

Chaper 1: E10+ PLCs Host-Link Command Format

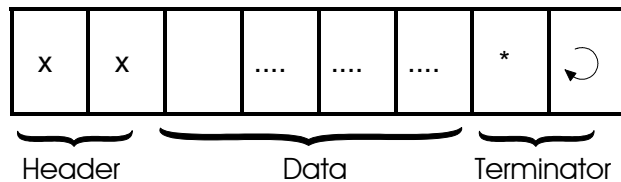
While an E10+ programmable logic controller is running, a host computer can send commands in the form of ASCII strings to the controller to read or write to the inputs, outputs, relays, timers and counters. These ASCII commands are known as the "host-link commands" and are to be serially transmitted via RS232C or RS485 port (if available) to and from the controller. The standard serial port settings for communication are: 9600 baud, 8 data bit, 1 stop bit, no parity. Future models of E10+ PLC may allow the serial port to be configured to another baud rate using the "BW" command described in the next Chapter.

All E10+ PLCs support both point-to-point (one-to-one) and multi-point (one-to-many) communication protocols and hardware. Each protocol has a different command structure. This chapter describex the format of these two protocols. The actual command lists are decribed in details in Chapter 2.

1. Point-to-point Communication

In a point-to-point communication system, the host computer's RS232C serial port is connected to the PLC. At any one time, only one controller may be connected to the host computer. The host-link commands do not need to specify any controller ID code and are therefore of simpler format, as shown below:

Command/Response Block Format (Point to Point)



Each command block starts with a two-byte ASCII character header, followed by a number of ASCII data and ends with a terminator which comprises an '*' character and a carriage return (ASCII value = 13₁₀). The purpose of the command is denoted by the header, e.g. RI for Read Input, WO for Write Output, etc. The data are usually the hexadecimal representation of numeric data. Each byte of binary data is represented by two ASCII characters (00 to FF).

To begin a communication session, the host computer must first send one byte of ASCII character: Ctrl-E (=05Hex) via its serial port to the controller. This informs the controller that the host computer wishes to send a (point-to-point) host-link command to it. Thereafter, the host computer must wait to receive an echo of the Ctrl-E character from the controller. Reception of the echoed Ctrl-E character indicates that the controller is ready to respond to the command from the host computer. At this moment, the host computer must immediately send the *command block* to the controller and then wait to receive the *response block* from the controller. The entire communication session is depicted in Figure 2-1.

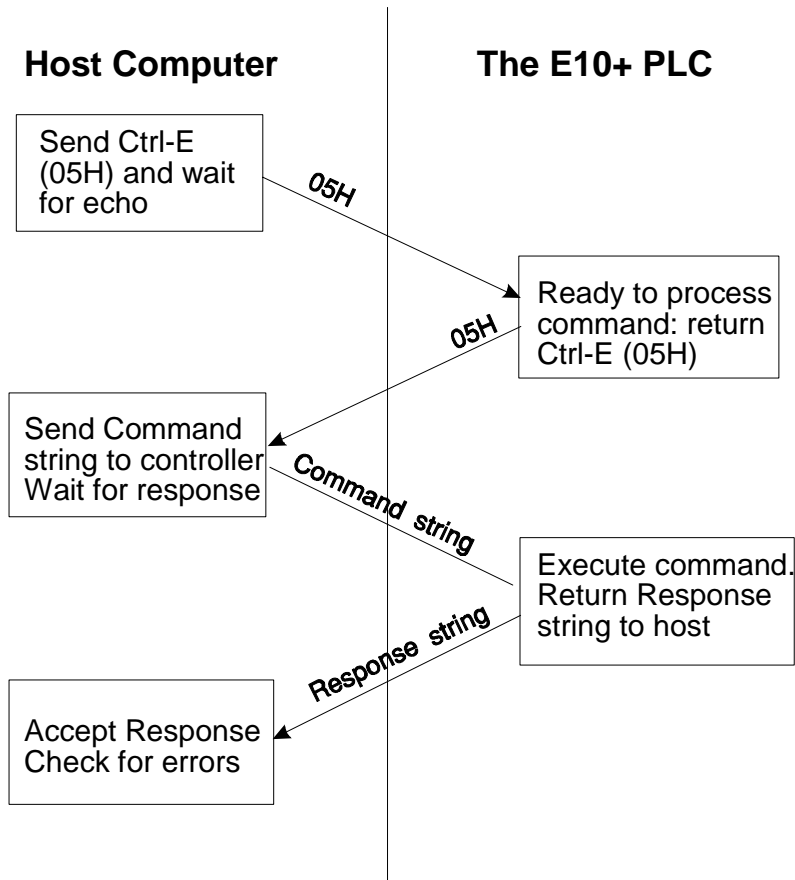
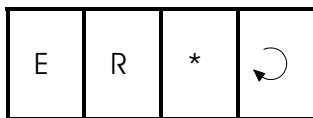


Figure 2-1

After the controller has received the command, it will send a response block back to the host computer and this completes the communication session. If the command is accepted by the controller, the response block will start with the same header as the command, followed by whatever information that is requested by the command and the terminator.

If an unknown command is received or if the command is illegal (such as access to an unavailable output or relay channel), the following **error response** will be received:

Error Response Format



The host computer program should always check the returned response for possibilities of errors in the command and take necessary actions.

2. Multipoint Communication system

2.1 Overview

E10+ PLCs also support the multi-point communication protocols found in their bigger siblings, namely the H and the M-series PLCs. Multi-point communication protocol has the advantage that error checks are included in the message to reduce the possibility of wrong actions being carried out due to corrupted messages. Future design of E10+ PLCs will include built-in RS485 interface which will allow multiple E10+ PLCs to be networked to a host PC or to a master M-series PLC.

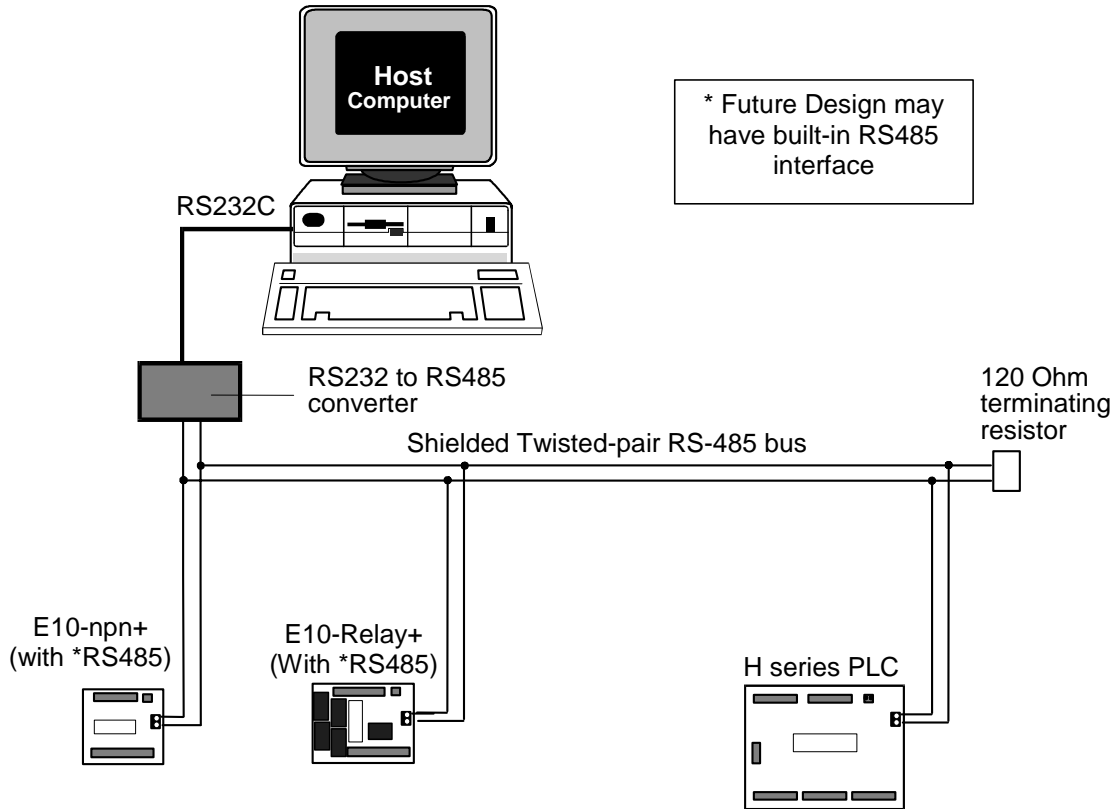
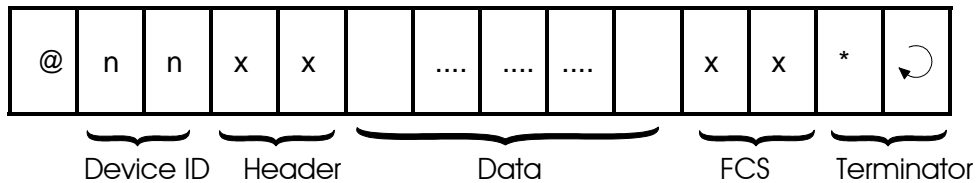


Figure 2-3 - A Multi-drop RS485 network

2.2 Command/Response Block Format (Multipoint)



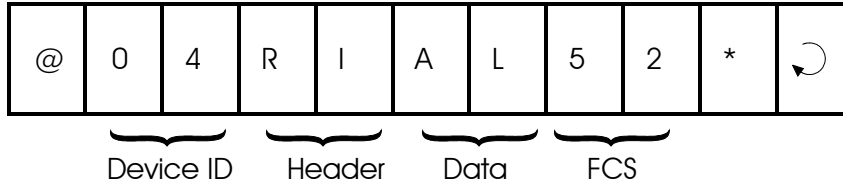
Each command block starts with the character "@" and two-byte hexadecimal representation of the controller's ID (00 to FF), and ends with a two-byte "Frame Check Sequence" (FCS) and the terminator. FCS is provided for detecting communication errors in the serial bit-stream. If desired, the command block

Chapter 1 – E10+ PLC Host-Link Command Format

may omit calculating the FCS simply by putting the characters "00" in place of the FCS.

Calculation of FCS

The FCS is an 8-bit data represented by two ASCII characters (00 to FF). It is a result of Exclusive OR sequentially performed on each character in the block, starting from @ in the device number to the last character in the data. An example is as follow:



@	0100 0000	
	XOR	
0	0011 0000	
	XOR	
4	0011 0100	
	XOR	
R	0101 0010	
	XOR	
I	0100 1001	
	XOR	
A	0100 0001	
	XOR	
L	0100 1100	
	XOR	
<hr/>		
	0101 0010	= 52 ₁₆
<hr/>		

Value 52₁₆ is then converted to ASCII characters '5' (0011 0101) and '2' (0011 0010) and placed in the FCS field.

FCS calculation program example

The following C function will compute and return the FCS for the "string" passed to it.

```
unsigned char compute_FCS(unsigned char *string)
{
    unsigned char result;
    result = *string++; /*first byte of string*/
    while (*string)
        result ^= *string++; /* XOR operation */
    return (result);
}
```

2.3 Communication Procedure

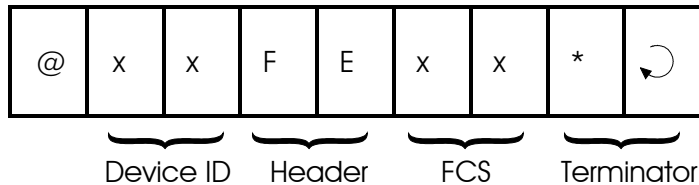
Unlike the point-to-point communication protocol, the host computer must **NOT** send the CTRL-E character before sending the command block. After the host computer has sent out the multi-point host link command block, only the controller with the correct device ID will respond. Hence it is essential to ensure that every controller on the RS485 network assumes a different ID. Otherwise contention may occur (i.e. two controllers simultaneously sending data on the receiver bus, resulting in garbage data being received by the host). On the other hand, if none of the controller IDs match that specified in the command block, then the host computer will receive no response at all.

The PLC automatically recognizes the type of command protocols (point-to-point or multi-point) sent by the host computer and it will respond accordingly. If a multi-point command is accepted by the controller, the response block will start with a character '@', followed by its device ID and the same header as the command, then the data requested by the command, a response block FCS and the terminator.

Framing Errors

When the controller receives a multi-point host link command block, it computes the FCS of the command and compares it with the FCS field received in the command block. If the two do not match, then a "framing error" has occurred. The controller will send the following Framing Error Response to the host:

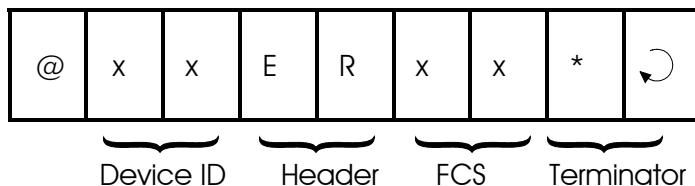
Framing Error Response Block (Multi-point only)



Command Errors

If an unknown command is received or if the command is illegal (such as an attempt access an unavailable output channel), the following **error response** will be received:

Error Response Format



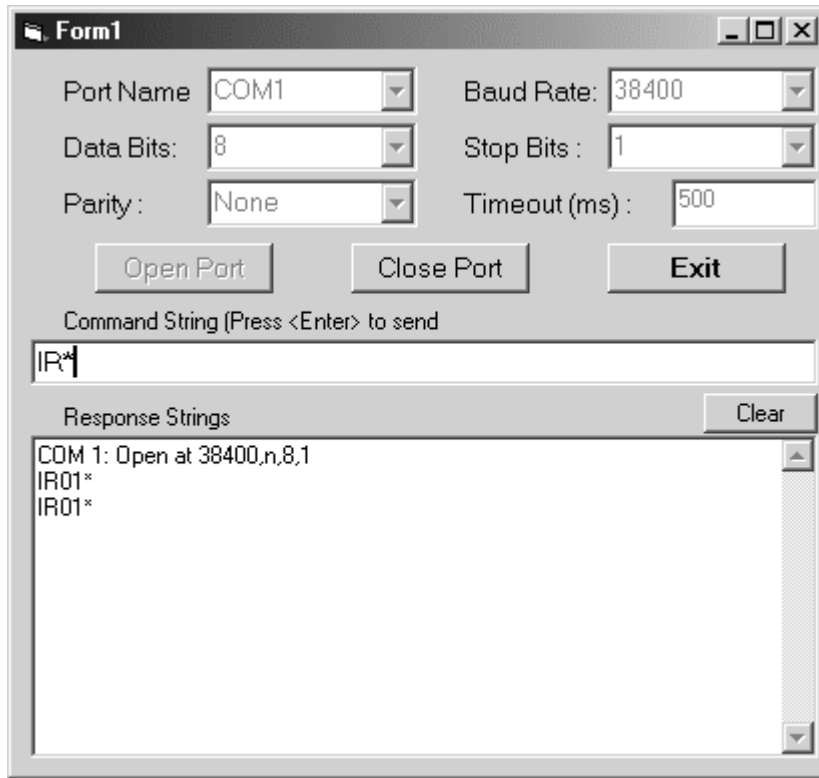
Chapter 1 – E10+ PLC Host-Link Command Format

The host computer program should always check the returned response for possibilities of errors in the command and take necessary action.

3. Sample Visual Basic Host Communication Program

To help you get started, a sample Visual Basic 6 sample program that runs under Windows operating system can be downloaded from the web page:

<http://www.tri-plc.com/applications/VBsample.htm>



The screenshot shows a Visual Basic 6.0 application window titled "Form1". It features several dropdown menus and text boxes for configuring serial communication parameters. The "Port Name" is set to "COM1", "Baud Rate" is "38400", "Data Bits" is "8", "Stop Bits" is "1", "Parity" is "None", and "Timeout (ms)" is "500". Below these are three buttons: "Open Port", "Close Port", and "Exit". A text box labeled "Command String (Press <Enter> to send)" contains the text "IR*". Below this is a "Response Strings" area with a "Clear" button. The response area displays the following text: "COM 1: Open at 38400,n,8,1", "IR01*", and "IR01*".

The above sample program is created using Visual Basic 6.0 that supports the MSComm controls. This program lets you define the serial port, baud rate and comm format. It will then take your command string and send it to the PLC and obtain a response string from the PLC and display on the response box. It takes care of time-out (user-definable) and will not block the program if the PLC is not present. We have tested it on a H-series as well as M-series PLC and it works well. Please feel free to email to support@tri-plc.com if you have any feedback or suggestions.

Chapter 2 Host Link Command/Response

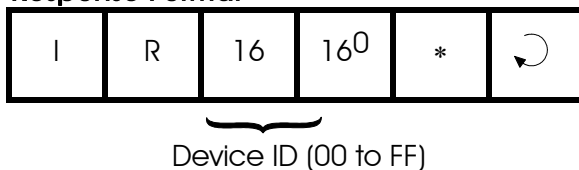
This chapter describes the detailed formats of the command and response blocks for the entire E10+ host-link commands. Only the formats for the point-to-point communication system are presented, but all these commands are available to the multi-point system as well. To use a command for multi-point system, simply add the device ID (@nn) before the command header and the FCS at the end of the data (See Section 2 for detailed description of multipoint communication command format).

1. Device ID Read

Command Format



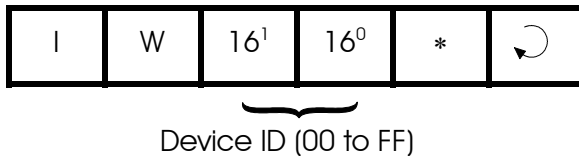
Response Format



The device ID is to be used for multi-point communication protocol where the host computer can selectively communicate with any controller connected to a common RS485 bus (see Chapter 1 for details). The ID has no effect for point-to-point communication. The device ID is stored in the PLC's EEPROM and therefore will remain with the controller until it is next changed.

2. Device ID Write

Command Format

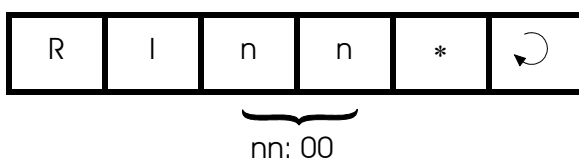


Response Format

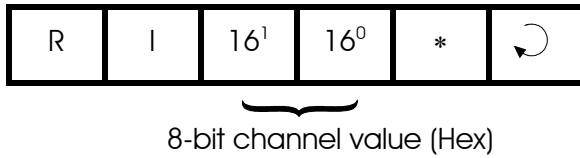


3. Read Input Channel

Command Format



Response Format

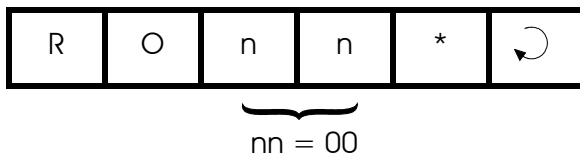


This command allows you to read 8 digital inputs (called a channel) in a single command. The 8-bit inputs of each channel is represented by two bytes ASCII text expression of its hexadecimal value. E.g. if inputs 1 to 3 are OFF and inputs 4 to 8 are ON, then CH0=11111000, which is represented as ASCII characters 'F' and '8'.

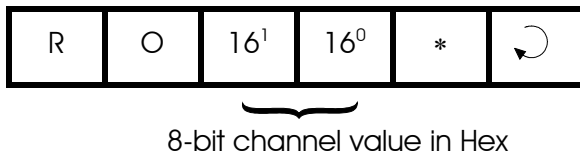
Since E10+ only support 8 digital inputs, there will only be one channel which is = 00. So only "RI00*" is supported by the E10+ PLC.

4. Read Output Channel

Command Format



Response Format

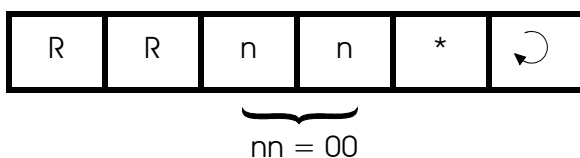


This command allows you to read 8 digital outputs (called a channel) in a single command. The 8-bit outputs of each channel is represented by two bytes ASCII text expression of its hexadecimal value. For example: if Outputs 1 to 3 are ON and Outputs 4 to 8 are OFF, then CH00=00000111₂, which is represented as ASCII characters '0' and '7'.

Since E10+ only support 6 digital outputs (4 physical and two internal memory bits, which can be used as additional internal relays), there will only be one channel, which is = 00. So only "RO00*" is supported by the E10+ PLC. In addition, the two highest bits (for non existent output 7 and 8) in the response data will always be 0.

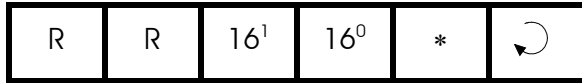
5. Read Relay Channel

Command Format



Response Format

Chapter 2 – Host Link Commands & Responses



8-bit channel value in Hex

This command allows you to read 8 internal relays (called a channel) in a single command. The 8-bit relay of each channel is represented by two bytes ASCII text expression of its hexadecimal value.

Since E10+ only support 8 internal relays, there will only be one channel, which is = 00. So only "RR00*" is supported by the E10+ PLC.

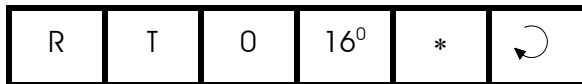
6. Read Timer Contact

Command Format



nn = 00

Response Format



4-bit channel value in Hex

This command allows you to read all four E10+'s timer contacts in a single command. Since the response data contains two bytes ASCII text expression of its hexadecimal value, the response string will always be RT0x, where x is from '0' to 'F'.

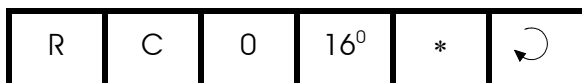
7. Read Counter Contact

Command Format



nn = 00

Response Format



4-bit channel value in Hex

This command allows you to read all four E10+'s counter contacts in a single command. Since the response data contains two bytes ASCII text expression of its hexadecimal value, the response string will always be RC0x, where x is from '0' to 'F'.

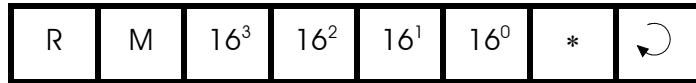
8. Read Timer Present Value (P.V.)

Command Format



nn: Timer1=00, Timer4=03

Response Format



Timer present value in Hexadecimal

You can read the Present Value (P.V.) of any of the four E10+ timers with this command. The response data contains 4-byte ASCII text expression of the timer's P.V. in hexadecimal format.

Note: The response data format differ from that of the H-series and M-series data which returns the data for RM command in decimal format. However, to keep coding simple, E10+ designer decided to represent the P.V. in hexadecimal value. An inactive timer will return the value "FFFF".

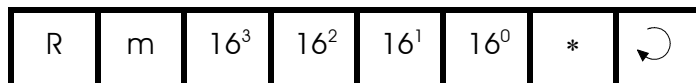
9. Read Timer Set Value (S.V.)

Command Format (Note: it is a lower case 'm')



nn: Timer1=00, Timer4=03

Response Format



Timer Set Value in Hexadecimal

Chapter 2 – Host Link Commands & Responses

You can read the Set Value (S.V.) of any of the four E10+ timers with this command. The response data contains 4-byte ASCII text expression of the timer's S.V. in hexadecimal format. The Timer's S.V.s are normally programmed into the PLC's EEPROM during program transfer, but may be altered by the Wm command described later.

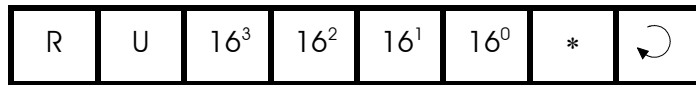
10. Read Counter Present Value

Command Format



nn: Counter =00, Timer4=03

Response Format



Counter present value in Hexadecimal

You can read the Present Value (P.V.) of any of the four E10+ counters with this command. The response data contains 4-byte ASCII text expression of the counter's P.V. in hexadecimal format.

Note: The response data format differ from that of the H-series and M-series data which returns the data for RU command in decimal format. However, to keep coding simple, E10+ designer decided to represent the P.V. in hexadecimal value. An inactive counter will return the value "FFFF".

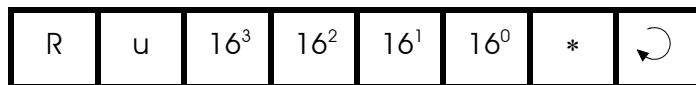
11. Read Counter Set Value (S.V.)

Command Format (Note: it is a lower case 'u')



nn: Counter 1=00, Counter r4=03

Response Format



Counter Set Value in Hexadecimal

Note: 'u' is lower case.

Chapter 2 – Host Link Commands & Responses

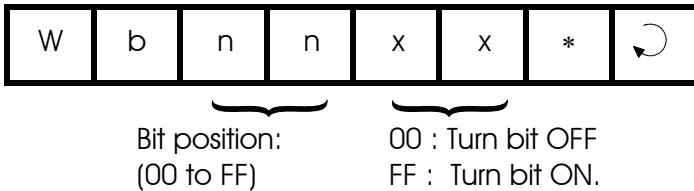
You can read the Set Value (S.V.) of any of the four E10+ counters with this command. The response data contains 4-byte ASCII text expression of the counter's S.V. in hexadecimal format. Counters' S.V.s are normally programmed into the PLC's EEPROM during program transfer, but may be altered by the Wu command described later.

12. Write Single I/O Bit

Most host link commands handle the I/O bits in groups of eight (called a 'channel') which is more efficient since each communication command read or write to all 8 I/O bits in the same channel. However, there are situation where you wish to change only a **single** input, output or relay bit without affecting the rest. Using a word oriented host link command means that you would have to read the whole channel, alter the desired single bit and then write back the channel to the PLC. This can result in two communication message exchanges just to affect one bit, which is inefficient.

The "Wb" command is perfectly suited for handling such a situation. The E10+ PLC's inputs, outputs, relays, timers and counters are all assigned bit position in the 8-bit address 00 to FF and the "Wb" command is then used to turn ON or turn OFF the desired bit.

Command Format (note: lower case 'b' here)



Response Format

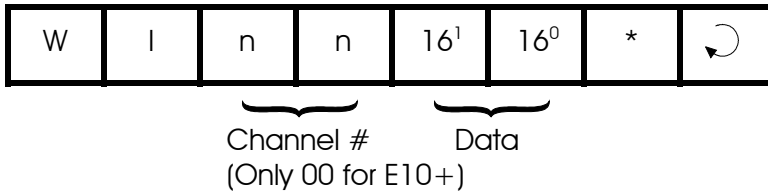


I/O Type	Bit Position nn - (Hex)
Input 1 to 8	00 to 07
Output 1 to 6	08 to 0D
Counter contact 1 to 4	10 to 13
Timer contact 1 to 4	14 to 17
Relay 1 to 8	18 to 1F

E.g. To turn ON output #4, send command "Wb0BFF*" to the PLC.
To turn OFF relay #1, send command "Wb1800*" to the PLC.

13. Write Input Channel

Command Format

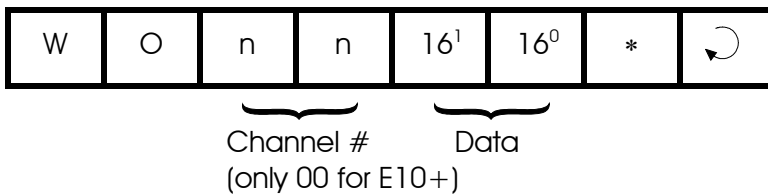


Response Format



14. Write Output Channel

Command Format

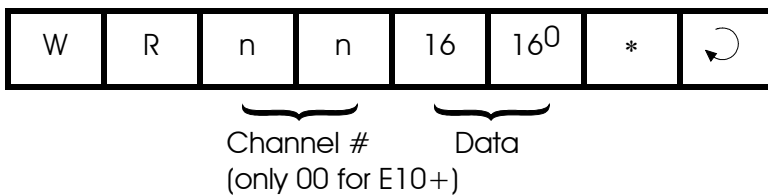


Response Format

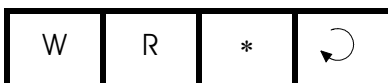


15. Write Relay Channel

Command Format

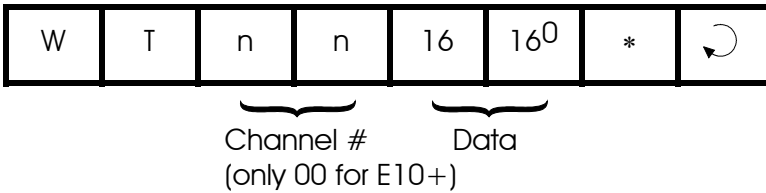


Response Format



16. Write Timer Contacts

Command Format

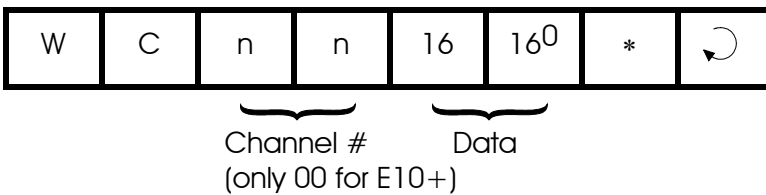


Response Format



17. Write Counter Contacts

Command Format

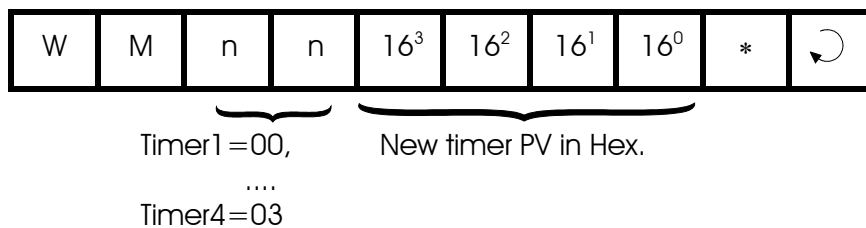


Response Format



18. Write Timer Present Value

Command Format

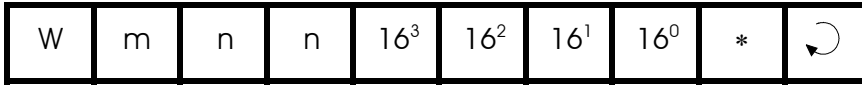


Response Format



19. Write Timer Set Value

Command Format



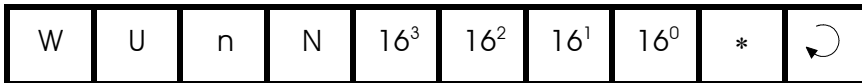
Timer1 =00, New timer S.V. in Hex.
.....
Timer4 =03

Response Format



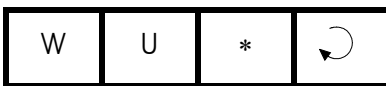
20. Write Counter Present Value

Command Format



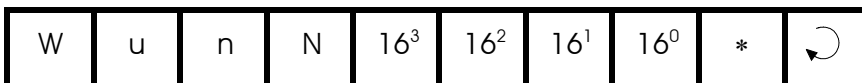
Counter 1 =00, New counter P.V. in Hex.
.....
Counter 4 =03

Response Format



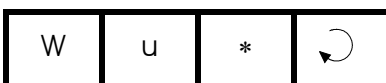
21. Write Counter Set Value

Command Format



counter 1 =00, New counter S.V. in Hex.
.....
counter 4 =03

Response Format

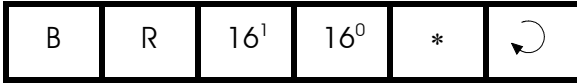


22. Baud Rate Read

Command Format



Response Format

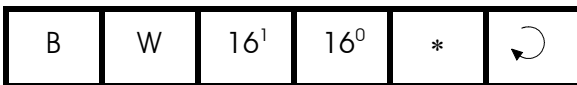



 Baud Rate No. (00 to 06)

Please see BW command for the definition of the baud rate number.

23. Baud Rate Write

Command Format




 Baud Rate No. (00 to 06)

Response Format



Future design of the E10+ PLC may be configurable to communicate at baud rate other than the standard 9600 bits per second. The BW command is used to define the desired baud rate. Each Baud Rate No (00 to 06) corresponds to a particular baud rate, as follow:

Baud Rate No.	Transmission Speed
0	1200 bps
1	2400 bps
2	4800 bps
3	9600 bps
4	19,200bps
5	31,500 bps
6	38,400 bps

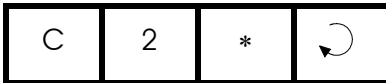
E10+ PLC **may not** support all the baud rate listed above. Please read the PLC's installation guide to find out whether your PLC supports baud rate number other than the standard "03" (which corresponds to 9600 bps). The PLC will send the error response "ER*" if you attempt to use an unsupported baud rate number.

Chapter 2 – Host Link Commands & Responses

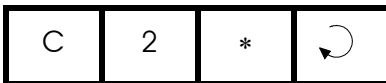
The baud rate number is stored in the PLC's EEPROM until it is overwritten by the "BW" command again. The PLC will assume the new baud rate only after a power-on reset.

24. Halting the PLC

Command Format



Response Format



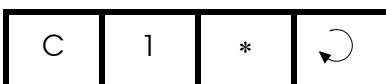
When the PLC receives this command, it temporarily halts the execution of the PLC's ladder program. The PLC continues to process host link commands sent to it until it receives a resume command.

25. Resume PLC Operation

Command Format



Response Format



When the PLC receives this command, it will resume execution of the ladder program if it has been halted previously by the "C2" command. Otherwise, this command has no effect.

26. Applications of Host Communication Capabilities

The E10+ PLC's host-link capability gives rise to a number of possibilities of using the controller in tandem with a host computer for control jobs that require a higher level of intelligence for decision making.

Since the host computer only needs one serial port to interface with the PLC, an inexpensive laptop, palmtop or notebook computer can be used both as a programming console as well as a high level decision maker for controlling the target machine.

Chapter 2 – Host Link Commands & Responses

The following are some possibilities and suggestions for using the host communication ability of the H-series controller:

i) **Combining ladder logic program and host computer processing**

Use ladder logic program to perform all the tedious, repetitive logic control tasks which often pose headaches to high-level language programmers. If a host computer decision is required to execute a certain logic sequence, then use any internal relay as the triggering input. The high level language program running on a host computer can then trigger the desired logic sequence by turning on or off the corresponding relay.

The ladder logic program can use any of its outputs or relays as a flag to indicate to the host processor that a certain logic state has been reached and requires the host computer's attention. The host computer will periodically monitor the corresponding flag(s) to make further program decisions.

ii) **Using the PLC as a remote I/O interface**

If you transfer a blank program to the PLC, all the inputs and outputs can be directly controlled by the host computer. The PLC thus effectively becomes a slave I/O processor, accepting only commands from the host computer and then reading from or writing to the corresponding input and output channels as specified in the command. While this may seem like a waste of the logic processing power of the CPU, it may be useful if the application needs the host computer to directly control all the inputs and the outputs.

iii) **I/O sharing: host computer utilizing the I/Os not used by ladder logic program**

The host computer may directly turn on or turn off any of the output points which are not controlled by the ladder logic program (i.e. these outputs do not appear as "coil symbol" in TRiLOGI). Thus it is possible for a host computer to make use of the unused inputs and outputs of the PLC for its own control purposes without affecting the underlying ladder logic program that is currently running.

Caution: If the host computer attempts to control an output that is already used by the ladder logic program and a conflict in the output logic state occurs, the output will change its state, but only momentarily and will return to the value decided by the outcome of the ladder logic program in the next scan time. This might result in a short pulse of width less than 1 scan time being sent to the output terminal.

If this is undesirable, then it is the responsibility of the host computer programmer to prevent this from happening. Since the host computer can only read or write to the controller in terms of an 8-bit channel at a time, bit-masking technique is needed if a channel of output is to be shared by the ladder logic program as well as the host computer.