Section 3 Command/Response Format

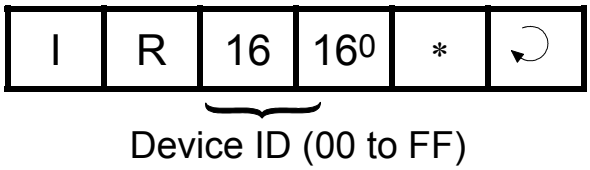
This section describes the detailed formats of the command and response blocks for the entire H-series PLC host-link commands. Only the formats for the point-to-point communication system are presented, but all these commands are available to the multi-point system as well. To use a command for multi-point 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).

Since different models of PLCs have different maximum number of inputs, outputs, internal relays, timers and counters, attempts to send commands to an invalid address will result in an error response from the controller. For commands that read ALL the channels such as "RIAL", the response string will contain up to the maximum number of channels available on that PLC. The maximum channel numbers stated in the subsequent parts of this manual is based on the ***assumption*** that the PLC has 96 inputs, 64 outputs, 256 internal relays, 64 timers and 64 counters. The actual limits on your PLC can be observed during the transfer of your TRiLOGI program into the controller. You should work with the actual limit of the PLC model you are using and not what's stated in the following sections.

1. Device ID Read

Command Format

Response Format

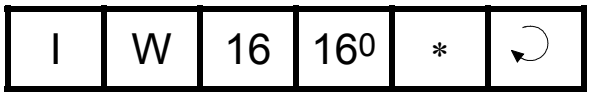


The device ID is to be used for multi-point communication protocol where the host computer can selectively communicate with any controller connected to a common RS485 bus (see Section 2 for details). The ID has no effect for point-to-point communication.

The device ID is stored in the PLC's EEPROM and therefore will remain with the controller until it is next changed.

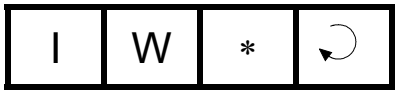
2. Device ID Write

Command Format



Device ID (00 to FF)

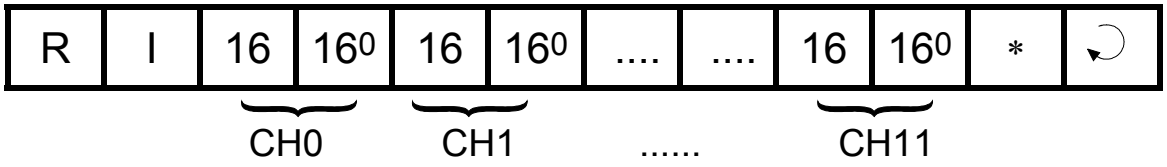
Response Format



3. Read All Inputs

Command Format

Response Format



The data part of the response string contains the hexadecimal values of all the input channels from channel #0 to the maximum channel#. The 8-bit inputs of each channel is represented by two bytes ASCII text expression of its hexadecimal value. For example: if inputs 1 to 3 are logic '0's, inputs 4 to 8 are logic '1's and all other inputs are logic '0's, then CH0=1111 1000, which is represented as ASCII characters 'F' and '8'.

Definition of Input Channels

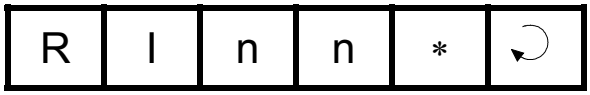
The following table shows the input numbers as defined in TRiLOGI's Input entry table corresponding to the input channel number.

	Bit7			Input Numbers				Bit0
CH0:	8	7	6	5	4	3	2	1
CH1:	16	15	14	13	12	11	10	9
CH2:	24	23	22	21	20	19	18	17
CH3:	32	31	30	29	28	27	26	25
CH4:	40	39	38	37	36	35	34	33
CH5:	48	57	56	45	44	43	42	41
CH6:	56	55	54	53	52	51	50	49

CH7:	64	63	62	61	60	59	58	57
CH8:	72	71	70	69	68	67	66	65
CH9:	80	79	78	77	76	75	74	73
CH10:	88	87	86	85	84	83	82	81
CH11:	96	95	94	93	92	91	90	89

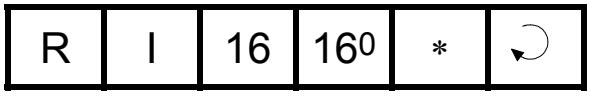
4. Read Input Channels

Command Format



nn: CH0=00, CH1=01, ... CH11=0B

Response Format

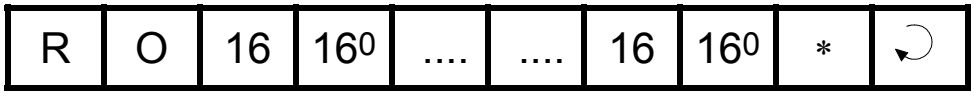


8-bit channel value (Hex)

5. Read All Outputs

Command Format

Response Format



CH0 CH7

Definition of Output Channels

The following table shows the physical output number corresponding to the output channels 0 and 1.

	bit7		Output numbers					bit0
CH0:	8	7	6	5	4	3	2	1
CH1:	16	15	14	13	12	11	10	9
CH2:	24	23	22	21	20	19	18	17
CH3:	32	31	30	29	28	27	26	25
CH4:	40	39	38	37	36	35	34	33
CH5:	48	57	56	45	44	43	42	41
CH6:	56	55	54	53	52	51	50	49
CH7:	64	63	62	61	60	59	58	57

6. Read Output Channels

Command Format

R	O	n	n	*	↵

nn: CH0=00, CH1=01, CH7=07

Response Format

R	O	16	16^0	*	↵

8-bit channel value in Hex

7. Read All Relays

Command Format

R	R	A	L	*	↵

Response Format

R	R	16	16^0	16	16^0	....	....	16	16^0	*	↵

CH0 CH1 ..... CH31

Definition of Relay Channels

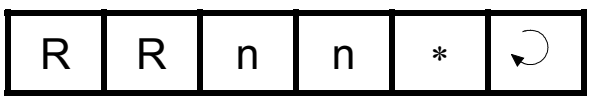
The table shows the relay numbers defined in TRiLOGI's Relay entry table and their corresponding channel numbers.

	bit7		Relay numbers					bit0

CH0:	8	7	6	5	4	3	2	1
CH1:	16	15	14	13	12	11	10	9
CH2:	24	23	22	21	20	19	18	17
CH3:	32	31	30	29	28	27	26	25
CH4:	40	39	38	37	36	35	34	33
CH5:	48	57	56	45	44	43	42	41
CH6:	56	55	54	53	52	51	50	49
CH7:	64	63	62	61	60	59	58	57
CH8:	72	71	70	69	68	67	66	65
CH9:	80	79	78	77	76	75	74	73
CH10:	88	87	86	85	84	83	82	81
CH11:	96	95	94	93	92	91	90	89
CH12:	104	103	102	101	100	99	98	97
CH13:	112	111	110	109	108	107	106	105
CH14:	120	119	118	117	116	115	114	113
CH15:	128	127	126	125	124	123	122	121
CH16:	136	135	134	133	132	131	130	129
CH17:	144	143	142	141	140	139	138	137
CH18:	152	151	150	149	148	147	146	145
CH19:	160	159	158	157	156	155	154	153
CH20:	168	167	166	165	164	163	162	161
CH21:	176	175	174	173	172	171	170	169
CH22:	184	183	182	181	180	179	178	177
CH23:	192	191	190	189	188	187	186	185
CH24:	200	199	198	197	196	195	194	193
CH25:	208	207	206	205	204	203	202	201
CH26:	216	215	214	213	212	211	210	209
CH27:	224	223	222	221	220	219	218	217
CH28:	232	231	230	229	228	227	226	225
CH29:	240	239	238	237	236	235	234	233
CH30:	248	247	246	245	244	243	242	241
CH31:	256	255	254	253	252	251	250	249

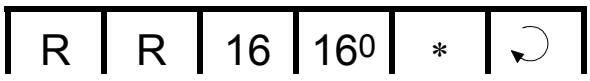
8. Read Relay Channels

Command Format



nn: CH0=00, CH1=01..... CH31=1F

Response Format



8-bit channel value in Hex

9. Read All Timer Contacts

Command Format

R	T	A	L	*	↵

Response Format

R	T	16	16^0	....	....	16	16^0	*	↵

CH0 CH7

If an energized timer has timed-out, its contact (completion flag) will be read as a "1", otherwise it is read as a "0".

Definition of Timer Channels

The following table shows the timer number defined in TRiLOGI's Timer entry table and their corresponding channel numbers.

	bit7			Timer numbers				bit0
CH0:	8	7	6	5	4	3	2	1
CH1:	16	15	14	13	12	11	10	9
CH2:	24	23	22	21	20	19	18	17
CH3:	32	31	30	29	28	27	26	25
CH4:	40	39	38	37	36	35	34	33
CH5:	48	57	56	45	44	43	42	41
CH6:	56	55	54	53	52	51	50	49
CH7:	64	63	62	61	60	59	58	57

10. Read Timer Contacts

Command Format

R	T	n	n	*	↵

nn: CH0=00, CH1=01, CH2=02

Response Format

R	T	16	16^0	*	↵

8-bit channel value in Hex

11. Read All Counter Contacts

Command Format

R	C	A	L	*	↵

Response Format

R	C	16	16^0	....	....	16	16^0	*	↵

CH0 CH7

If a counter has counted down to zero, its contact will be read as a "1", otherwise it is read as a "0".

Definition of Counter Channels

The following table shows the counter numbers defined in TRiLOGI's Counter entry table and their corresponding channel numbers.

CH0:	8	7	6	5	4	3	2	1
CH1:	16	15	14	13	12	11	10	9
CH2:	24	23	22	21	20	19	18	17
CH3:	32	31	30	29	28	27	26	25
CH4:	40	39	38	37	36	35	34	33
CH5:	48	57	56	45	44	43	42	41
CH6:	56	55	54	53	52	51	50	49
CH7:	64	63	62	61	60	59	58	57

12. Read Counter Contacts

Command Format

R	C	n	n	*	↵

nn: CH0=00, CH1=01, CH2=02

Response Format

R	C	16	16^0	*	↵

8-bit channel value in Hex

13. Read All Timer Present Values

Command Format

R	M	A	L	*	↵

Response Format

R	M	10^3	10^2	10^1	10^0	....	....	10^3	10^2	10^1	10^0	*	↵

Timer 1 ..... Timer 64

The present values of all the timers are returned in decimal form as four- byte ASCII text characters from 0000 to 9999.

14. Read Timer Present Value

Command Format

R	M	n	n	*	↵

nn: Timer1=00, ... Timer16=0F.... Timer64=3F

Response Format

R	M	10^3	10^2	10^1	10^0	*	↵

Timer present value in Decimal

15. Read All Counter Present Values

Command Format

R	U	A	L	*	↵

Response Format

R	U	10^3	10^2	10^1	10^0	....	....	10^3	10^2	10^1	10^0	*	↵

Counter 1 ..... Counter 64

The present values of all the counters are returned in decimal form as four byte ASCII text characters from 0000 to 9999.

16. Read Counter Present Value

Command Format

R	U	n	n	*	↵

nn: Counter1=00, ... Counter16=0F.... Counter64=3F

Response Format

R	U	10^3	10^2	10^1	10^0	*	↵

Counter present value in Decimal

17. Write Inputs

Command Format

W	I	n	n	16	16^0	*	↵

Channel # (00 to 0B)

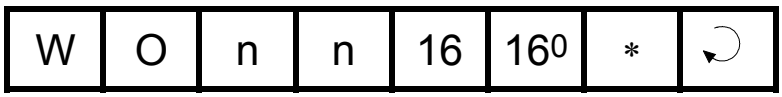
Data

Response Format

W	I	*	↵

18. Write Outputs

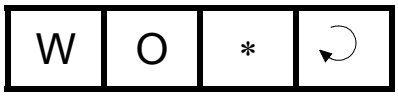
Command Format



Channel # (00 to 08)

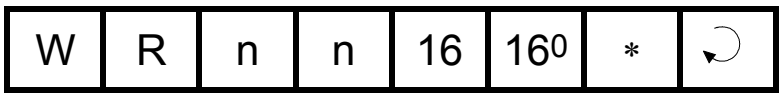
Data

Response Format



19. Write Relay Channel

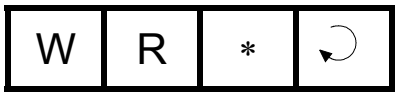
Command Format



Channel # (00 to 0F)

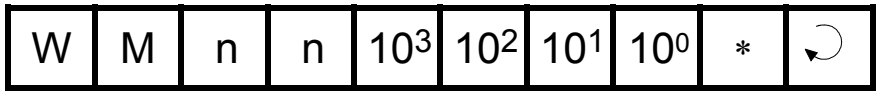
Data

Response Format



20. Write Timer Present Value

Command Format



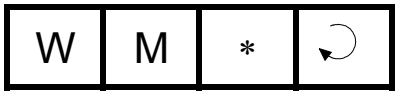
Timer1=00,

....

Timer64=3F (Hex)

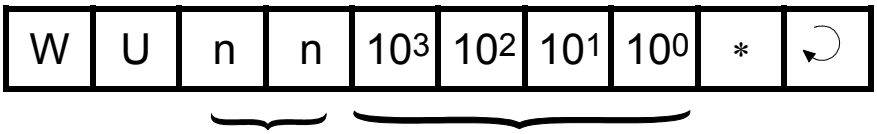
New timer PV

Response Format



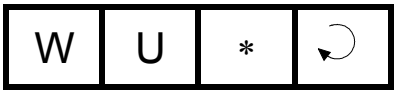
21. Write Counter Present Value

Command Format



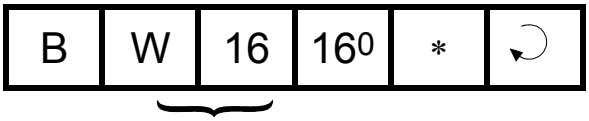
Counter1=00, New PV
....
Counter64=3F (Hex)

Response Format



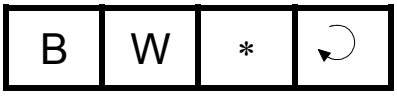
22. Baud Rate Write

Command Format



Baud Rate No. (00 to 06)

Response Format



Some of the H-series PLC can be configured to communicate at baud rate of between 1200 and 38,400 baud. The BW command is used to defined 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

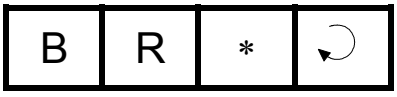
The baud rate number is stored in the PLC's EEPROM until it is overwritten by the "BW" command again.

Note that for T20H, T28H and T44H controller which has only one serial port, the new baud rate will only take effect if their respective DIP switch SW1-3 is set to the "ON" position when the PLC is first turned ON. Otherwise, the PLC is automatically configured to communicate at 9600 baud. If it is desirable to temporarily use the standard 9600 baud rate, you may turn OFF DIP switch SW1-3 and then reset the processor.

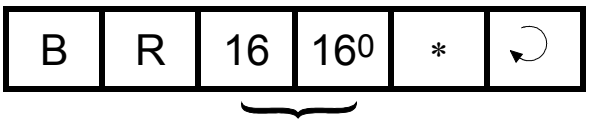
For PLCs with two serial ports such as the T64H-Relay, the RS232C port is always fixed to communicate at 9600 baud. The "BW" command defines the baud rate setting for the RS485 port. This ensure that you can always communicate with the PLC via the RS232C port, regardless of the baud rate setting of the RS485 port.

23. Baud Rate Read

Command Format



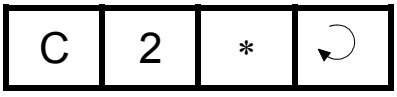
Response Format



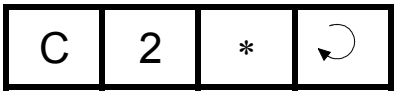
Baud Rate No. (00 to 06)

24. Halting the PLC

Command Format



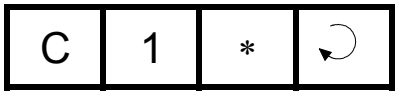
Response Format



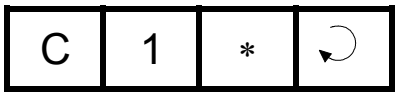
When the PLC receives this command, it temporarily halts the execution of the PLC's ladder program. The PLC continues to process host link commands sent to it until it receives a resume command.

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. Host Communication Program Examples

Two sample programs, one written in Microsoft GWBASIC (HOSTCOMM.bas) and the other written in Borland International's Turbo C (HOSTCOMM.C), are provided in the TRiLOGI distribution diskette to help programmers get started. Both programs essentially perform the same functions, as follows:

(a) Prompt user to enter the desired command block via the PC's keyboard.

(b) Initiate a communication session (for point-to-point protocol only) and send the command string to the controller.

(c) Wait to receive the response block from PLC and display the response block on the PC's screen.

These two programs incorporate all the codes needed to communicate successfully with the T20/28/44H in either BASIC or C language using the point-to-point protocol. Programmers can therefore build their applications using either of the programs as building blocks.

For those who wish to experiment with the RS485 network, A program "Host485.C" in Turbo C language has also been included. This program assumes that an RS232-to-RS485 adapter is used such that the direction of communication of the RS485 bus is controlled by the state of /RTS line of the RS232C. This program accepts both point-to-point and multipoint commands from the keyboard and automatically initiates the correct communication protocol with the control. If your RS485 adapter works differently then you must modify the functions "transmit485()" and "receive485()"

to control the direction of the half-duplex RS485 bus. Please refer to the technical manual of your RS485 adapter for details.

Although written for RS485 bus, you can also experiment with this program using PLC's RS232 interface, you would however not be able to link more than one H-series PLC to the host PC in this case.

27. Applications of Host Communication Capabilities

The H-series 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.

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 be 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.