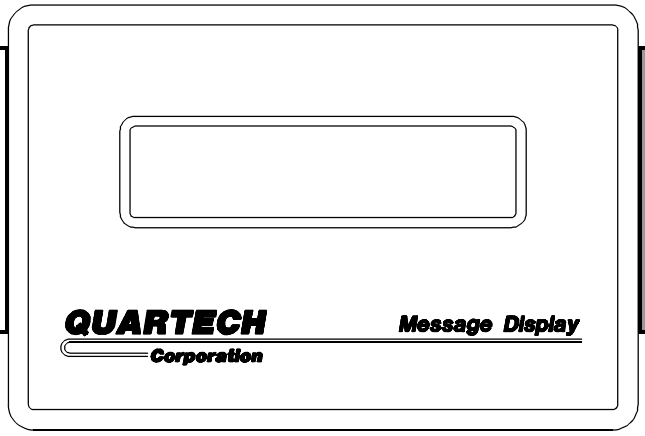
9701AP

Alphanumeric Display
for Allen-Bradley
SLC 500



PRODUCT

PM-9701
Revision 3

QUARTECH
Corporation

The product described in this document can have a variety of uses, the user and those responsible for applying this equipment must satisfy themselves as to the acceptability of each application and the use of the unit. Under no circumstances will QUARTECH CORPORATION be responsible or liable for any damage, including indirect or consequential losses resulting from the use, misuse, or application of the unit.

The text, illustrations, charts, and examples included in this document are intended solely to help explain applications of the product. Due to the many variables associated with specific uses or applications, QUARTECH CORPORATION cannot assume responsibility or liability for actual use based upon the data provided in this document.

No patent liability is assumed by QUARTECH CORPORATION with respect to the use of circuits, information, equipment, or software described in this document.

This document is printed in the U.S.A. and is subject to change without notice.

SLC is a trade mark of the Allen-Bradley Company.

Quartech Corporation
15943 Angelo Drive
Macomb Twp, MI 48042

Technical Assistence: (586) 781-0373
Fax: (586) 781-0378
www.QuartechCorp.com

Table of Contents

Section 1: Introduction

The model 9701 is a 40 character, alphanumeric message display, designed for use with Allen-Bradley's SLC 500 programmable controller. 32k bytes of EEPROM memory is available to store up to 327 messages. Messages may be displayed on the 9701's 2 line by 20 character vacuum fluorescent display, or as an option, messages may be transmitted out an RS-232 serial communication port to a printer, computer, or other device. Variable data (such as a counter's accumulator) may be embedded within a message.

The 9701 communicates directly with the SLC 500 programmable controller through its programming port, eliminating the need for communication modules, I/O modules and discrete wiring.

1.1 Compatibility:

- The 9701 can communicate directly with any member of the SLC 500 family, or on the DH-485 network along with other 9701's, other Quartech peripherals, Allen-Bradley's Advanced Programming Software (APS), other programmers, and any other device that strictly follows the DH-485 protocol.

- The 9701 can communicate with any one, but only one, SLC 500 at one time. The operator cannot select between multiple SLC's that may be on the network.

1.2 Features

- Ruggedly constructed for harsh industrial environments.

- Integral gasket to maintain NEMA 4/12 enclosure rating.

- Extremely reliable EEPROM memory will save messages until you want to change them.

- Messages are programmed using an IBM PC/XT/AT compatible keyboard. The keyboard plugs directly into the 9701.

- Optional computer program allows messages to be programmed using a personal computer.

- Optional RS-232 asynchronous communication port allows the ability to send messages to a printer, computer, or other device.

- 9701 may be mounted up to 1000 meters away from SLC.

- Up to four 9701's can communicate with the same SLC 500 at the same time.

1.3 Basics about Messages

The 9701 holds 327 messages. Each message is assigned a reference number from 0 to 326. The displayed message must not exceed 40 characters; however, 98 bytes of memory is allocated for each message. This extra memory is needed to store the message because variable data (such as the accumulator of a counter) may be embedded within a message. These embedded variables (called "variable definitions") often take up many more bytes of memory, then they occupy on the display screen.
We will find it convenient to make a distinction between the two forms of the message.

1. The "displayed message" is the message that will be seen on the display when triggered by the SLC 500. All variable definitions will be removed and replaced with the data from the SLC. The displayed message must not exceed 40 characters.

2. The "message definition" is the text of the displayed message as it is stored in memory. The message definition contains: 1) "destination flags" (which indicate the destination of the message, i.e. printer, stack, or display), 2) the text of the message, 3) "variable definitions" (which define the location and type of embedded variables), 4) printer control codes, if the message is printed, and 5) a checksum.

1.4 Initial Power Up Message

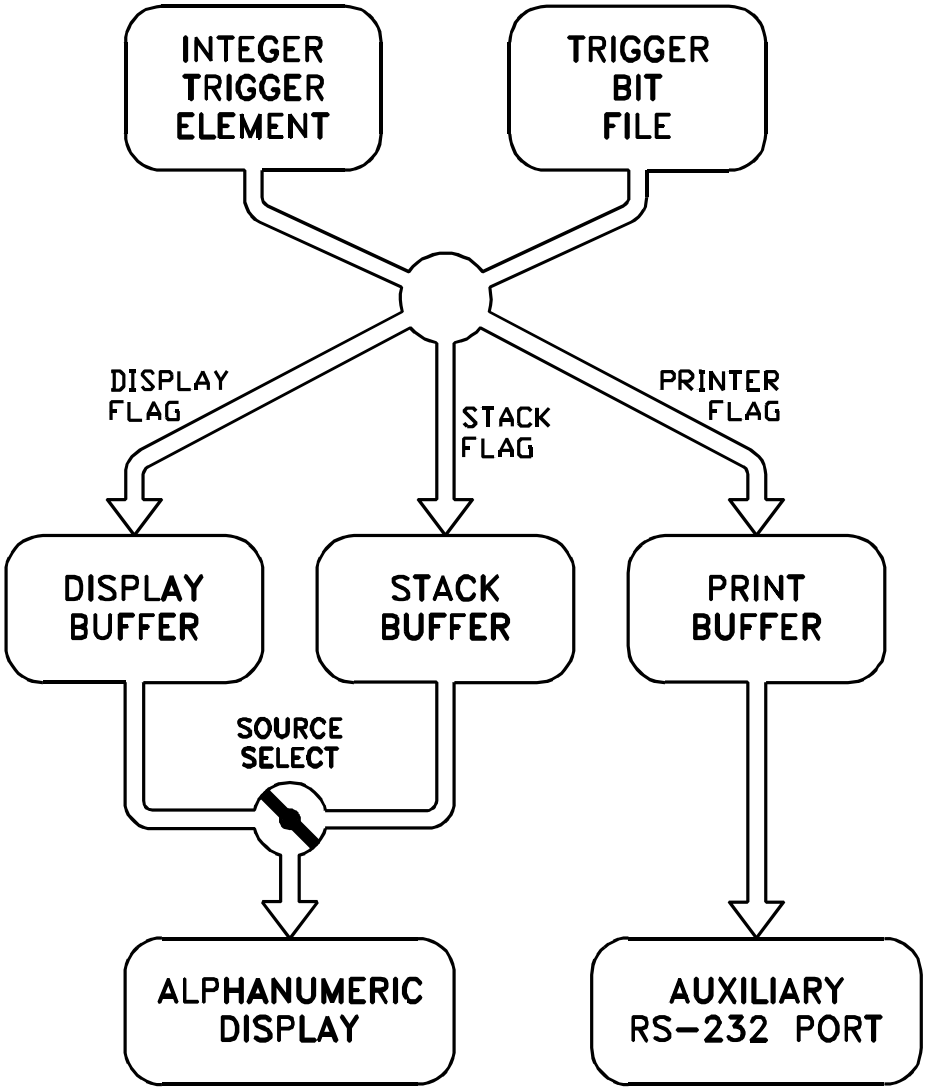
On power up, before any messages are triggered, the 9701 will display its "sign on" message that reads "Quartech Corporation 9701 Version x.x". This message may be replaced by the end user. To do this, you must create message 325 and put your own "sign on" message there. Message 325 must contain text only; no variables are allowed.

After communication is established with the SLC 500, message number 0 will be triggred by default. This message will remain on the display until some other message is triggered. The text of this message is set at the factory to read: "9701 IS ON LINE...."; however, it may be reprogrammed by the end user. This message is treated like any other message in that it may contain embedded variables, and its destination may be modified by the end user. It will always go to the display, regardless of the state of its Display destination flag.

Section 2: Triggering Messages

Since the 9701 communicates directly with the SLC 500's programming port, messages may be triggered simply by moving data around inside the SLC. In general, there are two methods available to trigger messages, one uses an integer file, the other uses a bit file. The SLC 500 may trigger messages using one method or the other, or the programmer may choose to use a combination of the two methods.

Once a message is triggered, it can go to one or more of three destinations: the **Display buffer**, the **Stack buffer**, or the **Printer buffer**. The **Source select** bit will determine which buffer feeds the alphanumeric display. The operator may switch back and forth between the two buffers, to alternately display two different messages. The Printer buffer is only available on the 9701-AP.



Flow of messages based on Message Destination Flags.

2.0.1 destination flags

Three destination flags may be adjusted independently for each message. The flags are labeled "Display", "Stack", and "Printer". See section 4.2.1 for more about programming the flags.

Display flag: If the Display flag is set when a message is programmed, the message will be sent to the Display buffer when it is triggered. If the Display buffer is selected as the message source, the triggered message will pass through the buffer and go immediately to the alphanumeric display. This is the most commonly used flag, and it is set by default.

Stack flag: If the Stack flag is set when a message is programmed, the message will be saved in the Stack buffer, inside the 9701. The message can then be displayed later, when requested by the operator. This could be useful in cases where multiple messages might be triggered in quick succession.

Note that the display flag could also be set. In this case the message would be sent to the display immediately, and also saved on the stack. (See Section 2.4 Message Stack, for more information).

Printer flag: If the Printer flag is set when the message is programmed, the message will be sent out the printer port. Note that this flag may be used in any combination with the other destination flags. Also note that this flag only applies to the 9701-AP. The standard 9701 does not have a printer port, and the Printer destination flag is not available.

2.1 Integer Trigger Element

Using the integer trigger method, the user will assign one element of an integer type file to be used as a Trigger Element. The 9701 will monitor that address, and any time the value of the data changes, a new message will be triggered. For example, to trigger message number 10, move the decimal number 10 into the Trigger Element.

- **!** The trigger element can be located within any integer file in the SLC 500.
- **!** The trigger element address must be programmed into the 9701 display, using the **F7 Configure Mode** of the 9701's **Message Editor**. (See Section 4.2.5).
- **!** The Integer Trigger Element can be disabled by programming its file number to be zero. If this is done the 9701 will not use an Integer Trigger Element. This might be desirable when using the bit trigger method.

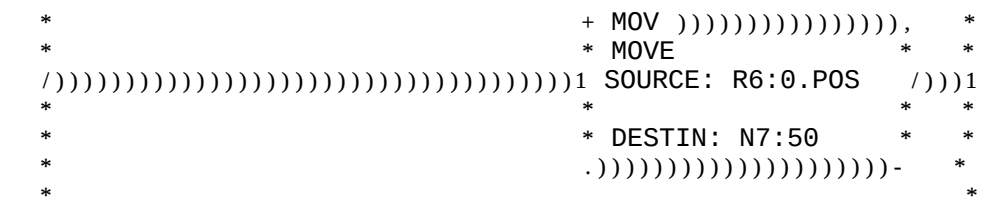
EXAMPLE 1:

Let's say your SLC 500 controls some process using a Sequencer Instruction, and you wish to use the 9701 display to monitor the status of that process. This is quite easily done using the 9701's Integer Trigger method.

First, choose the address of the Trigger Element. Since this can be any element in any integer file, let's make it N7:50 for this example.

Next, you must tell the 9701 what address you have chosen for the Trigger Element. Use the **F7 Configure Mode** in the **Message Editor** to do this. Set the Integer File number to 7, and the Element number to 50. Then, since we won't be using the bit trigger method, set the Trigger Bit File to zero. This will disable bit triggering.

Finally you must program the SLC 500 to make it trigger the messages. Let's assume the sequencer instruction is programmed at R6:0, the position register is R6:0.POS.



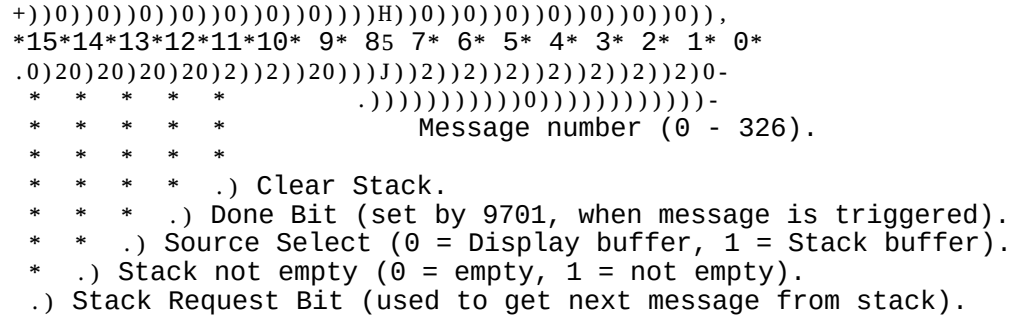
This instruction will copy the value from the position register into the Trigger Element. Each time the sequencer advances to the next step, the position register will change, and the 9701 will trigger the next message. This way the message display will follow the sequencer, with each step in the sequence having a message attached to it.

Obviously, for this example to work, you must also program some messages into the 9701. Since each message can have its own, independent destination, it is important to adjust the destination flags for each message. For this example, each message will have only one destination - the Display buffer. See Section 4: "Programming Messages". Also, it is likely that you will want to embed variables into your messages. This is a powerful feature, but still very easy to use. See Section 3: "Embedded Variables".

2.2 More About Integer Triggering

The Integer Trigger method will use one 16-bit element of an integer file. The message number has a range of 0 to 326, therefore it will occupy a maximum of nine bits in the Trigger Element. The remaining (upper 7) bits may be used by the 9701 to indicate status, or by the SLC 500 to pass commands to the 9701. These will be referred to as "control bits".

Integer Trigger Element



Operation of bits 11, 13, 14, and 15 (the "stack control bits") is discussed in Section 2.4 "Message Stack". The Done Bit is described in Section 2.2.1 "Done Bit".

The ladder logic shown in Example 1 of section 2.1 is the simplest method to trigger a message. Most programmers will probably choose to use similar logic in their programs. However, the MOV instruction used in that example, writes to all 16 bits of the Trigger Element. This destroys the status of the upper bits of the Trigger Element (the control bits).

In that example the control bits are not used, so it is acceptable to use the MOV instruction. A simple modification could be made to protect the control bits if they are needed:

```
*                                  + MVM ))))))))))))))),    *
*                                  * MASKED MOVE        *    *
/)))))))))))))))))))))))))))))))))1 SOURCE:     R6:0.POS/)))1
*                                  * MASK:       01FF     *    *
*                                  * DESTINATION: N7:50 *    *
*                                  .)))))))))))))))))))))-    *
*                                                             *
```

The Masked Move instruction will copy the value from the SOURCE address, then pass it through the MASK and into the DESTINATION. Only the bits marked with ones in the MASK will be modified in the DESTINATION address. Using a constant value of 01FF hex, for the MASK will protect all bits in the Trigger Element that are not used to hold the message number. This will protect the upper control bits as well as all bits reserved for future use.

2.2.1 Done Bit (Trigger Element bit 12)

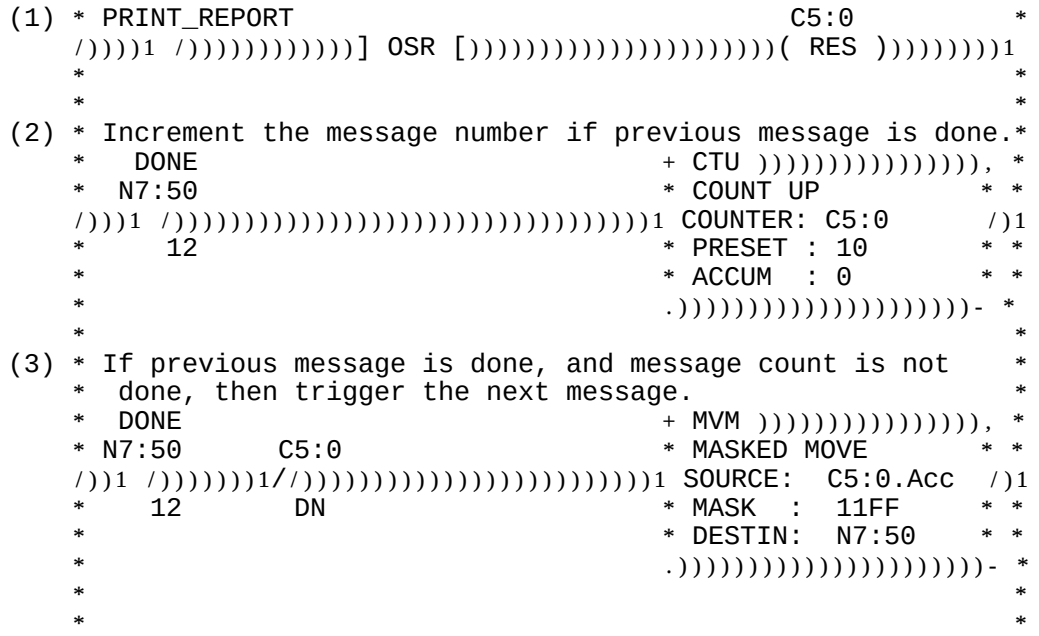
The communication between the 9701 and SLC 500 takes some (small amount of) time. This means that there will undoubtedly be some delay between the time that the SLC 500 triggers a message, and the time that the 9701 senses the message trigger and actually displays the message. This delay time can vary greatly, depending on a number of factors; such as, the communication baud rate, the number of nodes on the network, and the number and type of variables embedded within a message. In the best case the delay time might be as low as 20 to 25mS. In the worst case it is likely to be higher than 3 sec.

In many programs (such as Example 1 of Section 2.1) this time delay can simply be ignored. However, there are cases where it would be helpful to know just how long a message number must be left in the Trigger Element to ensure that the message is triggered. This can be accomplished by using the Done Bit. The 9701 will set the Done Bit every time it triggers a message. If the SLC 500 clears the Done Bit when it writes the message number to the Trigger Element, then when the 9701 sets the Done Bit, the SLC will have positive confirmation that the message was triggered (see Example 2).

Note that the Done Bit is set even if the message is sent to the stack or to the printer. In this case the message may not actually have been displayed yet, but it has been triggered and is waiting to be displayed.

The Done Bit could also be used to resolve the conflict that would arise if the SLC 500 attempts to trigger several messages at the same time; however, the ladder logic to do this can get a bit tricky. A more efficient solution would be to use the Bit Trigger method described in Section 2.3 of this manual.

EXAMPLE 2:
The Done Bit will come in handy when using the 9701 to print a report to a printer. In this example, ten messages are chained together to form a report that will be periodically printed. Each of the ten messages will be triggered, one after the other, as quickly as possible.



Rung 1: The process begins when the PRINT_REPORT contact picks up (this could be controlled by an externally wired push-button). The one shot contact prevents counter C5:0 from being reset for more then one scan.

RUNG 2: Counter C5:0 is used to hold the message number. It will count the number of messages that we trigger. The first pass through, assume the DONE bit is set, so the counter will not increment.

RUNG 3: If the previous message is DONE and the message counter is not done (we still have more messages to print) then we use the accumulator of the message counter to trigger the next message. **Notice the value of the MASK (11FF hex), this will clear the Done bit when we write the message number!** When the 9701 is done it will set the Done bit. This will increment the message count and cause the next message to be triggered.

2.3 Bit Trigger Address

To trigger messages using a bit type address, the user will assign a bit type file within the SLC, to be used exclusively for triggering messages. Each bit in the Trigger Bit File will be assigned a message in the 9701. To display a particular message, simply latch its corresponding trigger bit. **After the 9701 displays the message, it will unlatch the trigger bit.**

! Any Bit type file in the current SLC 500 application program may be designated as the Trigger Bit File. However, the bits in this file MUST only be used for triggering messages.

! The Trigger Bit File number must be programmed into the 9701, using the F7 Configure Mode of the 9701's message editor. (See Section 4.2.5 for more details).

! The maximum number of elements allowed in the Trigger Bit File is 21 (i.e. elements 0 - 20). There may be less then 21 elements but not more. Better performance will be

achieved if the Trigger Bit File is as small as possible. If you only need to use two elements to trigger all of the messages used by your program, don't create a Trigger Bit File with more than two elements in it.

- All bits in the Trigger Bit File are used to trigger messages. Bit 1 will trigger message 1, bit 2 will trigger messages 2, and so forth up to bit 326.

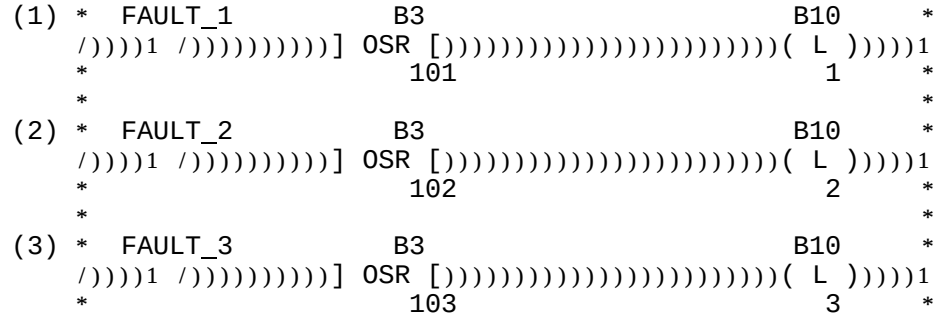
- The 9701 will unlatch the trigger bit after the message is displayed. If a message is sent to the stack or to the printer, its trigger bit will not be cleared until it is pulled out of the stack or printed.

- The Trigger Bit File can be disabled by programming its file number to be zero. If this is done the 9701 will not use a Trigger Bit File. This might be desirable when using the Integer Trigger method.

EXAMPLE 3:
The Bit Trigger method is particularly useful for triggering fault messages. Fault messages are usually unpredictable: one might pop up at any time, or several might be triggered at the same time. The bit trigger method can handle cases like this, because each message has its own separate control bit. This example will show how to trigger three different fault messages, in response to three different faults.

First, choose the Trigger Bit File number. Since all of the bits in this file must be used to trigger messages, let's create a new bit type file - say file number 10. We will get better performance if the Trigger Bit File is as small as possible. Since we only need to use three bits for this example, it will be best to only create address B10:0 (use the SLC 500 programmer to do this).
Next, you must tell the 9701 what file number you have chosen for the Trigger Bit File. Use the F7 Configure Mode in the Message Editor (see Section 4.2.5 "F7 Key"). Make the Integer Trigger Element file number zero, to disable Integer Triggering. Then set the Trigger Bit File number to 10.

Finally you must program the SLC 500 to make it trigger the messages. The logic to do this follows:



Let's take a closer look at rung 1 of the ladder logic above. When the FAULT_1 contact closes, the one-shot coil B3/101 will solve true, and latch B10/1. This is the trigger bit for message number 1. B10/1 will remain latched until the 9701 actually displays the message, then the 9701 will unlatch the trigger bit B10/1. The one-shot coil B3/101 will prevent the message from triggering again, until the FAULT_1 contact drops out. The address of the one-shot B3/101 is arbitrary. We use B3/101 in this example simply to make it clear that this bit must not be placed in the Trigger Bit File. Even though there are lots of unused bits left in the Trigger Bit File, they must not be used for any purpose other than triggering additional messages.

2.4 Message Stack

When a message is triggered it may be displayed immediately, or if you choose, the 9701 can save the messages in a FIFO stack to be reviewed later. The ability to save the messages in a FIFO stack could be useful in cases where several messages might be triggered in quick succession, or where an operator is not always able to watch the display.

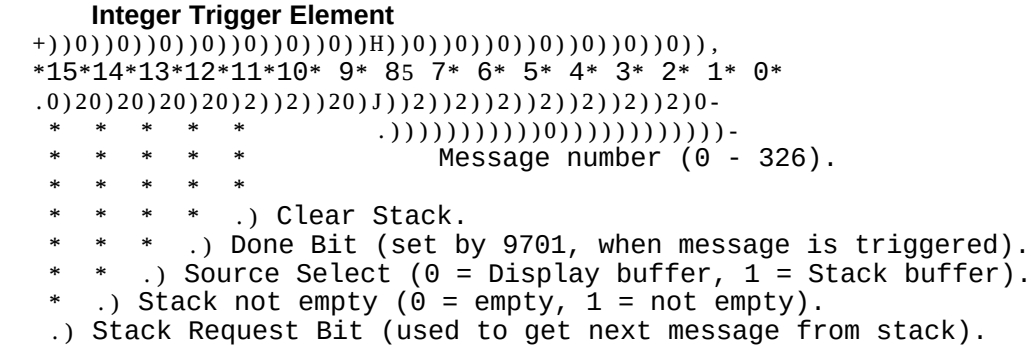
Each message can be sent to the display, or to the stack, or both. The destination of the message is determined by adjusting destination flags in the Message Editor, when the message is programmed. Destination flags can be set for each message independently, using the F7 key in the Message Editor. (See Section 4.2.1 for details).

- **!** The message stack is a First-In, First-Out (FIFO) type. The first message put in the stack when it is triggered, will be the first message pulled out of the stack when the Stack Request Bit is set.

- **!** The stack can hold a maximum of 50 messages at any one time. Once the stack is filled, an attempt to push a new message into the stack will simply be ignored. When a message is pulled off the stack using the Stack Request Bit, a new message can be added.

As far as the stack is concerned, it makes no difference if messages are triggered using a Trigger Bit File or an Integer Trigger Element. However, an Integer Trigger Element must be assigned to provide the Stack Control Bits.

2.4.1 Stack Control Bits



Bit 13, Source Select: When the Source Select bit is off (0), the 9701 will fetch messages from its Display buffer and send them to its alphanumeric display. In this state, when a message is triggered, and its Display destination flag is set, it will go to the display immediately. When a new message is triggered, it will immediately overwrite the old message.

When the Source Select bit is on (1), the 9701 will fetch messages from its Stack buffer. It will display the first message that is waiting in the Stack. It will not display the next message until a transition of the Stack Request Bit. If the Stack buffer is empty it will display message 326 by default.

The Source Select bit may be switched off and on to alternately display a message from the Display buffer or from the Stack buffer.

Bit 14, Stack not Empty: The 9701 will set the Stack not Empty bit whenever it pushes a message onto the stack. The 9701 will clear this bit when the Stack Request Bit is used to pull the last message off the stack (i.e. when the stack is empty).

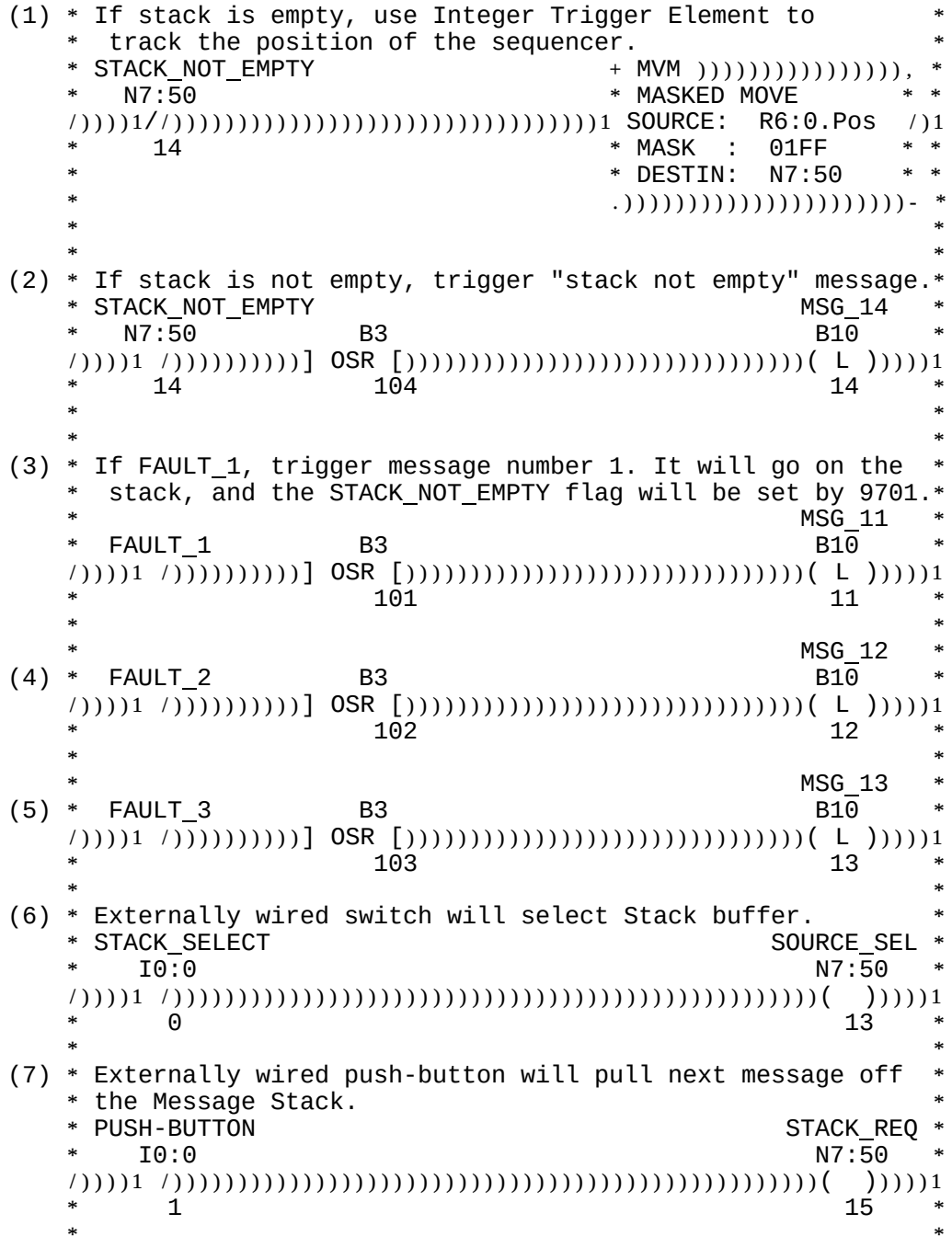
Bit 15, Stack Request: Messages may be removed from the stack and sent to the display, by using the Stack Request Bit. A low to high transition of the Stack Request Bit will cause the next available message to be pulled from the stack and sent to the display. The message that was previously on the display will be lost.

The 9701 will only examine the Stack Request bit when the Source Select bit is on (that is, when the Stack buffer is selected).

Bit 11, Clear Stack: A low to high transition of the Clear Stack Bit will cause all messages to be removed from the stack. The Stack not Empty bit will be turned off. If the Source Select bit is set to view the stack, message 326 will be triggered ("Stack is empty" message) to indicate that the stack is empty.

EXAMPLE 4:
This example will demonstrate the use of a message stack, as well as the idea of using an Integer Trigger Element and a Trigger Bit File to trigger messages in the same program.

A Trigger Bit File will be used to trigger fault messages. The technique used is similar to Example 2, but here the fault messages will go onto the Message Stack. The Integer Trigger Element will be used to track a sequencer controlled process, much like Example 1, only this time the sequencer messages may be interrupted by a fault message.
Assume the Integer Trigger Element is N7:50, and the Trigger Bit File is B10. Message numbers 1 through 10 will be triggered using the Integer Trigger method, while messages 11 through 14 will be triggered using bits.



Ladder logic used in Example 4.

Section 3: Embedded Variables

Perhaps the most powerful feature of the 9701, is the ability to display variable data (such as a counter's accumulator) in real time. To do this you need only supply the address of the data, and specify how the data is to be displayed (i.e. decimal, hex, or bar graph). The address information is simply included in the message when it is typed in using the message editor. Then, when the message is triggered, the 9701 pulls the address out of the message and replaces it with the data from the SLC 500.

Several characters have been reserved for the purpose of specifying how the data is to be displayed; these characters also act as a delimiter for the address information. For example, to display the accumulator of counter C5:0, a message might look like this:

Counter C5:0 = $C5:0.A$

The $ delimiter specifies 5 digit decimal data, and also denotes the beginning and ending of the address information. Once the message is triggered it might look like this:

Counter C5:0 = 03752

The number on the display will change as the count in the accumulator changes.

The correct entry format (syntax) must be used when specifying embedded variables. The following examples indicate the proper syntax for the various data displays.

3.1 Decimal Data Display

Any word address in the SLC 500 data table may be displayed as a decimal number using two, three, four, or five digits. A negative sign and decimal point may also be displayed, if desired.

The following variable delimiters are used for decimal data display:

2 digit data: * 4 digit data: !
3 digit data: # 5 digit data: $

A typical variable definition for a decimal display is as follows:

```
$-.2T4:0.A$
```

3.1.1 negative sign

A negative sign may be included (optional) in the display by immediately following the start delimiter by a negative sign. If this is the case, the data will be interpreted as a 16 bit signed number. If the data is negative, a negative sign will be displayed in front of the number. If the data is not negative, a blank space will be left in front of the number, as a place keeper.

If the negative sign is not included in the variable definition, the data will still be interpreted as a 16 bit signed number, however the negative sign will not be displayed. For example, the number -32767 will simply be displayed as 32767.

3.1.2 DECIMAL POINT

A decimal point may be included (optional) in the display by immediately following the start delimiter (or negative sign if it is used) by a decimal point and then a single digit that indicates position.

Decimal point field	Decimal point position
.0	XXXXX.
.1	XXXX.X
.2	XXX.XX
.3	XX.XXX
.4	X.XXXX
.5	.XXXXX

3.1.3 Word Address

Any word address in the SLC 500's data table can be displayed as a decimal number, including addresses in Bit type files, and the Status file.

When defining variables of Timer, Counter, and Control type files, remember to specify the address down to the word level. For timers and counters use the following word identifiers:

.S or .s = Status word.
.A or .a = Accumulator.
.P or .p = Preset.

For control type files, use these word identifiers:

.S or .s = Status word.
.L or .l = Length.
.P or .p = Position.

EXAMPLE 3.1:

In this example we will display the preset and accumulator of timer T4:50 . Since the preset value is 9.99 seconds, we will save space in the message by defining a three digit decimal variable (use the # delimiter). We will also need to include a decimal point in the display. The Message Definition below, shows how the message will look when it is typed into the Message Editor. The displayed message below, shows how the message would look when it is triggered by the SLC 500. The '**#**' character represents one digit of a variable.

Message Definition:

```
Wash cycle is #.2T4:50.P#
Elapsed time= #.2T4:50.A#
```

Displayed message:

```
Wash cycle is #.##
Elapsed time= #.##
```

EXAMPLE 3.2:

Here we demonstrate a very simple four digit decimal type variable from a counter, and a more complex two digit decimal with a negative sign, from an integer file. Counter C5:23 is counting parts produced so far today. This is likely to be a large number so we use a four digit decimal variable. The SLC 500 will perform the necessary calculation to convert this count to a percent change in production when compared to yesterday, and save this new variable in integer N7:198. Since the percent change variable is likely to be a smaller number, we will use only a two digit decimal; however, the percent change may be positive or it may be a negative value, so we will include a negative sign in the variable definition.
Message definition:

```
Parts today: !C5:23.A!
Percent change: *-N7:198*
```

Displayed message:

```
Parts today: ####
Percent change:  ##
```

If the percent change is a positive value, the message will be as shown above. However, if it is a negative value the message will be as shown below.

Displayed message:

```
Parts today: ####
Percent change: -##
```

3.2 Hexadecimal Data Display

Any word address in the SLC 500 data table may be displayed as a four digit hexadecimal number. The variable delimiter character to use is the % .

Four digit hex: %

A typical variable definition for a hexadecimal display is as follows: `%B3:12%`

When defining variables of Timer, Counter, and Control type files, remember to specify the address down to the word level. Use the word identifiers shown in section 3.1.3 Word Addresses.

3.3 Bit Status

Any bit address in the SLC 500 data table may be assigned a user defined label for the 0 and 1 state of the bit. The first user defined label (up to 40 characters) will be displayed if the bit is 0. If the bit is 1, the second label (up to 40 characters) is displayed. The delimiter character to use for this type variable is the ^ .

Bit status: ^

In general the variable definition syntax is:

^ Bit address | Off Label | On Label ^

A typical variable definition for the Bit Status display is as follows:

```
64444444444444444444444444444444444444444444444L4444444L44444444444444444447
5               VARIABLE                      *B3:0/0 *    RESULTING     5
5              DEFINITION                     * VALUE *     DISPLAY      5
:444444444444444444444444444444444444444444444P4444444P444444444444444444<
5 ^B3:0/0|The bit is zero|The bit is one^ *   0   * The bit is zero 5
5 ^B3:0/0|The bit is zero|The bit is one^ *   1   * The bit is one  5
94444444444444444444444444444444444444444444444N4444444N444444444444444444448
```

Note that the '|' character must be used to separate the bit address from the first label, and also to separate the first label from the second label. Labels may use any character except the '|'.

EXAMPLE 3.3

Message definition:

```
Fault at station 2
Pump 2 is ^O0:0/3|Off|On^
```

Displayed message:

```
Fault at station 2
Pump 2 is Off
```

If bit O0:0/3 is off the message will be displayed as shown above. Else the message will be displayed as shown below.

Displayed message:

```
Fault at station 2
Pump 2 is On
```

EXAMPLE 3.4

A clever application of the Bit Status variable is to use it to flash a message or perhaps only one word within a message. Here we demonstrate the technique with a message that is too long to fit on the display all at one time. What we will do is break the message in two, then alternately display each part for a few seconds. To do this we will define the status of bit S2:4/7. This is one of the fine time base bits in the SLC 500's status file.

Message definition:

```
^S2:4/7|9701 Message Display     For SLC 500|
     Uses No I/O!^
```

Displayed message:

```
9701 Message Display
    For SLC 500
```

If bit S2:4/7 is off the message will be displayed as shown above. Else the message will be displayed as shown below. Since S2:4/7 will be turning on and off at a steady rate, the two displayed messages will alternate back and forth at a steady rate.

Displayed message:

```
Uses No I/O!
```

3.4 Bar Graph

Any word address in the SLC 500 data table may be displayed as a 20 segment bar graph on the lower line of the two line display. The variable delimiter character to use is the @ .

Bar graph: @

A typical Bar Graph variable definition is as follows:

```
@N7:25@
```

The full scale value of the bar graph is 20. Values greater than 20 will be considered full scale. The 9701 does not scale the value. To scale, divide the full scale raw data value by 20 to obtain the scaling factor. The raw data is then divided by the scaling factor to obtain the data that is assigned to the bar graph register.

The bar graph is always displayed on the lower line. The top display line may be used for additional information and/or data displays.

3.5 Displaying Delimiter Characters

The delimiter characters are also characters which can be displayed in a message. To do this, put a '°' degree symbol in front of the delimiter character. When the 9701 scans the message looking for variables, if it sees a ° in front of a delimiter character it will not treat it as a variable definition. Instead it will print the delimiter character in the message just like any non-delimiter character.

Since the standard IBM keyboard, needed by the Message Editor, does not have a [°] key on it, we use the [ALT] key instead. When the [ALT] key is pressed in the Message Editor, the ° character is produced on the display.

EXAMPLES:

Message definition:

`Fault at Station #2` (Incorrect).

The message above contains a syntax error, because the '#' character is a variable delimiter.

Message definition:

`Fault at Station °#2` (Correct).

Displayed message:

```
Fault at Station #2
```

Message definition:

`Temp is 100 °°C`

Displayed message:

```
Temp is 100 °C
```

The above message shows how to print a '°' degree symbol in a message.

3.6 Variable Definition Syntax Table

The following table lists all the variable delimiter and some typical variable definition syntax.

Delimiter Character	Function	Variable Definition	Resulting Display
*	2 Digit Display - least significant 2 digits of specified word address.	*N7:0*	XX
		.1T4:0.A	X.X
#	3 Digit Display - least significant 3 digits of specified word address.	#C5:0#	XXX
		#.2T4:0.P#	X.XX
!	4 Digit Display - least significant 4 digits of specified word address.	!-C5:0!	-XXXX
		!.3T4:0.A!	X.XXX
$	5 Digit Display - least significant 5 digits of specified word address.	$R6:0.P$	XXXXX
		$-.4B3:0$	-X.XXXX
%	Hexadecimal Display - display 4 digit hexadecimal data from specified word address.	%I1:0%	XXXX
		%R6:0.S%	XXXX
^	Bit Status Defined - User defined label for the two states of the given bit (Off Label, and On Label).	*^B3:0/0|OFF|ON^*	AAA
@	Bar Graph Display - 20 segment bar graph base on the value in a specified word address.	@N7:0@	-----

Other Special Characters

Character	Function	Example	Resulting Display
°	Allows a variable delimiter character to be displayed in a message.	°#	#
		°°	°
		°\	\
\	Insert two digit hexadecimal printer code.	\1B	
		\0D	

Section 4: Programming Messages

Messages may be entered into the 9701's memory using a standard IBM PC/XT/AT compatible keyboard. With the keyboard plugged into the 9701's keyboard port, you may use the built in message editor to edit and save messages, and setup all necessary configuration parameters.

A computer program is also available to allow you to program the messages using a personal computer. With this software you can create or edit message files, save the messages to diskette, print a listing of messages, and download the message file to the 9701 via the computer's serial port.

4.1 Invoking the Message Editor

To gain access to the 9701's resident Message Editor:

1) Remove power from the 9701.

2) Plug your IBM PC/XT/AT, or PS/2 compatible keyboard into the 5-pin DIN connector located on the bottom of the 9701.

3) Apply power to the 9701. After a few seconds the 9701 should display "Keyboard Active Select Function Key".

   Note that some keyboards (especially XT type) will require a key to be pressed on the keyboard, in order for the 9701 to detect that the keyboard is present. With such a wide variety of keyboard manufacturers, it is impossible to tell which keyboards have this trait. If you have trouble making a keyboard work, try pressing the [Space] bar on the keyboard when you power up the 9701.

4.2 Main Menu Function Keys

When the lower line of the display reads "Select Function Key", this is the prompt of the Main Menu. The only keys that are active on the keyboard are the function keys. They are defined as follows:

F1	Create a new message.
Shift+F1	Copy source message to destination.
F2	Delete a message.
F3	Edit an existing message.
F4	Review a message without variable definitions.
F5	Send/Receive message file from computer.
Shift+F5	Send a message file to another 9701.
F6	
F7	Configure Trigger address, Station number, ...
F8	Configure Printer.
F9	Go on line with the SLC 500.
F10	Clear Memory.

4.2.1 F1 create/F3 edit a message

The F1 key and F3 key perform the same function - they both allow you to create or edit a message. Once inside the message editor the following keys have special functions:

Enter	Exit and save the message.
Esc	Exit without saving, and return to main menu.
Tab	Display the message number of current message.
Insert	Toggle insert/overwrite mode.
Delete	Delete one character under the cursor.
Back Space	Delete one character left of cursor.
Alt	Print '°' symbol in message. See Section 4.5 .
F4	Test variable definition syntax.
F7	Modify current message "Destination flags".

F4 KEY:

The F4 key will test the syntax of any variable definitions that may be included within a message. If an error is found, an error message will signal the type of error. Press any key to bring your work back, and the cursor will be located at the position that the error was detected.

Since variable definitions take more space than the actual variable data, the F4 key is also very useful for checking the position of variables within a message. The F4 key will toggle between the "Message Definition" and a simulated version of the "Displayed Message", with '"' characters in place of variables. This will allow you to see the message as it will appear when it is triggered by the SLC 500.

F7 KEY:

The message destination flags are described in Section 1.3.1 . Basically these flags determine the destination of the message when it is triggered (i.e. display, stack, or printer). Use the F7 key to modify these flags.

the 9701 will ask a series of questions, such as
"Send to DISP_BUFF? >Yes No" .
If you choose yes it will set the Display flag, else it will clear the Display flag.

The last question it asks is
"Make this the new default? >Yes No" .
If you choose yes the 9701 will use the current set of destination flags as the default setting when a new message is created. This will only affect subsequent messages that are created. Messages that already exist will not be changed.

4.2.2 F4 key: review a message

When the F4 key is selected from the main menu, it performs similar functions as described in the Section 4.2.1. That is, it tests the syntax of variable definitions, and displays the message as it would be seen by an operator. However, it also performs one other useful function: when it is through with one message, it will search for the next message that exists in memory. It will flash this message number on the screen, so if you wish to view this message, press the [Enter] key. This function allows you to browse through your list of messages while skipping those messages that are not used.

4.2.3 F5 key: send/receive

A computer program, model number 9209, is available to allow you to program messages using a personal computer. The F5 key may be used to make the 9701 ready to transfer a message file to or from the computer. Follow the instructions provided with the 9209 software for trouble free operation.

4.2.4 Shift+F5: initiate transfer

The 9701 can transfer its message file to another 9701 display. This could be useful in cases where several 9701s need to be identically configured, or in the event of some failure of the 9701, you may wish to replace the defective unit with an identically configured one.

To transfer the message file between two 9701's:

- Connect Model 2102 (or equivalent) communication cable between the two 9701's.

- Attach a keyboard to the source 9701, and apply power. From the main menu, press Shift+F5. The display will read

```
Initiate Transfer
    >Yes   No
```

- Choose yes. The display will read

```
Initiate Transfer
Trying to sync...
```

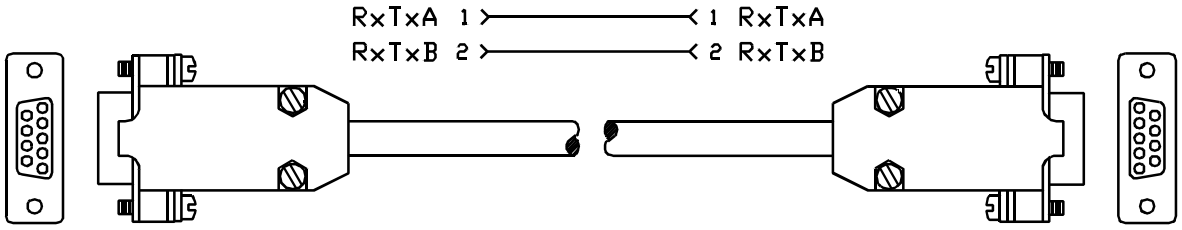
- Now apply power to the destination 9701. Its display will read

```
Loading from master:
Please wait...
```

The transfer will take only a few seconds. When completed both 9701's will read

```
Operation Complete.
Select Function Key
```

Model 2102 Communication Cable:



Typical cable used to transfer messages from one 9701 to another.

4.2.5 F7 key: configure

Select the F7 key from the main menu to modify the following parameters:

Integer Trigger File (07)
Integer Trigger Element (00)
Trigger Bit File (00)

See Section 2: "Triggering Messages" for more details.

9701 Station address (00)
SLC 500 Station Address (01)
Maximum Poll Address (31)
Baud Rate (19.2k)

See Section 5: "Communication (DH-485)".

The numbers in parentheses are the default settings, set at the factory.

4.2.6 F8 key: configure printer

Select the F8 key from the main menu to modify the following parameters needed by the Auxilliary Port:

Printer Baud Rate (9600)
Data Bits (8)
Stop bits (2)
Parity bit (None)

Append CRLF to message (Yes)
Lines per page (60)
Title message number (0)

4.2.7 F9 key: On-line with SLC 500

The F9 key allows the 9701 to go on line with the SLC 500 without unplugging the keyboard. This could be helpful when testing or troubleshooting messages.

If the 9701 is powered up without a keyboard attached, it will automatically attempt to establish communication with the SLC 500.

4.2.8 F10 key: Clear Memory

The F10 key will perform the following functions:

1) Set all messages to unused.

2) Set default values for all user programmable parameters (i.e. Trigger Address, Station Address, etc.).

3) Set default text for message number 0, 325, and 326.

Section 5: Communication (DH-485 data link)

The 9701 will communicate with Allen-Bradley's SLC 500 using the **DH-485 data link**. This communication network uses a multiple master, token passing protocol.

Compatibility:

- The 9701 can communicate on the DH-485 network along with other 9701's, other Quartech peripherals, Allen-Bradley's APS programmer, other programmers, and any other device that strictly follows the DH-485 protocol.
- Up to four Quartech peripherals along with up to one programming device can communicate with the same SLC 500 at the same time.
- The 9701 can communicate with any one, but only one, SLC 500 at one time. The operator cannot select between multiple SLC's that may be on the network.

5.1 Protocol Basics

The 9701 will communicate with Allen-Bradley's SLC 500 using the **DH-485 data link**. This communication network uses a multiple master, token passing protocol. For this protocol, the stations on the network form a logical ring; that is, the stations assume an ordered sequence, with the last member of the sequence followed by the first. Each station knows the identity of the station following it (its successor). The physical ordering of the stations on the medium is irrelevant and independent of the logical ordering.

Each station on the network has the responsibility of periodically granting an opportunity for new stations to enter the network. Anytime there exists a gap between a given station's address and its successor, that station must periodically issue a "**solicit successor**" message to each potential node between itself and its successor. The solicit successor message allows an opportunity for a new station to enter the network.

To reduce network maintenance, a **maximum poll address** may be assigned. The 9701 will send a "solicit successor" command only to those nodes that are in the gap between itself and its successor, and have an address less than or equal to its maximum poll address. Use of the maximum poll address will relieve the highest station on the network from the burden of soliciting a large number of unused nodes.

5.2 Choosing Station Addresses

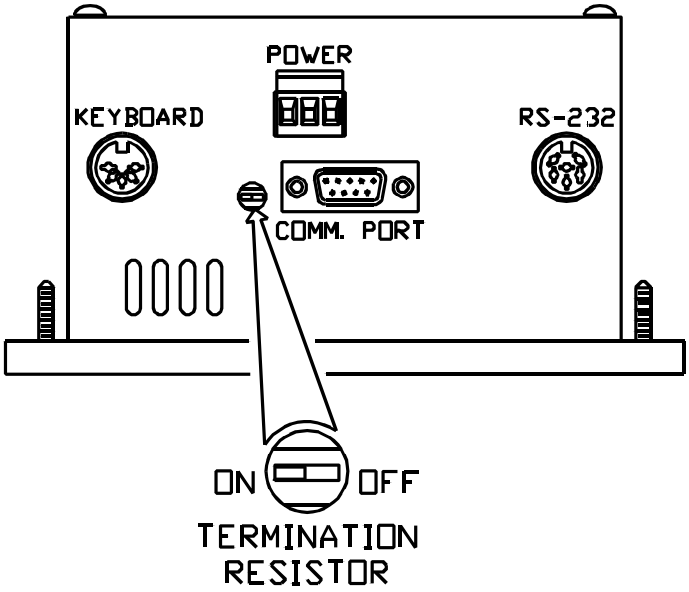
It is imperative that each device on the network is assigned a unique station address. The best performance will be realized if station addresses start at 0 and increase sequentially, with no gaps between successive addresses. The SLC 500s should be assigned the lowest addresses. Any program loaders or other peripherals should have the next higher addresses. The highest address on the network should be a 9701, and its maximum poll address should be set equal to its own station address.

For example: A typical DH-485 network might include one programming terminal, two 9701 displays, and one SLC 500. The SLC 500 should have station address 00. The programming terminal should have station address 01. The 9701s should have station addresses 02 and 03, and the last 9701's maximum poll address should be set at 03.

The above discussion has been for best performance. Sometimes it is desirable to leave gaps between station addresses so that other devices (perhaps a program loader) can be easily added after the data link has been initialized. This is acceptable, however, it will result in a reduction of performance because some device on the network will have to issue a solicit successor message to each unused address in the gap. If you do wish to leave a gap between station addresses, the gap should be as small as possible to reduce the link initialization time. However, a gap of even just one address will reduce network performance.

5.3 Transmission Line Termination

The 9701 is equipped with a 150Ω receiver termination resister that may be inserted or removed from the receiver circuit with a switch located near the communication port. This resister must be applied across the very end of the transmission line. In most cases the end of the transmission line is at the 9701's receiver, so the terminating resister should be left in the circuit. However, in case the 9701 is installed in a communication network where it is not the last device on the transmission line, its termination resister should be switch off. Only the last device on the line should have its termination resister turned on.



Location of 9701's Receiver Termination Resistor.

5.4 Communication Setup

Select the F7 key from the main menu of the message editor to set the following parameters needed by the communication hardware. (Note that the default values, set at the factory, will be acceptable for most applications).

9701 Address: This is the station address that the 9701 will assume on the DH-485 network. This address must be set even if the 9701 is plugged directly into the SLC 500.

SLC address: This is the station address of the SLC 500 that the 9701 will communicate with. The 9701 can communicate with any one, but only one, SLC 500 that may be on the network.

Baud Rate: This is the baud rate that the 9701 will use to communicate with the SLC 500. You must set the 9701's baud rate to match that of the SLC 500.

Max. Poll Address: This is the highest station address that the 9701 will poll when soliciting a successor. It has a range of 0 to 31.

Token Skip Number: The 9701 allows the user to adjust the number of times the 9701 will receive the token

before transmitting a message over the communication link. If the 9701 receives the token, and it has not yet reached the programmed "number of skips", it will simply pass the token without sending any other message. This, in effect, allows the user to adjust the speed of communication: If the number of token skips is increased, the 9701 will read data from the SLC less often. This will slow the 9701 down, but since it uses the communication link less often, it will speed up other devices that may be using the same link.

This feature may also be used to speed up the 9701: If the number of token skips is decreased, the 9701 will read data from the SLC 500 more frequently, and thus it will be more responsive to changes inside the SLC. It will trigger messages faster, and update variables more frequently.

To implement this feature the user must adjust the 9701's "# of Token Skips" variable. This variable tells the 9701 the number of times it must receive the token before sending a message to the SLC.

- If the "# of Token Skips" is set to zero, the 9701 will actively adjust the number of skips for optimum performance, based on current communication link conditions. This is the way that 9701 version 1.0 handles its communication.

- If the "# of Token Skips" is set to something other then zero, that value will be used regardless of other communication link conditions. For example, if the "# of Token Skips" is set to 1, the 9701 will read data from the SLC every time it gets the token. The 9701 version 1.0 sets its "# of Token Skips" variable to 2 by default. The highest value that can be entered is 255, but the 9701's communication will be very slow (and probably make the 9701 useless) with values greater than about 10 or 15.

- The "# of Token Skips" variable is located in the 9701's Communication Setup, [Shift+F7] in the 9701's message editor. This variable can only be adjusted using the 9701's built in editor, it cannot be adjusted with the 9209 computer software. However, if messages are created on the 9701 then saved on the computer, the "# of Token Skips" variable will also be saved. In fact, once the message file is in the computer, it can be copied or edited using the 9209 software, and the "# of Token Skips" variable will be still be there. In other words, it is part of the file, but it cannot be edited by the computer.

5.5 Communication Errors

The 9701 will display one of the following codes if it encounters a communication error that it cannot recover from without help from an operator. After displaying the error message of a few moments, the 9701 will retry.

-E6- No Communication
The 9701 has not received the token. That is, it is not able to join the communication network.

Possible Cause:

- **!** Disconnected, broken, or incorrect communication cable. Incorrect or damaged wiring (physical link) of the DH-485 network.
- **!** Inconsistent baud rate. The 9701 may be assigned a different baud rate than the SLC 500 (or the network). Use F7 key to check and adjust baud rate.
- **!** Bad station address. The 9701 may have the same station address as the SLC 500 (or some other device on the DH-485 network) or its station address may be higher than the maximum poll address. The 9701 MUST have a unique station address. Use F7 key to check station addresses.

-E7- No Acknowledge
The SLC 500 has failed to acknowledge a message sent by the 9701.

Possible Cause:

- **!** There is no SLC 500 located at the 9701's "SLC address". That is, the station address that the 9701 is expecting to find its SLC 500, is not the address of an SLC 500.
- **!** Noisy communication link. Check cable routing and look for possible sources of electrical interference. See Appendix B: Installation.
- **!** Improper transmission line termination. Each end of the transmission line should be terminated with a 150Ω receiver termination resister. See Section 6.3: "Transmission Line Termination".

-E8- No Reply
A message was acknowledged by the SLC 500, but no reply was received.

Possible Cause:

- **!** The SLC 500 or the 9701 was removed from the network.

-E9- Illegal Reply
The SLC 500's reply to a message was not the reply expected by the 9701.

Possible Cause:

- **!** The 9701's "SLC address" is not the address of an SLC 500, but it is the address of some other device.
- **!** The 9701 attempted to perform some operation that is not allowed by the SLC 500. For example: An external device is not allowed to monitor the SLC's data table, while the data table is being downloaded from a program loader. This error code will often be displayed if you download a program to the SLC 500 while the 9701 is on line (it will not affect the program transfer).

Section 6: Auxiliary Port

The 9701-AP has an auxiliary serial communication port that may be used to send messages to a printer or other device. This section will describe the options available when a message is printed.

6.1 General Operation

Each of the 9701's messages may be independently sent to the Auxiliary Port as well as other destinations, such as, the display or the stack. If a message is to be sent out the Auxiliary Port, its Printer Destination flag must be set. If this is the case, once the message is triggered its message number is saved in a buffer inside the 9701. This Print Buffer is a First-In First-Out (FIFO) type stack, with room enough for up to 50 messages at any one time.

When the 9701 is ready to print it will fetch the message from the Print Buffer, and replace any variable definitions with the data from the SLC 500. Then it will assert its **RTS** (Request To Send) line and wait for the **CTS** (Clear To Send) signal from the printer. Once this is satisfied, the 9701 will transmit the message using ASCII codes. The serial data frame is defined using the F8 key in the Message Editor.

The transmission may be interrupted at any time by removing the **CTS** signal. Once this signal is satisfied again, the 9701 will resume transmitting again. After a message is completely transmitted, the **RTS** signal will be set inactive while the next message is prepared.

6.2 Printer Control Codes

The 9701 provides the ability to directly control the printer by inserting printer control codes within a message. Most printers require some sort of control codes; these are most often non-printable ASCII codes such as 'Carriage Return' (0Dh), Line Feed (0Ah), and so forth. These codes and indeed any 8-bit hexadecimal code can be inserted into a message by using the 9701's printer code delimiter '\'. For example, to insert a Carriage Return into a message, use the following sequence of characters:

\0d

When the 9701 finds a 'back slash' character '\', within a message definition, it will use the two characters immediately following the 'back slash' to form an 8 bit hexadecimal number. It will transmit this number out to the printer. The 'back slash' and the two characters following it will not be included in the printed message, or the displayed message (if the message is also sent to the display).

More Examples:
The 'Esc' character is 1B hex **\1B**
The 'Form Feed' character is 0C hex **\0C**

Many printers use sequences of control codes (escape sequences) to select print options such as bold print or alternate font styles. For example, one printer uses the sequence 'Esc G' to turn on bold print, and the sequence 'Esc H' to turn bold print off. ASCII character 'G' = \47, and 'H' = \48.
Message definition:

```
\1B\47This will be bold.  \1B\48This will be normal.
```

Printed message:

This will be bold. This will be normal.

6.2.1 Append CRLF to Message:

The 9701 can perform several functions that will help automate the process of printing messages. One such

function will append a Carriage Return and Line Feed (CRLF) to each message.

If this function is selected, the 9701 will append a Carriage Return (0Dh) and a Line Feed (0Ah) character to the end of each message, when it is sent to the printer. The result is that each message will be printed on a new line.

This is probably best for most applications, however, there are cases where this is undesirable. For example, you may wish to chain a number of messages together to form a report. In this case you would probably want the messages to follow one after the other, like sentences in a paragraph. To do this, select "No" at the "Append CRLF to message" prompt. Once this is done you must insert Carriage Return and Line Feed characters manually by embedding them in a message using the Printer Code delimiter (\).

6.2.2 Number of Lines Per Page

Another function that the 9701 provides to help automate the message printing process is the ability to insert a page break, after a number of lines have been printed.

If this function is selected, the user must provide the number of lines per page. The 9701 will count the number of 'Line Feed' characters (0Ah) that it transmits out the printer port. When the number of 'Line Feeds' equals the number of lines per page, the 9701 will insert a 'Form Feed' character (0Ch), immediately following the last 'Line Feed'. The next message that is printed will start at the top of the next page.

A 'Title message' may be inserted following the 'Form Feed' character. This will result in a title at the top of each page, as described in the next section.

If the number of lines per page is assigned to be zero, the automatic page break function will be disabled. If this is the case, page breaks will not be inserted.

6.2.3 Title Message

If the automatic page break function is selected, as described above, a 'Title message' may be inserted into the print transmission, following the 'Form Feed' character. This will result in a title at the top of each page.

The 'Title message' will be one of the 9701's 327 messages. It will be treated like any other message, except that it cannot contain embedded variables. It may, however, contain printer control codes.

The 'Title message' function will be disabled if a message number of zero is assigned. If message number zero is assigned as the 'Title message' number, the 'Title message' will not be printed.

6.3 More About the Auxiliary Port

The 9701's auxiliary port is an RS-232C compatible asynchronous serial communication port, with the following parameters possible:

Baud Rate:
19.2 k
9600
4800
2400
1200
300

Data Bits: 7 or 8

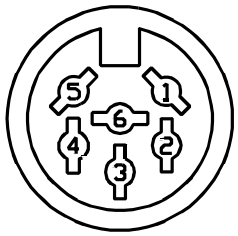
Stop Bits: 1 or 2

Parity Bits: Even, Odd, or None

Use the F8 key in the 9701's Message Editor to adjust these parameters.

6.3.1 Auxiliary Port Pin Assignments

The interface into the 9701's auxiliary port is via a 6 pin DIN connector. The pin assignments are as follows:

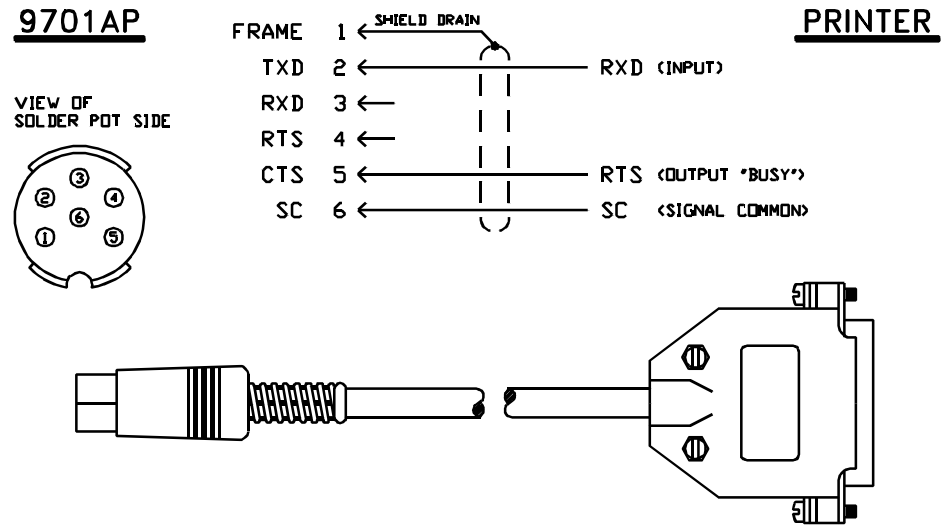


Pin numbers on 9701's Auxiliary Port (looking into female DIN connector on 9701).

Pin #	Description
1	Earth ground
2	Txd: Transmit data. RS-232C transmitter (output).
3	Rxd: Receive data. RS-232C receiver (input).
4	RTS: Request to send (output). The 9701 will make this line active (-10v) when it has data to send. It will go inactive only after the last byte of data has been completely transmitted.
5	CTS: Clear to send (input). The 9701 will not transmit data unless this line is active (-v). This will usually be connected to the 'busy' signal from the printer.
6	SC: Signal common

6.3.2 Typical Cable Configurations

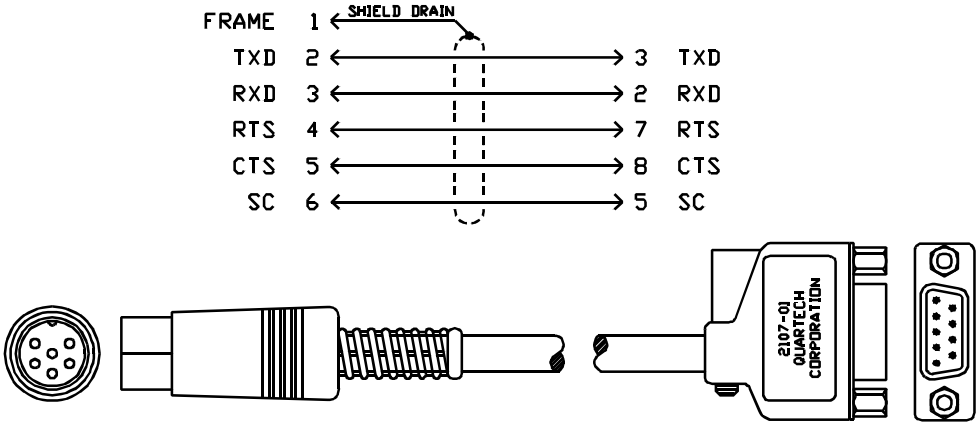
Serial Printer Cable:



Typical
cable from 9701AP to a serial printer.

Model 2107 Transition Cable:

One end of the 2107 cable is plugged into the 9701AP, the other end looks like the 9-pin serial port connector on an IBM compatible personal computer. Any cable that is designed for an IBM type serial port can be plugged into the 2107.



Mode
l 2107 transition cable. The 9-pin connector looks like an IBM PC serial port.

Appendix A: Error Messages and Troubleshooting

BASIC ASSURANCE TEST FAILURE

On power up, the 9701 will perform a basic assurance test of its RAM, ROM, and other hardware. If the display does not pass this test, it must be repaired or replaced.

CHECKSUM ERROR.

A checksum is calculated over each message individually. When a message is triggered, its checksum is verified. If the checksum is not valid, the message will be rejected, and this error message will be displayed instead. This problem must be corrected by editing or reloading the messages.

EEPROM CKSUM ERROR

A checksum is calculated over the entire EEPROM memory space. On power up, this checksum is verified. If the checksum is not valid, the 9701 will not continue. This problem must be corrected by editing or reloading the messages.

EEPROM WRITE FAILURE

The 9701 is not able to write to EEPROM memory. The unit must be repaired or replaced.

ILLEGAL TRIGGER ADDR ELEMENT NUMBER

The Integer Trigger Element does not exist in the current SLC 500 application program. The 9701's Integer Trigger Element number must be changed, or that element must be created inside the SLC 500.

ILLEGAL TRIGGER ADDR INTEGER FILE NUMBER

The Integer Trigger File number is not the number of an integer type file, or that file does not exist in the current SLC 500 application program. The 9701's Integer Trigger File number must be changed, or that file must be created as an integer type file.

ILLEGAL TRIGGER ADDR TRIGGER BIT FILE

The Trigger Bit File number is not the number of a bit type file, or that file does not exist in the current SLC 500 application program. The 9701's Trigger Bit File number must be changed, or that file must be created as a bit type file.

Another possibility is that the Trigger Bit File has too many elements. The maximum number of elements in any Trigger Bit File is 21 (i.e. elements 0 through 20).

ILL. VARIABLE: FILE

A variable definition within the given message has an illegal file number. That is, the file type and file number are inconsistent, or the file does not exist.

ILL. VARIABLE: ELEM

A variable definition within the given message has an illegal element number. The element does not exist.

MESSAGE NOT FOUND.

The given message was triggered, but it does not exist.

SYNTAX ERROR

A variable definition within the given message does not use the correct variable definition syntax. See Section 3: "Embedded Variables" for examples of proper syntax.

If the message does not use any variables, it is possible that a variable delimiter character was inadvertently used within the message. See Section 3.5 "Displaying Delimiter Characters", and Section 3.6 "Variable Definition Table" for a list of all variable delimiter characters.

TOO MANY VARIABLES

The given message contains too many variable definitions. The maximum number of variables within any message is 9.

Appendix B: Instalation

The 9701 is designed to be mounted in the door of an enclosure or on an operators console for ease of use. A template is provided to assist in the drilling and cutting of the mounting holes for the unit.

Care should be taken to protect the unit from metal chips and conductive particles. Failure to protect the unit may cause failure when power is applied and may void the warranty.

A minimum clearance of six inches should be kept between the unit and any other device that generates heat. In the event that the internal enclosure temperatures periodically exceeds 60 Deg. C (140 Deg. F), fans or a purge air system should be used to increase the air flow and eliminate "Hot Spots" that occur within the panel.

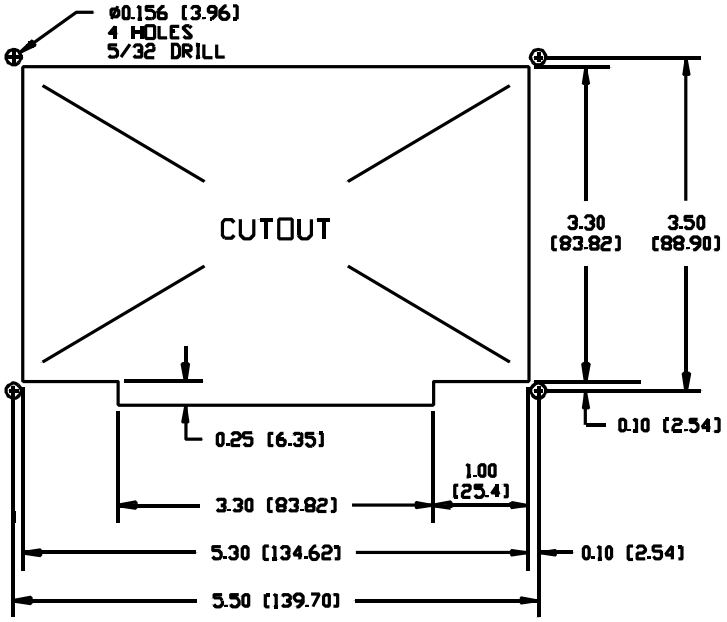
Electrical Requirements:

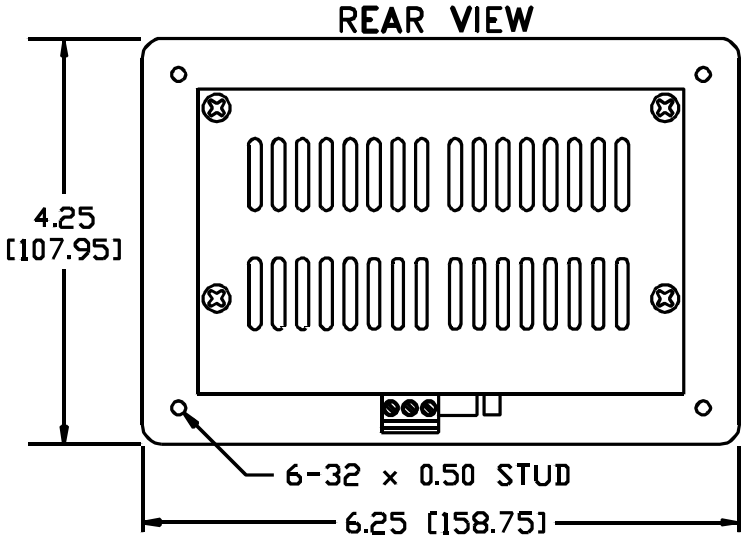
9701AP:	120 VAC, 10%, 60Hz, 1 Amp.
9701-240:	240 VAC, 10%, 60Hz, .5 Amp.
9701-12/24:	12 - 24 VDC, 1.5 Amp.

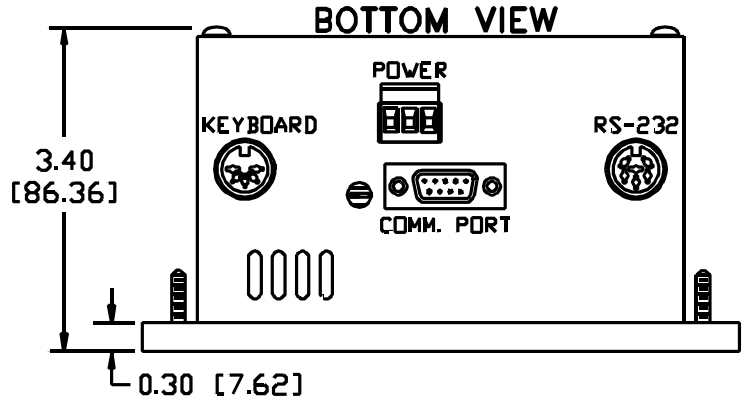
Wiring Considerations:

Care should be taken when routing the the communication cable. Follow these guidelines for a trouble free installation.

- **!** Keep the cables away from AC power lines. Keep the cables at least one foot from 120 VAC lines, and at least two feet away from higher voltage lines.
- **!** If the cables must cross AC power lines, cross them at right angles.
- **!** If you route the cables through conduit, the conduit should contain only other communication cables or low voltage DC signals. Do not run the cables in conduit that contains AC power lines.
- **!** Keep the cables away from sources of high energy fields such as arc welders, AC motors, motor starters, servo controllers, generators, induction heaters, and transformers.


ø0.156 [3.96]
4 HOLES
5/32 DRILL
CUTOUT
3.30
[83.82]
3.50
[88.90]
0.25 [6.35]
0.10 [2.54]
1.00
[25.4]
3.30 [83.82]
5.30 [134.62]
0.10 [2.54]
5.50 [139.70]


REAR VIEW
4.25
[107.95]
6-32 x 0.50 STUD
6.25 [158.75]


BOTTOM VIEW
POWER
KEYBOARD
RS-232
COMM. PORT
3.40
[86.36]
0.30 [7.62]