Adaptive Micro Systems Inc

Adaptive Micro Systems
ALPHA Industrial Protocol

Revision 3.2

9711-8814A

Adaptive Micro Systems Inc

Table of Contents

List of Figures

List of Tables

Adaptive Micro Systems Inc

1.0 REVISION HISTORY

Date	Author	Description	Revision
3/15/01	Greg Byzewski	Original document.	1.0.0
3/21/01	Greg Byzewski	Changed name of document. Changed marquee to display.	1.0.1
3/29/01	Greg Byzewski	Removed ^V update Variable 0 when not specified. Added broadcast to all displays for ^T and ^V. Removed Section 5.3 and 5.4.	1.0.2
4/6/01	Greg Byzewski	Changed document name. Changed software name	2.0.0
4/10/01	Greg Byzewski	Changed \FF\FF to \2D\31 for –1 ASCII in Sections 3.1, 4.5, 4.6 Added other options for removing all messages section 4.5, 4.6 Changed Document Revision	2.1
8/7/01	Greg Byzewski	Changed name of document Added Ctrl-T and Ctrl-V Added Heartbeat enable/disable Added changing display address Change text to be more consistent	3.0
1/12/02	Greg Byzewski	Changed pause from 300 ms to 350ms in sections 8.8 and 8.9. Added the word ‘of’ in section 7.3 Change E (46h) to E (45h) in Section 9.0	3.1
7/2/02	Greg Byzewski	Changed checksum from 7E to 82 in Figure 48. Change ^] (28h) to ^] (27h) in Section 9.0	3.2

2.0 INTRODUCTION

The purpose of this document is to show the protocol that is required to trigger messages and update variables on Adaptive Micro Systems Alpha displays with the ALPHA Industrial Protocol upgrade. This protocol will be in addition to Adaptive's current ALPHA Sign Communication firmware and will utilize some of the features already built into the existing firmware.

In general, messages are downloaded through the ALPHA Automation Software and stored within the display's memory. Up to 4000 messages and 100 variables can be created with the ALPHA Automation Software. Messages and variable data are sent to the display memory utilizing the Modbus ASCII protocol and modified ALPHA Sign Communication protocol. See section 9.0 for further explanation of the ALPHA Sign Communication Protocol being used.

Messages can be displayed using the Control-T function or Modbus ASCII. One of three different methods can be used; 1) Priority Messaging, 2) Add a Message, 3) Remove a Message. Variables can be updated using the Control-V function or Modbus ASCII.

This document is broken up into three basic sections: The Control-T and Control-V functions, Modbus ASCII, and Modified ALPHA Sign Communication Protocol.

3.0 CONTROL-T AND CONTROL-V FUNCTIONS

The two functions that can be used to trigger messages and update variables are the Control-T (CRTL-T) and the Control-V (CTRL-V). The CTRL-T function allows for two different ways of displaying messages: Priority Messaging and Add/Remove Messages. The CTRL-V function is used to update variables on all displays or a specific display.

Priority Messaging will display the message number that was just transmitted to the display. Using this function will have precedence over any priority levels that are used with Add/Remove Messages. Add/Remove Messages will allow for up to 64 concurrently running messages to be shown on the display. Messages can be cleared one at a time or all at once. These messages can have priority levels assigned to them that will be used to determine how they are displayed.

Adaptive Micro Systems Inc

3.1 The CTRL-T Function Frame

The CTRL-T function is used to trigger messages. This function requires an ASCII decimal value to be used to trigger the desired message number. The CTRL-T function has the ability for Priority Messaging, Add a Message, or Remove a Message. Figure 1 shows the format for the Control-T function and Table 1 shows the acceptable values.

			Optional[1]				
Name	[CTRL][T]	MSG #	Backslash	Function	Backslash	Display address	<CR>

Figure 1. Control-T Frame Format

[1] If not used; the message will be shown as a Priority Message on all displays.
When using a Function, a display address must be included, even if it is a broadcast address of 255.

Data	Acceptable Values		Description
	ASCII	Hex	
[CTRL][T]	^T	\14	Start of Header
MSG #	1 to 4000 (Decimal)	\31 to \34\30\30\30	Message number
	4095 (Decimal)	\34\30\39\35	
	-1 (Decimal)	\2D\31	
Backlash	\	\5C	Backslash
Function	1 (Decimal) = Priority message 2 (Decimal) = Add message 3 (Decimal) = Remove message	\31 \32 \33	Message function
Backlash	\	\5C	Backslash
Display Address	1 to 255	\31 to \32\35\35	Display address, where 255 is a broadcast address
<CR> or CTRL][M]	^M	\0D	Carriage Return

Table 1. Control-T Frame Description

Using a Priority message will clear all concurrently running messages and display the message number just sent. This will supersede any message that has been added to the queue regardless of priority levels assigned to them.

Add/Remove messages allows for the ability for up to 64 concurrently running messages to be displayed. Messages are added to the queue and will be displayed for the length of time based on the message pause setting (1-5 seconds). Messages can have three different priority levels assigned to them; Low, Medium, and High. Messages with the highest priority level in the queue will be displayed until they are cleared. When all messages are removed or cleared, the Background Message (message # 4095) is displayed.

Note: If Message pause of zero (0) is used, the message will be shown only briefly.
Note: If no background message is programmed, the default message "**NO BACKGROUND MESSAGE**" will appear.

3.3 The CTRL-V Function Frame

The CTRL-V function is used to update variables that are embedded within messages. The value of the variable number will determine which variable register will be updated in the display. If no display address is used, it will update all displays with the variable data. Figure 2 shows the format for updating variables and Table 2 shows the acceptable values for the CTRL-V function.

			Optional[2]				
Name	[CTRL][V]	Variable Data	Backslash	Variable #	Backslash	Display address	<CR>

Figure 2. Control-V Frame Format

[2] If the Variable # and Display address are not used; only variable 0 on all displays will be updated. If the Variable # is used and not the Display address, that variable number will be updated on all displays.

Where:

Data	Acceptable Values		Description
	ASCII	Hex	
[CTRL][V]	^V	\16	Start of Header
Variable Data	-32768[3] to 32767 (ASCII decimal)	\2D\33\32\37\36\38 to \33\32\37\36\37	Data
Backlash	\	\5C	Backslash
Variable #	0 to 99 (ASCII decimal)	\30 to \39\39	Variable ID number
Backlash	\	\5C	Backslash
Display Address	1 to 255	\31 to \32\35\35	Display address where 255 is a broadcast address
<CR> or [CTRL][M]	^M	\0D	Carriage Return

Table 2. Control-V Frame Description

[3] **Note:** To display negative values for a variable, +/- variable format must be selected in the software or sent via protocol (See section 9.0 for further explanation of the ALPHA Sign Communcation Protocol being used).
Note: To send floating point numbers, you will need to use 2 variables; one for the integer portion, and one for the decimal portion. i.e {var1;}.{var2;}

4.0 EXAMPLES OF USING THE CONTROL-T FUNCTION

There are two basic ways in which messages can be displayed: Priority Messaging or Add/Remove Messages. When using either of these methods, the messages can be shown on one or all displays. The following examples show the different ways in which the Control-T function can be used.

4.1 Trigger a message on all displays using Priority Messaging.

A message can be shown on all displays one of three different ways. These examples show the Control-T function being used to show message number 45 on all displays.

Name	[CTRL][T]	MSG #	Return
ASCII	^T	45	^M
Hex	\14	\34\35	\0D

Or

Name	[CTRL][T]	MSG #	Backslash	Display address	Return
ASCII	^T	45	\	255	^M
Hex	\14	\34\35	\5C	\32\35\35	\0D

Or

Name	[CTRL][T]	MSG #	Backslash	Function	Backslash	Display address	Return
ASCII	^T	45	\	1	\	255	^M
Hex	\14	\34\35	\5C	\31	\5C	\32\35\35	\0D

Figure 3. Triggering a message on all displays

Note: Allen-Bradley PLC's requires two backslashes (\\) to be used as a delimiter. Other PLC manufactures may only require one backslash (\).

4.2 Trigger a message on a specific display using Priority Messaging.

The following are examples of triggering message 39 on display address 031.

Name	[CTRL][T]	MSG #	Backslash	Display address	Return
ASCII	^T	39	\	31	^M
Hex	\14	\33\39	\5C	\33\31	\0D

Or

Name	[CTRL][T]	MSG #	Backslash	Function	Backslash	Display address	Return
ASCII	^T	39	\	1	\	31	^M
Hex	\14	\33\39	\5C	\31	\5C	\33\31	\0D

Figure 4. Triggering a message on a specific display

4.3 Adding a message on all displays.

This is an example of adding message 2011 to all displays.

Name	[CTRL][T]	MSG #	Backslash	Function	Backslash	Display address	Return
ASCII	^T	2011	\	2	\	255	^M
Hex	\14	\32\30\31\31	\5C	\32	\5C	\32\35\35	\0D

Figure 5. Adding a message on all displays

4.4 Adding a message on a specific display.

The following will add message 348 to display address 055.

Name	[CTRL][T]	MSG #	Backslash	Function	Backslash	Display address	Return
ASCII	^T	348	\	2	\	55	^M
Hex	\14	\33\34\38	\5C	\32	\5C	\35\35	\0D

Figure 6. Adding a message on a specific display

Adaptive Micro Systems Inc

4.5 Remove all messages on all displays.

The following will remove all messages on all displays and will automatically display the Background Message (4095) without adding the Background Message number to the Message Queue.

Name	[CTRL][T]	MSG #	Backslash	Function	Backslash	Display address	Return
ASCII	^T	-1	\	3	\	255	^M
Hex	\14	\2D\31	\5C	\33	\5C	\32\35\35	\0D

The following will remove all messages on all displays and add the background message (4095) to the Message Queue using Priority Messaging.

Name	[CTRL][T]	MSG #	Return
ASCII	^T	4095	^M
Hex	\14	\34\30\39\35	\0D

Or

Name	[CTRL][T]	MSG #	Backslash	Display address	Return
ASCII	^T	4095	\	255	^M
Hex	\14	\34\30\39\35	\5C	\32\35\35	\0D

Or

Name	[CTRL][T]	MSG #	Backslash	Function	Backslash	Display address	Return
ASCII	^T	4095	\	1	\	255	^M
Hex	\14	\34\30\39\35	\5C	\31	\5C	\32\35\35	\0D

Figure 7. Removing a message on all displays

4.6 Remove all messages on a specific display.

The following will remove all currently running messages on a display address 024 and will automatically display the Background Message (4095) without adding the Background Message number to the Message Queue.

Name	[CTRL][T]	MSG #	Backslash	Function	Backslash	Display address	Return
ASCII	^T	-1	\	3	\	24	^M
Hex	\14	\2D\31	\5C	\33	\5C	\32\34	\0D

The following will remove all messages on display address 024 and add the Background Message (4095) to the Message Queue using Priority Messaging.

Name	[CTRL][T]	MSG #	Backslash	Display address	Return
ASCII	^T	4095	\	24	^M
Hex	\14	\34\30\39\35	\5C	\32\34	\0D

Or

Name	[CTRL][T]	MSG #	Backslash	Function	Backslash	Display address	Return
ASCII	^T	4095	\	1	\	24	^M
Hex	\14	\34\30\39\35	\5C	\31	\5C	\32\34	\0D

Figure 8. Removing all messages on a specific display

4.7 Remove a message on a specific display.

The following will remove message 367 on a display address 4.

Name	[CTRL][T]	MSG #	Backslash	Function	Backslash	Display address	Return
ASCII	^T	367	\	3	\	4	^M
Hex	\14	\33\36\37	\5C	\33	\5C	\34	\0D

Figure 9. Removing a message on a specific display

5.0 EXAMPLES OF USING THE CONTROL-V FUNCTION

This function will allow for the updating of variables that are embedded in messages. If no display address is used, it will update all displays with the variable data. The following example will update variable 0 in all displays.

5.1 Update a variable on all displays.

Update variable 0 with the value of 2395 on all displays

Name	[CTRL][V]	Variable Data	Return
ASCII	^V	2395	^M
Hex	\16	\32\33\39\35	\0D

Or

Name	[CTRL][V]	Variable Data	Backslash	Variable #	Return
ASCII	^V	2395	\	0	^M
Hex	\16	\32\33\39\35	\5C	\30	\0D

Or

Name	[CTRL][V]	Variable Data	Backslash	Variable #	Backslash	Display address	Return
ASCII	^V	2395	\	0	\	255	^M
Hex	\16	\32\33\39\35	\5C	\30	\5C	\32\35\35	\0D

Figure 10. Updating a variable on all displays

5.2 Update variable on a specific display.

Update variable 5 with the value of 87 on display address 006.

Name	[CTRL][V]	Variable Data	Backslash	Variable #	Backslash	Display address	Return
ASCII	^V	87	\	5	\	6	^M
Hex	\16	\38\37	\5C	\35	\5C	\36	\0D

Figure 11. Updating a variable on a specific display

Adaptive Micro Systems Inc

6.0 MODBUS ASCII PROTOCOL

Modbus ASCII is an industrial data communications protocol that has been implemented in Adaptive's ALPHA Industrial Protocol. It is a master and slave protocol providing for one master and up to 247 slaves. Each slave is assigned a unique address, because only the master can initiate a transaction. The protocol controls the query and response that takes place between master and slave devices as in Figure 12

The Modbus ASCII protocol is incorporated into our ALPHA Industrial Protocol that will allow displays to be connected to a Modbus ASCII communications network. These displays can receive applicable Modbus ASCII protocol and limited ALPHA Sign Communication protocol (used during downloading and previewing of messages). Messages are pre-loaded into the displays, and then triggered by writing Modbus ASCII instructions to specific registers in the display.

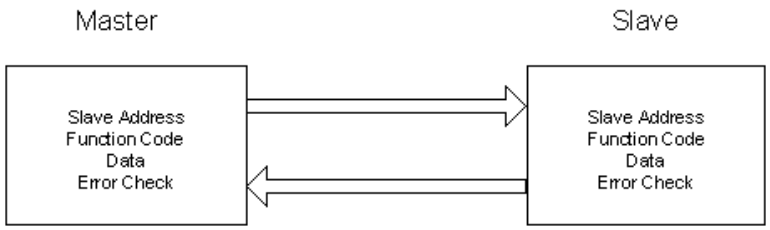


Figure 12. Master and Slave Query/Response Cycle

Below are several terms used throughout the Modbus ASCII section.

Terms:

ASCII - American Standard Code for Information Interchange.
LRC - Longitudinal Redundancy Check used for error checking.
RTU - Remote Terminal Unit.
Query – Sending information to a node address where a response is expected.
Response – A response is given to a Query.
Transmission – Sending information to all node addresses where no response is expected.

Adaptive Micro Systems Inc

7.0 HOW ALPHA SIGN COMMUNICATION PROTOCOL IS USED WITH MODBUS ASCII PROTOCOL.

Adaptive Micro Systems uses only a small portion of the Modbus protocol. There are two different modes of transmission used with Modbus protocol; Modbus ASCII and Modbus RTU. In our case, Modbus ASCII is used. Modbus ASCII protocol is used to accomplish such functions as Priority Messaging, Add a Message, Remove a Message, Update Variables, and reading register in the Message Queue or Variable.

The basic framing format that is used in the Modbus ASCII is shown in Figure 5

Beg of Frame	Address	Function	Data	Error Check	EOF	Ready to rec. resp
:	2-char 16 bits	2-char 16 bits	N x 4-Char N x 16-Bits	2-char 16 bits	CR	LF

Figure 13. ASCII Message Frame Format.

Downloading messages to the displays using Modbus ASCII protocol requires a special function in order to pass Adaptive's ALPHA Industrial Protocol to the displays. To accomplish this, a Modbus 01 Function or Frame is specifically used to used which encapsulates our ALPHA protocol. The downloading of messages that are created using ALPHA Automation Software automatically encapsulates our protocol in the Modbus ASCII string.

To encapsulate the Alpha protocol a Modbus ASCII header and trailer are used. The ALPHA Automation Software has message length range from 50 to 450 Bytes. This is only the maximum length of the Alpha protocol that is encapsulated within the Modbus ASCII protocol. When downloading messages on a Modbus network, the maximum length of the Modbus ASCII and the Alpha protocol is 500 bytes. Figure 6 shows the format of the 01 Frame used for the downloading of messages to the displays.

Beg of Frame	Address	Function	Starting Register Lo	Starting Register Hi	Data	Error Check	EOF	Ready to rec. resp
:	2-char 16 bits	2-char 16 bits	2-char 16 bits	2-char 16 bits	ALPHA Protocol	2-char 16 bits	CR	LF

Figure 14. Function 01 used for Downloading of Messages.

Messages are downloaded to all displays on the Modbus ASCII display network. A broadcast address is used that will require no response from the displays. Modbus ASCII protocol uses address 00h for broadcasting information from one device to the next. This is generally reserved for PLC to PLC communications. For this reason, address 255 (0xff) is used.

7.1 Mode of Transmission

Mode of transmission is the format which messages are transmitted over the network. Characteristics of the Modbus ASCII system are described as follows:

Coding Systems - hexadecimal (uses ASCII printable characters: 0-9, A-F).

Modbus allows the format for data transmission to be as follows:

Baud rate: user selectable
Format: Number of bits per character - 1 start bit, 7 data, 1 (optional) parity bit, 1 or 2 stop bits.

For our products, the data baud rate and format shall be **9600** baud (maximum), **1** start bit, **7** data bits, **even** parity, **2** stop bits.

7.2 ALPHA Automation Display Memory Map

Information transmitted to the displays will write or read information into holding registers. The holding registers used in our products are 40001 to 40167. Holding registers 40001 to 40103 can be written to. Registers 40001 to 40100, and 40103 to 40167 can be read from.

Our product has three basic areas for registers: 100 registers for variables, 3 registers for message control, and 64 registers for the Message Queue. When referencing a holding register, Modbus ASCII requires that 40001 be subtracted from the holding register address. For example, when using Priority Messaging, messages are sent to holding register 40103. The register number that is transmitted would be (40103-40001=102) or 102 (0x66) would be transmitted at 100 (64h). The following shows the memory map for the Adaptive Displays:

Display Memory Map		
Modbus Holding Registers	Registers	Description
40001	001	Variables registers Variables 001-100 (00-99 with ALPHA Automation Software)
…	…	
40100	100	
40101	101	Add a Message/Downloading of messages
40102	102	Remove a Message
40103	103	Message Queue/Priority Messaging
…	…	
40167	167	

Table 3. Memory Map

7.3 Methods of Transmission

There are eight different methods of transmission of data to and from the display. Each one of these use 1 of 4 different Modbus function codes. Table 2 explains each of the four Modbus ASCII function codes used and Table 3 shows the eight methods of using these codes with our product.

Function Code	Function Code Hex	Modbus Meaning	Action.
01	01	Read Coil status	Used to signal the downloading of messages to the display
03	03	Read Output Registers Query	Used to read registers in a display
06	06	Preset Single Register	Used to preset a single register in a display (Priority Messaging, Update Variables, and Add/Remove a Message)
16	10	Preset Multiple Registers	Used to preset multiple registers in a display (Add/Remove a Message or Update Variables)

Table 4. Modbus Function Codes used in Adaptive Industrial Products

Method	Description	Modbus Function code	Display Action
Add/Remove a Message using a Broadcast transmission	Triggers a message(s) on all displays using address 255	10	Writes information into registers 101 and 102. The display will give no response to transmission.
Add/Remove a Message using a Guaranteed transmission	Triggers a message(s) on a specific display address	10	Writes information into register 101 and 102. The display will respond to the query.
Priority Messaging using a Broadcast transmission	Triggers a message on all displays using address 255	06	Write information into register 103. The display will give no response to transmission.
Priority Messaging using a Guaranteed transmission	Triggers a message on a specific display address	06	Write information into register 103. The display will respond to the query.
Read register data	Request information from a group of registers in the display.	03	Transmit back the information in the registers queried.
Variables Broadcast	Update variables in all displays using address 255	06 or 10	Update variable registers. The display will give no response to the transmission.
Variable Guaranteed	Update variable in a specific display address	06 or 10	Update variable registers. The display will respond to the query.
Message Download	Download messages to all display(s) using a specific function code using address 255	01	Signals the displays that the following data will be Alpha protocol. The display will give no response to the transmission.

Table 5. Methods of Transmission

7.4 Message Format

Messages are transmitted at 9600 baud 7e2, and starts with a Beginning of Frame ":" and ends with a carriage return (CR) line feed (LF) to indicate the end of frame. The line feed character also serves as a synchronizing character to indicate that the transmitting station is ready to receive an immediate reply.

The Modbus ASCII message frame format consists of an address field, a function field, a data field, an LRC or error check field, an end of frame field (EOF), and a ready to receive response field (LF). The EOF is a carriage return (0x0d), and the LF is a line feed (0x0a). The following shows the ASCII message frame format.

Beg. Of Frame	Address	Function	Data	Error Check (LRC)	EOF	LF
:	2 – char 16 – bits	2 – char 16 – bits	N x 4 – char N x 16 – bits	2 – char 16 – bits	0x0d	0x0a

Figure 15. ASCII Message Frame Format.

7.4.1 Beginning of Frame Field

Each transmission will start with a ":" and is used to signal the receiving device that message packet shall follow.

7.4.2 Address Field

Each slave must be assigned a unique address. When a master sends to a specific slave address (query), the slave sends a response (response) message back to the master. When a master sends a message to a specific display address (001 to 247), the slave will give a response if the message data is correct. All other slaves will ignore the data transmission.

When a master sends a broadcast message addressed of 255 (0xff), then all slaves interpret this as an instruction to read and take action on the message. No response message is required.

Modbus allows 001-247 for unique addresses that can be used on a network. Our products allow for address 001-255, with 255 being the broadcast address. Since some networks may not support address 248-255, then the broadcast address may not be used, and message queries will always be guaranteed. If Modbus message transmissions are being generated from a serial port, then addressing from 001-255 are valid.

Note: When a Query/Response is used for sending messages to the display, the display will respond back to the host in less than 10 ms after receiving the Line Feed field (0x0a).

7.4.3 Function Field

The function field tells the address slave what function to perform. There are only four functions that are applicable to our display application. They are described in Table 4.

7.4.4 Data Field

The data field contains information on the specific action that the slave must perform.

7.4.5 Error Check Field

The error checking is the LRC of the message and allows for the master and slave to detect message errors. A response message is only sent, if the original message was received correctly. The following shows a simulated query and response.

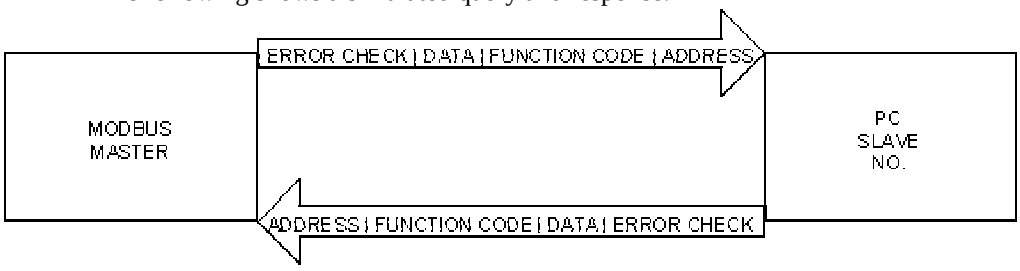


Figure 16. Simulated Query and Response

7.4.6 End Of File Field

This field is used to signify the end of file for the transmission. It uses a Carriage Return (0x0d).

7.4.7 Line Feed Field

This is the ready to respond field file (LF) and uses a line feed (0x0a).

7.5 Longitudinal Redundancy Check (LRC) Error Detection and Calculation

Some sort of error detection is needed, because communication errors can occur in an industrial environment due to machinery noise and electromagnetic interference. The method for error detection used with Modbus ASCII is Longitudinal Redundancy Check (LRC). The LRC is an 8-bit binary number represented and transmitted as two ASCII Hexadecimal characters. The LRC is produced by adding the message characters (ignoring the carry bit) and taking the two's compliment of the result. The error check byte is done from the address up to the data field. The following is an example of how to calculate the LRC in a Modbus ASCII transmission.

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA	LRC Error	EOF	Ready to rec. resp
:	05	06	00	1F	0264	70	CR	LF

Message	Transmitted Bytes	Binary
Address	05	0000 0101
Function	06	0000 0110
Start Add H.O.	00	0000 0000
Start Add L.O.	1F	0001 1111
Data H.O.	02	0000 0010
Data L.O.	64	0110 0100
Sum	90	1001 0000
1's Complement		0110 1111
+1		+0000 0001
2's Complement		0111 0000
Transmitted as	70	

Figure 17. Example of LRC Calculations

8.0 EXAMPLES OF USING MODBUS ASCII FUNCTIONS

The following sections show examples of Modbus ASCII protocol Query/Response transmissions to/from a display for each command instruction listed above. Modbus ASCII will write/read to the holding registers in the display. These registers are in the 40000 range. When writing to register 40101, 40001 is dropped from the address leaving 100 (0x64). The following example shows the transmission of data to holding register 40102 in all displays. The display will add 40001 to the starting address upon receiving the transmission before updating the appropriate holding register.

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA	LRC Error	EOF	Ready to rec. resp
:	FF	06	00	65	FFFF	98	CR	LF

Target holding register 40102 – 40001 = 101 (0x65)

Figure 18. Example of a Target Holding Register

Format for the following sections will be the use of a Query/Response or a Transmission. Each section will show the complete transmission and response (if applicable). In some examples, there are multiple steps that are needed to complete the function requested. Within these steps, there are pause or delay times that must be used in-between each step.

The data that is sent will be ASCII information and the CR and LF will represent 0x0d and 0x0a respectively. The Beg of Frame (:) will represent (0x3a).

8.1 Heartbeat Function

The Heartbeat Function, when enabled, will allow the display to determine if it is no longer on the network or if the host device is not functioning properly (not sending data or a heartbeat). Should the display not see any serial activity (valid or invalid) within 3 seconds, an error message "No Network Activity" will be displayed. The Heartbeat can either be enabled or disabled via protocol or software.

Factory Default: **DISABLED**.

8.1.1 Enabling the Heartbeat Function

The displays will be shipped with the Heartbeat disabled. Should the Heartbeat function be required, the following string of information will enable the display to look for a Heartbeat or serial activity.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA**	EOF	Ready to rec. resp
:	FF	01	00	64	^AZFF^BE01^D9C	CR	LF

Figure 19. Transmission for Enabling the Heartbeat Function

****Note:** See section 9.0 for further explanation of the ALPHA Sign Protocol being used.

Response: **NONE**

8.1.2 Disabling the Heartbeat Function

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA**	EOF	Ready to rec. resp
:	FF	01	00	64	^AZFF^BE00^D9D	CR	LF

Figure 20. Transmission for Disabling the Heartbeat Function

****Note:** See section 9.0 for further explanation of the ALPHA Sign Protocol being used.

Response: **NONE**

8.1.3 Heartbeat transmission

The following is the transmission for the Heartbeat that is required for the display once the Heartbeat function has been enabled.

Beg of Frame	Data	EOF	Ready to rec. resp
:	00	CR	LF

Figure 21. Transmission for the Heartbeat

Note: Recommended interval to be transmitted: once every 500 ms.
Note: Variable data or Message data that is being updated in the displays will also act as a Heartbeat for the "No Network Activity" error message.

Response: **NONE**

8.2 Clearing the Display Memory

This command is used to clear *all* of the memory (messages) in the displays, resize the memory partitions to 100 bytes (2,000 messages), and load each memory slot with a default message number (e.g. Message #0002)

Note: The "Clearing Memory" string may be required prior to the downloading of messages.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA**	EOF	Ready to rec. resp
:	FF	01	00	64	^AZ00^BE$^D	CR	LF

Figure 22. Transmission for Clearing Display Memory

****Note:** See section 9.0 for further explanation of the ALPHA Sign Protocol being used.

Response: **NONE**

8.3 Clearing the Message Queue using Modbus ASCII in Broadcast mode (Recommended)

The Adaptive display uses a Message Queue for all currently running messages on the display. This command is used to remove *all* currently running messages being displayed on all displays. This is also required when switching between Priority Messages and Add/Remove Messages modes of operation.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA	LRC Error	EOF	Ready to rec. resp
:	FF	06	00	65	FFFF	98	CR	LF

Figure 23. Transmission for Clearing the Message Queue with 06 Frame Broadcast

Response: **NONE**

Adaptive Micro Systems Inc

8.4 Clearing the Message Queue using Modbus ASCII in Guaranteed mode

The following transmission is used to remove all currently running messages on a specific display address. For example, clearing all messages being displayed for display address 001:

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA	LRC Error	EOF	Ready to rec. resp
:	01	06	00	65	FFFF	96	CR	LF

Figure 24. Query for Clearing the Message Queue with 06 Frame Guaranteed

Response:

The normal response to a function 06 is to echo (or re-transmit) the query after the holding register is updated.

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA	LRC Error	EOF	Ready to rec. resp
:	01	06	00	65	FFFF	96	CR	LF

Figure 25. Response from Clear Queue with a 06 Frame Guaranteed

Adaptive Micro Systems Inc

8.5 Setting Time in Broadcast mode with AM/PM Format

This command is used to set the time and the format for the time in all displays. It is used primarily with the ALPHA Automation Software to synchronize the time in the displays with the computer. This requires two separate transmissions. First, setting the time and format for the time. Second, clearing the queue.

The following is an example of setting the time to 1034 and the format for the time to be AM/PM.

Transmission:

Step 1. Set the time and time format.

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA**	EOF	Ready to rec. resp
:	FF	01	00	64	^AZ00^BE 1034^C^BE'S^C^D	CR	LF

Figure 26. Transmission for Setting Time with AM/PM Format

Note: See section 9.0 for further explanation of the ALPHA Sign Protocol being used.

Response: **NONE**
300 ms pause

Step 2. Clear the Message Queue.
This transmission is used to remove the time that was just loaded into the Message Queue.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA	LRC Error	EOF	Ready to rec. resp
:	FF	06	00	65	FFFF	98	CR	LF

Figure 27. Transmission for Clearing the Message Queue

Response: **NONE**

8.6 Setting Time in Broadcast mode with 24 Hour Format

This command is used to set the time and the format for the time in all displays. This is primarily done using the ALPHA Automation Software to synchronize the time in the displays with the computer. This requires two separate transmissions. First, setting the time and format for the time. Second, clearing the queue. The following is an example of setting the time to 1035 and the format for the time to be 24 hour (Military) format.

Step 1. Set the time and time format.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA**	EOF	Ready to rec. resp
:	FF	01	00	64	^AZ00^BE 1035^C^BE'M^C^D	CR	LF

Figure 28. Transmission for Setting Time with 24 Hour Format

****Note:** See section 9.0 for further explanation of the ALPHA Sign Protocol being used.

Response: **NONE**
300 ms pause

Step 2. Clear the Message Queue.
This transmission is used to remove time that was just loaded into the Message Queue.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA	LRC Error	EOF	Ready to rec. resp
:	FF	06	00	65	FFFF	98	CR	LF

Figure 29. Transmission for Clearing the Message Queue

Response: **NONE**

8.7 Setting Day and Date in Broadcast mode

This command is used to set the date in all displays. This is primarily done using the ALPHA Automation Software to synchronize the date in the displays with the computer. This requires three separate transmissions. First, setting the date. Second, setting the day of week. Third, clearing the queue. The following is an example of setting the day and date to Thursday, January 4, 2001.

Step 1. Set the Date.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA**	EOF	Ready to rec. resp
:	FF	01	00	64	^AZ00^BE;01040^D	CR	LF

Figure 30. Transmission for Setting Date

Note: See section 9.0 for further explanation of the ALPHA Sign Protocol being used.

Response: **NONE**
300 ms pause

Step 2. Set the Day of week.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA**	EOF	Ready to rec. resp
:	FF	01	00	64	^AZ00^BE&5^D	CR	LF

Figure 31. Transmission for Setting Day of Week

Note: See section 9.0 for further explanation of the ALPHA Sign Protocol being used.

Response: **NONE**
300 ms pause

Step 3. Clear the Message Queue.
This transmission is used to remove Date and Day of week that was just loaded into the Message Queue.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA	LRC Error	EOF	Ready to rec. resp
:	FF	06	00	65	FFFF	98	CR	LF

Figure 32. Transmission for Clearing the Message Queue

Response: **NONE**

Adaptive Micro Systems Inc

8.8 Previewing a Message

Previewing a Message requires four separate types of transmissions. This is primarily done using the ALPHA Automation Software to preview a message. First, the Message Queue must be cleared. Second, the message must be downloaded to the appropriate memory partition. Third, the message must be triggered for viewing. Fourth, a Heartbeat must be generated (if enabled) so that there is some serial network activity allowing the message to be previewed. The ALPHA Automation Software automatically performs these steps.

Note: Previewing of message causes the loss of data in the Memory partition where the message is going to be stored.

This is an example of previewing Message #0001 as a target memory position.

Step 1. Clearing the Message Queue
This transmission is used to remove all currently running messages on the display for all displays.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA	LRC Error	EOF	Ready to rec. resp
:	FF	06	00	65	FFFF	98	CR	LF

Figure 33. Transmission for Clearing the Message Queue

Response: **NONE**
350 ms pause

Step 2. Downloading Message #0001 to the appropriate memory position
This will store the message in the message number used within the data field. It will overwrite any data already stored in the memory position.

Note: Messages are downloaded to register 101 (40101), which then moves the Alpha message to the appropriate memory position.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA**	EOF	Ready to rec. resp
:	FF	01	00	64	^AZ00^BA2001^[”b^I^^1^\1 Hello^D	CR	LF

Figure 34. Download Message 1 to be Previewed

Note: See section 9.0 for further explanation of the ALPHA Sign Protocol being used.

Response: **NONE**
350 ms pause

Step 3. Triggering the message that was just downloaded

Trigger the message that was just downloaded by activating the message number that you want to preview in the Message Queue.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA	LRC Error	EOF	Ready to rec. resp
:	FF	06	00	66	0001	94	CR	LF

Figure 35. Trigger Message for Viewing

Response: **NONE**
350 ms pause

Step 4. Generate a Heartbeat to view the message that was downloaded (if the Heartbeat function has been enabled).

Use the Heartbeat to provide serial activity so the message may be previewed. Disabling the Heartbeat causes the previewed message to turn off after a 3 second internal time-out.

Transmission:

Beg of Frame	Data	EOF	Ready to rec. resp
:	00	CR	LF

Figure 36. Transmission for Heartbeat

Note: Send the Heartbeat every 500 ms until no longer needed for the previewing of the message.

Response: **NONE**
500 ms pause between each Heartbeat transmission

8.9 Downloading Messages

Messages can be downloaded to the display so that they can be triggered (or activated) at a later time. Downloading of messages is generally done using the ALPHA Automation Software, but can be done as shown below. There are three basic steps that must be done to accomplish this. First, set the Memory size. Second, Messages are then downloaded. Third, clear the queue.

It may be necessary to clear the memory in the display prior to downloading of messages. This can be done either through a serial transmission or using the Adaptive's infrared (IR) Keyboard.

8.9.1 Example 1. Downloading three messages

This example shows the downloading three messages (with three different priority levels). Setting the message size to 200 bytes (0xC8) for each message.

Step 1. Set message partition size in the display(s) to 200 bytes.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA**	EOF	Ready to rec. resp
:	FF	01	00	64	^AZ00^BEa00C8^D	CR	LF

Figure 37. Transmission for Setting Message size to 200 Bytes (Example 1)

****Note:** See section 9.0 for further explanation of the ALPHA Sign Protocol being used.

Response: **NONE**
2000 ms pause

Step 2. Download three messages.

This step is accomplished with three separate transmissions to the display, one for each message.

Note: The total number of bytes for each transmission (from the Beg of Frame to Ready to rec. resp.) must not exceed *500* bytes.

Download message file #0001

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA**	EOF	Ready to rec. resp
:	FF	01	00	64	^AZ00^BA2001^[”b^I^^1^\1Priority High^D	CR	LF

Figure 38. Transmission for Downloading Message 1 (Example 1)

****Note:** See section 9.0 for further explanation of the ALPHA Sign Protocol being used.

Response: **NONE**

350 ms pause

Download message to file #0002

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA**	EOF	Ready to rec. resp
:	FF	01	00	64	^AZ00^BA1002^["b^I^^1^\1Priority Medium^D	CR	LF

Figure 39. Transmission for Downloading Message 2 (Example 1)

****Note:** See section 9.0 for further explanation of the ALPHA Sign Protocol being used.

Response: **NONE**
350 ms pause

Download message to file #0003

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA**	EOF	Ready to rec. resp
:	FF	01	00	64	^AZ00^BA0003^["b^I^^1^\1Priority Low^D	CR	LF

Figure 40. Transmission for Downloading Message 3 (Example 1)

****Note:** See section 9.0 for further explanation of the ALPHA Sign Protocol being used.

Response: **NONE**
350 ms pause

Step 3. Clear the Message Queue.
This transmission is used to remove all currently running messages downloaded to the Message Queue.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA	LRC Error	EOF	Ready to rec. resp
:	FF	06	00	65	FFFF	98	CR	LF

Figure 41. Transmission for Clearing the Message Queue

Response: **NONE**

8.9.2 Example 2. Downloading two messages with a variable

This example shows downloading two messages (one with a variable). Setting the message size to 60 bytes (0x3C) for each message.

Step 1. Set message partitions in the display(s) to 60 bytes.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA**	EOF	Ready to rec. resp
:	FF	01	00	64	^AZ00^BEa003C^D	CR	LF

Figure 42. Transmission for Setting Message Size to 60 Bytes (Example 2)

****Note:** See section 9.0 for further explanation of the ALPHA Sign Protocol being used.

Response: **NONE**
2000 ms pause

Step 2. Download two messages.

This step is accomplished with two separate transmissions to the display, one for each message.

Note: The total number of bytes for each transmission (from the Beg of Frame to Ready to rec. resp.) must not exceed *500* bytes.

Download message file #0001

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA**	EOF	Ready to rec. resp
:	FF	01	00	64	^AZ00^BA2001^[”b^I^^1^\1Hello^D	CR	LF

Figure 43. Transmission for Downloading Message 1 (Example 2)

****Note:** See section 9.0 for further explanation of the ALPHA Sign Protocol being used.

Response: **NONE**
350 ms pause

Download message to file #0002

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA**	EOF	Ready to rec. resp
:	FF	01	00	64	^AZ00^BA1002^[b^I^^1^\2Part count = ^]A0^]B0^P00^D	CR	LF

Figure 44. Transmission for Downloading Message 2 (Example 2)

Note: See section 9.0 for further explanation of the ALPHA Sign Protocol being used.

Response: **NONE**
350 ms pause

Step 3. Clear the Message Queue.
This transmission is used to remove all currently running messages downloaded to the Message Queue.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA	LRC Error	EOF	Ready to rec. resp
:	FF	06	00	65	FFFF	98	CR	LF

Figure 45. Transmission for Clearing the Message Queue

Response: **NONE**

Adaptive Micro Systems Inc

8.10 Add/Remove a Message using a Modbus ASCII 10 frame Query (Recommended)

This method uses both registers 40101 and 40102 in the display to Add and Remove a Message from the Message Queue respectively. By using the Guaranteed mode, only the display that is addressed on the network will accept and process the information it receives. The following is an example of adding message #0015 and removing message #0045 from the Message Queue on display number 019.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	QUANTITY	BYTE CNT	DATA	
:	13	10	00	64	0002	04	000F	002D

LRC Error	EOF	Ready to rec. resp
37	CR	LF

Figure 46. Query for Add/Remove a Message

Response:

The normal response to a function 10 is to echo the address, function code, starting address and the number of registers that were loaded.

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	QUANTITY	LRC Error	EOF	Ready to rec. resp
:	13	10	00	64	0002	77	CR	LF

Figure 47. Response for Add/Remove a Message

8.11 Add/Remove a message using Modbus ASCII 10 frame Transmission

This method uses both registers 40101 and 40102 in the display to activate and de-activate messages respectively. By using the Broadcast mode, all displays on the network will accept and process the information received. The following is an example of adding message #0003 and removing message #0002 from the Message Queue.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	QUANTITY	BYTE CNT	DATA	
:	FF	10	00	64	0002	04	0003	0002

LRC Error	EOF	Ready to rec. resp
82	CR	LF

Figure 48. Transmission for Add/Remove a Message

Response: **NONE**

8.12 Priority Messaging using a Modbus ASCII Query (Recommended)

This method writes to the first register in the Message Queue (40103). When this happens, the previous message in the queue is replaced with the new message to being loaded.

This is an example of sending a Priority Message #0099 to display address 001.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA	LRC Error	EOF	Ready to rec. resp
:	01	06	00	66	0063	30	CR	LF

Figure 49. Query for a Priority Message

Response:

The normal response to a function 06 is to echo (or re-transmit) the query after the holding register is updated.

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA	LRC Error	EOF	Ready to rec. resp
:	01	06	00	66	0063	30	CR	LF

Figure 50. Response for a Priority Message

8.13 Priority Messaging using a Modbus ASCII Transmission

This method writes to the first register in the Message Queue (40103). When this happens, the previous message in the queue is replaced with the new message to be activated. This is an example sending a Priority Message #0800 to all displays that are on the network.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA	LRC Error	EOF	Ready to rec. resp
:	FF	06	00	66	0320	72	CR	LF

Figure 51. Transmission for Priority Messaging

Response: **NONE**

8.14 Updating a variable using a Modbus ASCII 06 frame in Broadcast mode (Recommended)

This method writes (or loads) variable data to the appropriate variable registers in all displays. The following is an example of loading variable 20 (register 40020) with the value of 3656 for all displays on the network.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA	LRC Error	EOF	Ready to rec. resp
:	FF	06	00	13	0E48	92	CR	LF

Figure 52. Transmission to Update a Variable Register with 06 Frame Broadcast

Response: **NONE**

8.15 Updating a variable using a Modbus ASCII 06 frame in Guaranteed mode

This method writes (or loads) variable data to the appropriate variable register in a specific display. The following is an example of loading variable 32 (register 40032) with the value 612 in display address 005.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA	LRC Error	EOF	Ready to rec. resp
:	05	06	00	1F	0264	70	CR	LF

Figure 53. Query for Updating a Variable Register

Response:

The normal response to a function 06 is to echo (or re-transmit) the query after the holding register is updated.

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA	LRC Error	EOF	Ready to rec. resp
:	05	06	00	1F	0264	70	CR	LF

Figure 54. Response for Updating a Variable Register

Note: Using the Guaranteed mode for variable updates may slow down variable updates due to the response transmission.

Adaptive Micro Systems Inc

8.16 Updating Variables using a Modbus ASCII 10 frame in Broadcast mode (Recommended)

This method allows the ability to update more than one variable register in all displays. A maximum of 60 registers can be updated in one transmission.

The following example shows how to update variables 1, 2, and 3 (registers 40001, 40002, and 40003) with values 24, 53, and 56 respectively in all displays.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	QUANTITY	BYTE CNT	DATA		
:	FF	10	00	00	0003	06	0018	0035	0038

LRC Error	EOF	Ready to rec. resp
63	CR	LF

Figure 55. Transmission for Updating Variable Registers with 10 Frame Broadcast

Response: **NONE**

Adaptive Micro Systems Inc

8.17 Updating Variables using a Modbus ASCII 10 frame in Guaranteed mode

This method will allow the ability to update more than one variable in a specific display address. A maximum of 60 registers can be updated in one transmission. Unused high order bits must be set to zero. The following is an example of updating variables 1, 2, and 3 with values 23, 734, and 7 respectively in display address 003.

Below is an example of a preset multiple register transmission.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	QUANTITY	BYTE CNT	DATA		
:	03	10	00	00	0003	06	0017	02DE	0007

LRC Error	EOF	Ready to rec. resp
E6	CR	LF

Figure 56. Query for Updating Variable Registers using a 10 Frame

Note: Using the Guaranteed mode for variable updates, messages may not be displayed as expected.

Response:

The normal response to a function 10 query is to echo the address, function code, starting address and the number of registers that were loaded.

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	QUANTITY	LRC Error	EOF	Ready to rec. resp
:	03	10	00	00	0003	EA	CR	LF

Figure 57. Response for Updating Variable Registers using a 10 Frame

Adaptive Micro Systems Inc

8.18 Reading the Message Queue using a Modbus ASCII 03 frame

This function allows the ability to read message numbers that are currently running in the Message Queue on a display. A maximum of 64 registers can be read since there is a maximum of 64 registers (or concurrently running messages) in the Message Queue. The following is an example of request for the first 3 registers in the Message Queue (Registers 40103-40105) which contain message data values 4, 6, and 11 respectively in display address 010.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA Number of register to read	LRC Error	EOF	Ready to rec. resp
:	0A	03	00	66	0003	8A	CR	LF

Figure 58. Query for Requesting Message Queue Data

Response:

The addressed slave responds with its address and the function code, followed by the information field. The information field contains 2 bytes describing the quantity of data bytes to be returned. The contents of the registers requested (DATA) are two bytes each, with the binary content right justified within each pair of characters.

The normal response to a function 03 query is to echo the address, function code, starting address and the data for the registers that were requested.

Beg of Frame	ADDR	FUNC	BYTE COUNT	DATA OUTPUT REG H.O. 0066	DATA OUTPUT REG L.O. 0066	DATA OUTPUT REG H.O. 0067	DATA OUTPUT REG L.O. 0067
:	0A	03	06	00	04	00	06

DATA OUTPUT REG H.O. 0068	DATA OUTPUT REG L.O. 0068	LRC	EOF	Ready to rec. resp
00	0B	D8	CR	LF

Figure 59. Response for Requesting Message Queue Data

Adaptive Micro Systems Inc

8.19 Reading variables in a display using a Modbus ASCII 03 frame

This function allows the ability to read variable data stored in a display. A maximum of 100 registers can be read since there is a maximum of 100 registers of data. Below is an example of the request for registers 40002-40004 in display address 020.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA Number of register to read	LRC Error	EOF	Ready to rec. resp
:	14	03	00	01	0003	E5	CR	LF

Figure 60. Query for Requesting Variable Data

Response:

The addressed slave responds with its address and the function code, followed by the information field. The information field contains 2 bytes describing the quantity of data bytes to be returned. The contents of the registers requested (DATA) are two bytes each, with the binary content right justified within each pair of characters. Below shows the response to Query for registers 40002-40004 having the decimal contents of 44, 63, and 1 respectively.

The normal response to a function 03 query is to echo the address, function code, starting address and the data for the registers that were requested.

Beg of Frame	ADDR	FUNC	BYTE COUNT	DATA OUTPUT REG H.O. 0002	DATA OUTPUT REG L.O. 0002	DATA OUTPUT REG H.O. 0003	DATA OUTPUT REG L.O. 0003
:	14	03	06	00	2C	00	3F

DATA OUTPUT REG H.O. 0004	DATA OUTPUT REG L.O. 0004	LRC	EOF	Ready to rec. resp
00	01	77	CR	LF

Figure 61. Response for Requesting Variable Data

Adaptive Micro Systems Inc

8.20 Changing the ALPHA Display Address

This function allows the ability to change the serial address in a display. For example change display address 001 to address 020. When doing this, you will need to know the address of the display you want to change. This is typically done using the ALPHA Automation Software.

The following is an example of changing a display (with the address of 005) to address 004.

Transmission:

Beg of Frame	ADDR	FUNC	H.O. ADDR	L.O. ADDR	DATA**	EOF	Ready to rec. resp
:	05	01	00	64	^AZ05^BE704^D	CR	LF

Figure 62. Transmission for Changing the Serial Address

Note: See section 9.0 for further explanation of the ALPHA Sign Protocol being used.

Response: **NONE**

Adaptive Micro Systems Inc

9.0 ALPHA DISPLAY COMMUNICATION PROTOCOL FUNCTIONS AND DESCRIPTIONS

The ALPHA Sign Communication protocol that is used in conjunction with the Modbus ASCII protocol is mostly limited due to the number of functions and features that required by the ALPHA Industrial products. The ALPHA Industrial Protocol is available on the 420, 4000, and 7000 series ALPHA products. There are certain rules that apply to how messages are displayed on these products.

9.1 Alpha Message format used within Modbus ASCII protocol

Information that is used within the Alpha Message is formatted differently than that of our standard protocol. The following table shows the basic message structure with a brief explanation of each part.

Figure 63. Basic Alpha Message Format

<SOH>	Start of Header
Type Code	What display type to communicate with.
Display Address	Address of display where information is written
<STX>	Start of Text
Command Code	Describes what type function to perform, Message priority, and Message number
Message Control	Describes the How the message is displayed, such as Position, Mode, Pause, and Justification
Message Attributes	Describes any Font, Color, Width/Height, and Flash
Insert Objects	Describes any special items that are inserted within a message such as Time, Date, and Extended Character set.
Insert Variables	Describes Variable number and formatting
Message Data	Data for the Message itself
<EOT>	End of Transmission

Table 6. Alpha Message Field Explanation

Items in Bold must be used in each transmission. The ALPHA Automation Software will automatically put this information into the message that is transmitted. Some of these functions are modified from the ALPHA protocol that is used in our standard product line. E.g. message number versus file labels. Below is a list of the valid values for each of the fields shown above. Each of these are concatenated together to make up the Alpha Message.

<SOH>			Type Code			Display Address		
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
Start of Header	^A	01	All displays	Z	5A	Display Address	00	3030

<STX>			Command Code					
			Command			Priority		
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
Start of Text	^B	02	Write Text file	A	41	Low	0	30
			Write Special Function	E[2]	45	Medium	1	31
						High	2	32

[2] See Section 4.2 for further explanation of the Special Function Command

Command Code (Continued)			Message Control					
Message #			<ESC>			Display Position		
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
0001	001	303031	Start of Mode Field	^[	27	Middle	(space)	20
Through						Top	“	22
4000	FA0	464130				Bottom	&	26
4095	FFF	464646				Fill	0	30

Adaptive Micro Systems Inc

Message Control (Continued)								
Mode			Speed			Justification		
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
Rotate left	a	61	Speed 0 (No Hold)	^I	09	Left	^^1	1E31
Hold	b	62	Speed 1 (5 Sec.)	^U	15	Center	^^3	1E33
			Speed 2 (4 Sec.)	^V	16			
			Speed 3 (3 Sec.)	^W	17			
			Speed 4 (2 Sec.)	^X	18			
			Speed 5 (1 Sec.)	^Y	19			

Message Attributes								
Character Font			Character Color			Character Width/Height		
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
5 High Standard	^Z1	1A31	Red	^\1	1C31	Standard	^Q^]10	111D3130
7 High Standard	^Z2	1A32	Green	^\2	1C32	Wide	^R^]10	121D3130
7 High Fancy	^Z5	1A35	Yellow	^\8	1C38	Dble Wide	^Q^]11	111D3131
10 High Standard	^Z6	1A36				Double High	^Q^\21	111D3231
16 High Fancy (Full Height)	^Z8	1A38						
16 High Standard (Full Height)	^Z9	1A39						

Message Attributes (Continued)			Insert Objects					
Character Flash			Time			Date		
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
Flash ON	^G1	0731	Time	^S	13	MM/DD/YY	^K0	0B30
Flash OFF	^G0	0730				DD/MM/YY	^K1	0B31
						MM-DD-YY	^K2	0B32
						DD-MM-YY	^K3	0B33
						MM.DD.YY	^K4	0B34
						DD.MM.YY	^K5	0B35
						MM DD YY	^K6	0B36
						DD MM YY	^K7	0B37
						MMM.DD YYYY	^K8	0B38

Insert Object (Continued)			Insert Variable		
Miscellaneous			Variable Format		
Description	ASCII	Hex	Description	ASCII	Hex
See Appendix A	^H+	08	No Padding, XX	^]A0^]B0	1D41301D4230
			Leading 0, 000XX	^]A1^]B0	1D41311D4230
			Leading Space, ___XX	^]A2^]B0	1D41321D4230
			_+/- No Padding, +/- XX	^]A0^]B1	1D41301D4231
			+/- leading 0, +/- 000XX	^]A1^]B1	1D41311D4231
			+/- Leading Space, +/-, ___XX	^]A2^]B1	1D41321D4231

Insert Variable			Message Field			<EOT>		
Call Variable			Message or Data					
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
Variable 1	^P00	103030	ASCII Data	(space) - DEL	20-7F	End of Transmission	^D	04
Variable 2	^P01	103031						
…	…	…						
Variable 99	^P98	103938						
Variable 100	^P99	103939						

Figure 64. Alpha Message Format.

9.2 Special Function Command

The Special Function Command is used to; Clear the Display Memory, Set Message Size, Set Time and Date, Read Memory from a display on the network, and set the serial address. The following figures show the valid information to perform each on the commands

9.2.1 Clear the Display Memory

<SOH>			Type Code			Display Address		
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
Start of Header	^A	01	All displays	Z	5A	Display Address	00	3030

<STX>			Command Code					
			Command			Function		
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
Start of Text	^B	02	Write Special Function	E	45	Clear Memory	$	24

<EOT>		
Description	ASCII	Hex
End of Transmission	^D	04

Figure 65. Special Function command to Clear Display Memory

9.2.2 Set Message Size

The Message Size is the number of bytes that each message can be stored in. This Message size will partition the displays memory based on the number of bytes used in the Message Size. When this command is used, it (generally, based on model of display) takes 200,000 Bytes and divides it by the number of bytes used for the Message Size. For Example, 2,000 message partitions would be available if the Message Size was set to 100 Bytes (200,000 Bytes/100 Bytes/Message = 2,000 Messages).

The number of bytes used in each message is the number of bytes used from the <SOH> to the <EOT> inclusive. Therefore, care must be used to insure that the number of bytes sent does not exceed the Message Size. The range for the Message Size is 50 Bytes to 450 Bytes. The following is the Message data used to set the Message Size.

<SOH>			Type Code			Display Address		
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
Start of Header	^A	01	All displays	Z	5A	Display Address	00	3030

<STX>			Command Code					
			Command			Function		
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
Start of Text	^B	02	Write Special Function	E	45	Set Message Size	a	61

Command Code (Continued)			<EOT>		
Description	ASCII	Hex	Description	ASCII	Hex
Message Size (minimum) 50 Bytes	0032	30303332	End of Transmission	^D	04
Message Size (maximum) 450 Bytes	01C2	30314332			

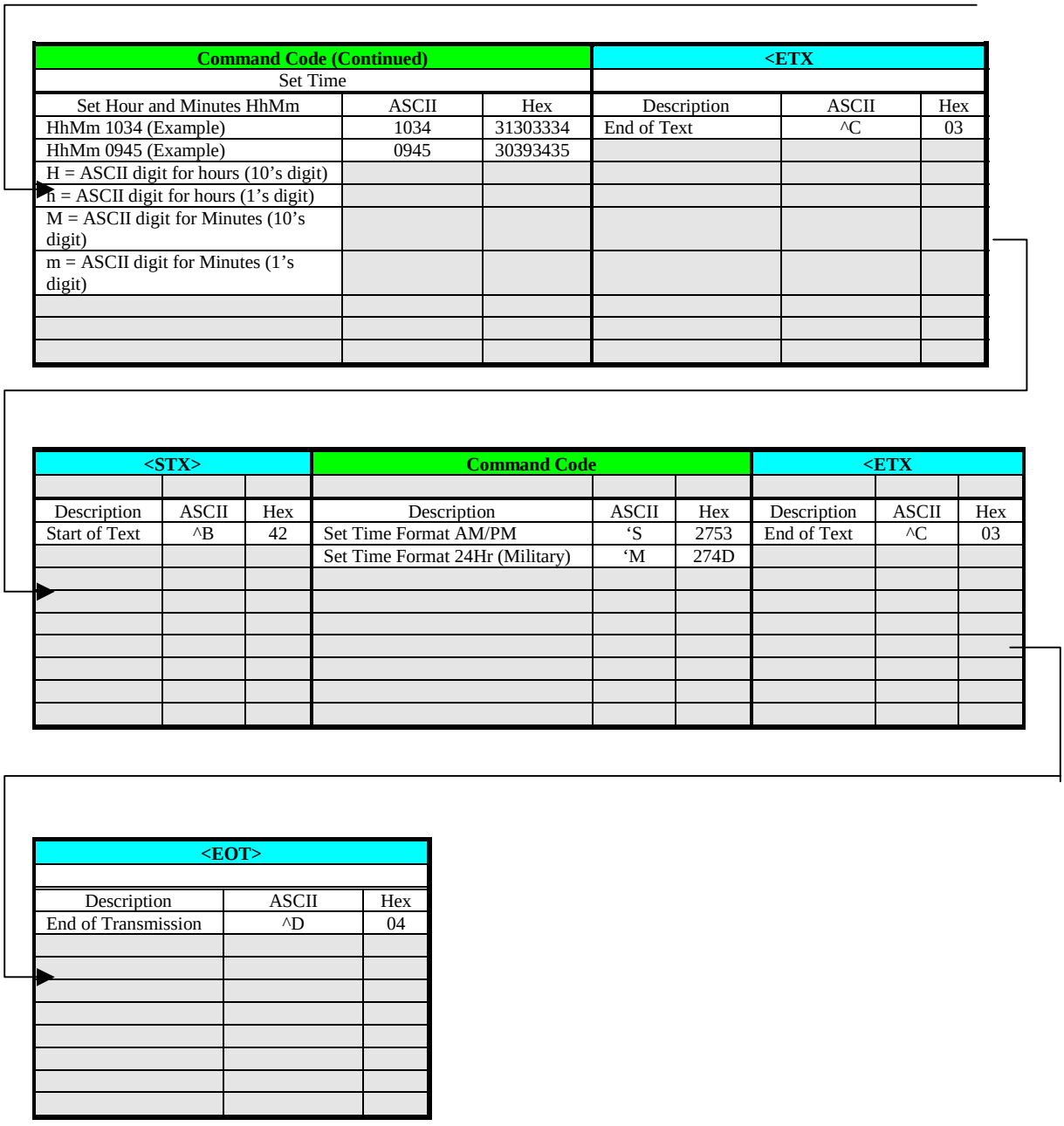
Figure 66. Special Function command to set Memory Size

9.2.3 Set Time

Time can be set in either AM/PM format or 24 Hour (Military) format. The following shows the valid values for this command

<SOH>			Type Code			Display Address		
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
Start of Header	^A	01	All displays	Z	5A	Display Address	00	3030

<STX>			Command Code					
			Command			Function		
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
Start of Text	^B	02	Write Special Function	E	45	Set Time	(Sp)	20



Command Code (Continued)			<ETX		
Set Time					
Set Hour and Minutes HhMm	ASCII	Hex	Description	ASCII	Hex
HhMm 1034 (Example)	1034	31303334	End of Text	^C	03
HhMm 0945 (Example)	0945	30393435			
H = ASCII digit for hours (10's digit)					
h = ASCII digit for hours (1's digit)					
M = ASCII digit for Minutes (10's digit)					
m = ASCII digit for Minutes (1's digit)					

<STX>			Command Code			<ETX		
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
Start of Text	^B	42	Set Time Format AM/PM	'S	2753	End of Text	^C	03
			Set Time Format 24Hr (Military)	'M	274D			

<EOT>		
Description	ASCII	Hex
End of Transmission	^D	04

Figure 67. Special Function command to set Time and Time Format

Adaptive Micro Systems Inc

9.2.4 Set Date

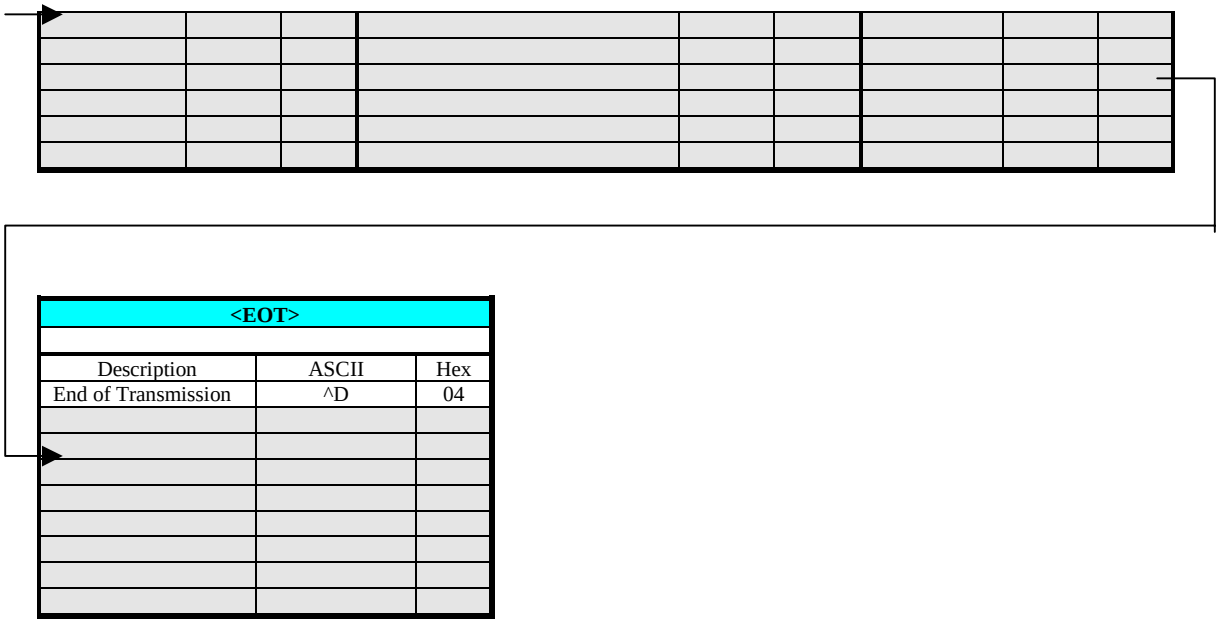
This command is used to set the Date. Format for the displaying the Date is shown under Insert Object in Figure 50. Alpha Message Format.

<SOH>			Type Code			Display Address		
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
Start of Header	^A	01	All displays	Z	5A	Display Address	00	3030

<STX>			Command Code					
			Command			Function		
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
Start of Text	^B	02	Write Special Function	E	45	Set Date	;	

Command Code (Continued)			<ETX		
Set Date					
Set Hour and Minutes HhMm	ASCII	Hex	Description	ASCII	Hex
January 4, 2001 (Example)	010401	303130343031	End of Text	^C	03
December 5, 2002 (Example)	120502	313230353032			

<STX>			Command Code			<ETX		
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
Start of Text	^B	42	Set Time Format AM/PM	'S	2753	End of Text	^C	03
			Set Time Format 24Hr (Military)	'M	274D			



<EOT>		
Description	ASCII	Hex
End of Transmission	^D	04

Figure 68. Special Function command to set Time and Date

9.2.5 Set Day of Week

This command will set the Day of week once the Date is set.

<SOH>			Type Code			Display Address		
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
Start of Header	^A	01	All displays	Z	5A	Display Address	00	3030

<STX>			Command Code					
			Command			Function		
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
Start of Text	^B	02	Write Special Function	E	45	Set Date	&	26

Command Code (Continued)			<EOT>		
Set Day of Week	ASCII	Hex	Description	ASCII	Hex
Sunday	1	31	End of Transmission	^D	04
Monday	2	32			
Tuesday	3	33			
Wednesday	4	34			
Thursday	5	35			
Friday	6	36			
Saturday	7	37			

Figure 69. Special Function command to set Day of Week

9.2.6 Set serial address

This command will change a serial address for a display.

<SOH>			Type Code			Display Address		
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
Start of Header	^A	01	All displays	Z	5A	Display Address	01-FF	3031-4646

<STX>			Command Code					
			Command			Function		
Description	ASCII	Hex	Description	ASCII	Hex	Description	ASCII	Hex
Start of Text	^B	02	Write Special Function	E	45	Set address	7	37

Command Code (Continued)			<EOT>		
Serial address	ASCII	Hex	Description	ASCII	Hex
New serial address	01-FE	3031-4645	End of Transmission	^D	04

Figure 70. Special Function command to set Serial Address

Adaptive Micro Systems Inc

9.3 Text Position Placement

Adaptive uses the 420, 4000, and 7000 series displays with ALPHA Industrial protocol. The 420 display is a single line unit, the 4000 series is a two-line display and the 7000 series is a 3-line display. Displaying of messages using protocol is slightly different based on which display is being used. Below are some rules for the 4000 and 7000 display. The 420 will ignore the Display Position and show (display) the text.

4000 series	
Display Position	Description
Top	Displays text on top line only.
Bottom	Displays text on the bottom line only
Middle	Used to display full height text (Fancy 16 Pixel or 16 Pixel)
Fill	Displays two lines of 7 pixel characters

Figure 71. Valid Modes Used in 4000 Series Displays

7000 series			
Display Position	Description	Comment 1	Comment 2
Top	Displays text on top line only.	If the top line only has one row of text, the remaining bottom rows are considered bottom	If the top line only has two rows of text, the remaining bottom row is considered bottom
Bottom	Displays text on the bottom line only	If the bottom line only has one row of text, the remaining top rows are considered top	If the bottom line has two rows of text, the remaining top row is considered top
Middle	Not used	Not used	Not used
Fill	Displays full screen of text	Can be used to display Fancy 16 Pixel or 16 Pixel characters	

Figure 72. Valid Modes Used in 7000 Series Displays

Adaptive Micro Systems Inc

APPENDIX A

You can enter extended ASCII characters directly within the text of a message. Some of the characters may not show up depending on the fonts on your PC.

Char	Hex	Dec	Char	Hex	Dec
Ç	80H	128	í	A1H	161
ü	81H	129	ó	A2H	162
é	82H	130	ú	A3H	163
â	83H	131	ñ	A4H	164
ä	84H	132	Ñ	A5H	165
à	85H	133	ª	A6H	166
å	86H	134	º	A7H	167
ç	87H	135	¿	A8H	168
ê	88H	136	⌐	A9H	169
ë	89H	137	¡	AAH	170
è	8AH	138	Note	ABH	171
ï	8BH	139	θ	ACH	172
î	8CH	140	Θ	ADH	173
ì	8DH	141	ć	AEH	174
Ä	8EH	142	Ć	AFH	175
Å	8FH	143	c	B0H	176
É	90H	144	C	B1H	177
æ	91H	145	d	B2H	178
Æ	92H	146	D	B3H	179
ô	93H	147	s	B4H	180
ö	94H	148	z	B5H	181
ò	95H	149	Z	B6H	182
û	96H	150	ß	B7H	183
ù	97H	151	S	B8H	184
ÿ	98H	152	ß	B9H	185
Ö	99H	153	Á	BAH	186
Ü	9AH	154	À	BBH	187
¢	9BH	155	Á´	BCH	188
£	9CH	156	á´	BDH	189
¥	9DH	157	É	BEH	190
%	9EH	158	Í	BFH	191
f	9FH	159	Õ	C0H	192
á	A0H	160	õ	C1H	193

Note: A single column space

Adaptive Micro Systems Inc

APPENDIX B

Reference material:

ALPHA™ Sign Communications Protocol Revision C.
Modicon Modbus Protocol Reference Guide PI-MBUS-300 Rev C.