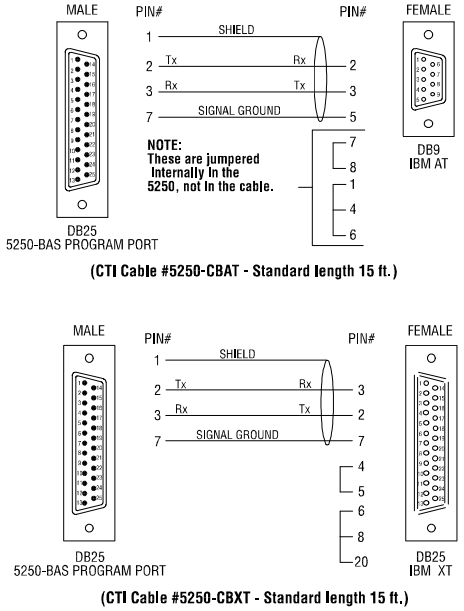


Figure 10 5250-BAS Auxiliary Port to a Programming Device

2.7 Cleaning the Front Panel

When dirt obscures the keypad or LCD window, the front panel should be carefully cleaned. CTI only recommends using a mild soap and water solution or a mild ammonia base window cleaning agent such as Windex (TM). Clean the front panel with a soft cloth or sponge using vertical strokes. Wipe off excess cleaner to prevent water spots.

> ***CAUTION:***
> *To avoid damage to the laminated plastic front panel, do not clean with abrasive, alkaline, or solvent cleaners. Do not scrape the front panel with razor blades, squeegees or other sharp instruments.*

CHAPTER 3. OPERATION

3.1 Initial Power Up

The 5250-BAS will display the following screen when power is first applied to the unit, provided a BASIC program has not been loaded:

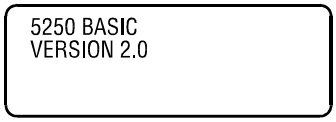


Figure 11 Initial Power Up Screen

If a program has been loaded into the unit, the display will be controlled by the BASIC program.

3.2 Keypad Entry

Operation of the Operator Panel is controlled through the keypad. Included on the keypad are digits 0-9, -(minus) and . (decimal point) for entering numeric data and addresses, user defined function keys, CANCEL, ENTER, and RESET keys.

3.2.1 SHIFT Key

The blue SHIFT key works in conjunction with each key on the keypad. When the SHIFT key is pressed, the SHIFT LED lights, and the SHIFT (blue) function is active. Two shifted keys are labeled NO, and - (minus). The remaining keys are user defined within the BASIC program. Keypad codes for each shifted key are listed in Appendix B.

After pressing the SHIFT key, the SHIFT function remains on only until the next key is pressed. The following example shows how the SHIFT key is used.

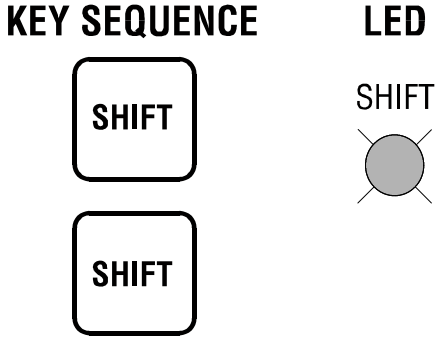


Figure 12 SHIFT Key

3.2.2 RESET Key

Both the RESET key and the SHIFT RESET key are trapped by the internal BASIC firmware and result in a restart of the loaded user program. Prior to restarting the program, the following message will be displayed on the display.

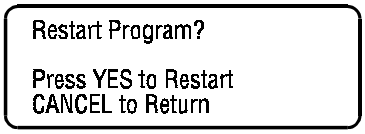


Figure 13 Reset Message

The indicated response will proceed to restart the program. Otherwise, the RESET key will be ignored and the program will resume execution.

The RESET key may be overridden by the executing program using CALL 3. See Section II.

3.2.3 Other Keys

The other keys return to the user program the keypad scan code indicated in Appendix B. The user program retrieves each code from the keypad input buffer and determines program response to operator keypad input.

3.2.4 LEDs

Three LEDs labelled L1, L2 and L3 are provided. The LEDs may be turned on and off under user program control.

L1 shows **GREEN**
L2 shows **RED**
L3 shows **GREEN**

3.3 TCTALK

TCTALK is an MS-DOS program designed to provide communications with the 5250-BAS Operator Panel and a personal computer. TCTALK runs on the personal computer and communicates with the Operator Panel through the serial port. The program port of the Operator Panel is used to communicate with the SIMATIC TI PLC or other appropriate serial device.

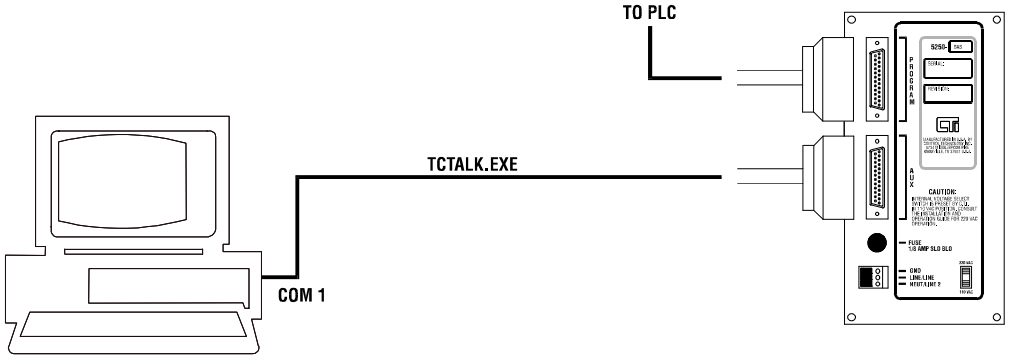


Figure 14 TCTALK Connection

TCTALK is invoked at the DOS prompt

```
>TCTALK
```

TCTALK options and configuration file specifications are discussed in Section II.

SPECIFICATIONS

Data Display:	1" x 3" 80 character (4x20) alphanumeric high temperature supertwist LCD display with wide viewing angle
LED Indicators:	SHIFT, L1, L2, and L3
Communications (2 ports):	5250-BAS Program Port to PLC Programming Port via RS-232 or RS-422 5250-BAS Aux Port: RS-232
Power Requirements:	120/220 VAC (switch selectable) (24 VDC available on special request)
Fuse:	Littlefuse #218.125 125mA @ 250 VAC CTI Part #80-62
Battery:	Tadrian TL-5242 3.6V Lithium (8 year shelf life) Current drain <20 µA with AC power removed
Storage Temperature:	-40° to 80°C (-40° to 185°F)
Operating Temperature:	0° to 60°C 32° to 140°F
Relative Humidity:	5% to 95% non-condensing
Agency Approvals:	UL, UL Canada, FM
Shipping Weight:	4 lbs. (1.8 Kg.)

Specifications subject to change without notice.

SECTION II. 5250-BAS OPERATOR PANEL PROGRAM ENVIRONMENT

CHAPTER 4. TCTALK

4.1 TCTALK Description

TCTALK provides control for the development of user programs for the 5250-BAS Operator Panel. Function keys are used as commands, HELP screens are provided online, and editing and resequencing of programs are supported.

TCTALK uses a configuration file at start up (default file is TCTALK.CFG). This file may be modified by the user.

4.2 Hardware Requirements

The configuration software is supplied on a 3.5" DSDD diskette and will run on any IBM PC, XT, AT, PS2 or IBM compatible computer system running PC or MS-DOS v2.0 or greater. The computer system must have 512K of RAM and a CGA, EGA, VGA or Hercules monochrome display. Hard disk storage is not a requirement but may be useful if multiple configurations are to be generated and stored. One serial port is required.

4.3 TCTALK Start Up

Be sure a cable to the target Operator Panel has been installed before starting TCTALK.

TCTALK is started by typing **TCTALK** at the DOS prompt.

Type CTRL-C to put TCTALK in control of the Operator Panel. The TCTALK prompt is the >symbol.

See 1.3.3 TCTALK Options for advanced usage features.

4.3.1 Status Line

TCTALK uses the top line of the display screen for its status line.

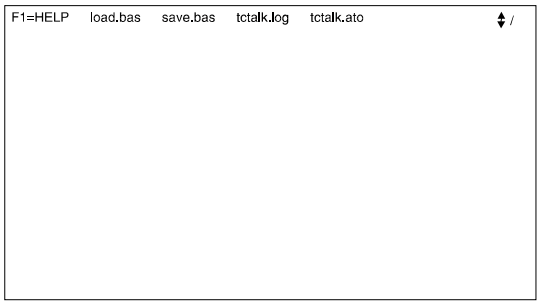


Figure 15 TCTALK Status Line

F1 = HELP	A reminder that F1 invokes HELP is located at the left side of the status line.
LOAD.BAS	The default file name for the user program either loaded or to be loaded into the Operator Panel from disk.
SAVE.BAS	The default file name for the user program either saved or to be saved from the Operator Panel to disk.
TCTALK.LOG	The default file name for logging of TCTALK activity.
TCTALK.ATO	The default file name for automatic data entry.

The upper right corner of the status line contains four symbols that describe the status of TCTALK.

Position 1	Blank or A	Indicates automatic data entry is either disabled or enabled
Position 2	Blank or L	Indicates logging is either disabled or enabled
Position 3	DOUBLE ARROW	TCTALK is in Interactive Mode
	DOWN ARROW	TCTALK is running Load Utility
	UP ARROW	TCTALK is running Save Utility
	Greek alpha	TCTALK is waiting for response from OPERATOR PANEL. This symbol is rarely seen except when an error such as timeout occurs.
Position 4	Spinner	TCTALK is working normally as long as the spinner is spinning. If it stops, TCTALK is waiting on disk access. If the spinner remains stopped, TCTALK is probably having communication problems with the Operator Panel.

4.3.2 Operating Modes

Program Mode - Operator Panel
The Operator Panel is in Program Mode while executing a BASIC program. To stop execution, enter Ctrl-C from the terminal keyboard while TCTALK is executing. The Operator Panel may now be controlled by TCTALK in Interactive Mode.

If the Operator Panel is outputting to the Auxiliary Port, Ctrl-C will not stop the output operation. Once the output has terminated, Interactive Mode may be entered with Ctrl-C.

Interactive Mode - TCTALK
Once TCTALK starts execution and the Operator Panel has responded to Ctrl-C, any key typed on the PC terminal keyboard will be sent to the Operator Panel except for the keys listed in Figure 16. These keys are command keys to TCTALK. A description of each command key for TCTALK follows:

F1 Pressing the F1 key on the keyboard will cause the first of several HELP displays to appear on the right side of the screen. These displays define the FUNCTION KEYS and CURSOR KEYS for Interactive Mode and list the CALL function library available for the Operator Panel.

Ctrl-F1 Pressing Ctrl-F1 will cause TCTALK to start a DOS Shell. TCTALK is reentered by typing EXIT at the DOS prompt.

During this processing, interaction with the Operator Panel will stop and any characters sent to the Operator Panel may be lost. It is recommended that the Operator Panel be in an idle or wait state before starting the DOS Shell.

F1:	Help Screens	**F2:**	RUN (cr)
Ctrl-F1:	Shell to DOS	**Ctrl-F2:**	RUN (space)
Shift-F1:	Change Parameters	**Shift-F2:**	not used
Alt-F1:	Change Filenames	**Alt-F2:**	not used
F3:	Enter Save Mode	**F4:**	Enter Load Mode
Ctrl-F3:	not used	**Ctrl-F4:**	not used
Shift-F3:	not used	**Shift-F4:**	not used
Alt-F3:	Change Save Filename	**Alt-F4:**	Change Load Filename
F5:	CONT (cr)	**F6:**	Enable Log File
Ctrl-F5:	not used	**Ctrl-F6:**	Clear Log File
Shift-F5:	not used	**Shift-F6:**	Disable Log File
Alt-F5:	not used	**Alt-F6:**	Comment Log File
F7:	Exit TCTALK	**F8:**	LIST (cr)
Ctrl-F7:	not used	**Ctrl-F8:**	LIST (space)
F9:	Enter EDIT Mode	**F10:**	Start Auto Command
Ctrl-F9:	not used	**Ctrl-F10:**	Restart Auto Command
Shift-F9:	not used	**Shift-F10:**	End Auto Command
Alt-F9:	not used	**Alt-F10:**	Single Step Command

Alt-e: Start/stop key for displaying HEX character value

Figure 16 Interactive Mode Function Keys

Shift-F1 Pressing Shift-F1 will cause TCTALK to display the current parameters and prompt for possible modification. A typical Change Parameters Menu follows.

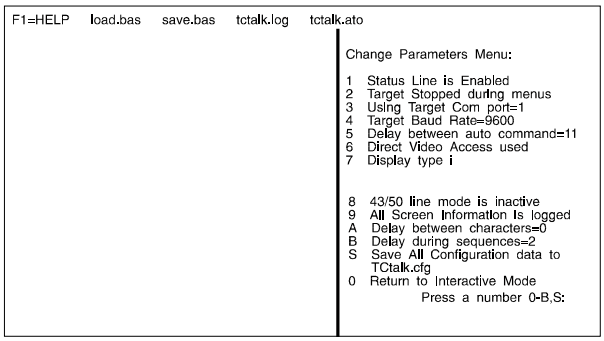


Figure 17 TCTALK Change Parameters Menu

Pressing the number of the option in the menu will toggle through the values. For example, pressing 1 will disable the status line. Pressing 1 again will enable it.

Item 2 is not functional.

Item 3 will toggle between COM1 and COM2.

Item 4 will toggle baud rates. Use 9600 baud as the communication rate for the Operator Panel.

Item 5 adjusts the number of seconds in tenths between processing of commands from the automatic data entry file.

Item 6 allows selection of Direct Video Access or BIOS Video Access. Use Direct Video Access for faster processing. If snow develops on your display, switch to BIOS Video Access.

Item 7 selects the class of display that is in use. Default is MONO, but EGA, VGA, and COLOR are available.

Item 8 selects 43 or 50 line mode for EGA and VGA displays. This item is ignored for other display types.

Item 9 selects the level of logging information to the log file (TCTALK.LOG). The minimum level is keystrokes only, the intermediate level is keystrokes and all characters received from the Operator Panel, and the maximum level includes screens and menus.

Items A and B adjust the time between characters being sent to the Operator Panel. Adjusting these values from the defaults provided by Control Technology may produce erratic results.

Item S saves the current menu parameters in the configuration file (TCTALK.ATO).

Press 0 to exit the Change Parameters Menu.

Alt-F1 Pressing Alt-F1 will display the current file names and prompt for changes. Five file names may be changed.

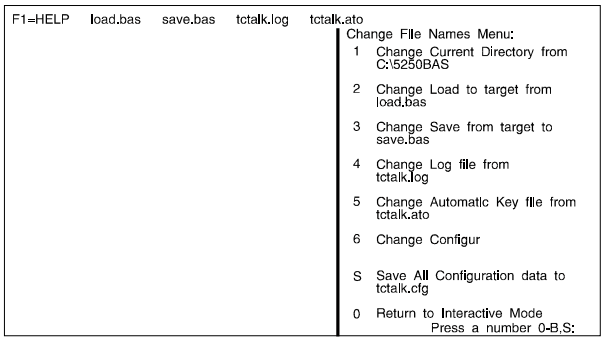


Figure 18 TCTALK Change File Names Menu

Press 1 to change the current directory.
Press 2 to change the filename of the program to be transferred to the Operator Panel.
Press 3 to change the filename to receive the program from the Operator Panel.
Press 4 to change the filename to receive the log data when logging is enabled.
Press 5 to change filename of the data that is used when automatic data entry is enabled.
Press 6 to change the filename for the configuration file.
Press S to save the configuration file.

F2 Pressing F2 causes the RUN command followed by a carraige return to be sent to the Operator Panel which will start execution of the program stored in the Operator Panel. Enter Ctrl-C to stop execution. See Program Mode - Operator Panel.

Ctrl-F2 The RUN command sent to the Operator Panel without a carraige return, The user may enter a line number followed by a carraige return. This will start program execution at the line number specified.

F3 Pressing F3 will cause TCTALK to start the Save Utility which results in the following sequence:

the current save file is opened (created if necessary) for write,

a message is displayed on the terminal screen identifying the existing save file along with a request for approval to destroy current contents of the file,

Ctrl-C is generated and sent to the Operator Panel to verify that it is ready to upload the program file,

a LIST command generated and sent to the Operator Panel to initiate the upload to disk (save) operation,

the program is stored on disk as it is received from the Operator Panel, and

the file is closed upon completion of the transfer and TCTALK returns to Interactive Mode.

Once F3 has been pressed in Interactive Mode and the Save Utility has been entered, these function keys become active:

F1 Help Screen
F3 Return to Interactive Mode
F4 Return to Interactive Mode
F7 Exit TCTALK

Other keys are ignored (and if pressed will cause the terminal bell to sound).

If an error occurs during the Save Utility processing, appropriate error messages are displayed, processing stops, and TCTALK returns to Interactive Mode.

Alt-F3 Pressing Alt-F3 will allow the filename that will receive the program from the Operator Panel to be changed.

F4 Pressing F4 will cause TCTALK to start the Load Utility which results in the following sequence:

the current load file is opened for read only,

Ctrl-C is generated and sent to the Operator Panel to verify that it is ready for download,

the contents of the file are sent one character at a time (waiting for echo before sending the next) until the complete file is downloaded,

the file is closed, and

TCTALK enters Interactive Mode

While in the Load Utility, only these function keys are active:

F1 Help Screen
F3 Return to Interactive Mode
F4 Return to Interactive Mode
F7 Exit TCTALK

Other keys are ignored (and if pressed will cause the terminal bell to sound).

If an error occurs during the Load Utility processing, appropriate error messages are displayed, processing stops, and TCTALK returns to Interactive Mode.

Alt-F4 Pressing Alt-F4 will allow the filename that will be transferred to the Operator Panel to be changed.

F5 Pressing F5 will cause the string CONT followed by a carraige return to be sent to the Operator Panel. This will restart the Operator Panel program from its point of interruption by a STOP statement.

F6 Pressing F6 will enable the disk loggin process. The setup option in the configuration file (see Shift-F1, Item 9) determines the information logged to the log file.

Ctrl-F6 Pressing Ctrl-F6 will clear the current information in the log file. TCTALK will continue to enter information to the log file.

Shift-F6 Pressing Shift-F6 will stop the collection of information in the log file. Logging can be resumed by pressing F6. New information will be added to the end of the existing log file.

Alt-F6 Pressing Alt-F6 will allow the user to add comments to the log file. Prompts are used to solicit comments.

F7 Pressing F7 will cause TCTALK to stop execution. The Operator Panel will continue normal operation. The communication port of the computer is reset and a normal exit is made to the computer operating system.

F8 Pressing F8 will cause the string LIST followed by a carraige return to be sent to the Operator Panel. If the Operator Panel is not executing, it will start listing the BASIC program to the computer terminal screen.

Ctrl-F8 Pressing Ctrl-8 will cause the strin LIST followed by a space to be sent to the Operator Panel. A line number may be entered to start th listing operation.

F9 Pressing F9 will cause TCTALK to start the Editing Utility. Typing a line number followed by ENTER will initiate editing the line. These function keys are active:

Cursor The SPACE bar moves the cursor right one position, and the BACKSPACE key moves the cursor left one position.

Insert Ctrl-A begins insertion of text including spaces in current line; a second Ctrl-A ends text insertion.

Delete Ctrl-D deletes character at the current cursor position.

Replace Typing a new character a the current cursor postion.

Retype Pressing the ENTER key retypes the line and positions the cursor at the beginning.

Exit Ctrl-Q exits the EDIT Utility and saves the editing changes.

Ctrl-Q abirts editing and returns the original line intact.

In addition, these typed characters perform these functions while in the EDIT Utility:

DEL Character deletion (Ctrl-D)

INS Character insertion (Ctrl-A)

END Exit Edit Utility saving changes (Ctrl-Q)

HOME Carraige Return

(right arrow) Moves cursor one postion to the right

(left arrow) Moves cursor one position to the left.

F10 Pressing F10 will cause TCTALK to read the automatic data entry file. The delay between commands initiated from the automatic data entry file is controlled in the configuration file (see Shift-F1, Item 5). The user can interact with the Operator Panel while automatic commands are being generated. When a manual command is initiated, additional delays are generated to ensure that the manual command is completed before the next automatic command is initiated.

Shift-F10 Pressing Shift-F10 will stop automatic command entry processing.

Alt-E Each character that is received from the Operator Panel is normally placed on the terminal screen. Alt-E starts/stops the display of the hexidecimal value of the character. This is useful for monitoring control characters and other non-printable characters.

Ctrl-C Entering Ctrl-C from the keyboard will cause the Operator Panel to stop execution of its BASIC program.

4.3.3 TCTALK Options

Options are:	/s	Force set up mode (Shift-F1). This option causes TCTALK to display Change Parameters Menu.
	/a	Start up in Interactive Mode with automatic data entry enabled using the data file defined in the system configuration file.
	/a=file.ext	Start up with automatic data entry enabled using the data file (file.ext)

For example,

```
TCTALK /s/a=TCTALK00.ABC
```

Space is required after TCTALK and before parameter options.

CHAPTER 5. BASIC LANGUAGE ENHANCEMENT

5.1 Basic Language Enhancement

5250 BASIC includes an enhancement to the syntax of BASIC-52 that makes the use of strings much more flexible. In 5250 BASIC, DATA statements can contain string initializers (character strings delimited by ""), and READ statements can correctly initialize strings from the corresponding DATA. This dramatically improves the speed of "data driven" programs that use DATA statements for string initialization, and also makes such programs easier to maintain.

Example:

```
>10 STRING 3000, 40
>20 DATA 4, "this is a test", 17
>30 READ A, $(0), B
```

In standard BASIC-52, this type of functionality would require initializing the string in a FOR loop that READs each byte from the DATA statement and uses the ASC operator to initialize the string.

The RESTORE command syntax has been extended to allow the form.

```
RESTORE line number
```

This reset the READ data pointer to the indicated line number, rather than the beginning of the program.

Example:

```
RESTORE 1000
```

After this command is executed, the next READ statement will look for data beginning with line 1000. This extension greatly enhances the flexibility of the READ/DATA usage.

The renumber command REN is used to remember BASIC programs. A description of this command is given in Section 2.14.

5.2 Implementation of Other 5250-BAS Enhancements

The BASIC-52 PRINT statement and its relatives PH0 and PH1 are sufficient to allow outputting character strings to either the serial ports or to the LCD display. Output is mapped as follows:

AUX Port - Output for the BASIC Interpreter in command mode and for PRINT, PH0, PH1, and LIST statements.

LCD Display - Output for PRINT#, PH0#, PH1#, and LIST#.

Program Port - Output for PRINT@, PH0@, PH1@, and LIST@.

The remaining functions are made available through the use of the CALL statement.

> ***NOTE:***
> *ASCII control characters (values less than 32) are not sent to the LCD. Cursor positioning (see below) should be used to control the location of any character string sent to the LCD unless the cursor is in a known location. The remainde of strings which are longer than the remaining space on a line will be displayed two lines further, with wrap from the fourth line to the first.*

5.3 Call Implementation

5250 enhancements implemented using the CALL statement are passed parameters (if necessary) using the PUSH statement. If a function returns values, they are placed into BASIC variables using the POP statement. The number of parameters and/or return values for each function are documented below. This must be adhered to, or programs will not behave properly. In some cases, the program will trap directly into command mode before the CALL executes.

CALL Statement Summary is in Appendix B.

5.4 Keypad Support Calls

The 5250 keypad is supported by three CALLs:

CALL 1 - Return Keypad Value

This call returns a keypress value (ASCII) to be POPed, if any, or -1 if no key has been pressed.

The keypad handler scans for keypresses using a timer interrupt, and keystrokes are buffered. Most of the keys on the keypad return the ASCII value corresponding to the non-shifted or shifted value indicated on the keypad (See Appendix B for ASCII values).

The following program illustrates the use of CALL 1 and allows for display of numeric values of all key combinations. ASCII characters are also printed for non-control characters.

```
>10PRINT "Press a key on the keypad" : PRINT
>20CALL 1 : POP A : IF A = -1 THEN 20 : REM LOOP WAITING FOR KEY
>30PRINT "Key value:" ,A
>40IF A>32 THEN PRINT "Key character" ,CHR(A) ELSE PRINT
>50GOTO 20
```

Type Ctrl-C to exit this program.

CALL 2 - Clear Keypad Input Buffer

This call clears the keypad input buffer of any keystrokes that were entered before the call is made. It has no parameters and does not return any values.

CALL 3 - Disable/Enable Program Restart

In 5250-BASIC Version 2.0, the RESET key on the keypad has a special function. It allows the user to optionally restart the program via a warm reset of the unit. Normally, if RESET is pressed, the current LCD display is saved, and the following is displayed:

```
Restart Program?
Press YES to Restart
CANCEL to Return
```

Any other keys other than YES or CANCEL are ignored. If CANCEL is pressed, the program resumes where it left off. This also happens if neither of these keys is pressed in 10 seconds.

CALL 3 can be used to disable or enable this function. It is automatically enabled at power-up or after a warm reset. To use this call:

```
PUSH 1 - disables program restart function
or
PUSH 0 - enables program restart function
CALL 3
There are no return parameters.
```

NOTE:
When this function is disabled, the RESET key returns an ASCII "r", shifted or not.

5.5 LCD Display Support Calls

As discussed in a previous section, PRINT# (and relatives) are used to display characters on the LCD display. The following CALLs are also implemented to support use of the LCD display.

CALL 10 - Output to LCD

This function outputs the value PUSHed onto the argument stack to the LCD Display at the current sensor location. The cursor position is advanced by one.

CALL 11 - Position Cursor

This function positions the cursor. PUSH an integer which is interpreted as a 16 bit integer. The high byte is used for the row number, and the low byte for column number, both one-relative, i.e.

```
PUSH 105H:CALL 11
```

This example puts the cursor at line 1, column 5. The values used must be between 1 and 4 for the row number, and 1 to 20 for the column number, for proper operation. The LCD hardware advances the cursor position by TWO rows if a character is output with the cursor at the last position in a line. In general, the cursor should be positioned before a character string is written, unless it is certain that the previous output did not cause the automatic wrap described above.

CALL 12 - Cursor ON

This function turns on the cursor. No arguments are PUSHed or POPed.

CALL 13 - Cursor OFF

This function turns off the cursor. No arguments are PUSHed or POPed.

CALL 14 - Read Cursor State

This call returns the position and on/off state of the cursor. Normally, it is meant to be used in conjunction with CALL 15 to save/restore the cursor state within programs that perform asynchronous timeout or error processing, or in other programs where it is desirable to quickly save and restore the cursor state without managing it fully.

CALL 14, then

POP CS, where CS is the BASIC floating point equivalent of a 16-bit unsigned integer that contains the cursor state, as follows:

The high order bit of the high order byte, if set (1), indicates that the cursor is on, and it is off if the bit is not set (0).

The remainder of the high order byte is the row number (1-4) OR'ed with the cursor bit.

The low order byte is the column position in the display row (1-20).

For example, 8103 hex indicates that the cursor is on, and is in row 1 column 3.

NOTE:
In Version 2.0, the value returned by CALL 14 can be used, if desired, with CALL 11 to position the cursor, but it will not turn the cursor on. Use CALL 15 with an explicit value if it is desired to combine these functions.

CALL 15 - Restore Cursor State

This call restores the cursor statefrom a value returned by CALL 14.

```
PUSH CS: REM cursor state

CALL 15
```

This call does not return any values.

CALL 16 - Backspace Cursor

This call performs a destructive backspace at the current cursor position. It does not return any values or require any parameters.

CALL 17 - LED Control

This call allows the BASIC program to turn the LEDs L1, L2, and L3 on or off as needed. When the unit powers up, L3 is turned on to indicate that the power-up diagnostics succeeded. It can be turned off at the beginning of a program if desired. This call works as follows:

```
PUSH LV: REM LED STATE
CALL 17
```

LV controls the state of all three LED's in a bitwise manner. It is treated as a 16-bit unsigned integer and only the low-order 3 bits are significant.

Bit 0 (low order) - turns on L3 if set, off if reset
Bit 1 - turns on L2 if set, off if reset
Bit 2 (high order) - turns on L1 if set, off if reset

Thus, if LV is 5, L1 and L3 are on, and L2 is off.

There are no return values.

CALL 19 - Clear Display

This function clears the display and homes the cursor. No arguments are PUSHed or POPed.

5.6 Program Port Support

The CALLs described in this section facilitate use of the program port for input in applications where the SIMATIC PLC Register Read/Write CALLs are not used. The program port operates with 8 data bits, no parity, and one stop bit at the baud rate selected by CALL 78.

CALL 78 - Program Port Baud Rate

CALL 78 permits changing the program port baud rate from its default value (1200 baud) to one of the following: 300, 600, 1200, 2400, 4800, 9600, or 19200. PUSH the desired baud rate and issue CALL 78. The program port remains at the new baud rate until it is changed again or CALL 73 is invoked followed by cycling power. Once battery-backup has been disabled, the baud rate is returned to its default value of 1200 baud.

CALL 114 - Poll for Program Port Input

This call returns -1 if no characters have been received from the program port, or decimal value of the oldest character received if character(s) are available.

Example:

```
>10CALL 114 : POP A : IF A = -1 THEN 10
>20IF A = 13 THEN END ELSE PRINT@"Character value",A
>GOTO 10
```

This program displays the values of characters received at the program port until the ENTER key is pressed.

CALL 115 - Disable Program Port Software Handshaking

Normally, the program port handles software handshaking automatically for both input and output, using the CTRL-S/CTRL-Q or XON/XOFF protocol. This CALL can be used in applications where this is not desirable.

This CALL also allows the program port to send and receive 8-bit characters. Normally, the high bit is masked off on input. Thus all 8-bit byte values can be sent and received when software handshaking is disabled.

No parameters need to be PUSHed and no values are returned. The handshaking status is NOT preserved across resets or power cycling.

CALL 116 - Clear Program Port Input Buffer

Input from the program port is buffered by the program port interrupt handler as received. This function allows you to clear any unwanted input from this buffer. No parameters are PUSHed or POPed.

5.7 SIMATIC TI 500 & 505 PLC Read and Write Register Support

The Model 5250-BAS contains support for reading and writing SIMATIC TI 500 & 505 series registers using a serial link (either RS-232 or RS-422) established on the 5250 program port. The following CALLs are supported:

CALL 20 - Initiate PLC Register Read

PUSH numeric value for register type. The following register types are supported in this release:

Value	Register
1	timer (reads current value)
2	timer (reads preset value)
3	counter (reads current value)
4	counter (reads preset value)
5	X
6	Y
7	Control Relay
8	WX
9	WY
10	V memory
11	V memory (result interpreted as 4 digit BCD)

PUSH register number (one-relative) CALL 20, then POP on of the following return values:

0 -command initiated successfully
-1 -command already pending (no other action taken, see CALL 22)
1 -bad register type argument

NOTE:
No range checking is done on the register number. If it is not in the range. 0-65535, however, the conversions routine will trap to the BASIC interpreter.

When the command is initiated successfully, the 5250-BAS program port operates with 7 data bits, one stop bit, and odd parity. The operational baud rate is that most recently set with CALL 78. Any input characters received before the command is sent are discarded. The program port handler operates as a simple state machine that is geared to receive a proper SIMATIC TI 500/505 task code response.

CALL 21 - Poll for Register Read Response

This function allows you to poll for a read command response. If the call completes successfully, the data is returned on the argument stack.

CALL 21

POP status, where:

0 -response not yet received
1 -successful response received
-1 -response error

If the status is other than 0,

POP value or error code.

If the status is response error, and the error code is ≥256, a communications error has occurred. Otherwise, the error is the result of receipt of an error response from the PLC, and the value is defined as in Appendix C. If a bad register number (i.e. 0) is passed, for example, error code 2 is returned.

The communications errors are:

256 - incomplete response received
257 - incorrect trailer or task code received
258 - invalid checksum received
259 - loop is disabled (See CALL 25)

If the status is successful, and the register type is other than X, Y, or CR, the value is the 16-bit value returned by the PLC. BASIC treats this as an unsigned integer (0-65535), so a conversion would be necessary for other data types.

NOTE:
If the register type is (V memory BCD) the value is assumed to be in 4 digit BCD format, rather than 4 HEX digits. It is converted to the proper BASIC floating point value as long as no HEX digits are present.

If the register type is X or Y, the returned value is the bitwise OR of the following:

10H	set if module present
8H	set if forced state
4H	indicates is output (set if module not present)
2H	set if word module
1H	set if ON

If the register type is CR, the returned value is the bitwise OR of the following:

10H	always set
8H	set if forced state
4H	always clear
2H	always clear
1H	set if ON

NOTE:
The BASIC boolean operator .AND. can be used to advantage to interpret these values.

CALL 22 - Reset PLC Communication Mode

This CALL requires no arguments and returns no values. It sets command-response processing state to its initial value, resets the serial port to no parity, and clears the serial port input buffer. It can be used in combination with the ONTIME and TIME operators to implement timeouts and/or retries for PLC communications.

CALL 22 must be used for "cleanup" whenever CALL 20 or CALL 23 returns a command in progress status.

CALL 23 - Initiate PLC Register Write

This function supports NON-FORCING writes only. All of the register types supported for READ are also supported for WRITE in this function.

PUSH numeric value for register type
PUSH register number (one-relative)
PUSH value to be written

The value is interpreted as an unsigned 16-bit number, unless the register type is for X, Y, CR, or V memory BCD.

If X, Y, or CR, the object is "turned on" if the low order byte is non-zero, and "turned off" if the low order byte is zero.

If V memory BCD, the value is converted from BASIC floating-point to a four-digit BCD value in the task code sent to the PLC.

CALL 23

POP one of the following return values:

0 -command initiated successfully
-1 -command already pending (no other action taken, see CALL 24)
1 -bad register type argument

NOTE:
No range checking is done on the register number or value. If either is not in the range 0-65535, will trap to BASIC interpreter.

When the command is initiated successfully, the 5250-BAS program port operates with 7 data bits, one stop bit, and odd parity. The operational baud rate is that most recently set with CALL 78. Any input characters received before the command is sent are discarded. The program port handler operates as a simple state machine that is geared to receive a proper SIMATIC TI 500/505 task code response.

CALL 24 - Poll for Register Write Response

This function operates in much the same way as CALL 21, except that register writes only "return" a status indication of whether the write completed successfully.

CALL 24

POP status, where:

0 -response not yet received
1 -successful response received
-1 -response error

If the status is -1 (error),

POP error code

If the error code is ≥256, a communications error has occurred. Otherwise, the error is an error response from the PLC, and the value is defined in Appendix C. For example, if a bad register number (i.e. 0) is passed, error code 2 is returned.

The communications errors are:

256 - incomplete response received
257 - incorrect trailer or task code received
258 - invalid checksum received

If the status is successful, the write completed successfully. No verification check is done, however.

CALL 25 and CALL 26 - Loop Variables for SIMATIC TI 545/565

CALL 25 and CALL 26 are used to read/write loop variables on the SIMATIC TI 545/565. These are similiar to the read/write register CALLs, respectively, but the parameters are different. CALL 21, Poll for Register Read Response, and CALL 24, Poll for Write Register Response, are used in conjunction with CALL 25 and CALL 26 to determine completion of the request, and return of data for CALL 25.

Each call takes a loop number parameter, which must be between 1 and 64, and loop data type. The supported data types and their numeric values are as follows:

Type	**Value**
Loop Gain	1
Loop Reset Time (minutes)	2
Loop Rate Time (minutes)	3
Loop High Alarm Limit	4
Loop Low Alarm Limit	5
Loop Process Variable	6
Loop Process Variable High Limit	7
Loop Process Variable Low Limit	8
Loop Orange Deviation Alarm Limit	9
Loop Yellow Deviation Alarm Limit	10
Loop Sample Rate (seconds)	11
Loop Setpoint	12

Loop Output (percent)	13
Loop V-Flags	14
Loop RAMP/SOAK Status Flags	16
Loop Error	17
Loop Bias	18
Loop High-High Alarm Limit	19
Loop Low-Low Alarm Limit	20
Loop Rate-of-Change Alarm Limit (eu./minute)	21
Loop Setpoint High Limit	22
Loop Setpoint Low Limit	23
Loop Alarm Deadband	24

All data values are floating point except for the V-Flags and the R/S Status Flags. The interface code does necessary data conversions between BASIC and SIMATIC TI 545/565 floating point formats. Since most of these values are floating-point numbers, extra care should be taken to be sure there is room to display the number of digits required for a value on the LCD display. The BASIC PRINT USING statement can be of assistance in specifying the data display format, and this works on the LCD display as well as on the TCTALK screen.

The loop operational mode can be set by writing data to the V-Flag as follows:

8000H	(32768)	Go to MANUAL Mode
4000H	(16384)	Go to AUTO Mode
2000H	(8192)	Go to CLOSED CASCADE Mode

The following status bit values are indicated by the Loop when the Loop V-Flags are read:

1000H	(4096)	Loop is in MANUAL Mode
800H	(2048)	Loop is in CLOSED CASCADE Mode
400H	(1024)	Error is negative
200H	(512)	PV in High-High Alarm
100H	(256)	PV in High Alarm
80H	(128)	PV in Low Alarm
40H	(64)	PV in Low-Low Alarm
20H	(32)	PV in Yellow Dev. Alarm
10H	(16)	PV in Orange Dev. Alarm
8H	(8)	PV in Rate-of-Change Alarm
4H	(4)	PV in Broken-Transmitter Alarm
2H	(2)	Loop is Overrunning

CALL 25 - Read Loop Variable

To read a loop variable,

PUSH loop number, PUSH variable type number, CALL 25

CALL 25 returns the same response status values as CALL 20, i.e., a response of 0 indicates that the call was initiated successfully.

Then, use CALL 21 (the same CALL used for "regular" register reads) to poll for the response. Error code 259 means that the loop is disabled.

CALL 26 - Write Loop Variable

To write a loop variable,

PUSH loop number, PUSH variable type number, PUSH value to write, CALL 26

CALL 26 returns the same response status values as CALL 23, i.e., a response of 0 indicates that the call was successfully initiated.

Then, use CALL 24 (the same CALL used for "regular" register writes) to poll for the response. The possible response values are unchanged from previous releases.

5.8 SIMATIC TI 405 PLC Read and Write Register Support

The following CALLs for reading and writing SIMATIC TI 405 series registers are supported:

CALL 30 - Initiate SIMATIC TI 405 Register Read

PUSH numeric value for register type. The following are register types supported in this release:

Value	Register
1	timer (reads current value)
3	counter (reads current value)
5	X
6	Y
7	Control Relay
10	V memory
11	V memory (result interpreted as 4 digit BCD)
16	Stage
17	Special Relay
18	Timer Relay
19	Counter Realy

PUSH register number (zero-relative)

CALL 30, then

POP one of the following return values:

0 -command initiated successfully
-1 -command already pending (no other action taken, see CALL 22)
1 -bad register type or register number argument

The register number must be within the legal range for the register type specified in Figure 19. All values are in hexidecimal:

When the command is initiated successfully, the 5250-BAS program port operates with 8 data bits, one stop bit, and odd parity. The operational baud rate is that most recently set with CALL 78. Any input characters received before the command is sent are discarded. The program port handler operates as a state machine that is geared to receive a proper SIMATIC TI 500/505 task code response.

CALL 31 - Poll for SIMATIC TI 405 Register Read Response

This function allows you to poll for a read command response. If the call completes successfully, the data is returned on the argument stack.

CALL 31

POP status, where:

0 -response not yet received
1 -successful rsponse received

If the status is 1,

POP value

If the status is successful, and the register type is Timer, Counter, or V memory, the value is the 16-bit value returned by the PLC. BASIC treats this as an unsigned integer (0-65535), so a conversion would be necessary for other data types.

> ***NOTE:***
> *If the register type is V memory BCD the value read from the PLC is assumed to be in 4 digit BCD format, rather than a 16-bit binary value. It is converted to the proper BASIC floating point value.*

If the register type is any other type, the returned value is 1 if the register is in the "ON" state and 0 if it is in the "OFF" state.

CALL 32 - Initiate SIMATIC TI 405 Register Write

All of the register types supported for READ are also supported for WRITE in this function.

PUSH numeric value for register type
PUSH register number (zero-relative)
PUSH value to be written

The value is interpreted as an unsigned 16-bit number if the register type is Timer, Counter, or V memory.

If the register type is V memory BCD, the value is converted from BASIC floating-point to a four-digit BCD value in the task code sent to the PLC.

For all other types, the object is "turned on" if the low order byte is non-zero, and "turned off" if the low order byte is zero.

CALL 32

POP one of the following return values:

0 -command initiated successfully
-1 -command already pending (no other action taken, see CALL 22)
1 -bad register type argument

NOTE:
No range checking is done on the register number or value. If either is not in the range 0-65535, the program will trap to BASIC interpreter.

When the command is initiated successfully, the 5250-BAS program port operates with 8 data bits, one stop bit, and odd parity. The operational baud rate is that most recently set with CALL 78. Any input characters received before the command is sent are discarded. The program port interrupt handler operates as a state machine that is geared to receive a proper SIMATIC TI 500/505 task code response.

CALL 33 - Poll for SIMATIC TI 405 Register Write Response

This function operates in much the same way as CALL 31, except that register writes only "return" a status indication of whether the write completed successfully.

CALL 33

POP status, where:

0 -response not yet received
1 -successful response received
-1 -response error

If the status is -1 (error),

POP error code

1 The PLC is password-locked
2 Attempt to write to read only program space
3 Keyswitch not set to TERM position

5.9 Time/Date Clock Support Calls

The battery backed time/date clock provides year, month, day of month, day of week, hours, minutes and seconds. The clock operates in 24 hour format. The support routines allow the setting of the clock and retrieval of clock values in numeric form.

The clock support routines use the BASIC argument stack to pass data between the BASIC program and the routines. Data is passed in both directions and consist of the actual clock data.

The time/date clock is separate from the real time clock also provided on the module. The real time clock is accessed by CLOCK1, CLOCK0, ONTIME and other statements, and has a resolution of 5 milliseconds. It should be used for all short time interval measurements because the greater resolution results in more accurate timing. The two clocks are not synchronized and comparison of times is not recommended.

CALL 40 - Setting the Clock Time (Hour, Minute, Second)

Use this routine to set the following time values:

H=hours (0 to 23)
M=minutes (0 to 59)
S=seconds (0 to 59)

These three values are PUSHed, and none are POPed.

Example:

Program the wall clock for 1:35 pm (13:35 on a 24 hour clock).

```
H=13:M=35:S=00:REM HOURS=13; MINUTES=35; SECONDS=00
PUSH H,M,S:REM PUSH HOURS, MINUTES, SECONDS
CALL 40:REM CALL THE ROUTINE TO SET THE CLOCK TIME
```

CALL 41 - Setting the Clock Date (Day, Month, Year)

Use this routine to set the following date values:

D=day
M=month
Y=year

These three values are PUSHed, and none are POPed.

Example:

Program the date to the 16th day of June, 1993

```
D=16:M=06:Y=93:REM DAY OF MONTH=16, MONTH=6, YEAR=93
PUSH D,M,Y:REM PUSH DAY OF MONTH, MONTH, YEAR
CALL 41:REM CALL THE ROUTINE TO SET THE DATE
```

CALL 42 - Set Day of Week

CALL 42 sets the day of the week. Sunday is day 1. Saturday is day 7.

Example:

```
PUSH 3: CALL 42 : REM DAY IS TUESDAY
```

CALL 43 - Date/Time Retrieve String

CALL 43 returns the current date and time as a string. PUSH the number of the string to receive the date/time (dd/mmm/yy HH:MM:SS). You must allocate a minimum of 18 characters for the string. This requires you to set the maximum length for all strings to at least 18 characters.

Example:

```
STRING 100,20
PUSH 1: CALL 43: REM put date/time in string 1
PRINT $(1)
```

CALL 44 - Date Retrieve Numeric (Day, Month, Year)

CALL 44 returns the current date on the argument stack as three numbers. There is no input argument to this routine and three variables are returned. The date is POPed in day, month and year order.

Example:

```
CALL 44: REM INVOKE THE UTILITY ROUTINE
POP D, M1,Y : REM GET THE DATA FROM THE STACK
PRINT "CURRENT DATE IS",Y,M1,D
```

CALL 45 - Time Retrieve String

CALL 45 returns the current time in a string (HH:MM:SS). PUSH the number of the string to receive the time. You must allocate a minimum of 8 characters for the string.

Example:

```
STRING 100,20
PUSH 1: CALL 45: REM put time in string 1
PRINT $(1)
```

CALL 46 - Time Retrieve Numeric

The time of day is available in numeric form by executing CALL 46 and POPing the three variables off of the argument stack on return. There are no input arguments. The time is POPed in hour, minute and second order.

Example:

```
CALL 46
POP H,M,S
PRINT "CURRENT TIME IS", H,M,S
```

CALL 47 - Retrieve Day of Week String

CALL 47 returns the current day of week as a three character string. PUSH the number of the string to receive the day of week. You must allocate a minimum of 3 characters/string. Strings returned are SUN, MON, TUE, WED, THU, FRI, SAT.

Example:

```
PUSH 0: CALL 47
PRINT "TODAY IS",$(0)
```

CALL 48 - Retrieve Day of Week Numeric

CALL 48 returns the current day of week on the argument stack as a number (i.e. Sunday = 1, Saturday = 7). This can be POPed into a variable.

Example:

```
CALL 48: REM INVOKE UTILITY TO GET D.O.W.
POP D
```

CALL 52 - Date Retrieve String

CALL 52 returns the current date in a string (dd/mmm/yy). PUSH the number of the string to receive the date. You must allocate a minimum of 9 characters for the string.

Example:

```
STRING 100,20
PUSH 1: CALL 52: REM put date in string 1
PRINT $(1)
```

5.10 String Support Calls

Several string manipulation routines have been included in order to facilitate programming with strings. Strings in 5250 BASIC are declared by the string statement. The STRING statement must be executed before your program can access or call any strings or string routines. The STRING statement has two arguments or numbers that follow it:

the total amount of space to allocate to string storage and,
the maximum size in characters of each string.

Since strings are terminated using a carraige return character, each string is given an extra byte of storage for its carraige return if it is the maximum length. You can determine the number of strings allowed by taking the first number and dividing it by one plus the second number. Note that you must use the strings consecutively starting with string 0 through the allowed number of strings. All strings are allocated the maximum number of characters regardless of the actual number used.

All of these routines use or modify strings as part of their operation. The mechanism for passing a string to the support routine is to PUSH its number or subscript onto the stack. The support routine can then get the string number from the BASIC argument stack, and locate it in string memory.

Many of these routines also require the length of a string as an input. This number must normally be inclusively between zero and the second number used in the last STRING statement which specifies the maximum size of a string. However, in all cases, if a string length argument of minus one (-1) is given, it is interpreted as the maximum allowable string length.

It is important to note that since the carraige return character is the string terminator, it cannot be used within a string as one of its characters. If the high bit is set in a carraige return character (use 141 instead of 13 as the decimal value of the carraige return character) the BASIC Module does not recognize it as the end of string character and passes it to the output device. Most devices use a seven bit ASCII code, ignore the top bit and treat the 141 as a normal carraige return.

NOTE:
All undefined characters of a string (i.e. characters following CR) are nulls. If data is input to a string using the ASC($(X),I) command, you must be sure the string is (CR) terminated properly. You can initialize the string or add a (CR) to terminate the string.

Example:

```
>10 STRING 100,10
>20 FOR I=1 TO 5
>30 ASC ($(0),I)=65
>40 NEXT I
>45 ASC($(0),6)=0DH
>50 PRINT $(0)
```

CALL 60 - String Repeat

This routine allows you to repeat a character and place it in a string. You can use the String Repeat when designing output formats. First PUSH the number of times to repeat the character, then PUSH the number of the string containing the character to be repeated. No arguments are POPed. You cannot repeat more characters than the string's maximum length.

```
>10REM STRING REPEAT EXAMPLE PROGRAM
>20STRING 1000,50
>30$(1)="*"
>40PUSH 40 :REM THE NUMBER OF TIMES TO REPEAT CHARACTER
>50PUSH 1 :REM WHICH STRING CONTAINS CHARACTER
>60CALL 60
>70PRINT $(1)
```

CALL 61 - String Append (Concatenation)

This routine allows you to append one string to the end of another string. The CALL expects two string arguments. The first is the string number of the string to be appended and the second is the string number of the base string. If the resulting string is longer than the maximum string length, the appended characters are lost. There are no output arguments.

```
>10STRING 200,20
>20$(1)="How are"
>30$(2)="you?"
>40PRINT "BEFORE:"
>50PRINT"$1=",$(1),"$2=",$(2)
>60PUSH 2 :REM STRING NUMBER OF STRING TO BE APPENDED
>70PUSH 1 :REM BASE STRING NUMBER
>80CALL 61 :REM INVOKE STRING CONCATENATION ROUTINE
>90PRINT "AFTER:",
>100PRINT "$1=",$(1),"$2=",$(2)
```

CALL 62 - Number to String Conversion

This routine converts a number or numeric variable into a string. You must allocate a minimum of 14 characters for the string. If the exponent of the value to be converted is anticipated to be 100 or greater, you must allocate 15 characters. Error checking traps string allocation of less than 14 characters only.

PUSH the value to be converted
PUSH the number of the string to receive the value.

Example:

```
>10STRING 100,14
>20INPUT "ENTER A NUMBER TO CONVERT TO A STRING",N
>30PUSH N
>40PUSH 1:REM CONVERT NUMBER TO STRING 1
>50CALL 62:REM DO THE CONVERSION
>60PRINT $(1)
```

CALL 63 - String to Number Conversion

This routine converts the first decimal number found in the specified string to a number on the argument stack. Valid numbers and associated characters are: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, ., E, +, -. The comma (,) is not a valid number character and terminates the conversion. Two bytes are POPed after the CALL:

Validity of the value
Actual value

If the string does not contain a legal value a 0 (zero) is returned. A valid value is between 1 and 255. PUSH the number of the string to convert. Two POPs are required. First POP the validity value, then POP the value. If a string contains a number followed by an E followed by a letter or non-numeric character, it is assumed that no number was found since the letter is not a valid exponent (UAB701EA returns a zero in the first argument popped indicating that no valid number was in the string).

Example:

```
>10STRING 100,14
>20INPUT "ENTER A STRING TO CONVERT",$(1)
>30PUSH 1:REM CONVERT STRING 1
>40CALL 63:REM DO THE CONVERSION
>50POP V,N
>60IF V0 THEN PRINT $(1),"",N : END
>70PRINT "INVALID OR NO VALUE FOUND"
```

CALL 64 - Find a String in a String

This routine finds a string within a string. It locates the first occurrence (position) of this string. This call expects two input arguments. The first is the string to be found, the second is the string to be searched for a match. One return argument is required. If the number is not zero then a match was located at the position indicated by the value of the return argument. This routine is similar to the standard BASIC INSTR$(findstr$,str$).

Example:

```
>10REM SAMPLE FIND STRING IN STRING ROUTINE
>20STRING 1000,20
>30$(1)="456"
>40$(2)="12345678"
>50PUSH 1 :REM STRING NUMBER OF STRING TO BE FOUND
>60PUSH 2 :REM BASE STRING NUMBER
>70CALL 64 :REM GET LOCATION OF FIRST CHARACTER
>80POP L
>90IF L=0 THEN PRINT "NOT FOUND"
>100IF L>0 THEN PRINT "FOUND AT LOCATION",L
```

CALL 65 - Replace a String in a String

This routine replaces a string within a string. Three arguments are expected. The first argument is the string number of the string which replaces the string identified by the second argument string number. The third argument is the base string's string number. There are no return arguments.

```
>10REM SAMPLE OF REPLACE STRING IN A STRING
>20STRING 1000,20
>30$(0)="RED-LINES"
>40$(1)="RED"
>50$(2)="BLUE"
>60PRINT "BEFORE: $0=",$(0)
>70PUSH 2 :REM STRING NUMBER OF THE STRING TO REPLACE WITH
>80PUSH 1 :REM STRING NUMBER OF THE STRING TO BE REPLACED
>90PUSH 0 :REM BASE STRING NUMBER
>100CALL 65 :REM INVOKE REPLACE STRING IN STRING ROUTINE
>110PRINT "AFTER:$0=",$(0)
```

CALL 66 - Insert String in a String

This routine inserts a string within another string. The call expects three arguments. The first argument is the position at which to begin the insert. The second argument is the string number of the characters inserted into the base string. The third argument is the string number of the base string. This routine has no return arguments.

```
>10REM SAMPLE ROUTINE TO INSERT A STRING IN A STRING
>20STRING 500,15
>30$(0)="1234590"
>40$(1)="67890"
>50PRINT "BEFORE: 0$=",$(0)
>60PUSH 6 :REM POSITION TO START THE INSERT
>70PUSH 1 :REM STRING NUMBER OF THE STRING TO BE INSERTED
>80PUSH 0 :REM BASE STRING NUMBER
>90CALL 66 :REM INVOKE INSERT A STRING IN A STRING
>91REM REM ROUTINE
>100PRINT "AFTER: 0$=",$(0)
```

CALL 67 - Delete String from a String

This routine deletes a string from within another string. The call expects two arguments. The first string is the base string number. The second is the string number of the string to be deleted from the base string. This routine has no return arguments.

> ***NOTE:***
> *This routine deletes only the first occurrence of the string.*

```
>10REM ROUTINE TO DELETE A STRING IN A STRING
>20STRING 200,14
>30$(1)="123456789012"
>40$(2)="12"
>50PRINT "BEFORE: $1=",$(1)
>60PUSH 1 :REM BASE STRING NUMBER
>70PUSH 2 :REM STRING NUMBER OF THE STRING TO BE DELETED
>80CALL 67 :REM INVOKE STRING DELETE ROUTINE
>90PRINT "AFTER: $1=",$(1)
```

CALL 68 - Determine Length of a String

This routine determines the length of a string. One input argument is expected. This is the string number on which the routine acts. One output argument is required. It is the actual number on non-carraige return (CR) characters in this string. This is similar to the BASIC command LEN(str$).

```
>10REM SAMPLE OF STRING LENGTH
>20STRING 100,10
>30$(1)="1234567"
>40PUSH 1 :REM BASE STRING
>50CALL 68 :REM INVOKE STRING LENGTH ROUTINE
>60POP L :REM GET LENGTH OF BASE STRING
>70PRINT "THE LENGTH OF",$(1),"IS",L
```

5.11 SIMATIC TI 305 Series PLC Read and Write Register Support

This section covers implementation on the SIMATIC TI 315, 325, 330, and 335 PLCs, as well as the stage versions (315S, 325S, and 330S).

Several 305 Series register types are defined for reading/writing by a BASIC program on the 5250-BAS Operator Panel. The presence of a register type depends on the PLC type and whether the PLC is a stage version.

The 305 Series PLC CPU type is determined when a BASIC program issues CALL 81 (see below). If this CALL is not done by the program, the assumed CPU type is 325/330/335.

For each register type, a range of addresses is "legal". Again, this range depends on the PLC type. Addresses specified from BASIC correspond to those specified when programming ladder logic. For example, on the 325/330/335, IO addresses start at 0, Internal Relays start st 112 (160 octal), and Timer/Counter Relays start at 384 (600 octal).

The following register types are defined:

Name	Numeric Value	Description
TIMER	1	Timer accumulator - binary 16-bit value - contains the current timer value. The register address must be between 384 (600 octal) and 447 (677 octal) for all 305 CPUs except 315 and 315S. For these, the limit is 404 (624 octal).
COUNTER	3	Counter accumulator - binary 16-bit value - fully equivalent to TIMER.
INPUT or OUTPUT	5	References an IO point, internal relay, or timer/counter relay. This register type is READ-ONLY. It allows the ON/OFF state of a single bit to be read, based upon the address specified. For example, CALL 83 returns a data value of 1 if input 5 is turned on and the address parameter is 5. No address checking is performed on this register type. If the PLC rejects the command, a communications error is returned to CALL 83.

WORD DATA REGISTER	10	This register type allows reading or writing of a pair of data registers as a 16-bit binary value. The address parameter, which must be between 256 (400 octal) and 383 (577 octal) is always adjusted to an even value. The low order byte of the return value contains the byte at the even address, and the high order byte contains the byte at the odd address. This register type is not legal if the CPU type is 315 or 315S. A bad argument error is returned for CALL 82 or 84.
WORD DATA REG BCD	11	This is the same as above, except CALL 83 returns a value converted from BCD to internal binary form when the register is read.
STAGE	16	This works just like type 5 above, except that stage numbers are converted to the correct internal I/O addresses. Address translation is not done if the CPU type is 315S.
TIMER RELAY	18	This register type is essentially the same as type 5, but includes address range check specifically for timer relays. The address parameter must be in the same range as for the TIMER register type indicated above. This register type is not legal for the 315S or 330S CPU types.
COUNTER RELAY	19	Fully equivalent to TIMER RELAY (18)
BYTE INPUT or OUTPUT	20	Generic read/write of 8-bit binary values at all I/O addresses. This includes byte-addressed I/O, internal relays, shifts registers, and timer/counter relays. In the stage CPUs, it also includes stages. No address checking or special address translation is performed. This function treats a group of 8 I/O points as an 8-bit value. Any passed address is rounded down to the nearest address divisable by 8. Thus, if input 5 is specified, CALL 83 returns the state of inputs 0-7 as an 8-bit value.

To write a single I/O point, you must first read in the value, then set/reset the relevant bit position, then write the value back out.

In the 330S stages can be accessed by adding 128 (200 octal) to the stage number. In addition, Internal Relays can only be accessed by adding 128 (200 octal) to the relay number.

CALL 80 - Set the 305 Series PLC Network Address

PUSH numeric value of the network address. This must be the same value set on the dip switch on the DCU. If this call is not issued by a BASIC program, address 1 is assumed.

CALL 80 - no return value

CALL 81 - Return and Store the 305 Series CPU Type

If this call is not performed by the BASIC program, the CPU type is interpreted as 325/330/335.

CALL 81, then

POP status indicator or CPU type

1 or greater - completed successfully - the CPU types are as follows:

1 - 330S (Stage)
2 - 323/330/335
3 - 315
4 - 315S (Stage)

-1 PLC command - response sequence already in progress;
0 PLC communications error or timeout

NOTE:
This function will automatically time out and return 0 if the network address is set incorrectly, as the PLC will not respond to an incorrect address.

CALL 82 - Initiate a 305 Register Read

PUSH register type number
PUSH register number (address)

CALL 82, then

POP status indicator:

0 -initiated successfully
-1 -PLC command - response sequence already in progress
1 -bad argument

CALL 83 - Poll for 305 Series Register Read Response

CALL 83, then POP result:

0 -not complete
1 -complete
-1 -communications error

If result is complete, POP register value
If result is not complete or there is a communications error, there is no second result
If based on the register type and the CPU type, the address is not valid, -1 is returned.

CALL 84 - Initiate a 305 Series Register Write

PUSH register type number
PUSH register number (address)
PUSH value to write

CALL 84, then POP status indicator:

0 -initiated successfully
-1 -PLC command - response sequence already in progress
1 -bad argument

CALL 85 - Poll for 305 Series Register Write Response

CALL 85, then POP result:

0 -not complete
1 -complete
-1 -communications error

If based on the register type and the CPU type, the address is not valid, -1 is returned.

5.12 Miscellaneous Support Routines

CALL 73 - Battery-backed RAM Disable

CALL 73 disables the battery-backed RAM, prints "Battery Backup Disabled" when executed and allows a purging reset. The next power loss destroys the contents of RAM. When power is reapplied, RAM is cleared and battery back-up is reenabled automatically.

> ***NOTE:***
> *CALL 73 erases any program currently in RAM when power is cycled.*

CALL 74 - Battery-backed RAM Enable

CALL 74 enables the battery-backed RAM and prints "Battery Backup Enabled" when executed. It is enabled on power-up and remains enabled until you execute a CALL 73 or until the battery fails.

CALL 77 - Protected Variable Storage

CALL 77 reserves space at the top of RAM memory for protected variable storage. Values are saved unless battery backup is disabled. BASIC variables can then be stored with the ST@ command and retrieved with the LD@ command. Each variable stored requires 6 bytes of storage space.

The system variable MTOP contains the current highest address available to BASIC. You must subtract 6 from MTOP for each variable to be stored in protected RAM memory. This value is PUSHed onto the stack as the new MTOP address. All appropriate variable pointers are reconfigured.

> ***NOTE:***
> *Do not let the ST@ address write over the MTOP address. This could alter the value of a variable or string.*

Example: For saving 2 variables.

```
>5PUSH MTOP-12 : CALL 77
>10K=678*PI
>15L=520 :PUSH K
>30ST@14335 :REM STORE K IN PROTECTED AREA
>40PUSH L
>50ST@14329
>55REM TO RETRIEVE PROTECTED VARIABLES
>60LD@14335 :REM REMOVE K FROM PROTECTED AREA
>70POP K
>80LD@14329
>90POP L
```

CALL 99 - Reset Print Head Pointer

BASIC will insert a CR/LF character pair in any output line whose length exceeds 79 characters. If this behavior is not desired, issue CALL 99 periodically to reset the internal character position pointer. This pointer is also reset whenever BASIC outputs a CR character.

CALL 110 - Convert Binary to BCD

This call converts the passed parameter from a 16-bit unsigned integer to a 16-bit BCD representation. If the converted number is printed, the number is correctly displayed if it is printed in hex format. The primary function of this call is to allow a BCD number to be passed to a PLC via a write register call to a location that is to be interpreted as a BCD value.

```
PUSH VAL
CALL 110
POP BCDV :REM converted value
```

If the passed value is greater than 9999 (270FH), a floating point -1 is returned instead.

CALL 111 - Convert BCD to Binary

This call converts the passed BCD value to its 16-bit unsigned integer equivalent. It is used to allow normal BASIC arithmetic on a value read from a PLC register in BCD format.

```
PUSH BCDV
CALL 112
POP VAL
```

CALL 120 - Set Bit in Integer

This call provides an efficient means of setting a specific bit in a BASIC variable that is a 16-bit unsigned integer (range 0-65535). This can be done in 5250 BASIC, but the CALL executes much more quickly.

```
PUSH VAL: REM variable in which to set bit
PUSH BN: REM bit number (0-15) to set (0 is low order)
CALL 120
POP VAL: REM converted variable
```

If the bit number is out of range, floating point -1 is returned.

CALL 121 - Clear Bit in Integer

This call provides an efficient means of clearing a specific bit in a BASIC variable that is a 16-bit unsigned integer (range 0-65535). This can be done in 5250 BASIC, but the CALL executes much more quickly.

```
PUSH VAL: REM variable in which to clear bit
PUSH BN: REM bit number (0-15) to clear (0 is low order)
CALL 120
POP VAL: REM converted variable
```

If the bit number is out of range, floating point -1 is returned.

5.13 BASIC Limitations

The following limitations are noted:

1. The BASIC interpreter supports one-dimensional arrays and the maximum dimension of any array is 254.
2. All outputs following a USING statement will be in the format evoked by the last USING statement executed.
3. The maximum number of characters in a USING statement is 8.
4. After memory is allocated for string storage, the only way to de-allocate this memory is to execute the STRING 0,0 statement. STRING memory allocation should be performed in the first statement of a program.
5. Version 2.0 of the Operator Panel supports one word transfers to/from the PLC.
6. A statement may contain no more than 79 characters.

5.14 REN Renumber Command

The renumber command REN allows the renumbering of BASIC programs loaded in the 5250-BAS Operator Panel. The command has these forms:

REN - the program is renumbered from the start of the program to the end. The new line numbers begin at 10 and increment by 10.

REN, NUM - the program is renumbered from the start of the program to the end. The new line numbers begin at 10 and increment by NUM.

REN, NUM1, NUM2 - the program is renumbered from the start of the program to the end. The new line numbers begin with NUM1 and increment by NUM2.

REN, NUM1, NUM2, NUM3 - the program is renumbered from line NUM2 to the end. The new line numbers begin with NUM1 and increment by NUM3. The command will not execute unless NUM1 is greater than NUM2.

Error processing for the REN Command

The following errors can occur in processing the REN command. In some cases, the program will no longer execute correctly. Thus it is recommended that you save your program by uploading with TCTALK before issuing the REN command, especially with a large program.

RENUMBER ERROR - BAD SYNTAX

This error occurs only if three numeric arguments are given when the REN Command is issued. This form of the command allows you to renumber only a part of the program. In this case, the first argument, which is the new first line number that is renumbered, must be larger than the second argument, which is the old first line number to be renumbered. If not, the bad syntax error is returned, as the command cannot be processed without possibly erasing part of the program.

RENUMBER ERROR - LINE NUMBER OVERFLOW

This error occurs if the generated new line number becomes greater than 65535. In this case, the program is only partially renumbered and will not execute correctly.

RENUMBER ERROR - LINE NUMBER NOT FOUND

This error occurs if a line number cannot be found in the program. This can happen in two cases:

1. The old starting line number is not in the program when the REN Command is invoked with three arguments. In this case, the program is still intact.
2. An old line number cannot be found when attempting to renumber a line number referenced by a GOTO, GOSUB, or other statement that references line number. In this case, all other renumbering is done correctly. When this error occurs, the program contains a syntax error that has not been detected because the line containing the error has never been executed by the interpreter.

RENUMBER ERROR - PROGRAM TOO LARGE

This error occurs if the REN Command processor runs out of memory while processing. This only occurs when a program is very large or contains a very large number of lines. To avoid this error, renumber frequently while in the process of building a large program, and keep the renumber increment large enough for future expansion when the renumber command can no longer be used.

The program MEMORY.BAS which is installed in the 5250-BAS Operator Panel at power up has approximately 882 lines of code. It renumbers successfully in less than 1 minute elapsed time. The program contains no arrays.

5.15 Editing Examples

Examples of editing programs with TCTALK Edit utility are shown:

To enter a line, type the line number followed by BASIC statements, ending the line with a Carraige Return (CR);

```
11  Call 12 : x=2 (CR)
```

To delete a line, type the line number and CR;

```
11  (CR)
```

To replace a character in a line;

Press F9 key followed by line number,
```
    11  Call 12 : x=2
```
position cursor to character with space bar, type the new character y followed by CR;
```
    11  Call 12 : y=2
```
type CTRL-Q to SAVE the edited line.

To insert a character in a line;

Press F9 key followed by line number,
```
    11  Call 12 : y=2
```
position cursor, enter CTRL-A, followed by character, then CTRL-A to end insertion,
```
    11  Call 12 : y(CTRL-A)t(CTRL-A)=2
```
type CR to view the corrected line,
```
    11  Call 12 : yt=2
```
then type CTRL-Q to SAVE the edited line.

Use CTRL-C to terminate the line edit and restore the original line.

To delete a character, position the cursor and type the DEL key or CTRL-D.

```
11  Call 12 : y(CTRL-D)=2(CR)
11  Call 12 : y=2
```
then type CTRL-Q to SAVE the edited line.

Other DOS Editors may be used to create BASIC programs. If using an editor other than TCTALK, remember to use line number with each statement including REMARK statements used for documentation.

APPENDIX A: 5250-BAS OPERATOR PANEL APPLICATIONS

A.1 Introduction

Three applications are included on the Version 2.0 application diskette along with TCTALK.

These applications are fully functional. The coding in the applications illustrate the ability of the 5250-BAS Operator Panel to be tailored specifically to a wide variety of operator interface requirements.

A description of each application follows.

A.2 Memory Access for SIMATIC TI 305, 405, 500 and 505 PLCs. (MEMORY.BAS)

This application is loaded in the 5250-BAS Operator Panel and may be used immediately upon installation of the Operator Panel. This application will provide read access to memory locations of the attached PLC.

This initial screen display request identification of the PLC.

```
Select SIMATIC PLC
Press 3 for 305
Press 4 for 405
Press 5 for 500/505
```

A message acknowledging your selection is displayed for a few seconds.

The next screen (Screen 1) presents the following memory types to be read.

```
SELECT Command  S1
F1 Timer        F3 X Input
F2 Counter      F4 Y Output
Press YES for next
```

The Fn function keys are located on the rightmost column of the 5250-BAS Operator Panel keypad. Press the appropriate function key to read the PLC memory.

For example, F1 displays a TIMER screen with the LCD cursor position to receive the TIMER address. Enter the address using the numeric keys only followed by the ENTER key. The TIMER status (on or off) and current value are displayed.

If a non-numeric key is entered or if the address entered is out of the acceptable address range for the PLC, a message "illegal timer/counter" is displayed and the cursor is reset to the TIMER address position.

TIMER values are updated continuously until either the CANCEL key is pressed for the next command or the ENTER key is pressed for write access.

The YES key is on line 4 of the LCD indicates additional memory types may be accessed. Press this key to move to the next screen display. Memory types on Screen 2 or Screen 3 are selected using the SHIFT Fn keys as displayed on the screen.

Programming conventions:

The YES key presents the next screen and the NO key presents the previous screen. The Sn, n=1, 2 or 3, appearing on the right of the LCD line 1 indicates which screen is displayed.

Write access is not enabled in this version of the application in order to protect PLC memory. Sample write access code is included in the application so that write access may be enabled by the user. See lines 900-999.

The RESET key is enabled and will allow a restart of the application in the event of trouble or if the user wants to restart the application.

The LEDs marked L1, L2, and L3 are not enabled by this application. See the description for CALL 17 for enabling the LEDs.

DRUMS and WX/WY memory access fir the SIMATIC TI 500 and 505 PLCs are not available in this version of MEMORY.BAS.

The application code is divided into 4 sections:

Lines 10 through 1000 contain setup and subroutines (GOSUB),
Lines 3000 through 3999 support the SIMATIC TI 305 PLCs,
Lines 4000 through 4999 support the SIMATIC TI 405 PLCs, and
Lines 5000 through 5999 support the SIMATIC TI 500 and 505 PLCs.

Should you encounter difficulty in using this application, please call CTI at (800) 537-8398 for assistance. Your comments and suggestions for improving the usefulness of this application are appreciated.

A.3 Loop Access Module for SIMATIC TI 545 PLC (LAM.BAS)

A description of this application is on the application diskette in the text file LAM.DOC.

To load the LAM.BAS application into the 5250-BAS Operator Panel, follow these steps:

a. Start TCTALK at the DOS prompt
b. Type Ctrl-C to enter Interactive Mode
c. Type NEW to clear the existing program in the Operator Panel
d. Type Alt-F4 to change the Load File name. Enter LAM.BAS as the Load File name, then enter F4 to exit.
e. Type F4 to start the load process.

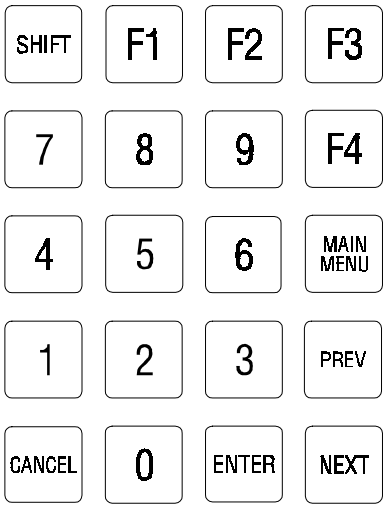
When the load operation completes, type F2 to initiate execution of the LAM.BAS program.

A.4 Operator Scripting (SCRIPT.BAS)

A description of this application is on the application diskette in the file SCRIPT.DOC.

Follow the procedures listed in A.3 to load this application into the Operator Panel.

APPENDIX B: KEYPAD SCAN CODES



B.1 KEYPAD SCAN CODES

Key	Decimal	Hex	ASCII Symbol
0	48	30	0
1	49	31	1
2	50	32	2
3	51	33	3
4	52	34	4
5	53	35	5
6	54	36	6
7	55	37	7
8	56	38	8
9	57	39	9
F1	68	44	D
F2	82	52	R
F3	10	0A	<LF>
F4	79	4F	O
ENTER	86	56	V
CANCEL	88	58	X
MAIN MENU	99	63	c
NEXT	13	0D	<CR>

KEYPAD SCAN CODES (USING SHIFT KEY)

Key	Decimal	Hex	ASCII Symbol
0	32	20	<SP>
2	34	22	"
3	35	23	#
4	65	41	A
5	66	42	B
6	67	43	C
7	68	44	D
8	69	45	E
9	70	46	F
F1	78, 68	4E, 44	N, D
F2	72	48	H
F3	97	61	a (see note)
F4	83	53	S
ENTER	115	73	s
CANCEL	89	59	Y
MAIN MENU	84	54	T
PREV	80	50	P
NEXT	45	2D	-

NOTE:

The F3 key will return a value of 97 only if breakthrough reset has been disabled using CALL 3 or the statement DBY(27H)=DBY(27H).OR.4. If breakthrough reset has not been disabled, pressing F3 will display a message which will allow the user to RESET the program or CONTINUE the program.

B.2 CALL Summary

Keypad Support Calls

CALL 1 - Get Keystroke
CALL 2 - Clear Keypad Input Buffer
CALL 3 - Disable/Enable Program Restart

LCD Display Support Calls

CALL 10 - Output to LCD
CALL 11 - Position Cursor
CALL 12 - Cursor ON
CALL 13 - Cursor OFF
CALL 14 - Read Cursor State
CALL 15 - Restore Cursor State
CALL 16 - Backspace Cursor
CALL 17 - LED Control
CALL 19 - Clear Display

Program Port Support Calls
CALL 78 - Program Port Baud Rate
CALL 114 - Poll for Program Port Input
CALL 115 - Disable Program Port Software Handshaking
CALL 116 - Clear Program Port Input Buffer

SIMATIC TI 500/505 PLC Register Support Calls
CALL 20 - Initiate 500/505 PLC Register Read
CALL 21 - Poll for 500/505 Register Read Response
CALL 22 - Reset 500/505 PLC Communications Mode
CALL 23 - Initiate 500/505 Register Write
CALL 24 - Poll for 500/505 Register Write Response
CALL 25 - Read S Memory Loop Variable
CALL 26 - Write S Memory Loop Variable

SIMATIC TI 405 PLC Register Support Calls
CALL 30 - Initiate 405 PLC Register Read
CALL 31 - Poll for 405 Register Read Response
CALL 32 - Initiate 405 PLC Register Write
CALL 33 - Poll for 405 Register Write Response

SIMATIC TI 305 PLC Register Support Calls
CALL 80 - Set 305 PLC Network Address
CALL 81 - Store 305 PLC Type
CALL 82 - Initiate 305 Register Read
CALL 83 - Poll for 305 Register Read Response
CALL 84 - Initiate 305 Register Write
CALL 85 - Poll for 305 Register Write Response

Time/Date Clock Support Calls
CALL 40 - Setting the Clock Time (Hour, Minute, Second)
CALL 41 - Setting the Clock Date (Day, Month, Year)
CALL 42 - Set Day of Week
CALL 43 - Date/Time Retrieve String
CALL 44 - Date Retrieve Numeric (Day, Month, Year)
CALL 45 - Time Retrieve String
CALL 46 - Time Retrieve Numeric
CALL 47 - Retrieve Day of Week String
CALL 48 - Retrieve Day of Week Numeric
CALL 52 - Date Retrieve String

String Support Calls

CALL 60 - String Repeat
CALL 61 - String Append (Concatenation)
CALL 62 - Number to String Conversion
CALL 63 - String to Number Conversion
CALL 64 - Find a String in a String
CALL 65 - Replace a String in a String
CALL 66 - Insert String in a String
CALL 67 - Delete String from a String
CALL 68 - Determine Length of a String

Miscellaneous Support Routines

CALL 73 - Battery-backed RAM Disable
CALL 74 - Battery-backed RAM Enable
CALL 77 - Protected Variable Storage
CALL 99 - Reset Print Head Pointer
CALL 110 - Convert Binary to BCD
CALL 111 - Convert BCD to Binary
CALL 120 - Set Bit in Integer
CALL 121 - Clear Bit in Integer

B.3 Basic Command Summary

RUN	CONT	NEW	NULL
EDIT	REN	LIST	

B.4 Basic Statement Summary

BAUD	CALL	CLEAR	CLOCK
DATA	READ	RESTORE	DIM
DO-WHILE	DO-UNTIL	END	FOR-NEXT
GOSUB	GOTO	ON-GOTO	ON-GOSUB
IF-THEN-ELSE		INPUT	LET
ONERR	ONTIME	PUSH	POP
REM	RETI	RETURN	STOP
STRING	LD@	ST@	GET
TIME			

B.5 Special Print Statement Summary

LCD:			
PRINT#	PH0#	PH1#	LIST#
AUX Port:			
PRINT	PH0	PH1	LIST
PROG Port:			
PRINT@	PH0@	PH1@	LIST@

B.6 Basic Operator Summary

ARITHMETIC			
ADD(+)	MULT(*)	DIVIDE(/)	SUB(-)
EXPONEN(**)	ABS( )	SQR( )	LOG( )
EXP( )	SGN( )	INT( )	
LOGICAL			
.AND.	.OR.	.XOR.	.NOT.
TRIG			
SIN( )	COS( )	TAN( )	ATN( )
OTHER			
RND	ASC( )	CHR( )	CBY( )
DBY( )	XBY( )		

See MCS®-BASIC 52 Manual in Appendix D for details.

APPENDIX C: COMMUNICATION ERROR CODES

These are the standard SIMATIC TI communication error codes. Refer to the proper SIMATIC TI manual for additional information. These error codes should be very rare when using the 5250-BAS Operator Panel. If any of these error codes do occur, please contact CTI with information on the error code type and the system conditions when the error code occurred.

CODE	ERROR DEFINITION
01	Reset Current Transaction
02	Address Out of Range
03	Requested Data Not Found
04	Illegal Task Code Request
05	Request Exceeds Program Memory Size
06	Diagnostic Fail upon Power Up
07	Fatal Error Detected
08	Keylock Protect Error
09	Incorrect Amount of Data Sent with Request
0A	Illegal Request in Current Operational Mode
0B	Network Was Not Deleted
0C	Attempted Write Operation Did Not Verify
0D	Illegal Number of ASCII Characters Received
0E	Illegal Write to Program Memory (Non volatile)
0F	Data Not Inserted
10	Data Not Written
11	Invalid Data Sent with the Command
12	Invalid Operation with NIM (obsolete)
13	Store and Forward Buffer Busy
14	No Response From the Special Function Module
15	Illegal Instruction Found in Program Memory
16	Attempted to Write to a Protected Variable
17	No Response From PC
18	Requested Memory Size Exceeds Total Memory
19	Requested Memory Size Incorrect Block Size
1A	Requested Memory Size Less Than Minimum Value
1B	Requested Memory Size Larger Than Max Value
1C	PLC Busy
1D	Comm Error in HOLD Mode
1F-20	Spare
21	I/O Configuration Error
22-3F	Spare
40-5F	SF/Loop Error
60-FF	Spare

APPENDIX D. The MCS®-BASIC 52 User Manual for Version 1.1

The MCS®-BASIC 52 User Manual for Version 1.1 is included with permission.
The MCS®-BASIC 52 User Manual for Version 1.1 follows this manual.

LIMITED PRODUCT WARRANTY

CTI warrants that this CTI Industrial Product shall be free from defects in material and workmanship for a period of one (1) year after purchase from CTI or from an authorized CTI Industrial Distributor. This CTI Industrial Product will be newly manufactured from new and/or serviceable used parts which are equal to new in the Product.

Should this CTI Industrial Product fail to be free from defects in material and workmanship at any time during this one (1) year warranty period, CTI will repair or replace (at its option) parts or Products found to be defective and shipped prepaid by the customer to a designated CTI service location along with proof of purchase date and associated serial number. Repair parts and replacement Product furnished under this warranty will be on an exchange basis and will be either reconditioned or new. All exchanged parts or Products become the property of CTI. Should any Product or part returned to CTI hereunder be found by CTI to be without defect, CTI will return such Product or part to the customer.

This warranty does not include repair of damage to a part or the Product resulting from: failure to provide a suitable environment as specified in applicable Product specifications, or damage caused by an accident, disaster, acts of God, neglect, abuse, misuse, transportation, alterations, attachments, accessories, supplies, non-CTI parts, non-CTI repairs or activities, or to any damage whose proximate cause was utilities or utility like services, or faulty installation or maintenance done by someone other than CTI.

Control Technology Inc. reserves the right to make changes to the Product in order to improve reliability, function, or design in the pursuit of providing the best possible Product. CTI assumes no responsibility for indirect or consequential damages resulting from the use or application of this equipment.

THE WARRANTY SET FORTH ABOVE IN THIS ARTICLE IS THE ONLY WARRANTY CTI GRANTS AND IT IS IN LIEU OF ANY OTHER IMPLIED OR EXPRESSED GUARANTY OR WARRANTY ON CTI PRODUCTS, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR OF FITNESS FOR A PARTICULAR PURPOSE AND IS IN LIEU OF ALL OBLIGATIONS OR LIABILITY OF CTI FOR DAMAGES IN CONNECTION WITH LOSS, DELIVERY, USE OR PERFORMANCE OF CTI PRODUCTS OR INTERRUPTION OF BUSINESS, LOSS OF USE, REVENUE OR PROFIT. IN NO EVENT WILL CTI BE LIABLE FOR SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES.

SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES FOR CONSUMER PRODUCTS, SO THE ABOVE LIMITATIONS OR EXCLUSIONS MAY NOT APPLY TO YOU.

THIS WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS, AND YOU MAY ALSO HAVE OTHER RIGHTS WHICH MAY VARY FROM STATE TO STATE.

REPAIR POLICY

In the event that the Product should fail during or after the warranty period, a Return Material Authorization (RMA) number can be requested verbally or in writing from CTI main offices. Whether this equipment is in or out of warranty, a Purchase Order number provided to CTI when requesting the RMA number will aid in expediting the repair process. The RMA number that is issued and your Purchase Order number should be referenced on the returning equipment's shipping documentation. Additionally, if under warranty, proof of purchase date and serial number must accompany the returned equipment. The current repair and/or exchange rates can be obtained by contacting CTI's main office at 1-800-537-8398.

When returning any module to CTI, follow proper static control precautions. Keep the module away from polyethylene products, polystyrene products and all other static producing materials. Packing the module in its original conductive bag is the preferred way to control static problems during shipment. **Failure to observe static control precautions may void the warranty.** For additional information on static control precautions, contact CTI's main office at 1-800-537-8398.