CTI GENERAL ASCII SUPPORT
PROTOCOL MANAGER

Version 1.4

CTI Part # 062-00182-014

GASPRM 051303

DOCUMENT DISCLAIMER STATEMENT

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

REVISION HISTORY

Version 1.0	1/31/95	Original Release
Version 1.1	6/15/95	Documented changes to numeric formats which allow user specified pad characters. Documented changes to signed numeric formats which allow the field to start with a decimal digit as well as a sign or blank. Added description of new checksum specification. Added description of new buffer positioning specification. Revised and corrected error code section.
Version 1.2	10/25/95	Added Modem Options to Create Connection Command Block Added selection for lower case hexadecimal to Format 1003 Clarified BCC starting position and message format.
Version 1.3	03/30/98	Added End-of-Message Interval Added Build and Parse commands Documented new Command Processing Flag bits. Renamed “Format Specifications Table” to “Format Table”
Version 1.4	05/05/99	Added commands and information about GAS extensions for the SEAbus protocol

PREFACE

This Reference Manual is intended for individuals who wish to develop applications using the CTI General ASCII Support Protocol Manager (GAS). It is a supplement to the *CTI 2573-TCM2 Installation and Operation Guide,* CTI Part Number 62-181 and to the *CTI 2573-MOD Installation and Operation Guide*, CTI Part Number 62-187.

Chapter 1 provides an overview of the GAS protocol manager operations and presents concepts of device communications.

Chapter 2 describes the PLC Command Blocks used with the GAS Protocol Manager. GAS uses the standard CTI 257x PLC Command Interface to implement commands. If you are not familiar with this interface, please refer to the *Installation and Operation Guide* for the module.

Chapter 3 describes Format Specifications which determine how the GAS Protocol Manager processes messages.

Chapter 4 provides an example of developing applications using GAS.

Chapter 5 provides general troubleshooting information.

For specific module hardware information, including module installation and checkout, please refer to the applicable *Installation and Operation Guide* for the module.

We assume you are familiar with the installation and operation of SIMATIC® 505 programmable controllers. Please refer to the appropriate SIMATIC® user documentation for specific information on SIMATIC® 505 programmable controllers and I/O modules.

USAGE CONVENTIONS

NOTE:
Notes alert the user to special features or procedures

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

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

TABLE OF CONTENTS

CHAPTER 1. OVERVIEW 1
- 1.1. Functional Overview 1
- 1.2. Device Communications Concepts 3
- 1.3. PLC Data Formats 6
- 1.4. Device Message Data Representation 6
- 1.5. GAS Format Table 8
- 1.6. GAS Command Blocks 9
- 1.7. Output Message Processing 9
- 1.8. Input Message Processing 10
- 1.9. Getting Started 13
- 1.10. Limitations and Restrictions 13

CHAPTER 2. COMMAND PROCESSING 15
- 2.1. Overview 15
- 2.2. Create Connection Command 18
- 2.3. Close Connection Command 19
- 2.4. Write Command 21
- 2.5. Read Command 22
- 2.6. Polled Read Command 24
- 2.7. Build Message Command 28
- 2.8. Parse Message Command 29

CHAPTER 3. FORMAT SPECIFICATIONS 33
- 3.1. Overview 33
- 3.2. Format Table 33
- 3.3. Format Specification General Information 34
- 3.4. Format Specification Number Index 38
- 3.5. Bypass Format Specification 40
- 3.6. Table End 40
- 3.7. PLC Unsigned Integer To Unsigned Decimal Integer ASCII String 41
- 3.8. Unsigned Decimal Integer ASCII String To PLC Unsigned Integer 43
- 3.9. PLC Real Number To Unsigned Decimal Fraction ASCII String 45
- 3.10. Unsigned Decimal Fraction ASCII String To PLC Real Number 47
- 3.11. PLC Unsigned Integer To Hexadecimal ASCII String 50
- 3.12. Hexadecimal ASCII String To PLC Unsigned Integer 52
- 3.13. PLC Signed Integer To Signed Decimal Integer ASCII String 54
- 3.14. Signed Decimal Integer ASCII String To PLC Signed Integer 56
- 3.15. PLC Real Number To Signed Decimal Fraction ASCII String 58
- 3.16. Signed Decimal Fraction ASCII String To PLC Real Number 60
- 3.17. PLC ASCII Character To ASCII Character String 62
- 3.18. ASCII Character String To PLC ASCII Character 64
- 3.19. PLC ASCII Character To ASCII Character String (Extended) 66
- 3.20. ASCII Character String To PLC ASCII Character (Extended) 68
- 3.21. PLC ASCII Characters To Variable Length ASCII Character String 70
- 3.22. Variable Length ASCII Character String To PLC ASCII Characters 72
- 3.23. Character Fill 74

3.24. PLC ASCII Character (Null Terminated) To ASCII Character String 76
3.25. Position Data Cursor 78
3.26. Modbus ASCII Master 83
3.27. Block Check Character (BCC) 91
3.28. PLC Word LSB to Message Byte 102
3.29. Message Byte to PLC Word LSB 104
3.30. PLC Word to Message Byte 105
3.31. Message Byte to PLC Word 108
3.32. PLC Word to Message Byte – Byte Swap 109
3.33. Message Byte to PLC word – Byte Swap 111

CHAPTER 4. APPLICATION EXAMPLES 115
4.1. Unsolicited Read Example 115
4.2. Polled Read Example 119
4.3. Parse Message Example 124

CHAPTER 5. PROTOCOL SPECIFIC EXTENSIONS 131
5.1. SEAbus 131
5.2. SEAbus Application Examples 133

CHAPTER 6. TROUBLESHOOTING 141
6.1. Hardware Related Problems 141
6.2. Protocol Problems 143
6.3. PLC Logic Problems 145
6.4. Development and Debugging Tips 147

APPENDIX A. GAS COMMAND ERROR CODES 151
System Errors 151
GAS Protocol Manager Errors 153
Protocol Specific Errors 157

APPENDIX B. FORMAT ERROR CODES 159

APPENDIX C. REFERENCE MATERIAL 169
Hexadecimal to ASCII Conversion Table 169
Command Block Summary 170
WX / WY Quick Reference 174

CHAPTER 1. OVERVIEW

1.1. Functional Overview

CTI General ASCII Support (GAS) is a member of the CTI 257x device support programs called *Protocol Managers*. In this context, *devices* are plant floor equipment which can accept commands and/or send data via communications facilities. The commands and data are transmitted in a group of characters called a *message*.

Protocol Managers perform tasks such as constructing output messages based on data contained in the PLC, storing selected data from input messages in PLC memory, and performing error checking on the message content. Since Protocol Managers execute in the CTI communications module, you do not have to construct complex logic or use valuable PLC CPU resources to perform these tasks.

Even though the Protocol Managers perform these functions, the locus of control still remains with the PLC. By manipulating the module WY words, PLC logic determines when messages are sent, when to place data in PLC memory, and what to do in case of errors. For a complete description of this interface please refer to the *Installation and Operation Guide* for your module.

Most Protocol Managers have been written to interface to a particular device or class of device. For example, the Task Code Master protocol manager communicates specifically with SIMATIC® 505 PLCs using the Non-Intelligent Terminal Protocol (NITP). While they provide excellent function and performance, device specific protocol managers require that a unique protocol manager be written for each device.

The GAS protocol manager is designed to provide a more general device support by allowing you to tailor GAS to match the device protocol. By entering data into tables, which are stored in PLC memory, you can define how an output message is constructed and how the characters in an input message are decoded and stored in PLC memory. Details on how to construct the tables may be found in Chapter 3 of this manual.

NOTE:
*In this manual, the term **input** is used to describe a message **from** the plant floor device to the CTI communications module. The term **output** is used to describe a message **to** the plant floor device from the CTI communications module*

GAS uses the standard PLC Command Interface used by all 257x Protocol Managers.

Information about the content of the Command Blocks may be found in Chapter 2.

1.2. Device Communications Concepts

Many plant floor devices, such as bar code readers, scales, and servo motors, support data communications. A physical *communications port* is used to provide communications access to the device. A typical physical communications port uses *electrical interface* standards such as RS-232 or RS-422. These standards help determine how to connect communications cabling to the device; however, they do not define how to actually communicate with the device.

In order to communicate with a device, you must adhere to its *communications protocol*. The communications protocol defines the format of the messages which the device will accept and the format of messages which will be sent by the device. Unlike the electrical interface standards such as RS-232, there are no commonly accepted industry standards for device communications protocols; every vendor invents whatever seems convenient. In most cases, especially with older designs, data communications appears to be an afterthought.

ASCII Data Representation

Even though every vendor implements a different protocol, most *use American Standard Code for Information Interchange (ASCII)* to represent characters in the message. ASCII defines a specific bit pattern for each character. For example, the letter A is represented by bit pattern 1000001; the character 9 is represented by 0111001. To simplify reading and writing strings of 0s and 1s, these bit patterns are often referred to in Hexadecimal notation. The ASCII character A = 0x41; the character 9 = 0x39. See Appendix C for an ASCII to Hexadecimal conversion table.

NOTE:
Throughout this manual, when hexadecimal notation is used, it will be preceded by the word "hex" or the characters "0x"; thus, hexadecimal 41 is indicated by hex 41 or 0x41.

A tutorial on the hexadecimal numbering system is beyond the scope of this manual. If you are not familiar with the Hexadecimal numbering system, you should obtain a suitable reference book.

ASCII defines not only printable characters but also special characters that may be used for message control. For example, hex 02 represents the ASCII *Start of Text* character. In this manual the ASCII special characters is shown by an abbreviation enclosed within < >. For example, the ASCII Start of Text character is presented as <STX>. See Appendix C for a table of ASCII characters and their hexadecimal equivalents.

It is very important to understand that when ASCII is used, numerical data is transmitted as *characters.* Thus, the decimal number 126 is sent as three characters (0x31, 0x32, and 0x36). This contrasts with a *binary* format, in which a numerical value is transmitted a group

of 8 bit bytes. In binary format, a single byte can represent any unsigned integer number up to decimal 255 (0xFF).

Because a binary representation uses fewer characters to represent numerical data than ASCII requires, some device protocols use the binary format to improve communications performance. This benefit requires a significant trade-off, however. Since any binary value can appear anywhere in the message, you cannot have any bit patterns reserved for special characters. Thus, you cannot use these characters to indicate where a message starts and where it ends. Instead you must use special bit sequences, character sequences, or character timing techniques, methods which complicate device communications considerably.

Protocols using a binary format are most useful when transmitting large amounts of data (e.g. file transfer). Since most plant floor devices transmit small amounts of data at a time, the performance improvement may not warrant the additional complexity of the binary format. As the name implies, the General ASCII Support (GAS) protocol manager handles only ASCII encoding, not binary format.

Device Message Requirements

Some devices, such as printer or a display, only accept messages, they do not generate them. These devices are often referred to as *output only*.

Some devices send a message when a particular device event occurs. For example, a bar code reader may send a message when a scan is triggered by the user. This method of sending a message is called *unsolicited input*; it requires that the communications module be ready to accept the entire message whenever it is sent.

Other devices require that a message containing a command be sent to the device before it will return a message containing the desired data. For example, you may be required to query a weigh scale in order to obtain the weight data. This is referred to as *solicited input* or *polled response*.

GAS provides commands that accommodate all of the above methods. The *WRITE* command sends an output message, the *READ* command accepts unsolicited input, and the *POLLED READ* command solicits input from a device.

Message Processing Requirements

A very simple application might require only that a fixed message be sent to the device and that a return message be stored in PLC memory in ASCII character format. In most cases, however, you will want to do more than this. Typically, you may wish to include variable data from the PLC in the output message or to convert a selected set of characters from the ASCII message to a numerical representation supported by the PLC. GAS provides the method for accomplishing these requirements. See the sections below for an overview of how numerical data is stored in the PLC and in the device message.

1.3. PLC Data Formats

Numerical data is stored in a SIMATIC® TI500/505 series PLC in one of the following formats:

- *Unsigned Integer Word*. All 16 bits of the word are used to represent the number. Therefore, the smallest number that can be represented is 0 and the largest is 65,535.
- *Signed Integer Word*. A signed integer uses one bit as an indication of the sign; negative numbers are stored in two's complement form. The smallest number which can be represented is -32,768 while the largest is +32,767.
- *Long Word*. A long word uses 32 bits (2 contiguous words) to represent a signed integer number. Long Word format can represent integers ranging from -4,294,967,295 to +4,294,967,294 .
- *Real Number*. A real number is represented by 32 bits (2 contiguous words) in ANSI/IEEE Standard 754-1985 format. The first word is used to store the sign bit and the exponent; the second word stores the fractional value (mantissa). A real number format is typically used where fractional values and/or values greater than the capability of an integer word format is needed. Using this format you can store:
  Positive Numbers ranging from $5.42101070 \times 10^{-20}$ to $9.22337177 \times 10^{18}$
  Negative Numbers ranging from $-2.71050535 \times 10^{-20}$ to $-9.22337177 \times 10^{18}$

  One advantage of the real number format it that it allows you to trade fractional precision for range. A disadvantage is that some decimal numbers cannot be precisely represented; therefore you may encounter some rounding errors.
- *Binary Coded Decimal.* Four bits of each word are used to store one decimal digit (0-9). One word can store a number ranging from 0 through 9999.

In addition, bit patterns representing ASCII characters may be stored in PLC memory. In general, PLC logic provides little direct support for character based data. Programming software such as TISOFT does allow you to enter and display the ASCII characters. See the appropriate SIMATIC® 505 documentation for more information on PLC data formats.

1.4. Device Message Data Representation

As mentioned previously, devices communicate by sending a stream of ASCII characters. While some characters convey text information or are used for message control, others typically represent a number. For example, consider a device which sends the following characters:

<STX>ccccccnnnnn,nnnnn<CR> where:

<STX>	is the ASCII Start of Text control character (0x02)
cccccc	are alphabetic characters representing text,
nnnnn	are characters representing a numeric values you want to store in the PLC.
<CR>	is the ASCII carriage return control character.

In this message there are three data *fields*, each representing a particular unit of information. The text field consists of six characters; the two numeric fields are separated by a comma in this example.

The characters representing the numeric value may take several forms including the following:

- *Unsigned Decimal Integer*. The characters 0-9 (0x30-0x39) are used to represent the value. For example, the value 1432 would be represented as the equivalent ASCII characters (0x31, 0x34, 0x33, 0x32).
- *Signed Decimal Integer*. The characters 0-9 (0x30-0x39), the + character (0x2B), and the - character (0x2D) are used to represent the value. For example, the value -1432 would be represented as the equivalent ASCII characters (0x2D, 0x31, 0x34, 0x33, 0x32).
- *Unsigned Decimal Fraction*. The characters 0-9 (0x30-0x39) and the decimal point character (0x2E) are used to represent the value. For example, the value 14.32 would be represented as the equivalent ASCII characters (0x31, 0x34, 0x2E, 0x33, 0x32).
- *Signed Decimal Fraction*. The characters 0-9 (0x30-0x39), the + character (0x2B), the - character (0x2D), and the decimal point character (0x2E) are used to represent the value. For example, the value -14.32 would be represented as the equivalent ASCII characters (0x2D, 0x31, 0x34, 0x2E, 0x33, 0x32).
- *Hexadecimal Integer*. This format is often called hex-ASCII. It uses the characters 0-9 (0x30-0x39) and the characters A-F) (0x41-0x46) to represent the number expressed in hexadecimal form. For example, the hexadecimal equivalent of 1432 is 0598. This value would be represented as the four ASCII *characters* 0598 (0x30, 0x35, 0x39, 0x38). Hex-ASCII is commonly found in PLC ASCII formats such as NITP and MODBUS.

1.5. GAS Format Table

GAS uses a *Format Table* to define how a device message will be processed. Each Format Specification in the table defines how a data field in the message (also called an ASCII *string*) will be converted to PLC data or how PLC data will be converted to an ASCII data field.

For output, a Format Specification defines how to *encode* each data field in the output message. The encoding process includes converting data stored in the PLC format to the appropriate message format and placing the resulting characters in a particular position in the output buffer.

For input, a Format Specification defines how to *decode* a data field in the input message. The decoding process includes locating the data field in the message, converting the characters to the appropriate PLC data format, and storing the result in a PLC memory location.

1.6. GAS Command Blocks

GAS uses the standard 257x Command Block structure to control reading and writing messages. If you are not familiar with Command Block concepts, please refer to *Installation and Operation Guide* for the module. The Command Blocks used with GAS provide:

- A WRITE command to send an output message to a device,
- A READ command to obtain unsolicited input from a device, and
- A POLLED READ command to solicit input from a device.
- A BUILD MESSAGE command which will construct an output message and store it in PLC memory
- A PARSE MESSAGE command which will decode an ASCII message stored in PLC memory.

The Command Blocks will reference the format specifications described above. The GAS Command Blocks are defined in Chapter 2 of this manual.

1.7. Output Message Processing

GAS includes commands that generate an output message to a device. These commands include the WRITE command and POLLED READ command. A POLLED READ is a WRITE followed immediately by a READ. During output processing, the output message is constructed and sent to the device.

Output processing consists of the following steps:

1. The output buffer is filled with the "?" character (0x3F) to aid in debugging. Subsequent characters placed in the output buffer by the user will overwrite the "?" characters. The output buffer can store up to 1080 characters.
2. The Format Table is read and the Format Specifications are processed in the order they appear in the table. Characters are placed in the output buffer in the position determined by the particular Format Specification. If a format error occurs, output processing is terminated and an error condition is returned.
3. When all Format Specifications in the Format Table have been successfully processed, GAS sends the complete message to the communications port. GAS keeps track of the highest buffer position written using the Format Table. This position becomes the last character in the message. See page 35 for an illustration.

Format Specifications are described in Chapter 3 of this manual.

The BUILD command functions in a similar manner, except that the output message is written to a specified V memory location rather than being sent to the serial port.

1.8. Input Message Processing

GAS also includes commands that process an input message from a device. These commands include the READ command and POLLED READ command. Characters arriving from the device are temporarily stored in a port buffer. When a *complete* message has been received, the characters in the port buffer are transferred to an input message buffer where they are decoded using the applicable Format Table.

Values in the READ and POLLED READ command blocks specify how to determine the start and end of a message. You may specify:

- *A Maximum length.* The length specifies the maximum number of characters (up to 1080) that will be included in the message (including any beginning or end delimiters).
- *A Start Delimiter*. A Start Delimiter is a unique character that will always be the first character in the message and repeated nowhere else in the message.
- *An End Delimiter.* An End Delimiter is a unique character that will appear at the end of the message and nowhere else within the message.

NOTE

GAS allows you to specify up to two pairs of beginning and end delimiters. If you specify two, GAS will look for either beginning delimiter in the incoming characters. Once the beginning delimiter is encountered, the corresponding ending delimiter will be used to determine the end of the message.

- *End-of- Message Interval*. The End-of-Message Interval specifies the maximum amount of time that can elapse between characters. If no additional characters are received after this interval, the message will be considered complete.

Note:

Receipt of XON, XOFF, or modem message characters will temporarily suspend timing. Timing will restart when another character, not included in the above, is received.

The following illustrates how these parameters are used.

Start of message

1. If you specify a start delimiter, the protocol manager will ignore all characters preceding the start character. If a duplicate start character is received after the first start character, all previous characters received will be ignored and the message will start with the most recently received start character.
2. If you do not specify a start character, the first character received will be considered as the first character in the message.

End of message

1. If you specify only a maximum length ***n***, the message will be considered complete after ***n*** message characters have been received.
2. If you specify a maximum length ***n*** and an end delimiter, the protocol manager will consider the message complete when *either* the end delimiter is encountered in the message or ***n*** characters have been received, whichever occurs first.
3. If you specify a maximum length ***n*** and an end-of-message interval, the protocol manager will consider the message complete when *either* ***n*** characters have been received or the end-of-message interval is reached, whichever occurs first.
4. If you specify a maximum length ***n***, an end delimiter, and an end-of-message interval, the protocol manager will consider the message complete upon the first occurrence of one of the following conditions:

    a. ***n*** message characters have been received,
    b. the end delimiter is received,
    c. the end-of-message interval is reached.

NOTE:
If you specify two sets of delimiters, both sets must include a start and an end delimiter.

NOTE:
If your device protocol provides a beginning and/or an end delimiter, you should use them. This is the best way to help ensure that you don't pick up spurious characters in the message. If the exact message length is fixed, set the maximum length to this number. If the length is variable, set the maximum length to the largest possible message.

Many protocols do not have an end delimiter; in this case, use the start delimiter and the maximum length. If the length is fixed, set maximum length to the fixed value. If the length

is variable, set maximum length to the largest possible length and use the end-of-message interval parameter.

Some protocols have only an end delimiter, especially those that are designed to send data to a simple display device. They often end with an ASCII Carriage Return character (hex 0D). For messages of this type you can specify only an end delimiter.

If the protocol has no message delimiters and is a fixed length, you can use the length parameter to specify when a complete message has been received. Alternately, you could use the end-of-message interval parameter. In general, be careful when implementing devices which have no message delimiters. If your communications link is noisy, it is easy to get spurious characters in the message.

1.9. Getting Started

The following steps should improve your effectiveness in getting your application up and running,

1. *Read Chapters 2 and 3 of this manual*. Make sure you understand how the module WX/WY words, Command Blocks, and Format Specifications are used. You may also refer to the *Installation and Operation Guide* for the module.
2. *Obtain all the necessary development equipment and supplies*. As a minimum you should have:
   - The actual device and its documentation.
   - A SIMATIC® 505 series PLC and CTI 2573 module.
   - A reliable communications cable.
   - A serial line analyzer or equivalent PC product. The serial line analyzer allows you to observe the actual data being sent on the serial line. Most PC-based serial line analyzer products can be obtained for around $300.00. Some shareware can be obtained for less than $100.00. Contact CTI for references.
3. *Verify that the equipment is functioning properly*. Set up the equipment; verify that the module is logged in correctly, that the device can communicate, and that the cabling is wired correctly. Many device manufacturers provide a PC test program or other method to test communications.
4. *Establish communications with the device*. Enter the applicable Command Block and Format Table into PLC V memory. If you are reading from the device, you should read the entire ASCII message into PLC memory. Do not attempt to decode the message fields yet. Rather than using PLC logic, you should manually trigger the command. See *Manual Triggering* on page 147 for more information. You may use the serial line analyzer to view the messages being sent by the CTI module and the device.
5. *Finalize the Format Specifications to match the actual device message format*. Once you have verified the actual message structure, you can use the numeric Format Specifications to decode selected message characters to a PLC numeric format.

1.10. Limitations and Restrictions

Although GAS is designed to process a wide variety of device protocols, there are limitations the kind of protocols GAS will process. In general, you cannot use GAS if:

- Your protocol uses a binary data format rather than ASCII character encoding.
- You require a block check character format not supported by this version of GAS.
- You must decode numeric formats not supported by this version of GAS (for example, scientific notation or Binary Coded Decimal).
- You need to encode PLC numeric formats not supported by this version of GAS (for example, Long Integer or Binary Coded Decimal).

CHAPTER 2. COMMAND PROCESSING

2.1. Overview

The following sections describe the command blocks used to control the GAS Protocol Manager functions. The use of command blocks is described in applicable *Installation and Operation Guide* for the module. If you are not familiar with this subject, please refer to the manual before proceeding.

Creating and Using Command Blocks

You should create and store each command block you wish to use in a unique location of V memory. You can create them directly in V memory using a tool such as TISOFT, create them at startup time using PLC logic, or store them in K memory and use PLC ladder to copy them into V memory. Then, when you want to execute a different command, change module word WY5-WY8 to point to the new command block. It has been CTI's experience that this approach seems to produce fewer logic errors, as compared to changing the data in a single Command Block "on the fly". Also, this approach always allows you to view the related command error word.

Command Block Conventions

The command blocks illustrations used in this manuals show the contents in both Hexadecimal and Decimal (integer) format. You can enter values in either format using your programming software.

NOTE:
*In the Command Block Description, a **boldface** entry in the value column designates a **required** value. Other entries are recommended values.*

Command Timeout

Some of the Command Blocks allow you to set a Command Timeout value. The timeout represents the number of seconds that can elapse from starting the command until the command is completed. If the command has not been completed within the elapsed time, the processing will halt and the CMD ERR bit will be set. The default timeout is approximately 9 seconds.

Command Control Flag

Some Command Blocks, including Write, Read, and Polled Read contain a Command Control Flag word. The Command Control Flag is a collection of bits which alter the command processing method. The bits are ordered as indicated below, where MSB = Most Significant Bit and LSB = Least Significant Bit.

MSB															LSB
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

The following Command Flag Bits are used with the GAS protocol manager.

Inhibit Buffer Clear

In normal operation, the input buffer is cleared at the beginning of each command. This clears the buffer of any extraneous characters that may have arrived earlier. However, there many be situations where you wish to read a part of the device message, make a logical decision, then issue another command to read additional characters in the same message. See *Application Examples* chapter in this manual.

Parsing Error Action

This flag allows you to continue parsing the input message, even after a parsing error has occurred. You may find this useful when debugging your application. In normal operation, once an error has occurred in decoding a field, the protocol manager immediately stops processing, writes an error code in the format specification and the error word in the Command Block, and sets the COMMAND ERROR bit. In most cases this action is appropriate; since the error could cause decoding for subsequent fields to fail.

If you choose to continue processing after an error occurs, GAS will attempt to decode all the remaining fields in the Format Table. If additional errors occur, the error words will be written to each format specification in error. When all format specifications have been processed, the COMMAND ERROR bit will be set and the Command Error word will be written.

NOTE:
When you choose to continue processing after an error occurs, a problem decoding one field may cause unpredictable results in decoding the remaining fields.

Cache Read

In normal operation, the Format Tables are read from PLC V memory at the beginning of each command cycle. This ensures that the most current data is used to build the output message and/or to decode the input message. However, reading the Format Tables from V memory consumes time because the module must wait at least one scan time before the PLC can deliver the data to the module.

Each time a Format Table is read from V memory, it is also stored (cached) in memory on the module. In situations where you need maximum performance and the Format Table does not change, you may instruct GAS to read the Format Tables from module cache rather from V memory by setting the CACHE READ bit. If you set the CACHE READ bit and the module cache is empty (no previous Format Tables have been read), GAS will automatically read the format table from V memory.

NOTE:
Be careful when using the Cache Read option for output messages (Write or Polled Read). Because the data is read from cache, any changes to format specifications made by PLC logic will not be recognized until the CACHE READ bit is cleared.

2.2. Create Connection Command

Before you can use the GAS protocol manager you must complete the CREATE CONNECTION command. This command starts a copy (instance) of the protocol manager and associates this copy with a physical port. After you have successfully completed this command, you then refer to this instance by the logical connection number in all subsequent commands.

Assuming you have not created any previous connections or automatically started any protocol managers via dipswitch settings, you can create up to four connections (one for each physical port).

NOTE:
Ensure that the physical port you select in offset 4 has been enabled for PLC select. See the module Installation and Operation Guide for information regarding switch settings.

Offset	Description	Hex Value	Decimal Value
0	Command Error Word	**0000**	**0**
1	Command (Create Connection)	**0001**	**1**
2	Connection Number (19221 -- 19299)		
3	Protocol Manager Number (GAS)	**0026**	**38**
4	Physical Port Number (1,2,3,4)		
5	Port Baud Rate (300, 600, 1200, 2400, 4800, 9600, 19200		
6	Bits Per Character (7 or 8)		
7	Parity (0=None, 1=Odd, 2= Even)		
8	Stop Bits (1 or 2)		
9	Handshake (0=None, 1=Software, 2=Hardware, 4 = RS-485). Note: Option 4 is not valid if modem bit in offset 10 below is set.		
10	Option Bits (0x0000 = None, 0x0001 = Modem)		
11	V Memory Address of Modem Configuration Table		
12-15	Unused - reserved for future use (Set to 0)	**0000**	**0**

Offset 0 *Command Error Word* - Your PLC logic should set this value to 0 to clear any previous error codes. The GAS protocol manager will write an error code here if an error is encountered during command processing.

Offset 1 *Command Code* - The command number for CREATE CONNECTION is 1.

Offset 2 *Connection Number* - Any valid number within the range 19221 to 19299 may be assigned as long as it is has not been used previously. For clarity, you may wish to set the lower digit to match the physical port number (e.g., 1922**1** for port 1, 1922**2** for port 2).

Offset 3 *Protocol Manager Number* - Set to 38 (Hex 0026) to select the GAS protocol manager.

Offset 4 *Physical Port Number* - Set to the physical port (1-4) you wish to use.

Offset 5-8 *Communications Parameters* - This must match the characteristics of your device. The values entered will override any switch settings on the module.

Offset 9 *Handshake*- Set to match your device requirements. Note: you cannot select RS-485 handshake when using the 2573 Modem Support Feature (see below).

Offset 10 *Option Bits* - Set bit 16 to 1 (set word = 0x0001) to select the 2573 Modem Support Feature (used with dial-up modems). See the applicable 2573 *Installation and Operation Guide* for a description of this feature.

Offset 11 V Memory Address Modem Configuration Table - If you are using the 2573 Modem Support Feature, this word must contain the V memory address of the first word of the Modem Configuration Table. This value is read when the Modem Option Bit is set.

Offset 12-15 *Unused* - Reserved for future use. Set to 0.

2.3. Close Connection Command

This command is used to close a connection that was previously created by a CREATE CONNECTION command. Once a connection has been closed, you can initiate another CREATE CONNECTION command which reuses the physical port and/or connection number.

Offset	Description	Hex Value	Decimal Value
0	Command Error Word	**0000**	**0**
1	Command Code (Close Connection)	**0002**	**02**
2	Connection Number (19221 - 19299)		
3 - 15	Unused (Set to 0)	**0000**	**0**

Offset 0 *Command Error Word* - Your PLC logic should set this value to 0 to clear any previous error codes. The GAS protocol manager will write an error code here if an error is encountered during command processing.

Offset 1 *Command Code* - The Command Code of the CLOSE CONNECTION command is 2.

Offset 2 *Connection Number* - Identifies the connection you wish to close.

Offset 3 -15 *Unused* - These words are unused and should be set to 0.

2.4. Write Command

The WRITE command sends an ASCII message to the device. The content of the message is determined by the Format Table located at the V memory address in offset 4.

Offset	Description	Hex Value	Decimal Value
0	Command Error Word	**0000**	**0**
1	Command Code (Write)	**2601**	**9729**
2	Connection Number (19221 - 19299)		
3	Command Control Flag	0000	0
4	V Memory Address of Format Table - Output		
5	Maximum # of Format Specifications (0 = Default = 6)	0000	0
6- 15	Unused (Set to 0)	**0000**	**0**

Offset 0 *Command Error Word* - Your PLC logic should set this value to 0 to clear any previous error codes. The GAS protocol manager will write an error code here if an error is encountered during command processing.

Offset 1 *Command Code* - The command code for the WRITE Command is 9729 (hex 2601).

Offset 2 *Connection Number* - The connection number must have been previously created using a CREATE CONNECTION command for the GAS protocol.

Offset 3 *Command Control Flag* - These bits are used to control command processing. See description on page 16.

Bits 2-14	Bit 16
Reserved. You should set these bits to 0.	**Cache Read** 0 = Read from V Memory 1 = Read from Cache

Offset 4 *V Memory Address of Format Table* - This value specifies the starting V memory address of the output Format Table. The Format Table is described in Chapter 3.

Offset 5 *Maximum Number of Format Specifications* - This word allows you to control the number of format specifications that GAS will read from V memory. If you set this to 0, GAS will default to reading a maximum of 6 format specifications (plus one TABLE END specification). If you have more than 6 format specifications, you must set this number to a value equal to or greater than the actual number of format specifications. The maximum value for this parameter is 24. If you set this to less than the number of format specifications in the table, GAS will return an error.

2.5. Read Command

When you trigger a READ command, GAS will wait for a message to be sent from the device. When a complete message has been received, as determined by the values in Offsets 6-9, the message will be processed using the Format Table located at the V memory address in offset 4. The amount of time the protocol manager will wait for a complete message before returning a timeout error is specified in offset 6.

Offset	Description	Hex Value	Decimal Value
0	Command Error Word	**0000**	**0**
1	Command Code (Read)	**2602**	**9730**
2	Connection Number (19221 - 19299)		
3	Command Control Flag (see below)	0000	0
4	V Memory Address of Format Table - Input		
5	Maximum # of Format Specifications (0 = default = 6).	0000	0
6	Command timeout in seconds (0 = default = 9 seconds).	0000	0
7	Input message maximum length		
8	Input message delimiters, first set		
9	Input message delimiters, second set		
10	End-of-Message Timeout in tenths of seconds (0 = Unused. 1 - 9 = 0.1 - 0.9 seconds)		
11-15	Unused (Set to 0)	**0000**	**0**

Offset 0 *Command Error Word* - Your PLC logic should set this value to 0 to clear any previous error codes. The GAS protocol manager will write an error code here if an error is encountered during command processing.

Offset 1 *Command Code* - The command code for the GAS READ command is 9730 (hex 2602).

Offset 2 *Connection Number* - The connection number must have been previously created using a Create Connection Command for the GAS protocol.

Offset 3 *Command Control Flag* - Bits used to control command processing. See detailed description on page 16.

Bit 1	Bits 2-14	Bit 15	Bit 16
Inhibit Buffer Clear 0 = Clear Input Buffer. 1 = Do not Clear Input Buffer	**Reserved.** You should set these bits to 0.	**Parsing Error Action** 0 = Stop processing on parsing error. 1 = Continue processing on parsing error.	**Cache Read** 0 = Read from V Memory 1 = Read from cache

Offset 4 *V Address of Format Specifications* - The first position of the input Format Table. The Format Table is described in Chapter 3.

Offset 5 *Maximum Number of Format Specifications* - This word allows you to control the number of format specifications that GAS will read from V memory. If you set this to 0, GAS will default to reading a maximum of 6 format specifications (plus one TABLE END specification). If you have more than 6 format specifications, you must set this number to a value equal to or greater than the actual number of format specifications. The maximum value for this parameter is 24. If you set this to less than the number of format specifications in the table, GAS will return an error.

Offset 6 *Command Timeout* - If the *entire* input message has not been received within the number of seconds specified, GAS will return a timeout error. A value of 0 results in a default timeout of 9 seconds. A value greater than 999 means infinite timeout (wait forever).

Offset 7 *Input Message Maximum Length* - the maximum number of characters (bytes) expected to be received in a message from the device. Valid entries are 1-1080.

Offset 8 *Input Message Delimiters, First Set* - Delimiters may be used to indicate the beginning and end of a device message. The character in the high byte specifies the start delimiter. The character in the low byte specifies the character used as the end delimiter. A byte value of 0x00 indicates that the particular delimiter is not used. If only one set of delimiters is required, you must use the first set to define them.

Offset 9 *Input Message Delimiters, Second Set* - Some device protocols may use more than one set of message delimiters. When you specify a second set using this word, GAS will look for either beginning delimiter in the incoming characters. Once the beginning delimiter is encountered, the corresponding ending delimiter will be used to determine the end of the message.

The character in the high byte specifies the start character. The character in the low byte specifies the character used as the end delimiter. A byte value of 0x00 indicates that the particular delimiter is not used.

NOTE:

*When two sets of input message delimiters are specified, incoming characters will be compared with both start delimiters to determine the beginning of the message. Once one of the delimiters has been used as the start character, the matching end delimiter will be used to determine the end of the message. If you specify two sets of delimiters**, both sets must have a beginning and end delimiter**.* *See Section 1.8. Input Message Processing, on page 10.*

Offset 10 *End-of-Message Interval* - Provides an alternate method for determining when all characters in the message have been received. You specify a maximum time interval between characters. If an additional character is not received before time interval expires, the message is considered complete. 0 indicates the timeout is not used. Maximum value is 9, representing 0.9 seconds.

2.6. Polled Read Command

A POLLED READ is a combination of a WRITE command and a READ COMMAND. This command will send a message to the device and wait for a response. The contents of the output message sent to the device is determined by the Format Table located at the V memory address in offset 4.

When a complete message has been received , as determined by the values in Offsets 9 - 11, the message will be decoded using the Format Table at the V memory address indicated in offset 6. The amount of time the protocol manager will wait for a message before returning a timeout error is determined by the value in offset 8.

Offset	Description	Hex Value	Decimal Value
0	Command Error Word	**0000**	**0**
1	Command Code (Polled Read)	**2603**	**9731**
2	Connection Number (19221-19299)		
3	Command Control Flag (see below)		
4	V Memory Address of Format Table - Output		
5	Maximum # of output Format Specifications (0 = default = 6)		
6	V Memory Address of Format Table - Input		
7	Maximum # of input Format Specifications (0= default = 6)		
8	Command timeout in seconds (0 = default = 9 seconds)		
9	Input message maximum length		
10	Input message delimiters, first set		
11	Input message delimiters, second set		
12	End-of Message Timeout in tenths of seconds (0 = Not used. 1-9 = 0.1 - 0.9 seconds)		
13 -15	Unused (Set to 0)	**0000**	**0**

The offsets are described as follows:

Offset 0 *Command Error Word* - Set to 0 so that any previous error code is cleared. The GAS protocol manager will write an error code here if an error is encountered during command processing.

Offset 1 *Command Code* - The command code for the POLLED READ command is 9731 (hex 2603).

Offset 2 *Connection Number* - The connection number must have been established previously using a CREATE CONNECTION command.

Offset 3 *Command Control Flag* - Bits used to control command processing. See detailed description on page 16.

Bit 1	Bits 2-14	Bit 15	Bit 16
Inhibit Buffer Clear 0 = Clear Input Buffer. 1 = Do not Clear Input Buffer	**Reserved.** You should set these bits to 0.	**Parsing Error Action** 0 = Stop processing on parsing error. 1 = Continue processing on parsing error.	**Cache Read** 0 = Read from V Memory 1 = Read from Cache

Offset 4 *V Memory Address of Output Format Specifications* - The V memory address of the first position of the Format Table used for output.

Offset 5 *Maximum number output Format Specifications* - This word allows you to control the number of output format specifications that GAS will read from V memory. If you set this to 0, GAS will default to reading a maximum of 6 format specifications (plus one TABLE END specification). If you have more than 6 format specifications, you must set this number to a value equal to or greater than the actual number of format specifications. The maximum value for this parameter is 24.

Offset 6 *V Memory Address of Input Format Specifications* - The V memory address of first position of the input Format Table.

Offset 7 *Maximum number of Input Format Specifications* - This word allows you to control the number of input format specifications that GAS will read from V memory. If you set this to 0, GAS will default to reading a maximum of 6 format specifications (plus one TABLE END specification). If you have more than 6 format specifications, you must set this number to a value equal to or greater than the actual number of format specifications. The maximum value for this parameter is 24.

Offset 8 *Command Timeout* - If the entire input message has not been received within the elapsed time specified, GAS will return a timeout error. For a POLLED READ, the elapsed time includes the time to build and send the output message as well as the time to fully receive the message. A value of 0 results in a default timeout of 9 seconds. A value greater than 999 means infinite timeout (wait forever).

Offset 9 *Input Message Maximum Length* - the maximum number of characters (bytes) expected to be received in a message from the device. Valid entries are 1 - 1080.

Offset 10 *Input Message Delimiters, First Set* - Delimiters may be used to indicate the beginning and end of a device message. The character in the high byte specifies the start character. The character in the low byte specifies the character used as the end delimiter. A byte value of 0x00 indicates that this delimiter is not used. See the examples in the following sections.

Offset 11 *Input Message Delimiters, Second Set* - Some device protocols may use more than one set of message delimiters. When you specify a second set using this word, GAS will consider a complete message received when either the first or second set is encountered.

The character in the high byte specifies the start character. The character in the low byte specifies the character used as the end delimiter. A byte value of 0x00 indicates that this delimiter is not used.

NOTE:
*When two sets of input message delimiters are specified, incoming characters will be compared with both start delimiters to determine the beginning of the message. Once one of the delimiters has been used as the start character, the matching end delimiter will be used to determine the end of the message. If you specify two sets of delimiters**, both sets must have a beginning and end delimiter**.* *See Section 1.8. Input Message Processing, on page 10.*

Offset 12 *End-of Message Interval* - Provides an alternate method for determining when all characters in the message have been received. Allows you to specify a maximum time interval between characters. If an additional character is not received before time interval expires, the message is considered complete. A value of 0 indicates the end-of message interval parameter is not used. The maximum value is 9, representing 0.9 seconds

2.7. Build Message Command

The BUILD MESSAGE command operates like a WRITE command except the ASCII message is written to PLC V memory rather than sent out the serial port. This command can be useful for development debugging and for building ASCII message strings which you want to use later. The content of the message is determined by the Format Table located at the V memory address in offset 4. The characters in the message will be written into V memory starting in the high byte of the V memory address indicated in offset 6.

Offset	Description	Hex Value	Decimal Value
0	Command Error Word	**0000**	**0**
1	Command Code (Build Message)	**2604**	**9732**
2	Connection Number (19221 - 19299)		
3	Command Control Flag	0000	0
4	V Memory Address of Format Table - Output		
5	Maximum # of Format Specifications (0 = Default = 6)	0000	0
6	Starting V Memory Address of Output String		
7- 15	Unused (Set to 0)	**0000**	**0**

Offset 0 *Command Error Word* - The GAS protocol manager will write an error code here if an error is encountered during command processing. Your PLC logic clear this value.

Offset 1 *Command Code* - The command code for the BUILD MESSAGE Command is 9732 (hex 2604).

Offset 2 *Connection Number* - The connection number must have been previously created using a CREATE CONNECTION command for the GAS protocol. Note that the Connection Number is required, even though you are not accessing a physical port. The module uses the Connection Number to reference a particular instance of a protocol manager.

Offset 3 *Command Control Flag* - Bits used to indicate unique command actions. This command uses the same flag bits as the WRITE command (see page 21) The CACHE READ bit can be set, if desired. However, except for some debugging operations, setting this bit is not particularly useful.

Offset 4 *V Memory Address of Format Table* - This indicates the starting V memory address of the output Format Table. The Format Table is fully described in Chapter 3.

Offset 5 *Maximum Number of Format Specifications* - This allows you to determine the number of format specifications that GAS will read from V memory. If you set this to 0, GAS will assume that the Format Table contains 6 format specifications (plus one TABLE END specification). If you have more than 6 format specifications, you must increase this number. The maximum value for this parameter is 24.

2.8. Parse Message Command

The PARSE MESSAGE command will use the Format Table identified in offset 4 to decode an ASCII message string stored in V memory. The operation is similar to the READ command, except that it operates on a message string stored in V memory, rather than from the serial port. This command is useful when handling complex protocols where you need to evaluate the contents of a message using different format specifications. See *Application Examples* chapter in this manual.

Offset	Description	Hex Value	Decimal Value
0	Command Error Word	**0000**	**0**
1	Command Code (Parse Message)	**2605**	**9733**
2	Connection Number (19221 - 19299)		
3	Command Control Flag (see below)	0000	0
4	V Memory Address of Format Table - Input		
5	Maximum # of Format Specifications (0 = default = 6).	0000	0
6	Command timeout in seconds (0 = default = 9 seconds).	0000	0
7	Input message maximum length		
8	Input message delimiters, first set		
9	Input message delimiters, second set		
10	Starting V memory address of input string		
11-15	Unused (Set to 0)	**0000**	**0**

Offset 0 *Command Error Word* - Your PLC logic should set this value to 0 to clear any previous error codes. The GAS protocol manager will write an error code here if an error is encountered during command processing.

Offset 1 *Command Code* - The command code for the PARSE MESSAGE command is 9733 (hex 2605).

Offset 2 *Connection Number* - The connection number must have been previously created using a Create Connection Command for the GAS protocol. Note that the Connection Number is required, even though you are not accessing a physical port. The module uses the Connection Number to reference a particular instance of a protocol manager.

Offset 3 *Command Control Flag* - Bits used to control command processing. See detailed description on page 16.

Bits 1-14	Bit 15	Bit 16
Reserved. You should set these bits to 0.	**Parsing Error Action** 0 = Stop processing on parsing error. 1 = Continue processing on parsing error.	**Cache Read** 0 = Read from V Memory 1 = Read from Cache

Offset 4 *V memory Address of Format Table* - The first position of the Format Table to be used for input. The Format Table is described in Chapter 3.

Offset 5 *Maximum Number of Format Specifications* - This word allows you to control the number of format specifications that GAS will read from V memory. If you set this to 0, GAS will default to reading a maximum of 6 format specifications (plus one TABLE END specification). If you have more than 6 format specifications, you must set this number to a value equal to or greater than the actual number of format specifications. The maximum value for this parameter is 24. If you set this to less than the number of format specifications in the table, GAS will return an error.

Offset 6 *Command Timeout* - The command will time out if the a complete message (based on the specified delimiters) cannot be found in V memory. A value of 0 results in a default timeout of 9 seconds. A value greater than 999 means infinite timeout (wait forever).

Offset 7 *Input Message Maximum Length* - the maximum number of characters (bytes) expected to be read from the V memory. Valid entries are 1-1080.

Offset 8 *Input Message Delimiters, First Set* - Delimiters may be used to indicate the beginning and end of a message. The character in the high byte specifies the start delimiter. The character in the low byte specifies the character used as the end delimiter. A byte value of 0x00 indicates that the particular delimiter is not used. If only one set of delimiters is required, you must use the first set to define them.

Offset 9 *Input Message Delimiters, Second Set* - Some device protocols may use more than one set of message delimiters. When you specify a second set using this word, GAS will look for either beginning delimiter in the incoming characters. Once the beginning delimiter is encountered, the corresponding ending delimiter will be used to determine the end of the message.

The character in the high byte specifies the start character. The character in the low byte specifies the character used as the end delimiter. A byte value of 0x00 indicates that the particular delimiter is not used.

NOTE:

When two sets of input message delimiters are specified, incoming characters will be compared with both start delimiters to determine the beginning of the message. Once one of the delimiters has been used as the start character, the matching end delimiter will be used

*to determine the end of the message. If you specify two sets of delimiters**, both sets must have a beginning and end delimiter**. See Section 1.8. Input Message Processing, on page 10.*

Offset 10 *Starting V Memory Address of Input String* - This specifies the first word in V memory which contains the ASCII message string which will be parsed.

CHAPTER 3. FORMAT SPECIFICATIONS

3.1. Overview

Device messages contain one or more data *fields* represented as an ASCII character string. A field is a contiguous group of characters which represent a unit of information. For messages received from the device, Format Specifications define how to decode data fields from the input message and where to store the results in PLC memory. For sending a message to a device, Format Specifications define how to encode data from specified PLC memory into fields in the output message. See *Section 1.3. PLC Data Formats* and *Section 1.4. Device Message Data Representation* on page 6 for an overview of formats.

3.2. Format Table

You may require several Format Specifications to describe how GAS will decode an input message or encode an output message. All of the format specifications used to encode or decode a particular device message are contained in a contiguous block of V memory called a *Format Table.* As noted in the previous chapter, the Command Blocks will reference the start location of the Format Table(s) used with the command. The following illustrates the structure of a Format Table.

Offset	Contents
0	Signature Value (0x4C00)
1 - 10	First Format Specification
11-20	Second Format Specification
21 - ?	Additional Format Specifications
? + 1	Table End Specification

The first word in a Format Table must contain a *signature value*. The signature value is used to verify that the memory location contains a Format Table. This word must contain the value 19456 (hex 4C00).

Following the Signature Value are the Format Specifications. Each Format Specification is exactly ten words long. GAS will process the Format Specification in the order in which they appear in the Format Table. You must include at least one Format Specification in the table. You may include a maximum of 24, a number which should be more than enough to process a single device message.

The entry in the Format Table must be a TABLE END specification - Format Number 65000 (0xFDE8). If you omit the TABLE END Specification, GAS will return an error. Format Specifications beyond the TABLE END specification will be ignored.

3.3. Format Specification General Information

Format Specification Structure

Format Specifications for data fields provide the information necessary to:

- Locate the beginning of field within the device message,
- Determine the size of the data field in characters,
- Convert the field contents between PLC format and message (ASCII Character) format.

Using the information in the format specifications, the GAS protocol manager can construct output messages to the device and process input messages from the device. See Chapter 1 for general terms and concepts related to message processing.

The general structure of a data format specification is shown below. The following sections describe individual format specifications in detail. Note that 10 words are reserved for a Format Specification.

Offset	Description	
1	Format Number	A unique number that determines how data is converted between PLC format and ASCII Character message format.
2	Start position	Position within the input or output message where the data field starts.
3	Length	Number of characters in the field. A value of -1 can be used with most numeric Format Specifications to process variable length fields.
4	Format Error Code	Used by GAS to report invalid specifications or errors interpreting the format specification. See Appendix B for a complete description of GAS specification errors.
5	Data Word 1	Usage depends on the Format Specification number.
6	Data Word 2	Usage depends on the Format Specification number
7	Data Word 3	Usage depends on the Format Specification number
8	Data Word 4	Usage depends on the Format Specification number
9	Data Word 5	Usage depends on the Format Specification number
10	Data Word 6	Usage depends on the Format Specification number

Format Specification - Output Processing

The WRITE command and the POLLED READ command both generate output messages consisting of ASCII characters. When these commands are initiated, the GAS protocol manager first fills the output buffer with the "?" character (hex 3F). Subsequent characters placed in the buffer using format specifications will overwrite the "?" character. When debugging the output message, you can look for embedded "?" to determine parts of the message that your format specifications have not written (assuming that the "?" character is not included in your output message).

Format specifications are processed in the order they appear in the Format Table. For example, a format specification at V2001 is processed before one at V 2011. Data Format Specifications positions may overlap. When they overlap, the last format specification encountered will overwrite any other characters in the overlapped positions of the output buffer. You will usually want to build the output message from left to right, but this is not required.

After all format specifications in the Format Table have been successfully processed, the output message is transmitted via the serial port. The last character in the output message will be the highest position in the buffer written using a format specification. See the example below.

Output Processing Example

When the output buffer is originally created, it contains the ASCII question mark character ("?"). *To conserve space, only a portion of the output buffer is shown.*

?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Assuming the first format specification writes the <STX> character in position 1, spaces (0x20) in positions 2 - 6 and the <ETX> character in position 7, the output buffer now looks like this:

STX						ETX	?	?	?	?	?	?	?	?	?
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Assume that the second format specification writes an unsigned decimal 34236 starting in position 2. The output buffer now contains:

STX	3	4	2	3	6	ETX	?	?	?	?	?	?	?	?	?
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

This is the highest position written by a format specification

The output message will be:

<STX>34236<ETX>

Format Specification - Input Processing

Both the READ command and the POLLED READ command will process input ASCII character messages. Input processing begins after a ***complete message*** is received from the device. The determination that a complete message has been received is based on the words in the command block which define maximum length and start/end delimiters. See Chapter 1 for information on message delimiters.

Format specifications are processed in the order they appear in the Format Table. Format Specifications may overlap. Only the fields that contain data you want to use in the PLC are required to have a format specification. Other characters can be ignored.

Input Example

Assume the following message was received from the device. The message contains one unsigned decimal integer field that you want to use in the PLC.

<STX>4146<ETX>

When a complete message has been received, the input buffer will contain the following:

STX	4	1	4	6	ETX										
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

You can decode only the data fields you wish to store in the PLC. In this case you would decode the field in positions 2 through 5. You don't have to create format specifications to decode the <STX> and <ETX> since you are not going to use them.

NOTE:
Since it requires time to transfer data to the PLC, your final application should decode only the data you need.

Format Error Processing

When the GAS protocol manager encounters an error processing a Format Specification, it:

1. Sets the Command Error Bit,
2. Places a format error code in the Format Error Word of the Format Specification, and
3. Places the command error code in the Command Error Word of the Command Block.

When an error occurs, you should first examine the Command Error Word to determine the type of error that has occurred. If the Command Error Code indicates a Format Error has occurred, then examine the Format Table for a non-zero Format Error Word. The Format Error Code should help you determine the cause of the problem. GAS does not reset the Format Error Code nor the Command Error Code to zero after the error is acknowledged. This must be done by PLC logic.

NOTE:
To prevent potential confusion, your PLC logic should reset the error codes to zero once you have corrected the error.

3.4. Format Specification Number Index

Following is an overview of the format numbers supported. Each format number is described in detail in the following sections. Format numbers are shown in decimal notation.

Fmt No.	Used for Output	Pg No	Used for Input	Pg No
1	Skip this format specification.	40	Skip this format specification.	40
1001	PLC unsigned integer to unsigned decimal integer ASCII string.	41	Unsigned decimal integer ASCII string to PLC unsigned integer.	43
1002	PLC real number to unsigned decimal fraction ASCII string.	45	Unsigned decimal fraction ASCII string to PLC real number.	47
1003	Unsigned PLC integer to hexadecimal ASCII string.	50	Hexadecimal ASCII string to unsigned PLC integer.	52
2001	PLC signed integer to signed decimal integer ASCII string.	54	Signed decimal integer ASCII string to PLC signed integer.	56
2002	PLC real number to signed decimal fraction ASCII string.	58	Signed decimal fraction ASCII string to PLC real number.	60
3001	PLC ASCII characters to ASCII character string. Up to 12 characters can be represented.	62	ASCII character string to PLC ASCII characters. Up to 12 characters can be represented.	64
3002	PLC ASCII characters to ASCII fixed string. Used to represent strings of 1-1080 characters.	66	ASCII character string to PLC ASCII characters. Used to represent strings of 1-1080 characters.	68
3003	PLC ASCII characters to ASCII variable length character string.	70	ASCII variable length character string to PLC ASCII characters.	72
4001	ASCII character fill	74	Not applicable	
4002	PLC ASCII characters (null terminated) to ASCII character string	76	Not applicable	
5001	Not Applicable		Position Data Cursor	78
6002	Modbus ASCII	84	Modbus ASCII	88
70xx	Block Check Character	93	Block Check Character	94
7031	Fletcher Checksum	98	Fletcher Checksum	101
8001	PLC Word LSB to Message Byte	102	Message Byte to PLC Word LSB	104
8002	PLC Word to Message Byte	105	Message Byte to PLC Word	108
8003	PLC Word to Message Byte – Byte Swap	109	Message Byte to PLC Word – Byte Swap	111
65000	Table End	40	Table End	40

NOTE:

Every Format Table must begin with a Signature Value and end with a Table End specification.

3.5. Bypass Format Specification

Bypass - Format 1

When offset 1 contains a value of 1, GAS will bypass the entire format specification. This can be useful during the development process. You can construct a format specification which you use for debugging, then disable it in the final application by replacing the format number you were using with the bypass format number.

Offset	Description	Comments
1	Format number 1	This tell GAS to skip processing this format specification. This enables you to logically delete a format specification without requiring you to physically clear the Format Table. This can be useful if you create some format specifications for debug which you will not use in the final version.
2-10	Unused	Ignored by this format specification.

3.6. Table End

Table End - Format 65000

Every Format Table must end with a Table End Format Specification number. If the GAS protocol manager does not find a *Table End* Format Specification (Format Number 65000), it will return an error code 0x2673 in the Command Error Word of the Command Block.

Offset	Description	Comments
1	Format number 65000	Signifies the end of the Format Table
2-10	Unused	Ignored by this specification.

NOTE:
The Table End Format Specification must begin in a standard Format Specification position (10 words after the previous Format Specification).

3.7. PLC Unsigned Integer To Unsigned Decimal Integer ASCII String

Output - 1001

This format specification encodes a PLC unsigned integer into an ASCII character string which represents an unsigned decimal integer (e.g. 14356) and places it in the output message buffer.

Offset	Description	Comments
1	Format number 1001	Converts between PLC Unsigned integer and Unsigned Decimal Integer ASCII string.
2	Start Position	Valid values are 1 - 1080 and -1. Specifies the start position of the field within the output buffer. A value of "-1" signifies to use the "next" position in the output buffer. If this is the first format specification in the table, the "next" position is position 1; else, it is the position immediately following the end of the previously defined field.
3	Length	Valid values are 1-9 and -1. A length value of 1-9 specifies the number of digits to place in the output message buffer. If the length exceeds the number of characters required to represent the number, the field is padded on the left with a pad character determined by Data Word 2. If the value for length is too small to represent the number, GAS will return an error code in the Format Error word. If a "-1" is used, the exact number of digits required to represent the numeric value is placed in the output message buffer.
4	Format Error Code	Format Error code. PLC logic should set to this zero when the command is initiated to clear any previous error codes. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Data Word 1 Unsigned Integer	Range is 0 - 65535. Place the unsigned integer value to output in Data Word 1.
6	Data Word 2 Pad Character	If the value of this word is 0, pad character will be the ASCII character 0 (0x30). Otherwise the pad character is determined by the ASCII character in the low byte of this word. **Do not specify a pad character that is a valid non-zero decimal number.**
7-10	Unused	Not used by this format specification.

Examples: PLC Unsigned Integer to Integer ASCII String

Example 1

<STX>DATA=123<ETX>

To encode the integer 123 into the ASCII string starting in position 7 of the above output message, the following output format specification could be used:

V2001	1001	PLC Unsigned Integer To Unsigned Decimal Integer ASCII String
V2002	7	Start Position
V2003	3	Length
V2004	0	Format Error Word - written only if an error is detected
V2005	123	Unsigned Integer Value To Place Into The Output String
V2006	0	Use default pad character of 0
V2007-2010	0	Unused By This Format Specification

Example 2

<STX>DATA=**123<ETX>

To encode the integer 123 into the ASCII string starting in position 7 of the above output message, the following output format specification could be used. Note that changing the length from 3 to 5 pads the number with 2 of the pad characters specified in V2006.

V2001	1001	PLC Unsigned Integer To Unsigned Decimal Integer ASCII String
V2002	7	Start Position
V2003	5	Length
V2004	0	Format Error Word - written only if an error is detected
V2005	123	Unsigned Integer Value To Place Into The Output String
V2006	0x002A	Pad Character is an asterisk (hex 2A).
V2007-2010	0	Unused By This Format Specification

3.8. Unsigned Decimal Integer ASCII String To PLC Unsigned Integer

Input - 1001

This format specification decodes an input ASCII character string representing an unsigned decimal integer to a PLC unsigned integer.

Offset	Description	Comments
1	Format number 1001	Converts between Unsigned Decimal Integer ASCII String and PLC Unsigned Integer.
2	Start Position	Valid values are 1 - 1080 and -1. Specifies the start position of the field within the input buffer. A value of "-1" signifies the "next" position in the input buffer. If this is the first format specification in the table, the "next" position is position 1. If the preceding format specification is a Format Number 5001, it is the specified data cursor position. Else, it is the position immediately following the end of the previously defined field.
3	Length	Valid values are 1-9 and -1. If you specify a value of 1-9, the length field specifies the number of characters which will be processed for input. If a character is encountered which is not a valid decimal digit (0-9) or a leading pad character specified in Data word 2, GAS will return an error. If you specify a value of -1, variable length is assumed. The first character after a decimal digit (0-9) that is *not* a valid decimal digit will delimit the field. If nothing is found which will delimit the field after 9 characters, GAS will return an error.
4	Format Error Code	PLC logic should set to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Data Word 1 Unsigned Integer	Range is 0 - 65535. Data 1 will contain the unsigned integer value obtained from the message field.
6	Data Word 2 Pad Character	If the value of this word is 0, pad character will be the ASCII character 0 (0x30). Otherwise the pad character is determined by the ASCII character in the low byte of this word. **Do not specify a pad character that is a valid non-zero decimal number.**
7-10	Unused	Not used by this format specification.

Examples: Unsigned Decimal Integer ASCII String to PLC Unsigned Integer

Example 1

<STX>DATA=123<ETX>

To decode the integer 123 from the ASCII string starting in position 7 of the above output message into V-memory, the following input format specification could be used:

V3001	1001	Unsigned Decimal Integer ASCII String To PLC Unsigned Integer
V3002	7	Start Position
V3003	3	Fixed Length of 3
V3004	0	Format Error Word - written only if an error is detected
V3005		Unsigned Integer Value Obtained From the message will be placed here.
V3006	0	Use default pad character of 0.
V3007-3010	0	Unused By This Format Specification

After the command has completed, V3004 will contain the integer value 123.

Example 2

The following input message has two spaces before the number in the above example:

<STX>DATA= 123<ETX>

Therefore you could use the following format specification to obtain the data.

V3001	1001	Unsigned ASCII Integer String To Integer Format Type
V3002	7	Start Position
V3003	5	Variable Length
V3004	0	Format Error Word - written only if an error is detected
V3005		Unsigned Integer Value obtained from the message will be placed here.
V3006	0x0020	Use a default pad character of ASCII blank (hex 20).
V3007-3010	0	Unused By This Format Specification

Note that, if you used a length of 3, you would read only a value of 1, ignoring the characters 2 and 3. The pad characters count in the length calculation.

Since a non decimal integer immediately follows the field (the ETX character) in the above examples, you could input the field as variable length data by changing the length specification to
-1.

3.9. PLC Real Number To Unsigned Decimal Fraction ASCII String

Output - 1002

This format specification encodes a PLC real number to an ASCII character string which represents an unsigned decimal fraction number (e.g. 143.56) and places it in the output message buffer.

Offset	Description	Comments
1	Format number 1002	Converts between PLC Real Number and Unsigned Decimal Fraction ASCII String.
2	Start Position	Valid values are 1 - 1080 and -1. Specifies the start position of the field within the output buffer. A value of "-1" signifies to use the "next" position in the output buffer. If this is the first format specification in the table, the "next" position is position 1; else, it is the position immediately following the end of the previously defined field
3	Length	Valid values for length are 2-16 and -1. Length values of 2-16 represent the number of decimal characters to place in the output message buffer, including the decimal point. If the length exceeds the number of characters required to represent the number, the field is padded on the left with a pad character determined by Data Word 4. If the value for length is too small, GAS will return an error code in the Format Error word. A value of -1 indicates variable length. The exact number of digits required to represent the numeric value is placed in the output message buffer.
4	Format Error Code	Format specification Format Error code. PLC logic should be set to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Data Word 1 Number of Decimal Fraction Digits	Valid values are 0-6 Data Word 1 specifies the number of digits to the right of the decimal. This value cannot exceed the value for length less 1. For example, if the length were 5, the maximum value would be 4.
6-7	Data Word 2-3 Real Value	A Real Number consists of 32 bits and requires 2 V memory words. You can output up to six significant decimal digits. Place the value to output in Data Word 2-3.
8	Data Word 4 Pad Character	If the value of this word is 0, pad character will be the ASCII character 0 (ox30). Otherwise the pad character is determined by the ASCII character in the low byte of this word. **Do not specify a pad character that is a valid non-zero decimal number or a decimal point.**
9-10		Unused by this format specification.

When converting between PLC real number format and decimal fraction format, some rounding errors may occur. Some decimal fractions cannot be represented exactly in a real number format.

Examples: PLC Real Number to Unsigned Decimal Fraction ASCII String

Example 1

<STX>DATA=100.1234<ETX>

To output the real number 100.1234 into the ASCII string starting in position 7 of the above output message, the following output format specification could be used.

V2001	1002	PLC Real Number To Unsigned Decimal Fraction ASCII String
V2002	7	Start Position
V2003	-1	Length (-1 means variable length)
V2004	0	Format Error Word - written only if an error is detected
V2005	4	4 digits to right of decimal point.
V2006-V2007	100.1234	Real Number Value To Place Into The Output String, (Real Numbers Use Two Words)
V2008	0	Use default Pad Character of ASCII 0 (0x30).
V2009-2010	0	Unused By This Format Specification

Example 2

<STX>DATA= 100.1234<ETX>

The following format specification will output the real number 100.1234 into the ASCII string at position 7 with two leading blanks.

V2001	1002	PLC Real Number To Unsigned Decimal Fraction ASCII String
V2002	7	Start Position
V2003	10	Length of 10
V2004	0	Format Error Word - written only if an error is detected
V2005	4	4 digits to right of decimal point.
V2006-V2007	100.1234	Real Number Value To Place Into The Output String.
V2008	0x0020	Use a pad character of ASCII blank (hex 20).
V2009-2010	0	Unused By This Format Specification

Example 3

However if you specified the length as 7 (see below) , GAS would return an error code in V2004, because the number and the decimal point cannot be represented in 7 characters.

V2001	1002	PLC Real Number To Unsigned Decimal Fraction ASCII String
V2002	7	Start Position
V2003	7	Length of 7
V2004	0	Format Error Word - written only if an error is detected
V2005	4	4 digits to right of decimal point.
V2006-V2007	100.1234	Real Number Value To Place Into The Output String
V2008	0x0020	Use Pad character of ASCII blank (hex 20).
V2009-2010	0	Unused By This Format Specification

3.10. Unsigned Decimal Fraction ASCII String To PLC Real Number

Input - 1002

This format specification converts an ASCII character string representing an unsigned decimal fixed point number to a PLC real number.

Offset	Description	Comments
1	Format number 1002	Converts between Unsigned Decimal Fraction ASCII String and PLC Real Number formats.
2	Start Position	Valid values are 1 - 1080 and -1. Specifies the start position of the field within the input buffer. A value of "-1" signifies the "next" position in the input buffer. If this is the first format specification in the table, the "next" position is position 1. If the preceding format specification is a Format Number 5001, it is the specified data cursor position. Else, it is the position immediately following the end of the previously defined field.
3	Length	Valid values for length are 1-16 and -1. For values of 1-16, the length specifies the number of characters which will be processed for input (including the decimal point). If a character that is not a decimal digit (0-9), leading pad character, or decimal point is found, GAS will return an error. If you specify a value of -1, variable length is assumed. The first character after a decimal digit (0-9) that is *not* a valid decimal digit or a decimal point will delimit the field. If GAS does not find something to delimit the field after examining 16 characters, it will return an error.
4	Format Error Code	PLC logic should set to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Data Word 1	Data Word 1 is unused for input.
6-7	Data Word 2-3 Real Value	Real numbers consist of 32 bits and require 2 data words. You can read up to 6 significant decimal digits. Data Word 2-3 will contain the value obtained from the message field.
8	Data Word 4 Pad Character	If the value of this word is 0, pad character will be the ASCII character 0 (0x30). Otherwise the pad character is determined by the ASCII character in the low byte of this word. **Do not specify a pad character that is a valid non-zero decimal number or a decimal point.**
9-10	Unused	Unused by this format specification.

When converting between PLC real number format and decimal fraction format, some rounding errors may occur. Some decimal fractions cannot be represented exactly in a real number format.

Example: Unsigned Decimal Fraction ASCII String to PLC Real Number

Example 1

<STX>DATA=100.1234<ETX>

Since the numeric field is followed by a non numeric character, you can use the following format specification to decode the real number 100.1234 from the ASCII string located at position 7 in the input message above.

V3001	1002	Unsigned Decimal Fraction ASCII String To PLC Real Number
V3002	7	Start Position
V3003	-1	Length (-1 means variable length)
V3004	0	Format Error Word - written only if an error is detected
V3005	0	This value is ignored
V3006-V3007		Real Value Placed Into V-Memory Two Words are Required To Represent a real number
V3008	0	Use default pad character of ASCII 0 (0x30).
V3009-3010	0	Unused

After the command completes successfully V3006-V3007 will contain the real number 100.1234.

Example 2

You could also explicitly specify the length as shown below:

V3001	1002	Unsigned Decimal Fraction ASCII String To PLC Real Number
V3002	7	Start Position
V3003	8	Length (including decimal point)
V3004	0	Format Error Word - written only if an error is detected
V3005	0	This value is ignored
V3006-V3007		Real Value Placed Into V-Memory.
V3008	0	Use default pad character of ASCII 0 (0x30).
V3009-3010	0	Unused

Example 3

However, if you entered the length as 9 (see below), GAS would return an error because it will include the <ETX> in the string it tries to decode.

V3001	1002	Unsigned Decimal Fraction ASCII String To PLC Real Number
V3002	7	Start Position
V3003	9	Length (including decimal point)
V3004	0	Format Error Word - written only if an error is detected
V3005	0	This value is ignored
V3006-V3007		Real Value Placed Into V-Memory.
V3008	0	Use default pad character of ASCII 0 (0x30).
V3009-3010	0	Unused

3.11. PLC Unsigned Integer To Hexadecimal ASCII String

Output - 1003

This format specification encodes a PLC unsigned integer into an ASCII character string which represents a hexadecimal number (e.g. 12AF).

Offset	Description	Comments
1	Format number 1003	Converts between PLC Unsigned Integer and Hexadecimal ASCII String.
2	Start Position	Valid values are 1 - 1080 and -1. Specifies the start position of the field within the output buffer. A value of "-1" signifies to use the "next" position in the output buffer. If this is the first format specification in the table, the "next" position is position 1; else, it is the position immediately following the end of the previously defined field
3	Length	Length can be a value between 1 - 4 or -1. A length value of 1-4 specifies the number of characters to place in the output message buffer. If the length exceeds the number of significant Hexadecimal characters, the output is padded with leading zeroes. If the length is too small to represent the number, and error is returned. A value of -1 indicates variable length. The exact number of hexadecimal characters required to represent the numeric value is placed in the output message buffer.
4	Format Error Code	PLC logic should set this word to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Data Word 1 Integer Value	Range is 0x0000-FFFF (0 - 65,535) Place the unsigned integer value to output in Data Word 1.
6	Data Word 2 Case Selection	If you set this value to 0, all hexadecimal characters will be output on upper case (e.g. 12AF). If you set the value to 1, all hexadecimal characters will be output in lower case (e.g. 12af).
6-10	Unused	Unused by this format specification.

Examples: PLC Unsigned Integer to Hexadecimal ASCII String

Example 1

<STX>DATA=FFB4<ETX>

To encode the hexadecimal number FFB4 into the ASCII string starting in position 7 of the above output message, the following output format specification could be used:

V2001	1003	PLC Unsigned Integer To Hexadecimal ASCII String
V2002	7	Start Position
V2003	4	Length of 4
V2004	0	Format Error Word - written only if an error is detected
V2005	65460	Unsigned Integer Value To Encode Into The Output String (decimal 65460 - 0xFFB4)
V2006	0	Select Upper Case Output
V2007-2010		Unused

Example 2

<STX>DATA=f4b<ETX>

The following format specification would encode the number into the output message shown above. Note the lower case output.

V3001	1003	Unsigned Integer To Hexadecimal ASCII String
V3002	7	Start Position
V3003	-1	Length (Variable Length)
V3004	0	Format Error Word - written only if an error is detected
V3005	3915	Unsigned Integer Value To Encode Into The Output String (decimal 3915 = 0xf4b)
V3006	1	Select Lower Case Output
V3007-3010	0	Unused

If you changed the length field from -1 to 4, the output would look like the following. Note the leading 0.

<STX>DATA=0f4b<ETX>

If you specified a length of 5, GAS would return an error because the maximum length is 4.

3.12. Hexadecimal ASCII String To PLC Unsigned Integer

Input - 1003

This format specification decodes an ASCII character string representing a hexadecimal number to a PLC unsigned integer. It accepts both upper and lower case hexadecimal characters.

Offset	Description	Comments
1	Format number 1003	Converts between Hexadecimal ASCII String and PLC Unsigned Integer.
2	Start Position	Valid values are 1 - 1080 and -1. Specifies the start position of the field within the input buffer. A value of "-1" signifies the "next" position in the input buffer. If this is the first format specification in the table, the "next" position is position 1. If the preceding format specification is a Format Number 5001, it is the specified data cursor position. Else, it is the position immediately following the end of the previously defined field.
3	Length	Length can be a value between 1 - 4 or -1. A length value of 1-4 specifies the number of characters which will be processed for input If a character is encountered which is not a valid hexadecimal character, GAS will return an error. A length value of -1 indicates variable length. The first character after the starting position that is not a valid ASCII hexadecimal character will delimit the field. If GAS finds nothing to delimit the field after 4 characters have been examined, it returns an error. Valid hexadecimal characters include ASCII 0-9 (0x30 - 0x39) ASCII A-F (0x41-0x46), and ASCII a-f (0x61-66).
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Data Word 1 Integer Value	Range: 0 - 65, 535 Data Word 1 will contain the unsigned integer value obtained from the message field.
6-10	Unused	Unused by this format specification.

NOTE:
Use caution when specifying a " -1" (variable length) input. If the following field is a character field, it could easily contain valid hexadecimal characters at times. If so, GAS could return an erroneous value or an error code. The effect will probably be intermittent, depending upon the specific input message characters.

Examples: Hexadecimal ASCII String to PLC Unsigned Integer

Example 1

To decode the hexadecimal number FFB4 from the ASCII string starting in position 7 of the above output message into V-memory, the following input format specification could be used:

V3001	1003	Hexadecimal ASCII String To Unsigned Integer
V3002	7	Start Position
V3003	4	Length
V3004	0	Format Error Word - written only if an error is detected
V3005	65460	Unsigned Integer Value Obtained From The ASCII String (0xFFB4=65460 decimal)
V3006-3010	0	Unused

This format specification would also decode lower case hexadecimal:
<STX>DATA=ffb4<ETX>

Example 2

You could also use -1 as the length (see below) because the string is always followed by a <STX>.

V3001	1003	Hexadecimal ASCII String To Unsigned Integer
V3002	7	Start Position
V3003	-1	Variable length
V3004	0	Format Error Word - written only if an error is detected
V3005	65460	Unsigned Integer Value Obtained Form The input message. (0xFFB4=65460 decimal)
V3006-3010	0	Unused

After the command completes successfully, V3004 will contain the hex value FFB4 (decimal value 65460).

Example 3

If the input message appeared as above, you could not use a -1 to decode the A34 (decimal 2612) from the message. GAS would return A34D (decimal 41805) from the message because the “D” is a valid Hexadecimal character but “I” is not.

3.13. PLC Signed Integer To Signed Decimal Integer ASCII String

Output - 2001

This format specification encodes a PLC signed integer to an ASCII character string which represents an signed decimal character integer (e.g. -14356).

Offset	Description	Comments
1	Format number 2001	Converts between PLC Signed Integer and Signed Decimal Integer ASCII string.
2	Start Position	Valid values are 1 - 1080 and -1. Specifies the start position of the field within the output buffer. A value of "-1" signifies to use the "next" position in the output buffer. If this is the first format specification in the table, the "next" position is position 1; else, it is the position immediately following the end of the previously defined field. If the number in Data Word 1 is positive, a "+" (0x2B) will be placed the start position. If the number is negative, a "-" (0x2D) will be placed in the start position.
3	Length	Valid values for length are 2 - 10 and -1. A value of 2-10 specifies of digits to place in the output message buffer, including the decimal and the sign. If the length exceeds the number of significant digits required to represent the number the field is padded on the left after the sign with the pad character specified in Data Word 2. If the value for length is too small, GAS will return an error code in the Format Error word. A value of -1 indicates variable length. The exact number of digits required to represent the numeric value is placed in the output message buffer.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Data Word 1 Signed Integer Value	Range: -32,768 to + 32,767 Place the value to output in Data Word 1.
6	Data Word 2 Pad Character	If the value of this word is 0, pad character will be the ASCII character 0 (0x30). Otherwise the pad character is determined by the ASCII character in the low byte of this word. **Do not specify a pad character that is a valid non-zero decimal number or sign.**
7-10	Unused	Unused by this format specification.

Example: PLC Signed Integer to Signed Decimal Integer ASCII String

Example 1

To encode the signed integer -123 into an ASCII string starting at position 7 in the above output message, the following output format specification could be used:

V2001	2001	PLC signed integer to signed decimal integer ASCII string
V2002	7	Start in position 7
V2003	4	Place 4 digits in the ASCII string (including sign)
V2004	0	Format Error Word - written only if an error is detected
V2005	-123	Signed integer value to place into the output string
V2006	0	Use default pad character of ASCII 0.
V2007 - V2010	0	Unused by this format

Example 2

If you wanted to encode the -123 into the message using 5 digits (6 characters including the sign) you could specify:

V2001	2001	PLC signed integer to signed decimal integer ASCII string
V2002	7	Start in position 7
V2003	6	Place 4 digits in the ASCII string (including sign)
V2004	0	Format Error Word - written only if an error is detected
V2005	-123	Signed integer value to place into the output string
V2006	0	Use default pad character of 0.
V2007 - V2010	0	Unused by this format

Assuming the format specification for the is set to a variable starting position, the output message would look like:

3.14. Signed Decimal Integer ASCII String To PLC Signed Integer

Input - 2001

This format specification decodes an ASCII character string representing a signed decimal integer to a PLC signed integer.

Offset	Description	Comments
1	Format number 2001	Converts between Signed Decimal Integer ASCII string and PLC Signed Integer.
2	Start Position	Valid values are 1 - 1080 and -1. Specifies the start position of the field within the input buffer. A value of "-1" signifies the "next" position in the input buffer. If this is the first format specification in the table, the "next" position is position 1. If the preceding format specification is a Format Number 5001, it is the specified data cursor position. Else, it is the position immediately following the end of the previously defined field. If the first character is a "+" (0x2B) or space (0x20), then the integer is interpreted as positive. If the first character is "-" (0x2D), then the integer is interpreted as negative
3	Length	Valid values for length are 2-10 and -1. A length value of 2-10 specifies the number of characters which will be processed for input, including the decimal and the sign. The first position must contain a "+", apace, decimal digit (0-9) or pad character. Following positions must contain a leading pad character or a decimal digit. If the above conditions are not met, GAS returns an error. A value of -1 indicates variable length. The first character after a decimal digit (0-9) that is *not* a valid decimal digit will delimit the field. If GAS finds nothing to delimit the field after 10 characters are examined, it returns an error.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Data Word 1 Signed Integer Value	Data Word 1 will contain the value obtained from the input message field.
6	Data Word 2 Pad Character	If the value of this word is 0, the pad character will be the ASCII character 0 (ox30). Otherwise the pad character is determined by the ASCII character in the low byte of this word. **Do not specify a pad character that is a valid non-zero decimal number or sign.**
7-10	Unused	Unused by this format specification.

Example: Signed Decimal ASCII String to PLC Signed Integer

To decode the signed integer -123 from the above input string into V-memory, the following input format specification could be used:

V3001	2001	Signed decimal integer ASCII string to PLC signed integer
V3002	7	Start position in the ASCII string to find the integer
V3003	4	Number of digits to include in the signed integer
V3004	0	Format Error Word - written only if an error is detected
V3005		Signed integer value obtained from the input message
v3006	0	Use the default pad character of ASCII 0.
V3007-3010	0	unused

After the command completes successfully, V3005 will contain the signed integer -123.

You could also use a "-1" (variable length) specification to decode the value.

NOTE:
In this example, if you set the length to 3 (forgetting to count the sign), GAS would retrieve a value of -12, ignoring the final 3. Although the application data is erroneous, GAS would ***not*** *return an error. Make sure the length includes the sign.*

3.15. PLC Real Number To Signed Decimal Fraction ASCII String

Output - 2002

This format specification converts a PLC real number to an ASCII character string which represents a signed decimal fixed point number (e.g. -143.56).

Offset	Description	Comments
1	Format number 2002	Converts between PLC Real Number and a Signed Decimal Fraction ASCII String.
2	Start Position	Valid values are 1 - 1080 and -1. Specifies the start position of the field within the output buffer. A value of "-1" signifies to use the "next" position in the output buffer. If this is the first format specification in the table, the "next" position is position 1; else, it is the position immediately following the end of the previously defined field. If the number in Data Word 1and 2 is positive, a "+" (0x2B) will be placed the start position. If the number is negative, a "-" (0x2D) will be placed in the start position.
3	Length	Valid values for length are 3 through 17, and -1. A length value of 3 through 17 specifies the number of digits to place in the output message buffer, including the decimal and the sign. If the length exceeds the number of characters required to represent the number (including the sign and decimal point), the field is padded on the left *after the sign* with the pad character specified in Data Word 4. If the value for length is too small, GAS will return an error code in the Format Error word. A value of -1 indicates variable length. The exact number of digits required to represent the numeric value is placed in the output message buffer.
4	Format Error Code	Set by GAS to non-zero value if an error is detected.
5	Data Word 1 Number of decimal fraction digits	Valid values are 0 - 6. Data Word 1 specifies the number of digits to the right of the decimal. Non significant decimal positions will be filled with zero. This value cannot exceed the value for length less 2. For example, if the length were 5, the maximum value would be 3 because of the sign and decimal point.
6-7	Data Word 2-3 Signed Real Value	Real Numbers are 32 bit and require 2 words. You can output up to 6 significant digits. Place the value to output in Data Words 2-3.
8	Data Word 4 Pad Character	If the value of this word is 0, the pad character will be the ASCII character 0 (ox30). Otherwise the pad character is determined by the ASCII character in the low byte of this word. **Do not specify a pad character that is a valid non-zero decimal number, sign. or decimal point.**
9-10	Unused	Unused by this format number

Example: PLC Real to Signed Decimal Fraction ASCII String

To output the real number +100.1234 into an ASCII string starting at position 7 in the output message above, the following output format specification could be used:

V2001	2002	PLC real to signed decimal fraction ASCII string
V2002	7	Start position in position 7
V2003	-1	-1 = variable length
V2004	0	Format Error Word - written only if an error is detected
V2005	4	Number of digits to the right of the decimal
V2006-V2007	100.1234	Real value to place into the output string, (a real number uses two words)
2008	0	Use the default pad character of ASCII 0.
V2009-2010	0	Unused by this format specification

You could also use a fixed length of 9 (7 digits + sign character + decimal point) to output the string. If you specified a value of 8 for the length, GAS would return an error, since the number could not be represented in 8 characters.

To output the data field with blank pad characters into the following message, see the example below:

V2001	2002	PLC real to signed decimal fraction ASCII string
V2002	7	Start position in position 7
V2003	11	Field Length is 11
V2004	0	Format Error Word - written only if an error is detected
V2005	4	Number of digits to the right of the decimal
V2006-V2007	100.1234	Real value to place into the output string, (a real number uses two words)
2008	0	Use the default pad character of ASCII 0.
V2009-2010	0	Unused by this format specification

3.16. Signed Decimal Fraction ASCII String To PLC Real Number

Input - 2002

This format specification converts an ASCII character string representing an unsigned decimal fixed point number to a PLC real number.

Offset	Description	Comments
1	Format number 2002	Converts between a Signed Decimal Fraction ASCII String and a PLC Real Number.
2	Start Position	Valid values are 1 - 1080. Start position within the input buffer. A "-1" signifies to use the next position in the input buffer. If this is the first format specification, a "-1" signifies position 1. If the first character is a "+" (0x2B) or space (0x20), then the integer is interpreted as positive. If the first character is "-" (0x2D), then the integer is interpreted as negative.
3	Length	Valid values for length are 2-17, and -1. A value of 2-17 specifies the number of characters that will be processed for input, including the decimal and the sign. If a character which is not a valid decimal digit (0-9), pad character, plus sign (+), minus sign (-), blank, or decimal point is encountered, GAS will return an error. A value of -1 indicates variable length. The first character after a decimal digit (0-9) that is *not* a valid decimal digit will delimit the field. If more than 17 characters have been examined without finding a delimiting character, GAS will return an error.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Data Word 1	Data Word 1 is unused.
6-7	Data Word 2-3 Signed Real Value	Real Numbers are 32 bit format and require 2 data words. You can output up to 6 significant digits. Data 2-3 will contain the signed real value decoded from the input message ASCII string.
8	Data Word 4 Pad Character	If the value of this word is 0, the pad character will be the ASCII character 0 (0x30). Otherwise the pad character is determined by the ASCII character in the low byte of this word. **Do not specify a pad character that is a valid non-zero decimal number, sign. or decimal point.**
9-10	Unused	Unused by this format specification.

When converting between PLC real number format and decimal fraction format, some rounding errors may occur. Some decimal fractions cannot be represented exactly in a real number format.

Example: Signed Fixed Point ASCII String to PLC Real Number

Example 1

To decode the real number +100.1234 from the input string starting in position 7 of the above input message into V-memory, the following input format specification could be used:

V3001	2002	signed fixed point ASCII string to real format type
V3002	7	start position in the ASCII string to find the real
V3003	-1	-1 = variable length (<ETX> will delimit)
V3004	0	Format Error Word - written only if an error is detected
V3005	0	this value is ignored
V3006 V3007		Real number obtained from the message, 2 words are required to represent real numbers.
V3008	0x0020	Set the pad character to ASCII blank (hex 20)
V3009-3010	0	Unused by this format specification

The -1 variable length specification works properly because the numeric ASCII string is followed by a non-numeric character . After the command completes successfully, V3006-V3007 will contain the real number 100.1234.

The above specification will also properly decode real number from the following messages:

The following message would cause an error using this specification because the + sign follows the pad character.

Example 2

If you use a fixed length specification and the field size actually varies you will get unpredictable results. The following message represents a device which returns a variable length field where the first character is a digit if the sign is positive and a minus sign if the field is negative.

If you use a fixed length of 9, then when you receive a positive number you will get a format error because GAS tries to include the <ETX> in the numeric field. Conversely, if you use a fixed length of 8, the when you receive a negative number you will not decode the last digit. Thus your data would be -100.123 not -100.1234.

3.17. PLC ASCII Character To ASCII Character String

Output - 3001

This format specification converts PLC ASCII Characters to an ASCII character string. The output ASCII Characters are contained in the Format Specification data words. A maximum of 12 characters can be written using this format specification.

Offset	Description	Comments
1	Format number 3001	Converts between PLC ASCII Character and ASCII Character String.
2	Start Position	Valid values are 1 - 1080 and -1. Specifies the start position of the field within the output buffer. A value of "-1" signifies to use the "next" position in the output buffer. If this is the first format specification in the table, the "next" position is position 1; else, it is the position immediately following the end of the previously defined field
3	Length	Valid values for length are 1-12. Length is the number of characters to be placed in the output message buffer. A value of -1 is invalid
4	Format Error Code	PLC logic should be set to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5-10	Data Word 1-6	Place the characters to output in Data Words 1-6. The first character is located in the high byte of Data word 1. The second character is located in the low byte of Data word 1. Data Word 2 contains characters 3 and 4; the remaining data words follow the same pattern. TISOFT provides an ASCII string entry which facilitates data entry.

NOTE:

When you are creating output strings of 12 characters or less, always use this format, rather than format number 3002. Since the data is contained in the format specification, it saves a PLC access which would be required to obtain the data from another V memory location.

Example: PLC ASCII Character to ASCII Character String

To output the characters “ into an ASCII character string starting at position 1 of the output message, the following output format specification could be used:

V2001	3001	ASCII character to ASCII character format type
V2002	1	start position in the ASCII string to place the characters
V2003	6	number of characters to include in the output string
V2004	0	Format Error Word - written only if an error is detected
V2005	0x0244	character data “<STX>D”
V2006	0x4154	character data “AT”
V2007	0x413D	character data “A=“
V2008-2010	0	Unused

3.18. ASCII Character String To PLC ASCII Character

Input - 3001

This format specification converts an ASCII character string within a message to a series of PLC ASCII characters. The input characters are stored in the Format Specification data words. This format allows a maximum of 12 characters to be read. If you specify less than 12 characters, a null character (0x00) will be placed in the remaining byte positions.

Offset	Description	Comments
1	Format number 3001	Converts between ASCII Character String and PLC ASCII Characters.
2	Start Position	Valid values are 1 - 1080 and -1. Start position within the input buffer. A "-1" signifies to use the next position in the ASCII string. If this is the first format specification, a "-1" signifies position 1.
3	Length	Valid values for length are 1-12. Length represents the number of characters to be processed as input. If the length exceeds the end of the message, GAS returns an error.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5-10	Data Word 1-6	Data 1-6 will contain the characters obtained from the message field. The first character is located in the high byte of Data word 1. The second character is located in the low byte of Data word 1. Data Word 2 contains characters 3 and 4; the remaining data words follow the same pattern.

Example: ASCII Character String to PLC ASCII Characters

To copy the character string **A12C** from the above input message into V-memory, the following input format specification could be used:

V3001	3001	ASCII character to ASCII character format type
V3002	7	start in position 7 to find the characters
V3003	4	read 4 characters
V3004	0	Format Error Word - written only if an error is detected
V3005		Character data obtained from the message
V3006		Character data obtained from the message
V3007-3010		GAS will set the bytes in these words to 0x00

After the command is successfully completed, V3005 will contain 0x4131 (ASCII A and 1) V3006 will contain 0x3243 (ASCII 2 and C). V3007-V3010 will contain 0x0000.

3.19. PLC ASCII Character To ASCII Character String (Extended)

Output - 3002

This format specification converts a series of PLC ASCII Characters to an ASCII character string. Output Characters are obtained from a V memory location outside the Format specification. This format allows you to read and write character strings from 1- 1080 characters in length. It will typically be used instead of 3001 when you want to output more than 12 characters.

Offset	Description	Comments
1	Format number 3002	Converts between PLC ASCII Character and ASCII Character String for large strings.
2	Start Position	Valid values are 1 - 1080 and -1. Specifies the start position of the field within the output buffer. A value of "-1" signifies to use the "next" position in the output buffer. If this is the first format specification in the table, the "next" position is position 1; else, it is the position immediately following the end of the previously defined field
3	Length	Valid values for length are 1-1080. Length is the number of characters to be placed in the output buffer. GAS will return an error if the size of the output buffer is exceeded.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Data Word 1	Set the Data Word 1 to the address of the V memory location that contains the first character to output. The first character is located in the high byte of V memory referenced in Data Word 1. The second character is located in the low byte. The remaining characters exhibit the same pattern in the following V memory words.

NOTE:
To output character strings of 12 characters or less, always use Format 3001 instead of 3002. After the format specification is processed, Format 3002 requires an additional PLC access to obtain the data referenced in Data Word 1. Format 3001 contains the data within the format specification, thus eliminating the time required for the extra access.

Example: PLC ASCII Character to ASCII Character String (Extended)

To output the character string " " into the above output message, the following output format specification could be used:

V2001	3002	PLC ASCII character to ASCII character string (Extended)
V2002	1	Start position 1
V2003	35	Place 35 characters in the output buffer
V2004	0	Format Error Word - written only if an error is detected
V2005	4000	V-memory location to find the extended character data
V2006-2010	0	Unused by this format specification

The extended character string at location V4000 should contain the following data:

V4000	0x024E	ASCII <STX> and N
V4001	0x554D	ASCII U and M
V4002	0x4245	ASCII B and E
.		etc.
.		etc.
V4016	0x4953	ASCII I and S
V4017	0x3A00	ASCII : and NULL

Since the length of the string in this example is an odd number, the low byte in V4017 is ignored. The hex character in the low byte of this example happens to contain an ASCII (0x00).

3.20. ASCII Character String To PLC ASCII Character (Extended)

Input - 3002

This format specification converts an ASCII character string to PLC ASCII Characters. The Input characters are stored in a V memory location other than the Format specification. This format allows you to process character strings from 1- 1080 characters in length. It will typically be used instead of 3001 when you are want to input more than 12 characters.

Offset	Description	Comments
1	Format number 3002	Converts between PLC ASCII Character and ASCII Character String for large strings
2	Start Position	Valid values are 1 - 1080 and -1. Start position within the input buffer. A "-1" signifies to use the next position in the input buffer. If this is the first format specification, a "-1" signifies position 1.
3	Length	Valid values for length are 1-1080. Length represents the number of characters to be processed as input. GAS will return an error if the message length is exceeded.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Data Word 1	Set Data Word 1 to the address of the V memory location that will contain the first character. The first character is located in the high byte of V memory referenced in Data Word 1. The second character is located in the low byte. The remaining characters exhibit the same pattern in the following V memory words.

Example: ASCII Character String to PLC ASCII Character (Extended)

To obtain the characters "<STX>NUMBER OF BOARD FEET OF LUMBER IS: " from the above input string and store into V-memory, the following input format specification could be used:

V3001	3002	extended ASCII character to ASCII character format type
V3002	1	start position in the ASCII string to find the characters
V3003	35	Read 35 characters
V3004	0	Format Error Word - written only if an error is detected
V3005	4000	Beginning V-memory location to store the characters
V3006-3010	0	Unused by this format specification

The extended character string would then reside at V4000 as follows:

V4000	0x024E	ASCII <STX> and N
V4001	0x554D	ASCII U and M
V4002	0x4245	ASCII B and E
.		etc.
.		etc.
V4016	0x4953	ASCII I and S
V4017	0x3A00	ASCII : and NULL

Since the character string length in this example is odd, the last character will be contained in the high byte of V4017. GAS will set the low byte to an ASCII null.

3.21. PLC ASCII Characters To Variable Length ASCII Character String

Output - 3003

This format specification converts a series of PLC ASCII Characters stored in V memory to an ASCII character string. The number of characters to output is stored in a V memory location immediately preceding the data. This format is typically used to output variable length character data which you have obtained via a previous input using this format. See the following section.

Offset	Description	Comments
1	Format number 3003	Converts between PLC ASCII Characters and a variable length ASCII Character String.
2	Start Position	Valid values are 1 - 1080 and -1. Specifies the start position of the field within the output buffer. A value of "-1" signifies to use the "next" position in the output buffer. If this is the first format specification in the table, the "next" position is position 1; else, it is the position immediately following the end of the previously defined field
3	Length	The length value is ignored. The number of characters is obtained from the V memory location specified in Data word 1.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Data Word 1	Set the Data Word 1 to the address of the V memory location that contains the length value. The character data should be placed contiguous V memory locations immediately following this word. The first character is located in the high byte of V memory location following the address referenced in Data Word 1. The second character is located in the low byte. The remaining characters exhibit the same pattern in the following V memory words.
6 - 10	Unused	You should set these words to 0.

NOTE:
Make sure that you have specified the correct V memory location in Data Word 1. If you specify the incorrect address and it contains a non zero value, you will not get an error code but you will probably get some verrrry interesting (and possibly unpleasant) results!

Example: PLC ASCII Characters to Variable Length ASCII Character String

Assuming V 4000 contained the following data:

V4000	10	Number of Characters to output
V4001	0x0231	ASCII <STX> and 1
V4002	0x4142	ASCII A and B
V4003	0x3534	ASCII 5 and 4
V4004	0x3337	ASCII 3 and 7
V4005	0x4344	ASCII C and D

You could use the following format specification:

V3001	3003	PLC ASCII character to variable length ASCII character string
V3002	1	Start in the first position of the output message buffer
V3003	0	Ignored (length is in V memory)
V3004	0	Format Error Word - written only if an error is detected
V3005	4000	Length is stored in V4000, characters follow
V3006-3010	0	Unused by this format specification

to generate the following output message:

3.22. Variable Length ASCII Character String To PLC ASCII Characters

Input - 3003

This format converts all characters in the input buffer beginning with the specified starting location through the end of the input message to PLC ASCII characters. GAS places a value representing the number of characters actually read in the V memory location specified in Data Word 1. The actual characters will then be placed in following contiguous V memory locations.

This format is very useful for troubleshooting, because you can read the entire message from the device, store it in V memory, then use TISOFT to display the message. You may also find this format specification valuable for acquiring variable length message data from one device so that you can output it later to another device. For example you could read from a bar code reader attached to port 1 using format 3003 for input then write it to a display attached to port 2 using 3003 for output.

Offset	Description	Comments
1	Format number 3003	Converts between a variable length ASCII character string and PLC ASCII characters.
2	Start Position	Valid values are 1 - 1080 and -1. Start position within the input buffer. A "-1" signifies to use the next position in the input buffer. If this is the first format specification, a "-1" signifies position 1.
3	Length	Valid values for length are 1-1080. Length represents the maximum number of characters which will be stored in V memory. This value is used to prevent GAS from unexpectedly overwriting V memory used for something else. You should ensure that you have reserved enough V memory to hold the maximum number of characters specified plus one word for the length data. GAS will determine whether the number of characters in the input buffer starting at the location specified in offset 2 through the end of the input message exceeds the value entered for length. If so, GAS will not store the characters in V memory and will return an error code.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Data Word 1	Set Data Word 1 to the address of the V memory location that will contain the length value. The first character is located in the high byte of V memory following the address referenced in Data Word 1. The second character is located in the low byte. The remaining characters exhibit the same pattern in the following V memory words.
6 - 10	Unused	You should set these words to 0.

NOTE:
To prevent accidental overwriting of V memory used for other purposes, make sure that the V memory value you have specified in Data Word 1 is correct
.

Example: Variable Length ASCII Character String to PLC ASCII Character

Example 1

To copy all of the characters from the above input message into V-memory, the following input format specification could be used:

V3001	3003	extended ASCII character to ASCII character format type
V3002	1	start position in the ASCII string to find the characters
V3003	12	Reserve memory for 12 characters (this message actually has 10)
V3004	0	Format Error Word - written only if an error is detected
V3005	4000	Beginning V-memory location to store the characters
V3006-3010	0	Unused by this format specification

The data stored in V memory, starting at V4000 would be:

V4000	10	Number of Characters actually stored
V4001	0x0231	ASCII <STX> and 1
V4002	0x4142	ASCII A and B
V4003	0x3534	ASCII 5 and 4
V4004	0x3337	ASCII 3 and 7
V4016	0x4344	ASCII C and D

Example 2

If you received the message above, GAS would return an error for the following specification.

V3001	3003	extended ASCII character to ASCII character format type
V3002	1	start position in the ASCII string to find the characters
V3003	8	Reserve memory for 8 characters (this message actually has 10)
V3004	0	Format Error Word - written only if an error is detected
V3005	4000	Beginning V-memory location to store the characters
V3006-3010	0	Unused by this format specification

The error will occur because the value for length is too small. 10 characters were detected but space for only 8 characters was reserved.

3.23. Character Fill

Output - 4001

This format specification places a specified number of fill characters in the output message.

Offset	Description	Comments
1	Format number 4001	Fill with Specified Character
2	Start Position	Valid values are 1 - 1080 and -1. Specifies the start position of the field within the output buffer. A value of "-1" signifies to use the "next" position in the output buffer. If this is the first format specification in the table, the "next" position is position 1; else, it is the position immediately following the end of the previously defined field
3	Length	Valid values for length are 1-1080. Length is the number of fill characters to be placed in the output message buffer.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Data Word 1	Place the fill character in the high byte of Data Word 1. Fill character will be replicated based on the length value in Offset 3.
6 - 10	Unused	You should set these words to 0.

Example: Character Fill

To output the characters “*****“ into the above output string, the following output format specification could be used:

V2001	4001	Character fill format type
V2002	2	Start in position 2
V2003	5	Place 5 characters in the output buffer
V2004	0	Format Error Word - written only if an error is detected
V2005	0x2A00	fill character is ASCII * (asterisk)
V2006-2010	0	Unused by this format specification

The Character Fill format is valid only on output messages.

You may find this format specification especially useful when constructing output print messages. It allows you to create a fixed length message of space characters, then use subsequent format specifications to overwrite the output buffer with data you want to print.

3.24. PLC ASCII Character (Null Terminated) To ASCII Character String

Output - 4002

This format specification places a series of ASCII characters located in PLC V memory in an output message. The number of characters placed in the message is controlled by the presence of a null terminator. GAS will start reading characters at the V memory location specified in Data Word 1. It will continue to read characters until a null terminator is detected or until the maximum number of bytes specified in the Length word have been read. If no null terminator is found, GAS will return an error.

This format can accomplish the same output result as Format number 3002. However, rather than specifying a fixed output length, you place a null character after the last character. It is useful in situations where you are building the character string using programming tools such as TISOFT.

Offset	Description	Comments
1	Format number 4002	PLC ASCII Character (Null Terminated) To ASCII Character String
2	Start Position	Valid values are 1 - 1080 and -1. Specifies the start position of the field within the output buffer. A value of "-1" signifies to use the "next" position in the output buffer. If this is the first format specification in the table, the "next" position is position 1; else, it is the position immediately following the end of the previously defined field
3	Length	Valid values for length are 1-1080. The value for length determines the maximum number of bytes of V memory that will be examined to locate the null character terminator. If no null character is encountered within this range, GAS returns an error. When a null character is encountered, the characters preceding the null, starting with the V memory location specified in Data Word 1, are placed in the output buffer.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Data Word 1	Specifies the address of the V memory location which contains the first character. The first character is stored in the high byte of the first word; the second character is stored in the low byte of the first word. The third character is stored in the high byte of the second word; etc.
6 - 10	Unused	You should set these words to 0.

Example: PLC ASCII (Null Terminated) to ASCII Character String

Example 1

<STX>ERR34<CR><LF>

To output the above output message, the following output format specification could be used:

V2001	4002	PLC ASCII Character (Null Terminated) To ASCII Character String
V2002	1	start position in the ASCII string to place the characters
V2003	10	Examine a maximum of 10 V memory bytes
V2004	0	Format Error Word - written only if an error is detected
V2005	200	Start in V Memory 200
V2006-2010	0	Unused by this format specification

You would enter the following in V memory:

Location	Value (hex)	Comments
V200	0245	ASCII <STX> and E
V201	5252	ASCII R and R
V202	3334	ASCII 3 and 4
V203	0D0A	ASCII <CR> and <LF>
V204	0000	Null

Example 2

Assuming that V200-V204 contained the same data as the example above, the following format specification would result in an error.

V2001	4002	PLC ASCII Character (Null Terminated) To ASCII Character String
V2002	1	start position in the ASCII string to place the characters
V2003	8	Examine a maximum of 8 V memory bytes
V2004	0	Format Error Word - written only if an error is detected
V2005	200	Start in V Memory 200
V2006-2010	0	Unused by this format specification

The error would occur because GAS would examine 8 characters, but would not find a null terminator within the eight characters.

3.25. Position Data Cursor

Input - 5001

Most data format specifications allow you to define a start position value of -1 to indicate the "next" position. The "next" position is determined by the location of a reference pointer called the *data cursor*. After a standard data format specification is processed, the data cursor is implicitly located one character past the end of the defined field. Thus, the subsequent specification can start immediately following the previously defined field.

There may be situations, however, where you need to explicitly set the data cursor. This format specification lets you accomplish this function by allowing you to define an anchor position and then to move the data cursor forward or backward relative to the anchor. A data format specification immediately following this format specification will start where you positioned the data cursor if it uses the "next" position.

Offset	Description	Comments
1	Format number 5001	Position Data Cursor
2	Anchor Position	1 = Beginning of Message 2 = End of Message 3 = Current Position 4 = Beginning of previously defined field 11= Next occurrence of character string defined in Data Word 2 - 6. 12 = 2nd next occurrence of character string 13 = 3rd next occurrence of character string 21 = Previous occurrence of character string defined in Data Word 2-6. 22 = 2nd previous occurrence of character string 23 = 3rd previous occurrence of character string
3	Relative Movement	+ n = Move forward n characters from anchor position defined above. - n = Move backward n characters from anchor position 0 = No movement from the anchor position.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Data Word 1	Reserved. This should be set to 0 in this release.
6	Data Word 2	Number of Characters in Character String. When the anchor position is 11 - 13 or 21 - 23, this value specifies the length of the character string. Valid values are 1 - 8. The characters in the character string are specified in Data Word 3 - 6. The value in this word is ignored when the anchor position is 1 - 4. For clarity, you should set it to 0.
7	Data Word 3	Characters 1 and 2. Character 1 is located in the high byte and character 2 (if used) is located in the low byte.
8 - 10	Data Word 4 - 6	Characters 3 and 4, 5 and 6, 7 and 8 (if used).

Anchor Position

Beginning of Message	-	The anchor position is set to the first character of the message.
End of Message	-	The anchor position is set to the last character in the message.
Current Position	-	The anchor position is set to the current data cursor position
Beginning of Previous Field		The anchor position is set to the start position of the field defined in the format specification immediately preceding this one.
Next Occurrence of Character String	-	The message buffer is searched forward from the current data cursor position for the occurrence of a specific character string. If the character string is located, the data cursor is set to the position of the first character in the character string. If the string cannot be located, an error is returned. The comparison character string can be entered in data words 3 - 6. The first character is located in the high byte of data word 3. Data word 2 defines the length of the character string. Characters in data words 3 - 6 beyond the length are ignored. Thus, if length were set to 3 and data word 3 contained 0x4142 (ASCII “AB”) and data word 4 contained 0x4344 (ASCII “CD”), then the search string would be ASCII “ABC”. The low byte of data word 4 is ignored since length is set to 3.
2nd Next Occurrence of Character String	-	Same as the above except the message buffer is searched for the second occurrence. If the second occurrence of the string cannot be located, an error is returned.
3rd Next Occurrence of Character String	-	Same as the above except the message buffer is searched for the third occurrence. If the third occurrence of the string cannot be located, an error is returned.

Previous Occurrence of Character String	-	The message buffer is searched backward from the current data cursor position for the occurrence of a specific character string. If the character string is located, the data cursor is set to the position of the first character in the character string. If the string cannot be located, an error is returned. The comparison character string is defined the same as above.
2nd Previous Occurrence of Character String	-	Same as the above except the message buffer is searched backward for the second occurrence. If the second occurrence of the string cannot be located, an error is returned.
3rd Previous Occurrence of Character String	-	Same as the above except the message buffer is searched backwards for the third occurrence. If the third occurrence of the string cannot be located, an error is returned.

Examples - Position Data Cursor

Assume the input message **<STX>123ABC45<ETX>** is stored in the input message buffer as:

1	2	3	4	5	6	7	8	9	10
<STX>	**1**	**2**	**3**	**A**	**B**	**C**	**4**	**5**	**<ETX>**

Example 1

The following Position Data Cursor Specification would place the data cursor at position 8. Anchor position starts at end of message; movement is backwards two character positions.

Offset	Value	Comments
1	5001	Position Data Cursor
2	2	2 = End of Message
3	-2	Move backward 2 characters from anchor position
4	0	Format Error Code
5	0	Reserved.
6-10	0	Not Used

Example 2

The following format specification also places the data cursor at position 8. The anchor position is at the ***first*** character in the string "ABC" and the relative movement is forward 3 positions from the beginning of the character string. This example uses a forward search, which assumes that the data cursor is before the character string prior to executing this format specification.

Offset	Value	Comments
1	5001	Position Data Cursor
2	11	Search Buffer forward for character string for anchor position
3	3	Move forward 3 characters from anchor position
4	0	Format Error Code
5	3	The character string is 3 characters long.
6	0x4142	ASCII Characters "A" and "B".
7	0x4300	ASCII Character "C" (low order byte is unused)
8	0	Unused

Example 3

This format specification produces an error because it attempts to move the data cursor beyond the end of the input message (1 + 10 = 11).

Offset	Value	Comments
1	5001	Position Data Cursor
2	1	1 = Beginning of message (position 1)
3	10	Move forward 10 characters
4	0	Format Error Code

5	0	Reserved.
6-10	0	Not Used

3.26. Modbus ASCII Master

Note:
The Modbus ASCII protocol manager is now provided on the 2573-MOD module from CTI. Where you need faster performance and more function, you should use this fixed function protocol managers rather than GAS. In addition, the 2573-MOD supports Modbus RTU.

The Modbus ASCII Master format specification allows you to communicate with plant floor devices which use the Modbus Protocol with ASCII message framing. This protocol is fully specified in the *Modicon Modbus Protocol Reference Guide* (PI-MBUS-300 Rev E) published by: MODICON, Inc., Industrial Automation Systems
One High Street
North Andover,
MA 01845

NOTE:
This format specification assumes you are familiar with the Modbus ASCII protocol. If you are not, you should obtain Modbus information before proceeding.

MODBUS is a command /response protocol. As a master, you send a command and the slave responds. You must use the POLLED READ command block to implement this protocol (see Chapter 2, page 24). A POLLED READ sends an output message, then waits for a response. The command block requires two Format Tables, one for output and one for input. For Modbus, the output specification describes how to build the Modbus query message; the input specification describes how to process the response message. In offset 10 of the command block, you must set the message start delimiter to 0x3A and the message end delimiter to 0x0A)

The general structure of a Modbus ASCII message is shown below:

Start	Address	Function	Data	LRC Check	End
: (0x3A)	2 Chars	2 Chars	n Chars	2 Chars	<CR> <LF>

Start: The message begins with an ASCII Colon (hex 3A).
Address: The address field designates the slave address on a multidrop line.
Function: The function code specifies the task to be performed (e.g. Read Input Register).
Data: This field contains the data to/from the device. It may also contain a byte count.

LRC Check: This field is a Longitudinal Redundancy Check used for error checking.
End: The message ends with an ASCII Carriage Return (0x0D) and Line Feed (0x0A).

On output, this specification will build the output message based on the data you supply for the address, function and data fields. Output processing includes calculating the LRC characters. including calculating and inserting the message checksum.

On input, this specification will validate the LRC, the slave address response, and the response function code. It will then place the response message data contents in PLC V memory.

Output

Offset	Description	Comments
0	0x4C00	Signature Word
1	Format Number 6002	MODBUS ASCII Master
2	Start Position	Must be set to a value of 1 or -1 indicating the first position.
3	Length	Must be set to a value of 1 or -1.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Data Word 1	Modbus Slave Address. Valid values are 0-247 (0x00-0xF7)
6	Data Word 2	Enter the Modbus function code. Valid values are 1-127 (0x01 through 0x7F). GAS will include this function code in the message.
7	Data Word 3	Starting V memory location which contains the Modbus message data . Some Modbus messages do not contain a data portion. In this case you can set Data Word 3 to 0.
8	Data Word 4	Number of bytes of V memory used to store message data. Some Modbus messages do not contain a data portion. In this case, set Data Word 4 to 0.
9-10	Unused	Unused by this format.
11	Format Number 65000	Table End Specification

NOTE:
The MODBUS ASCII Master specification allows only one format specification per Format Table (not including the TABLE END specification). Therefore, the entire table ***including*** *the beginning signature word and the Table End Format specification are shown above.*

NOTE:
See the Modbus Master example in Chapter 4 for a further illustration of GAS Modbus processing.

Message Data Storage (Output)

You should place the entire data portion of the Modbus message in V memory, starting at the address you specified in offset 7 of the format specification. A Modbus message data is byte oriented. You should place the first byte of data in the high byte of the first word, the second byte of the data in the low byte of the first word, the third byte of the data in the high byte of the second word, etc. See the following example.

For function code 03 (Read Holding Registers), assume that you want the data portion to be:

Starting Address Hi 00
Starting Address Lo 6B
No. of Registers Hi 00
No. of Registers Lo 02

Then V memory should contain:

Offset	Value (hex)
0	006B
1	0002

NOTE:
Modbus Function Codes 15 and 16 are treated as a special case by GAS. See explanation and example below.

For most commands, the above technique will align word oriented data with V memory boundaries. However, two commonly used Modbus functions (function code 15 - Force Multiple Coils and function code 16 - Preset Multiple Registers) contain a byte count field which would cause the data following the byte count to be non-aligned with V memory boundaries. GAS solves this problem by allowing you to use a ***full word*** for the byte count field. When GAS creates the output message, it will not include the high byte in the message. See the following example:

Example - Function Code 16 Query

For function code 16 (Preset Holding registers)
assume that you want the data portion to be:

Starting Address Hi	00
Starting Address Lo	01
No. of Registers Hi	00
No. of Registers Lo	02
Byte Count	04
Data1 Hi	00
Data1 Lo	0A
Data2 Hi	01
Data2 Lo	02

Then V memory should contain:

Offset	Value	Comments
0	0001	Address Hi/Lo
1	0002	No. of Registers Hi/Lo
2	0004	Byte Count
3	000A	Data1 Hi/Lo
4	0102	Data2 Hi/Lo

Input

Offset	Description	Comments
0	0x4C00	Signature Word
1	Format Number 6002	MODBUS ASCII Master
2	Start Position	Must be set to a value of 1 or -1 indicating the first position.
3	Length	Must be set to a value of 1 or -1.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Data Word 1	Modbus Slave Address. Valid values are 0 -247 (0x00-0xF7)
6	Data Word 2	Enter the Modbus function code which was used in the command. Valid values are 1-255 (0x01 through 0xFF) GAS will check for a valid function code in the response message. A valid response code contains either an echo of the function code sent in the command or an exception code corresponding to the function code sent in the command.
7	Data Word 3	Starting V memory location where the response data will be stored.
8	Data Word 4	Number of ***bytes*** of V memory to reserve for storing the Modbus response. 1 word = 2 bytes.
9-10	Unused	Unused by this format.
11	Format Number 65000	Table End Specification

NOTE:
The MODBUS ASCII Master specification allows only one format specification per Format Table (not including the TABLE END specification). Therefore, the entire table ***including*** *the beginning signature word and the Table End Format specification are shown above.*

Message Data Storage (Input)

Modbus Message Data is stored in V Memory starting in the V memory address you specified in offset 7. Offset 0 will contain the function code from the response message. This is provided so that you can evaluate whether you received a message containing an exception code. If the response to the function contained a byte count, Offset 1 will contain the byte count from the message data area. Otherwise the byte count will be set to 0. This approach is used to force the alignment of numeric data on V memory word boundaries for most response formats.

NOTE:
See the Modbus Documentation for response formats containing a byte count.

Offset	Contents
0	Function Code from Response message
1	Byte Count from Response message
2	Message data
..	Message Data
n	Message Data

For example, the response to a function code 03 query which contained the following data:

Byte Count 04
Data1 High 02
Data1 Low 2B
Data2 High 00
Data2 Low 64

would be stored in V memory as:

Offset	Value (hex)	Comments
0	0003	Function code from response
1	0004	Byte Count
2	022B	Data 1
3	0064	Data 2

The response to a function code 16 (hex10) which contained the following data:

Starting Address Hi 00
Starting Address Lo 01
No. Registers Hi 00
No. Registers Lo 02

would be stored in PLC V memory as:

Offset	Value (hex)	Comments
0	0010	Function Code from response
1	0000	Byte Count (0 = not used)
2	0001	Starting Address
3	0002	No. of Registers

3.27. Block Check Character (BCC)

Overview

Typical serial data communications applications must allow for electrical "noise", interference which can alter the bit sequence transmitted on the wire. To avoid using erroneous data, the participants rely on certain error checking procedures. For example, it is a common practice to use parity checking to verify the integrity of the bit sequences which define individual characters. In addition, many device protocols attempt to validate the integrity of the characters in the message by including one or more *Block Check Characters (BCC).* The sending device calculates the BCC based on the output and places it in the message. The receiving device performs an identical calculation and compares the result to the BCC placed in the message.

Unfortunately, there is no standard method for computing a BCC. In fact, even the terminology varies. You will also find this field called a "checksum" , an "Error Check Character (ECC)", a "Longitudinal Redundancy Check (LRC)", and various other names. For consistency, this document uses the ANSI 3.28 terminology of Block Check Character (BCC).

Although some device protocols provide an option to select whether a BCC is used; others require that the BCC be included in the message. Since it is difficult, if not impossible, to perform most BCC calculations in the PLC, the CTI communications module must perform this function. The BCC Format Specifications provide a table oriented method for defining how the module will produce and evaluate a BCC.

GAS BCC support allows you to define the BCC field by specifying:

The method used to **represent** the BCC in the ASCII message which includes:

1. The location of the BCC in the message buffer, and
2. The format of the BCC field in the message (typically Hex-ASCII or Binary)

The method used to **compute** the BCC, which includes:

1. The technique for "accumulating" the result. There are two common methods, Arithmetic Summation and Exclusive Or (Xor). The summation adds the binary value of the characters to an accumulator register, ignoring carries beyond the accumulator size. Accumulator sizes are typically 7, 8 or 16 bits. The Xor performs an exclusive or calculation on corresponding bits in each character (identical to a bitwise sum of the characters ignoring carries between bits). GAS does not support Cyclical Redundancy Check (CRC).
2. Determining which message characters are included in the accumulation.
3. The processing performed upon each character or group of characters prior "accumulating". For example, some BCC calculations require you to mask off the high bit. Others may group characters together and treat the result as a

hexadecimal representation of a byte or word.

4. Processing performed upon the accumulated result, such as performing two's complement.

GAS Processing Logic (Output)

The calculation of the BCC for output will be based on the contents of the message buffer at the time the BCC specification is encountered in the Format Table. Thus, your BCC Format Specification must *follow* all of the specifications which write the characters used in the calculation to the message buffer. After the calculation is complete, the computed character value will be written to the output message buffer in the specified location. If required, you can include more than one BCC format specification in a single Format Table.

GAS Processing Logic (Input)

For input, BCC calculation and comparison will also be performed at the time the BCC format specification is encountered in the Format Table. GAS compares the calculated BCC characters with the actual characters contained in the BCC field. If the characters do not match, GAS will raise the error bit, skip interpretation of the remaining format specifications, and write a specific error message in the *Command* Error Word.

Since you usually want to check the BCC before you attempt to decode the fields in the message, you should place the BCC Format Specification at the *beginning* of the Format Table. If required, you can include more than one BCC format specification in a single Format Table.

General BCC Format Specifications (Output)

This format specification is used to produce a BCC in the output message. In the Format Table, it should follow all Format Specifications which generate the characters used in the BCC calculation.

Offset	Description	Comments
1	Format Number	Accumulation Method. (See Parameter Description, page 95)
2	BCC Start Position	Position in which to place the first character of the BCC within in the output message buffer. A positive number (1-1080) indicates an absolute position from the beginning of the message. 0 indicates the "next" position (e.g. 1 past the end of the previous field specified). A negative number indicates an absolute position from the end of the message. Use negative with care since the "end" of an output message is determined by the highest position written by a previous format specification. See page 35. *Note that this differs from the standard data format spec which uses -1 for the "next" position.*
3	Reserved	You should set this value to 0.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Character Format	Specifies the character representation of the BCC in the message 1001 = Decimal ASCII Integer 1003 = HexASCII 1004 = Binary
6	Calculation Start Position	Valid Values are 1-1080. This represents the position within the output message buffer of the *first* character that will be included in the BCC calculation.
7	Calculation End Position	This represents the position in the output message buffer of the *last* character that will be included in the BCC calculation. All characters between the positions defined in Offset 6 and Offset 7 will be included. Positive values indicate the position relative to the beginning of the output message buffer. For example, the value 5 represents the 5th character. Negative values indicate the position relative to the beginning of the BCC field moving backwards. For example, -3 defines a position three characters before the beginning of the BCC field. A value of 0 is invalid.
8	Pre-Processing Post-Processing	Defines bit manipulations performed before and after BCC calculations. The code defining pre-processing is located in the high byte of this word. The code representing the post-processing method is located in the low byte of this word. (See Parameter Description, page 95)
9-10	Reserved	You should set this value to 0.

General BCC Format Specifications (Input)

Offset	Description	Comments
1	Format Number	Accumulation Method. (See Parameter Description, page 95)
2	BCC Start Position	Location of the first character of the BCC within the message buffer. A positive number (1-1080) indicates an absolute position from the beginning of the message. 0 indicates the "next" position (e.g. 1 past the end of the previous field specified). *Note that this differs from the standard **data** format spec which uses -1 for the "next" position.* A negative number is the relative location from the **end of the message** backwards. For example, -1 is the last character in the message, -2 is the next to the last character, etc.).
3	Reserved	You should set this value to 0.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Character Format	Specifies the character representation of the BCC in the message See the input BCC Format Specification.
6	Calculation Start Position	Valid Values are 1-1080. This represents the position within the input message buffer of the *first* character that will be included in the BCC calculation.
7	Calculation End Position	This represents the position in the input message buffer of the *last* character that will be included in the BCC calculation. All characters between the positions defined in Offset 6 and Offset 7 will be included. Positive values indicate the position relative to the beginning of the message buffer. For example, the value 5 represents the 5th character. Negative values indicate the position relative to the beginning of the BCC field moving backwards. For example, -3 defines a position three characters before the beginning of the BCC field. A value of 0 is invalid
8	Pre-Processing Post-Processing	Defines bit manipulations performed before and after BCC accumulation. (See Parameter Description, page 95)
9	Reserved	You should set this value to 0.
10	Debug - BCC Calculated Value V Memory Location	Can be used during debug to display the calculated value of the BCC. If this is set to zero, nothing will be written to V memory. If set to a valid V memory address, it will be interpreted as the starting V memory location where the calculated BCC will be written. BCC characters will be stored with the first character in the high byte of the first word and the second (if applicable) in the low order byte. BCCs longer than one word in length will occupy successive words in PLC memory. **For improved performance, this value should set to 0 in normal operation**.

Parameter Description

Accumulation Method

This parameter describes how the BCC is accumulated. Accumulation is performed after pre - processing.

7001	*Arithmetic Summation (7 bit accumulator)*	The binary result of pre- processing is added to a seven bit accumulator, ignoring carries that exceed the accumulator size.
7002	*Arithmetic Summation (8 bit accumulator)*	The binary result of pre- processing is added to an eight bit accumulator, ignoring carries that exceed the accumulator size
7003	*Arithmetic Summation (16 bit accumulator*	The binary result of pre- processing is added to a sixteen bit accumulator, ignoring carries that exceed the accumulator size
7011	*Exclusive Or (X0R) (7 bit accumulator)*	The binary result of pre- processing is XORed with the contents of a seven bit accumulator.
7012	*Exclusive Or (X0R) (8 bit accumulator)*	The binary result of pre- processing is XORed with the contents of an eight bit accumulator.
7013	*Exclusive Or (X0R) (16 bit accumulator)*	The binary result of pre- processing is XORed with the contents of a sixteen bit accumulator.

Character Format

The character format specifies which format will be used to represent the checksum. The length of the checksum field is implied by the character format and the size of the accumulator. See the following table.

Character Format	**Format Description**	**Accumulator Size**	**Length in Characters**	**ASCII Character Range**
1001	Decimal ASCII	7 or 8 bits	3	000-255
		16 bit	5	00000-65535
1003	HexASCII	7 or 8 bits	2	00-FF
		16 bit	4	0000-FFFF
1004	Binary	7 or 8 bit	1	* Binary Number
		16 bit	2	* Binary Number

***** The binary output is not encoded into ASCII characters.

Pre-Processing

This parameter defines processing performed prior to accumulation. The parameter code is located in the ***high order*** byte of Offset 8 (0x**nn**nn)

0	*None*	Computations are performed on an 8 bit character
1	*Mask High Bit*	Computations are performed on a 7 bit character
2	*Two Character Hex-ASCII*	Treat groups of 2 characters as a hex representation of a byte. Convert to a binary equivalent before computation.
4	*Four Character Hex-ASCII*	Treat groups of 4 characters as a hex representation of a word. Convert to a binary equivalent before computation. If the number of characters in the range included in the calculation of the BCC is not a multiple of four, the last group will be padded in the low order byte with 00.

Post -Processing

This parameter defines processing performed after accumulation. The parameter code is located in the ***low order*** byte of Offset 8 (0xnn**nn**).

0	*None*	No Post Processing
1	*One's Complement*	Take the one's complement of the accumulator value
2	*Two's Complement*	Take the two's complement of the accumulator value
3	*ASCII Offset*	If the value of the accumulator is less than 0x20, add 0x20 .

BCC Format Specification Examples

Example 1 - Output

The BCC is calculated as follows:
1. Exclusively OR the binary values of all characters excluding the start delimiter but including the end delimiter.
2. If the result is less than 0x20, then add 0x20.
3. Represent the accumulator as a single character (0x20 - 0xFF).
4. Place the BCC after the end delimiter.

Assuming the output buffer contained:

<STX>	1	2	3	4	<ETX>

from previous format specifications. The following will produce an output message of :

<STX>1234<ETX><0x27> - where **0x27** is the binary BCC expressed in hexadecimal notation.

Accumulation Method	7012	Xor - 8 bit
BCC Start Position (Output)	7	
Unused	0	
Error Word	0	
Character Format	1004	Binary
Calculation Start Position	2	
Calculation End Position	-1	last char before BCC field
Pre/Post Comp	0x0003	Pre = None, Post = add 0x20 if
Unused	0	
Unused	0	

Note that the character length in bytes is implied as one character because you are using an 8 bit accumulator and a binary character format.

Example 2 - Output

The BCC is calculated as follows:
1. Sum the binary values of all characters beginning with the character just after the start delimiter and ending with the last character in the message.
2. Divide the sum by 256 and use remainder. *Note: this is the same as using an 8 bit accumulator (ignoring carries).*
3. Represent the accumulator as two Hex-ASCII characters.
4. Place the BCC as the last characters in the message.

Assuming the output buffer contained:

<STX>	1	2	3	4

from previous format specifications, the following Format Specification will produce an output message of :

<STX>1234CA - where **CA** is the calculated BCC in ASCII**.**

Accumulation Method	7002	Summation - 8 bit
BCC Start Position (output)	6	Next position. This format specification would be the last one in the table.
Unused	0	
Error Word	0	
Character Format	1003	Hex ASCII
Calculation Start Position	2	
Calculation End Position	5	
Pre/Post Comp	0x0000	Pre = None, Post = None
Unused	0	
Unused	0	

Note that the character length of two is implied by an 8 bit summation and a HexASCII character format.

Example 3 - Input

The BCC is calculated as follows:

1. Sum the binary value of all characters, excluding the start and stop delimiters.
2. Take the two's complement of the sum.
3. Represent the two's complement total as four ASCII characters (i.e. a BCC of 0xFF12 would be 0x46, 0x46, 0x31, 0x32).
4. BCC is located just prior to the end delimiter.

Assuming the input message was **<STX>1234FF36<ETX>** where **FF36** is the calculated BCC, the following Format Specification will process the BCC.

Accumulation Method	7003	Summation - 16 bit
BCC Start Position	-5	Start at 5th character from the end
Unused	0	
Error Word	0	
Character Format	1003	Hex ASCII
Calculation Start Position	2	
Calculation End Position	-1	All characters up to the BCC
Pre/Post Comp	0x0002	Pre = None, Post = 2's Complement
Unused	0	
Debug BCC	0	

Note that the BCC character length of 4 is implied by a 16 bit summation and a HexASCII character format.

Fletcher Checksum (output)

The Fletcher Checksum is a unique calculation used by some devices.

Offset	Description	Comments
1	Format Number	Fletcher Checksum = 7031
2	BCC Start Position	Location of the first character of the BCC within in the output message buffer. A positive number (1-1080) indicates an absolute position from the beginning of the message. 0 indicates the "next" position (e.g. 1 past the end of the previous field specified). *Note that this differs from the standard data format spec which uses -1 for the "next" position.*
3	Reserved	You should set this to 0
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Character Format	Ignored by this BCC format.
6	Calculation Start Position	Valid Values are 1-1080. This represents the position within the output message buffer of the *first* character that will be included in the BCC calculation.
7	Calculation End Position	This represents the position in the output message buffer of the *last* character that will be included in the BCC calculation. All characters between the positions defined in Offset 6 and Offset 7 will be included. Positive values indicate the position relative to the beginning of the output message buffer. For example, the value 5 represents the 5th character. Negative values indicate the position relative to the beginning of the BCC field moving backwards. For example, -3 defines a position three characters before the beginning of the BCC field. A value of 0 is invalid.
8	Pre/Post Computation Method	Ignored by this BCC format.
9	Reserved	You should set this value to 0.
10	Not Used	Ignored for Output. You should set this value to 0.

Fletcher Checksum (input)

Offset	Description	Comments
1	Format Number	Fletcher Checksum = 7031
2	BCC Start Position	Location of the first character of the BCC within the message buffer. A positive number (1-1080) indicates an absolute position from the beginning of the message. 0 indicates the "next" position (e.g. 1 past the end of the previous field specified). *Note that this differs from the standard **data** format spec which uses -1 for the "next" position.* A negative number is the relative location from the **end of the message** backwards. For example, -1 is the last character in the message, -2 is the next to the last character, etc.).
3	Reserved	You should set this value to 0.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to non-zero value if an error is detected. See Appendix B.
5	Character Format	Ignored by this BCC format. You should set this value to 0.
6	Calculation Start Position	Valid Values are 1-1080. This represents the position within the input message buffer of the *first* character that will be included in the BCC calculation.
7	Calculation End Position	This represents the position in the input message buffer of the *last* character that will be included in the BCC calculation. All characters between the positions defined in Offset 6 and Offset 7 will be included. Positive values indicate the position relative to the beginning of the message buffer. For example, the value 5 represents the 5th character. Negative values indicate the position relative to the beginning of the BCC field moving backwards. For example, -3 defines a position three characters before the beginning of the BCC field. A value of 0 is invalid
8	Pre/Post Computation Method	Ignored by this BCC format. You should set this value to 0.
9	Reserved	You should set this value to 0.
10	Debug - BCC Calculated Value V Memory Location	Can be used during debug to display the calculated value of the BCC. If this is set to zero, nothing will be written to V memory. If set to a valid V memory address, it will be interpreted as the starting V memory location where the calculated BCC will be written. BCC characters will be stored with the first character in the high byte of the first word and the second (if applicable) in the low order byte. BCCs longer than one word in length will occupy successive words in PLC memory. **For improved performance, this value should set to 0 in normal operation.**

3.28. PLC Word LSB to Message Byte

Output –8001

This format specification copies the least significant byte (LSB) from one or more PLC words to the output message buffer. Each byte is considered to be unsigned (value = 0 – 255) unless the signed option bit is set.

Offset	Description	Comments
1	Format Number 8001	Copies the LSB of one or more PLC words to the output message buffer.
2	Start Position	Valid Values are 1 through 1080 and –1. Specifies the start position of the field within the output buffer. A value of –1 signifies to use the "next" position in the output buffer. If this is the first format specification in the table, the "next" position is position 1; else, it is the position immediately following the end of the previously defined field.
3	Length	Valid Values for Length are 1 thorough 1080. Length is the number of bytes to be placed in the output buffer. GAS will return an error if the size of the output buffer is exceeded.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to a non-zero value if an error is detected. See *Appendix B*.
5	Data Word 1	Set Data Word 1 to the address of the first V memory location containing data to be copied.
6	Data Word 2	Processing Options. Not used in this release. Set to 0 to ensure future compatibility.
7-10	Unused	Reserved for future use. Set to 0.

Example: PLC Word LSB to Message Byte

To transmit the hex byte string F0 D2 06 00 01 38 0D 0A stored in V memory as immediately follows, the format table would be set up as seen below:

V Memory Location	Hex Value	Decimal Value
V20	00F0	240
V21	00D2	210
V22	0006	6
V23	0000	0
V24	0001	1
V25	0038	56
V26	000D	13
V27	000A	10

V Memory Location	Description	Hex Value	Decimal Value
V100	Signature Value	4C00	19456
V101	Format Number	1F41	8001
V102	Start Position – *position to place first character*	0001	1
V103	Length – *Number of bytes to be placed in buffer*	0008	8
V104	Format Error Code – *Initially set to 0*	0000	0
V105	Data Word 1 – *Address of first V memory location containing data to be copied*	0032	20
V106	Data Word 2 – *Unused - Set to 0*	0000	0
V107	*Unused – Set to 0*	0000	0
V108	*Unused – Set to 0*	0000	0
V109	*Unused – Set to 0*	0000	0
V110	*Unused – Set to 0*	0000	0
V111	Table End Specification	FDE8	65000

3.29. Message Byte to PLC Word LSB

Input –8001

This format specification copies one or more bytes from the input message to V memory. Each byte is placed in a separate word in the least significant byte (LSB) position. Each byte is considered to be unsigned (value = 0 – 255) unless the applicable processing option is set.

Offset	Description	Comments
1	Format Number 8001	Copies a byte string from a message to a PLC word, one byte per word.
2	Start Position	Valid Values are 1 through 1080 and -1
3	Length	Valid Values for Length are 1 thorough 1080. Length represents the number of bytes to be processed as input. GAS will return an error if the message length is exceeded.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to a non-zero value if an error is detected. See *Appendix B.*
5	Data Word 1	Set Data Word 1 to the address of the V memory location that will contain the first byte of data.
6	Data Word 2	Processing Options. Not used in this release. Set to 0 to ensure future compatibility.
7-10	Unused	Reserved for future use. Set to 0.

Example: Message Byte to PLC Word LSB

To copy the hex byte string F0 D2 06 00 01 38 0D 0A from the input buffer and store it in V memory as immediately follows, the format table would be set up as seen below:

V Memory Location	Hex Value	Decimal Value
V20	00F0	240
V21	00D2	210
V22	0006	6
V23	0000	0
V24	0001	1
V25	0038	56
V26	000D	13
V27	000A	10

V Memory Location	Description	Hex Value	Decimal Value
V100	Signature Value	4C00	19456
V101	Format Number	1F41	8001
V102	Start Position – *position of first character in input buffer*	0001	1
V103	Length – *Number of bytes to be processed*	0008	8
V104	Format Error Code – *Initially set to 0*	0000	0
V105	Data Word 1 – *Address of first V memory location where data is to be copied*	0032	20
V106	Data Word 2 – *Unused - Set to 0*	0000	0
V107	*Unused – Set to 0*	0000	0
V108	*Unused – Set to 0*	0000	0
V109	*Unused – Set to 0*	0000	0
V110	*Unused – Set to 0*	0000	0
V111	Table End Specification	FDE8	65000

3.30. PLC Word to Message Byte

Output –8002

This format specification copies bytes from one or more PLC words to message bytes. The least significant byte (LSB) of the PLC word precedes the most significant byte (MSB) in the output buffer.

Offset	Description	Comments
1	Format Number 8002	Copies bytes from one or more PLC words in little Endian order (LSB precedes MSB).
2	Start Position	Valid Values are 1 through 1080 and –1. Specifies the start position of the field within the output buffer. A value of –1 signifies to use the "next" position in the output buffer. If this is the first format specification in the table, the "next" position is position 1; else, it is the position immediately following the end of the previously defined field.
3	Length	Valid Values for Length are even numbers, 2 thorough 1080. Length is the number of bytes to be placed in the output buffer. GAS will return an error if the size of the output buffer is exceeded or the length value is not an even number.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to a non-zero value if an error is detected. See *Appendix B*.
5	Data Word 1	Set Data Word 1 to the address of the V memory location that contains the first two bytes of data.
6	Data Word 2	Processing Options. Not used in this release. Set to 0 to ensure future compatibility.
7-10	Unused	Reserved for future use. Set to 0.

Example: PLC Word to Message Byte

To transmit the hex byte string F0 D2 06 00 01 38 0D 0A that is stored in V memory as immediately follows, the format table would be set up as seen below:

V Memory Location	Hex Value	Decimal Value
V20	D2F0	54000
V21	0006	6
V22	3801	14337
V23	0A0D	2573

V Memory Location	Description	Hex Value	Decimal Value
V100	Signature Value	4C00	19456
V101	Format Number	1F42	8002
V102	Start Position – *position to place first character*	0001	1
V103	Length – *Number of bytes to be placed in buffer*	0008	8
V104	Format Error Code – *Initially set to 0*	0000	0
V105	Data Word 1 – *Address of first V memory location containing data to be copied*	0032	20
V106	Data Word 2 – *Unused - Set to 0*	0000	0
V107	*Unused – Set to 0*	0000	0
V108	*Unused – Set to 0*	0000	0
V109	*Unused – Set to 0*	0000	0
V110	*Unused – Set to 0*	0000	0
V111	Table End Specification	FDE8	65000

3.31. Message Byte to PLC Word

Input –8002

This format specification copies one or more bytes from the message to V memory. Two bytes are placed in each word. The first byte is placed in the least significant byte (LSB) and the second byte is placed in the most significant byte (MSB).

Offset	Description	Comments
1	Format Number 8002	Copies a byte string from a message to a PLC word in little Endian order (first byte in LSB, next byte in LSB).
2	Start Position	Valid Values are 1 through 1080 and –1.
3	Length	Valid Values for Length are even numbers, 2 thorough 1080. Length represents the number of bytes to be processed as input. GAS will return an error if the message length is exceeded.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to a non-zero value if an error is detected. See *Appendix B.*
5	Data Word 1	Set Data Word 1 to the address of the V memory location that will contain the first byte.
6	Data Word 2	Processing Options. Not used in this release. Set to 0 to ensure future compatibility.
7-10	Unused	Reserved for future use. Set to 0.

Example: Message Byte to PLC Word

To copy this hex byte string from the input buffer F0 D2 06 00 01 38 0D 0A and store it in V memory as immediately follows, the format table would be set up as seen below:

V Memory Location	Hex Value	Decimal Value
V20	D2F0	54000
V21	0006	6
V22	3801	14337
V23	0A0D	2573

V Memory Location	Description	Hex Value	Decimal Value
V100	Signature Value	4C00	19456
V101	Format Number	1F42	8002
V102	Start Position – *position of first character in input buffer*	0001	1
V103	Length – *Number of bytes to be processed*	0008	8
V104	Format Error Code – *Initially set to 0*	0000	0
V105	Data Word 1 – *Address of first V memory location where data is to be copied*	0032	20
V106	Data Word 2 – *Unused - Set to 0*	0000	0
V107	*Unused – Set to 0*	0000	0
V108	*Unused – Set to 0*	0000	0
V109	*Unused – Set to 0*	0000	0
V110	*Unused – Set to 0*	0000	0
V111	Table End Specification	FDE8	65000

3.32. PLC Word to Message Byte – Byte Swap

Output –8003

This format specification copies bytes from one or more PLC words to message bytes. The most significant byte (MSB) of the PLC word precedes the least significant byte (LSB) in the output buffer.

Offset	Description	Comments
1	Format Number 8003	Copies bytes from one or more PLC words to the output message buffer in big Endian order (MSB precedes LSB)
2	Start Position	Valid Values are 1 through 1080 and –1. Specifies the start position of the field within the output buffer. A value of –1 signifies to use the "next" position in the output buffer. If this is the first format specification in the table, the "next" position is position 1; else, it is the position immediately following the end of the previously defined field.
3	Length	Valid Values for Length are even numbers, 2 thorough 1080. Length is the number of bytes to be placed in the output buffer. GAS will return an error if the size of the output buffer is exceeded.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to a non-zero value if an error is detected. See *Appendix B*.
5	Data Word 1	Set Data Word 1 to the address of the first V memory location that contains data to be copied.
6	Data Word 2	Processing Options. Not used in this release. Set to 0 to ensure future compatibility.
7-10	Unused	Reserved for future use. Set to 0.

Example: PLC Word to Message Byte – Byte Swap

To transmit the hex byte string F0 D2 06 00 01 38 0D 0A that is stored in V memory as immediately follows, the format table would be set up as seen below:

V Memory Location	Hex Value	Decimal Value
V20	F0D2	61650
V21	0600	1536
V22	0138	312
V23	0D0A	3338

V Memory Location	Description	Hex Value	Decimal Value
V100	Signature Value	4C00	19456
V101	Format Number	1F43	8003
V102	Start Position – *position to place first character*	0001	1
V103	Length – *Number of bytes to be placed in buffer*	0008	8
V104	Format Error Code – *Initially set to 0*	0000	0
V105	Data Word 1 – *Address of first V memory location containing data to be copied*	0032	20
V106	Data Word 2 – *Unused - Set to 0*	0000	0
V107	*Unused – Set to 0*	0000	0
V108	*Unused – Set to 0*	0000	0
V109	*Unused – Set to 0*	0000	0
V110	*Unused – Set to 0*	0000	0
V111	Table End Specification	FDE8	65000

3.33. Message Byte to PLC word – Byte Swap

Input –8003

This format specification copies one or more bytes from the message to V memory. Two

bytes are placed in each word. The first byte is placed in the most significant byte (MSB) and the second byte is placed in the least significant byte (LSB).

Offset	Description	Comments
1	Format Number 8003	Copies a byte string from a message to a PLC word in Big Endian order (first byte in MSB, next byte in LSB).
2	Start Position	Valid Values are 1 through 1080 and -1
3	Length	Valid Values for Length are even numbers, 2 thorough 1080. Length represents the number of bytes to be processed as input. GAS will return an error if the message length is exceeded (see Data Word 2.)
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to a non-zero value if an error is detected. See *Appendix B.*
5	Data Word 1	Set Data Word 1 to the address of the V memory location that will contain the first byte.
6	Data Word 2	Processing Options. Not used in this release. Set to 0 to ensure future compatibility.
7-10	Unused	Reserved for future use. Set to 0.

Example: Message Byte to PLC Word – Byte Swap

To copy this hex byte string F0 D2 06 00 01 38 0D 0A from the input buffer and store it in V memory as immediately follows, the format table would be set up as seen below:

V Memory Location	Hex Value	Decimal Value
V20	F0D2	61650
V21	0600	1536
V22	0138	312
V23	0D0A	3338

V Memory Location	Description	Hex Value	Decimal Value
V100	Signature Value	4C00	19456
V101	Format Number	1F43	8003
V102	Start Position – *position of first character in input buffer*	0001	1
V103	Length – *Number of bytes to be processed*	0008	8
V104	Format Error Code – *Initially set to 0*	0000	0
V105	Data Word 1 – *Address of first V memory location where data is to be copied*	0032	20
V106	Data Word 2 – *Unused - Set to 0*	0000	0
V107	*Unused – Set to 0*	0000	0
V108	*Unused – Set to 0*	0000	0
V109	*Unused – Set to 0*	0000	0
V110	*Unused – Set to 0*	0000	0
V111	Table End Specification	FDE8	65000

CHAPTER 4. APPLICATION EXAMPLES

4.1. Unsolicited Read Example

This example demonstrates how the GAS protocol manager can be used to read unsolicited messages sent from a weigh scale. The term "unsolicited" means that the device sends data without being asked; it does not require a command from the PLC before it will send data. This mode is also typical of many bar code devices.

Assumptions

The weigh scale will periodically send a message with the following format:

<STX>ssswwwwwwttttt<CR><BCC>

where:

<STX>	is the ASCII Start of Text character (hex 02),
sss	is 3 decimal characters representing device status information,
wwwwww	is 6 decimal characters representing the display weight,
ttttt	is 6 decimal characters representing the tare weight,
<CR>	is the ASCII Carriage Return character (hex 0D),
<BCC>	is the message Block Check Character.

Therefore, a message with a device status of 589, a display weight of 2000, and a tare weight of 1000 would be sent as follows:

<STX>589002000001000<CR><BCC>

For this example, we decide that we only want to retrieve the display weight and the tare weight; we will ignore the status word. In addition, we will ignore the BCC.

The weigh scale is configured for the following communications parameters:

9600 baud
7 data bits
No parity
One Stop bit
No handshaking

We will store the Command Block for the Create Connection Command at V200, the command block for the READ command at V100, and the input Format Table at V3000. The example will use port 1 to communicate with the device.

Command Blocks

The CREATE CONNECTION command block for the GAS protocol manager (Protocol Manager Number 38) using the communications parameters of the weigh scale would be:

Location	Description	Hex Value	Decimal Value
V200	Command Error Word	0000	0
V201	Command (Create Connection)	0001	1
V202	Connection Number (19221)	4B15	19221
V203	Protocol Manager Number for GAS	0026	38
V204	Physical Port Number (Use Port 1)	0001	1
V205	Port Baud Rate = 9600	2580	9600
V206	Bits Per Character = 7	0007	7
V207	Parity = 0 = None	0000	0
V208	Stop Bits =1	0001	1
V209	Handshake = 0 = None	0000	0
V210 - V215	Unused - reserved for future use (Set to 0)	0000	0

The following READ command block would be used to retrieve the data from the device. The message is assumed to be a maximum of 17 characters, start with <STX> and end with <CR>.

Location	Description	Hex Value	Decimal Value
V100	Error Word	0000	0
V101	Command Code (Read)	2602	9730
V102	Connection Number 19221	4B15	19221
V103	Command Control Flag	0000	0
V104	V memory address of Format Table used for input = V3000	0BB8	3000
V105	Maximum # of input Format Specifications to read (0 = default = 6).	0000	0
V106	Read timeout in seconds = infinite	03E8	1000
V107	Input message maximum length = 17 characters	0011	17
V108	Input message delimiters, first set (<STX> <CR>)	020D	525
V109	Input message delimiters, second set (not used)	0000	0
V110-115	Unused (Set to 0)	0000	0

Note that the maximum message length is set to 17 in V107, the beginning and ending delimiters are set to hex 02 and hex 0D in V108. GAS will start placing characters in the input buffer when the <STX> is received and will consider the message complete when the <CR> is received. The BCC is ignored. V106 is set to an infinite timeout so that the 2573 will continue to wait until the device sends a message. Once a message is received, the PLC logic should immediately trigger the command again to wait for the next message.

Format Table

The following illustrates the Format Table located at V3000. Note that V3000 is referenced in V104 of the Command Block.

Location	Value	Comments
V3000	19456	Signature Value
V3001	1001	Integer ASCII to PLC Integer
V3002	5	Start in Position 5
V3003	6	Process 6 Characters
V3004	0	Format Error Word
V3005		Display Weight Value Retrieved (Set to 0 initially)
V3006 - V3010	0	Unused
V3011	1001	Integer ASCII to PLC Integer
V3012	11	Start in Position 11
V3013	6	Process 6 characters
V3014	0	Format Error Word
V3015		Tare Value Retrieved (Set to 0 initially)
V3016 - V3020	0	Unused
V3021	65000	Table End Format Specification

V3001 through V3010 contain the format specifications for the field containing the display weight. V3011 through V3020 contain the format specifications for the field containing the tare weight. Note that the fields which contain the status word and the BCC are not specified. Therefore, they are not decoded or checked by GAS.

Assuming the scales sends the following message:
<STX>589002000001000<CR><BCC> , after the command executes, V3005 will contain the display weight of 2000 and V 3015 will contain the tare value of 1000 .

4.2. Polled Read Example

This example demonstrates how the POLLED READ command can be used to obtain data from a hypothetical servo drive. This particular device is typical of one that must be polled before it will send a data message. To poll the device you send a command, then wait for the answer. This particular drive uses a device number field to select a particular device when it is used in multi-drop applications.

Assumptions

Before the drive will send data, it must receive a message with the following format:

<EOT>ddddtppcc><ENQ> where:

<EOT>	is the ASCII End of Transmission Character (hex 04)
dddd	is the device number
t	is the parameter Type
pp	is the parameter number
cc	are the Block Check Characters. Assume this BCC is an arithmetic summation of the characters in positions 2 - 8 using an 8 bit accumulator with no pre or post processing and that the BCC is expressed as 2 HexASCII characters.
<ENQ>	is the ASCII Enquiry Character (hex 05)

To retrieve data from device 0011 with parameter type 0 and parameter number of 09 you would send the following message:

<EOT>00110095B<ENQ> where 5B is the calculated BCC is expressed as HexASCII.

The response to this poll has the following format

<STX>tppsnnnnn<ETX>cc<CR> where:

<STX>	is the ASCII Start of Text Character (hex 02)
t	is the parameter type (echoed from the poll command)
pp	is the parameter number (echoed from the poll command)
s	is the sign. A + indicates a positive number, - is negative,
nnnnn	is a number which may contain decimal fraction,
<ETX>	is the ASCII End of Text Character (hex 03)
cc	are the Block Check Characters returned by the device. See BCC calculation assumptions above.
<CR>	is the ASCII Carriage Return character (hex 0D).

Thus, a valid response to the poll might be: **<STX>009+0150.<ETX>B8<CR>** where B8 is the calculated BCC.

The servo motor is configured with communications parameters as follows: Baud Rate = 9600, Data Bits = 7, Parity = odd, Stop Bits = 1, Handshake = 0. We will store the Command Block for the CREATE CONNECTION Command at V200, the command block for the POLLED READ command at V100, , the output Format Table at V2000, and the input Format Table at V3000.

Command Blocks

The following command block is used to create a connection to Port 2, using the GAS protocol manager (Number 38).

Location	Description	Hex Value	Decimal Value
V200	Command Error Word	0000	0
V201	Command (Create Connection)	0001	1
V202	Connection Number = 19222	4B16	19222
V203	Protocol Manager Number (GAS = 38)	0026	38
V204	Physical Port Number 2	0002	2
V205	Port Baud Rate = 9600, 19200	2580	9600
V206	Bits Per Character = 7	0007	7
V207	Parity = 1 = Odd	0001	1
V208	Stop Bits = 1	0001	1
V209	Handshake = 0 = None	0000	0
V210 -215	Unused - reserved for future use (Set to 0)	0000	0

A POLLED READ to obtain the data; the command block is shown below.

Location	Description	Hex Value	Decimal Value
V100	Error Word	0000	0
V101	Command Code (Polled Read)	2603	9731
V102	Connection Number = 19222	4B16	19222
V103	Command Control Flag (see below	0000	0
V104	V Memory Address of Output Format Table	07D0	2000
V105	Maximum # of output Format Specifications (0 = default)	0000	0
V106	V memory Address of Input Format Table	0BB8	3000
V107	Maximum # of input Format Specifications (0 = default)	0000	0
V108	Command timeout in seconds (0= default)	0000	0
V109	Input message maximum length	000C	12
V110	Input message delimiters, first set (<STX> <CR>)	020D	525
V111	Input message delimiters, second set	0000	0
V112 -115	Unused (Set to 0)	0000	0

Format Table (Output)

The following Format Table is used for the output message of the POLLED READ command. Note that V104 in the POLLED READ command block contains the V memory address of this table.

Location	Value	Comments
V2000	19456	Signature Value
V2001	3001	PLC ASCII Characters to ASCII Character String
V2002	1	Start in Position 1 of output buffer
V2003	1	Output 1 Character
V2004	0	Format Error Word
V2005	0x0400	ASCII <EOT> (0x04) located in the high byte.
V2006 - V2010	0	Unused
V2011	1001	PLC Unsigned Integer to Unsigned Decimal ASCII Character String
V2012	-1	Next Position in output buffer
V2013	4	Output 4 Digits (will pad w/ 0's)
V2014	0	Format Error Word
V2015	11	Device Address
V2016 - V2020	0	Unused
V2021	1001	PLC Unsigned Integer to Unsigned Decimal ASCII Character String
V2022	-1	next position
V2023	3	Output 3 characters (pad w/ 0's)
V2024	0	Format Error Word
V2025	9	Parameter Type / Number
V2026 - V2030	0	Unused
V2031	7002	BCC Arithmetic Summation into 8 bit accumulator
V2032	0	Start in next position
V2033	0	Reserved - Not Used
V2034	0	Format Error Word
V2035	1003	Hex ASCII Output Format
V2036	2	Start Calculation with second character in the message
V2037	-1	End Calculation with the character just before the BCC field
V2038	0	No pre- or post-processing
V2039	0	Reserved - Not Used
V2040	0	Unused
V2041	3001	ASCII Character to ASCII String
V2042	-1	Next Position
V2043	1	Output 1 character
V2044	0	Format Error Word
V2045	0x0500	ASCII <ENQ> hex 05 in high byte
V2046 - V2050	0	Unused
V2051	65000	Table End Format Specification

With this specification you can change device number and/or the parameter number by using PLC logic to change V2015 and V2025 respectively. Using the values in this Format Table, you will send the message: **<EOT>00110095B<ENQ>** where 5B is the calculated BCC.

Format Table (Input)

The following Format Table is used for the input message of the POLLED READ command. Note that V106 in the POLLED READ command block contains the V memory address of this table. To obtain the parameter value from the response message, use the following Format Table.

Location	Value	Comments
V3000	19456	Signature Value
V3001	7002	BCC Arithmetic Summation into 8 bit accumulator
V3002	-3	BCC Field starts 3 characters from the end
V3003	0	Reserved - Not Used
V3004	0	Format Error Word
V3005	1003	Hex ASCII Output Format
V3006	2	Start Calculation with second character in the message
V3007	-1	End Calculation with the character just before the BCC field
V3008	0	No pre- or post-processing
V3009	0	Reserved - Not Used
V3010	0	Unused
V3011	2002	Signed Decimal Fraction ASCII String to PLC Real
V3012	5	Start in Position 5
V3013	6	input length of 6 (including sign and decimal)
V3014	0	Format Error Word
V3105	0	Unused
V3106		Signed Real Value of Parameter
V3017		Signed Real Value of Parameter
V3018 - V3020	0	Unused
V3021	65000	Table End Format Specification

Since a PLC real number uses two words, the parameter value will be stored V3006 and V3007.
Notice that the BCC format specification appears at the beginning of the Format Table so that it will be checked first. An example input message which this Format Table would decode is:

<STX>009+0150.<ETX>B8<CR>

where B8 is the BCC calculated by the device.

4.3. Parse Message Example

There may be times when you need to decode a message in several different ways in order to obtain the desired result. The PARSE MESSAGE command provides you with a flexible way to accomplish this task.

The following example illustrates how you could read the device message, store the input message string in V memory, and then use the PARSE MESSAGE command to decode it in multiple ways. This example also illustrates the use of the POSITION DATA CURSOR Format Specification and the Command Control Flag bit to continue processing on parsing error.

Assumptions

The example device normally responds to a poll with:

<STX>nnnnnn,nnnnnn<ETX>

(where **nnnnnn** represents a variable length integer value up to six characters)

If the value were 234 and 5678, then the device would respond with:

<STX>234,5678<ETX>

However, if an error occurs the device responds with an **Ennn** in the field in error (where nnn is a numeric error code. If the first field had error number 23 and the second field contained the data value 5678, the device would respond with:

<STX>E023,5678<ETX>

Your application requires that you write the numeric data to V memory for a normal response. However, if you get an error, you want to write the error message in ASCII text to V memory for subsequent display. The following steps are used to accomplish this application.

Transfer the device message to V memory

First, use the POLLED READ Command with a format specification using only format specification 3003 to get the message into V memory. For the sake of brevity, the Command Block and Format Table for this operation are not shown, since they have been illustrated elsewhere. This example assumes that the format specification 3003 stored the result in V memory starting at V2000. Thus V2000 will contain the message length and the actual ASCII Characters will start in V2001.

Parse the First Message Field

Command Block

Use the PARSE MESSAGE command block to decode the message as shown below.

Offset	Description	Hex Value	Decimal Value
0	Command Error Word	0000	0
1	Command Code (Parse Message)	2605	9733
2	Connection Number (19221 - 19299)	4B17	19223
3	Command Control Flag (0x0002 = Continue processing on parsing error)	0002	2
4	V Memory Address of Format Table	0064	100
5	Maximum # of input Format Specifications to read (0 = default = 6).	0000	0
6	Command timeout in seconds (0 = default)	0000	0
7	Input message maximum length	000F	15
8	Input message delimiters, first set (<STX> and <ETX> characters	0203	515
9	Input message delimiters, second set	0000	0
10	Starting V memory address of input string	07D1	2001
11-15	Unused (Set to 0)	0000	0

Since the <STX> and <ETX> characters are included with the message, they are used as delimiters when reading the message from V memory. Alternately, you could set the message length in Offset 7 to the value stored in V2000 (which is the number of ASCII characters actually stored).

Format Table

The following Format Table is used with the first pass:

V Memory	Value	Description
100	19456	Signature Word
101	1001	Format Number 1001
102	2	Start in Position 2
103	-1	Variable Length
104	0	Format Error Code
105		Converted Value
106	0	unused
107	0	unused
108	0	unused
109	0	unused
110	0	unused
111	3001	Format Number 3001
112	2	Start in Position 2
113	4	Length of 4
114	0	Format Error Code
115		Input ASCII Characters
116		Input ASCII Characters
117 - 120		unused
121	65000	Table End

The first format specification (1001) tries to decode the first data field, starting in position 2 (after the <STX>) as a variable length integer.

The next format specification (3001) decodes the field again as four ASCII characters.

Note that, if the first field contains an error code, the attempt to decode the first field as an integer will fail, since it will contain the letter "E". However, since bit 15 of the command control flag (0x002) is set, GAS will not halt on a parsing error. Instead, decoding will continue with the second format specification, which captures the ASCII error code.

If the Command Error bit is not set, then you know that V105 will contain the actual data. If the Command Error bit is set, then V115-V116 contains the ASCII error code.

Therefore, if the reply were **<STX>234, 5678<ETX>** then V105 would contain the integer value 234. However, if the reply were **<STX>E023, 5678<ETX>** then V115-V115 would contain the ASCII characters E023 (0x 45 30 32 33).

Parse the Second Message Field

Command Block

For brevity, the command block for the PARSE MESSAGE command is not shown. It is essentially the same as the previous example except that offset 4 points to the Format Table at V130.

Format Specification

The Format Table used to decode the second field is shown below.

V Memory	Value	Description
130	19456	Signature Word
131	5001	Format 5001
132	11	Anchor Position
133	1	Relative Movement
134	0	Format Error Code
135	0	Reserved
136	1	Number of Characters
137	0x2C00	Search Character
138	0	Unused
139	0	Unused
140	0	Unused
141	1001	Format Number 1001
142	-1	Start in current Position
143	-1	Variable Length
144	0	Format Error Code
145		Converted Value
146	0	unused
147	0	unused
148	0	unused
149	0	unused
150	0	unused
151	5001	Format 5001
152	4	Anchor Position
153	0	Relative Movement
154	0	Format Error Code
155	0	Reserved
156	0	Unused
157	0	Unused
158	0	Unused
159	0	Unused
160	0	Unused
161	3001	Format Number 3001
162	-1	Start in Current position
163	4	Length of 4
164	0	Format Error Code
165		Input ASCII Characters
166		Input ASCII Characters
167 - 170		unused
171	65000	Table End

The first format specification (5001) positions the data cursor to the beginning of the second data field. This is the field following the comma delimiter (0x2C).

The next format specification (1001) tries to decode the second data field as a variable length integer.

The next format specification (5001) positions the data cursor to the beginning of the previous field.

The last format specification will decode the field again as four ASCII characters.

Note that, if the first field contains an error code, the attempt to decode the first field as an integer will fail, since it will contain the letter “E”. However, since bit 15 of the command control flag (0x002) is set, GAS will not halt on a parsing error. Instead, decoding will continue with the second format specification, which captures the ASCII error code.

If the COMMAND ERROR bit is not set, then you know that V105 will contain the actual data. If the Command Error bit is set, then V115-V116 contains the ASCII error code.

Multiple Input Buffer Read Example

NOTE:
You could accomplish the following example by using the PARSE MESSAGE command shown in the previous example. This example simply illustrates another way of deciding how to decode a complex protocol. As compared to using the Parse Message command, this technique may be faster, since it avoids the step of reading the message to V memory first.

In normal operation, the port buffer is cleared of all characters when the READ Command is initiated. Subsequent characters arriving at the port buffer are checked for a match with the message delimiters specified in the command block. When a complete message has arrived at the port buffer (based on the delimiter(s) and maximum length specified), the module transfers the complete message to the input buffer, where GAS can decode the message.

For most applications, the process above works fine. However, some protocols may return one of several responses, depending upon the state of the device. For example, assume a device which could reply to a query message with:

<NAK>

if the device could not send the data, or

<**STX>dddddddd**

if the data were available (where dddddddd represents the data returned).

For this application, you must read the first character of the message, and then decide how to process the data. To accomplish this you need to execute multiple READ commands. The first READ obtains only the first character. After you examine the first character using PLC logic, then you decide whether to read the rest of the message (containing data). If you choose to read the rest of the data you do not want to clear the port input buffer. Instead you should set the Command Control flag to (0x8000). This will cause GAS to read the existing characters in the port buffer.

Following is an example of this technique. The first command block reads the first character of the message.

Offset	Description	Hex Value	Decimal Value
0	Command Error Word	0000	0
1	Command Code (Polled Read)	2603	9731
2	Connection Number (19221)	4B15	19221
3	Command Control Flag	0000	0
4	V Memory Address of Format Table - Output	0064	100
5	Maximum # output Format Specifications to Read (0 = default = 6)	0001	1
6	V Memory Address of Format Table - Input	0128	200
7	Maximum # input Format Specifications to Read (0= default = 6)	0001	1
8	Command timeout in seconds	0000	0
9	Input Message maximum length	0001	1
10	Input Message delimiters, first set <NAK>	0000	0
11	Input Message delimiters, second set <STX>	0000	0
12	End-of-Message Timeout (0 = not used)	0000	0
13 -15	Unused (Set to 0)	0000	0

After the character is obtained, if a <NAK> is returned your logic may choose to retry the command. If a <STX> is returned, then your logic could trigger the following command block. The command control flag is set to 0x8000 to indicate that the port buffer is not cleared. No message delimiters are used. Length is set to read the next 8 characters in the port buffer. The actual message data is processed based on the Format Table located at V300.

Offset	Description	Hex Value	Decimal Value
0	Command Error Word	0000	0
1	Command Code (Read)	2602	9730
2	Connection Number (19221)	4B15	19221
3	Command Control Flag	8000	32768
4	V Memory Address of Format Table - Input	012C	300
5	Maximum # of input Format Specifications to read (0 = default = 6).	0001	1
6	Read timeout in seconds (0 = default = 9 seconds).	0002	2
7	Input message maximum length	0008	8
8	Input message delimiters, first set	0000	0
9	Input message delimiters, second set	0000	0
10-15	Unused (Set to 0)	0000	0

CHAPTER 5. PROTOCOL SPECIFIC EXTENSIONS

This chapter is intended to provide you with information regarding extensions to the GAS protocol that are added for specific serial protocol requirements. The information is grouped by protocol.

5.1. SEAbus

Overview

SEAbus is a serial protocol developed by Siemens®. It is used to communicate with a variety of equipment used in the power industry, including motor controllers, relays, and power meters. SEAbus is an asynchronous byte oriented binary protocol. It has a well-defined header and checksum (LRC) and a variable length data area. The format of the data portion varies with each device/command combination. Unlike fixed format protocols like Modbus and CAMP, the data area may contain numeric values represented by one byte intermixed with numeric values represented by two bytes (word data).

Implementation

The CTI 2573 can act as a master to slave devices that conform to the SEAbus protocol. This product enhancement will assume that the user is familiar with the SEAbus protocol. This protocol is documented in Siemens® publication SG-6213-00 *SEAbus and SEAbus Plus Protocol Reference Manual.*

The standard GAS Polled Read command will be used to initiate SEAbus requests and to process SEAbus responses. The GAS facilities include:

- A format specification which verifies the SEAbus response packet.
- A format specification that transfers bytes between a message and the least significant byte (LSB) of one or more PLC words. This facility allows the PLC to easily process numeric data represented by a single byte.
- A format specification that transfers data between groups of two message bytes and one or more PLC words using little Endian order. Using little Endian, the first byte of the pair is stored in the Least Significant Byte (LSB) of the PLC word and the second byte is stored in the Most Significant Byte (MSB) of the PLC word.
- A format specification that transfers data between groups of two message bytes and one or more PLC words using big Endian order. Using big Endian, the first byte of the pair is stored in the Most Significant Byte (MSB) of the PLC word and the second byte is stored in the Least Significant Byte (LSB) of the PLC word. Note: this format is equivalent to

format specification 3002, except that it requires an even multiple of bytes be transferred.

Limitations

Note:
These facilities do NOT support SEAbus Plus protocol, which imbeds a CRC and inverted Synch byte in the data area.

- Inter-character timing specifications are relaxed to 100ms rather than the 50ms timing dictated by the protocol specification. Although this difference could have a slight effect on error detection in some circumstances, it should have no serious consequences when the CTI module is acting as a master device.
- The maximum specified delay between a master request and a slave response is 500 ms. The minimum GAS timeout value is one second. When the CTI module is acting as a master, the only detrimental effect would be that it takes slightly longer (500 ms) to detect that a device did not answer a request.

GAS Format Specifications for SEAbus

SEAbus Received Message Verification

This format specification verifies the received packet is a valid SEAbus packet. It should be the first data format specification in the format table. This format specification performs the following checks:

- Verify that the LRC value is correct
- Verify the Synch Byte matches the expected value
- Verify that the data length matches the length value in the data length field
- Verify that the overall packet length is correct

Input –6003

Offset	Description	Comments
1	Format Number 6003	SEAbus Message Verification
2	Start Position	Unused field. Must be set to 1 or –1.
3	Length	Unused field. Must be set to 1 or –1.
4	Format Error Code	PLC logic should set this to zero when the command is initiated. Set by GAS to a non-zero value if an error is detected. See *Appendix B.*
5	Data Word 1 Expected Synch Byte Value	Set this to 27h if the message is a response from a slave. Set this to 14h if the message is a request from a master device.
6	Data Word 2	Message Verification Options. Not used in this release. Set to 0 to ensure future compatibility.
7	Data Word 3	Inter-Character timing value. Not used in this release. Set to 0 to ensure future compatibility.
8	Data Word 4	Data Length Value. Contains the byte count from the response message.
9-10	Unused	Reserved for future use. Set to 0.

5.2. SEAbus Application Examples

Assumptions

The examples assume you are communicating with a Siemens® Advanced Motor Master system using SEAbus protocol at 9600 baud. Assume the device address is to be set to 5.

Example 1 – Simple Command

The first example illustrates a single byte command with a slave response that contains no data. This particular command sets the Process Current to 50 %.

Note:
This example does not work with older SAMMS-LVX product. The older product does not echo the response. In that case, use the write command instead of the polled write.

The following data is stored in V memory:

Address	Value Hex	Value Decimal	Comments
V300	14	20	Synch Byte value for master request
V301	05	5	Device Type (address) of slave device

V302	47	71	Message Type value for Set Process Current request
V303	01	1	Data Length of 1
V304	32	50	Data Value of 50 %

We will store the Command Block for the CREATE CONNECTION Command at V200, the command block for the POLLED READ command at V100, the output Format Table at V2000, and the input Format Table at V3000.

Command Blocks

The following command block creates a connection to Port 2, using the GAS protocol manager (Number 38). The baud rate is selected as 9600. SEAbus uses an asynchronous protocol with eight data bits, no parity, and one stop bit.

Location	Description	Hex	Decimal
V200	Command Error Word	0000	0
V201	Command (Create Connection)	0001	1
V202	Connection Number = 19222	4B16	19222
V203	Protocol Manager Number (GAS = 38)	0026	38
V204	Physical Port Number 2	0002	2
V205	Port Baud Rate = 9600	2580	(1) 9600
V206	Bits Per Character = 8	0008	8
V207	Parity = 0 = None	0000	0
V208	Stop Bits = 1	0001	1
V209	Handshake = 0 = None	0000	0
V210 -215	Unused – reserved for future use (Set to 0)	0000	0

(1) Other baud rates are available. The baud rate must match the baud rate of the device.

A POLLED READ command, which is used to send the request and process the response, is shown below. Note that the command timeout is set to 1 second, the input message delimiter is hex 27 (the expected value of the response synch byte), and the length is set to 5 bytes (the response has no data).

The end of message timeout is set to 0.1 seconds (100 ms) to facilitate error detection based on inter-character timing. SEAbus inter-character timing is a maximum of 50ms. In this example, if no character were received within 100ms, end of message would be detected. When the SEAbus protocol format specification evaluated the packet, it would detect and report an error.

Location	Description	Hex	Decimal
V100	Error Word	0000	0
V101	Command Code (Polled Read)	2603	9731
V102	Connection Number = 19222	4B16	19222
V103	Command Control Flag	0000	0
V104	V Memory Address of Output Format Table	07D0	2000
V105	Maximum # of output Format Specifications (0 = default)	0000	0
V106	V memory Address of Input Format Table	0BB8	3000
V107	Maximum # of input Format Specifications (0 = default)	0000	0
V108	Command timeout in seconds (0= default)	0001	1
V109	Input message maximum length	000C	12
V110	Input message delimiters, first set	2700	9984
V111	Input message delimiters, second set	0000	0
V112	End-of-Message Timeout in tenths of seconds	0001	1
V113 -115	Unused (Set to 0)	0000	0

Format Table (Output)

The following Format Table is used for the output message of the POLLED READ command. Note that V104 in the POLLED READ command block contains the V memory address of this table. The first format specification builds the message header and the data field by extracting the low byte from V300 through V304 and placing them in the first five bytes of the request message. The second format specification creates the LRC.

Location	Value	Comments
V2000	19456	Signature Value
V2001	8001	PLC Word LSB to Message Byte
V2002	1	Start in Position 1 of output buffer
V2003	5	Output 5 bytes (Synch, Devt, Msgt, Len, data)
V2004	0	Format Error Word
V2005	300	V memory 300 contains the first byte
V2006 - V2010	0	Unused
V2011	7002	BCC Arithmetic Summation into 8 bit accumulator
V2012	0	Start in next position
V2013	0	Reserved - Not Used
V2014	0	Format Error Word
V2015	1004	Binary Output Format
V2016	2	Start Calculation with second character in the message
V2017	-1	End Calculation with the character just before the BCC field
V2018	1	Post-processing (one's complement)
V2019	0	Reserved - Not Used
V2020	0	Unused
V2021	65000	Table End Format Specification

Format Table (Input)

The following Format Table is used for the input message of the POLLED READ command. Note that V106 in the POLLED READ command block contains the V memory address of this table. Since the response has no data, the only requirement is to ensure that the device did respond with a valid packet. Format 6003 accomplishes this purpose.

Location	Value	Comments
V3000	19456	Signature Value
V3001	6003	SEAbus Message Verification
V3002	1	Unused Field - set to 1
V3003	1	Unused Field – set to 1
V3004	0	Format Error Word
V3005	0x27	Hex 27 is the expected response synch byte
V3006	0	Default Verification Options
V3007	0	Default Timing Options
V3008	0	Will contain data length value after processing
V3009	0	Reserved - Not Used
V3010	0	Reserved – Not Used
V3011	65000	Table End Format Specification

Example 2 –Command with Complex Response

This example illustrates a command that contains no data, with a slave response whose data section contains both byte-oriented and word-oriented data. This particular command requests the ambient temperature and pickup currents. The following data is returned:

Data Byte No.	Description
1	Ambient Temperature setting (0-70)
2	Jam Pickup current percentage (4 – 60)
3	Loss of Load pickup current percentage (20 – 95)
4 - 5	Ground Fault Pickup Current (up to 720 amps) Byte 4 contains the LSB of the data word and byte 5 contains the MSB of the data word.

The following data is stored in V memory:

Address	Value Hex	Value Decimal	Comments
V300	14	20	Synch Byte value for master request
V301	05	5	Device Type (address) of slave device
V302	04	4	Message Type value for Ambient Temperature and Pickup Currents
V303	0	0	Data Length of 0

We will store the Command Block for the CREATE CONNECTION Command at V200, the command block for the POLLED READ command at V100, the output Format Table at V2000, and the input Format Table at V3000.

Command Blocks

A Create Connection command block is used to create a connection to Port 2, using the GAS protocol manager (Number 38). See the previous example for details.

Location	Description	Hex Value	Decimal Value
V200	Command Error Word	0000	0
V201	Command (Create Connection)	0001	1
V202	Connection Number = 19222	4B16	19222
V203	Protocol Manager Number (GAS = 38)	0026	38
V204	Physical Port Number 2	0002	2
V205	Port Baud Rate = 9600	2580	9600
V206	Bits Per Character = 8	0008	8

V207	Parity = 0 = None	0000	0
V208	Stop Bits = 1	0001	1
V209	Handshake = 0 = None	0000	0
V210 -215	Unused - reserved for future use (Set to 0)	0000	0

A POLLED READ command is used to send the request and process the response as shown below. Note that the command timeout is set to 1 second and the input message delimiter is hex 27 (the expected value of the response synch byte). The length is set to 10 bytes, since the response has five bytes of data in addition to five bytes of overhead.

The end of message timeout is set to 0.1 seconds (100 ms) to facilitate error detection based on inter-character timing. SEAbus inter-character timing is a maximum of 50ms. In this example, if no character were received within 100ms, end of message would be detected. When the SEAbus protocol format specification evaluated the packet, it would detect and report an error.

Location	**Description**	**Hex Value**	**Decimal Value**
V100	Error Word	0000	0
V101	Command Code (Polled Read)	2603	9731
V102	Connection Number = 19222	4B16	19222
V103	Command Control Flag	0000	0
V104	V Memory Address of Output Format Table	07D0	2000
V105	Maximum # of output Format Specifications (0 = default)	0000	0
V106	V memory Address of Input Format Table	0BB8	3000
V107	Maximum # of input Format Specifications (0 = default)	0000	0
V108	Command timeout in seconds (0= default)	0001	1
V109	Input message maximum length	000C	12
V110	Input message delimiters, first set	2700	9984
V111	Input message delimiters, second set	0000	0
V112	End-of-Message Timeout in tenths of seconds	0001	1
V113 -115	Unused (Set to 0)	0000	0

Format Table (Output)

The following Format Table is used for the output message of the POLLED READ command. Note that V104 in the POLLED READ command block contains the V memory address of this table. The first format specification builds the message header and the data field by extracting the low byte from V300 through V304 and placing them in the first five bytes of the request message. The second format specification creates the LRC.

Location	Value	Comments
V2000	19456	Signature Value
V2001	8001	PLC Word LSB to Message Byte
V2002	1	Start in Position 1 of output buffer
V2003	4	Output 4 bytes (Synch, Devt, Msgt, Len)
V2004	0	Format Error Word
V2005	300	V memory 300 contains the first byte
V2006 - V2010	0	Unused
V2011	7002	BCC Arithmetic Summation into 8 bit accumulator
V2012	0	Start in next position
V2013	0	Reserved - Not Used
V2014	0	Format Error Word
V2015	1004	Binary Output Format
V2016	2	Start Calculation with second character in the message
V2017	-1	End Calculation with the character just before the BCC field
V2018	1	Post-processing (one's complement)
V2019	0	Reserved - Not Used
V2020	0	Unused
V2021	65000	Table End Format Specification

Format Table (Input)

The following Format Table is used for the input message of the POLLED READ command. Note that V106 in the POLLED READ command block contains the V memory address of this table. Format 6003 is used to validate the response message. Format specification 8001 reads the single byte data into V400 – V 402. Format specification 8002 reads the final two bytes into V403.

Location	Value	Comments
V3000	19456	Signature Value
V3001	6003	SEAbus Message Verification
V3002	1	Unused Field - set to 1
V3003	1	Unused Field – set to 1
V3004	0	Format Error Word
V3005	0x27	Hex 27 is the expected response synch byte
V3006	0	Default Verification Options
V3007	0	Default Timing Options
V3008	0	Will contain data length value after processing
V3009	0	Reserved - Not Used
V3010	0	Reserved – Not Used
V3011	8001	Message bytes to PLC Word LSB
V3012	5	Start in first data position (byte 5)
V3013	3	Read three bytes
V3014	0	Format Error Word
V3105	400	V memory 400 will contain first data byte
V3016	0	Use Normal Processing Options
V3017 – V3010	0	Unused
V3021	8002	Message Byte to PLC Word - Little Endian
V3022	-1	Start in the byte following the last field
V3023	2	Read two bytes
V3024	0	Format Error Code
V3025	403	V403 will contain the data
V3026	0	Normal Processing Options
V3027 – V3030	0	Unused
V3031	65000	Table End Format Specification

When the command is completed successfully, V memory will contain the following data:

V400	Ambient Temperature Setting
V401	Jam Pickup Current Percentage
V402	Loss of Load Current Percentage
V403	Ground Fault Pickup Current

CHAPTER 6. TROUBLESHOOTING

This chapter is intended to provide you with general guidelines for identifying the source of problems and making corrections. Also see the Troubleshooting Chapter in the *Installation and Operation Guide* for the particular CTI module you are using.

If you are having communications problems, you should first examine the Command Error Word in the command block. If an error has been reported, you will see an error code in this word which can help determine the cause of the problem. In addition, if there is an error in processing the format specifications, a format error code will be written to the format specification Format Error word.

6.1. Hardware Related Problems

Communications Cabling

A common source of problems is the cabling between the device and the CTI communication module. Make sure that your cable is wired properly. You can use the port LED's on the module to determine whether you are sending and receiving a signal. See the module *Installation and Operation Guide* (IOG) more detail.

Multi-drop Applications

Ensure that termination resistors are installed as required. See the documentation for your device. Check that each device is operating properly; one malfunctioning device can cause signal problems that affect all units on the line.

Communications Parameter Setting

Another common source of problems is communications settings that do not match. Make sure that baud rate, data bits, stop bits, etc. are set properly on both the device and the module. If hardware handshake is required, make sure it is enabled. When your communications parameters are not set properly and the plant floor device is sending data, you will probably encounter a parity error or framing error returned by GAS. See the module *Installation and Operation Guide* for additional information.

Power Supply

Make sure that each device has adequate power; low power can cause signal problems. Also, if you are obtaining power for multiple devices from separate circuits, ensure that you have not created a ground loop. Ground loops are caused when the ground potential of one circuit is different from another circuit. If you experience communications problems when multiple

devices are connected, but each device operates properly when only that device is connected, you may have a ground loop. To correct ground loop problems, install isolation equipment.

6.2. Protocol Problems

Protocol errors can result from several factors:

1. Communications signal problems are causing a garbled signal,
2. Your format specifications did not match the protocol described in the device documentation,
3. The documentation for the device is incorrect or incomplete.

Using Module LEDs for Troubleshooting

If you are expecting to receive a response from the device, ensure that you are actually getting some signal from the device. You can tell that a signal has been received, because the 2573 RCV LED will flash. If you are not getting a response and the device is supposed to send *unsolicited* messages, check the device setup (and recheck the hardware).

If you are not getting a response (RCV LED does not flash) and the device sends the message only after receiving a command from the 2573 module, observe the XMT LED to determine if the 2573 is sending anything to the device. If the LED does not flash, you are not sending a message from the 2573. Ensure the Command Block is correct and that there are no command errors. Trigger the command manually in this test (see *Development and Debugging Tips* on page 147).

If the XMT LED flashes, the 2573 is sending something, although the message content may not be correct. Use a serial line analyzer or GAS troubleshooting techniques to examine the actual contents of the message. Compare the actual message with the device documentation. If they do not match, correct the Format Table. If they are the same, then the device documentation is incorrect or the device is malfunctioning. Contact the device manufacturer for technical support.

If you are getting a response from the device (2573 RCV LED flashes), but GAS returns a timeout error, you may have specified the message delimiters or message maximum length incorrectly. Check your entries carefully. Use a serial line analyzer or GAS troubleshooting techniques to display the actual message received.

Using a Serial Line Analyzer

A valuable tool in debugging protocol problems is a *serial line analyzer*. This equipment lets you "see" the actual characters being sent down the line in both directions. You can use a dedicated hardware analyzer or you can obtain a PC hardware/software product such as *SerialTest* from Frontline Test Equipment Inc. and *COM-Watch* by CER International. These PC products can be obtained through the typical sources of PC software for around $300.00. In addition, shareware packages are available for under $50.00. Contact CTI for references. In a pinch, you can also use *Microsoft Terminal* to read, display, and save messages sent from either the 2573 or the plant floor device

GAS Troubleshooting Techniques

If you are having problems communicating with the device, GAS provides some options that can help.

Observing Output Messages

The GAS BUILD command allows you to use a Format Table to encode the output message and to write the resulting string to V memory. Use this command with the same Format Table used to send the device message. Then use TISoft or an equivalent programming package to observe the results.

Observing Input Messages

Format number 3003 is an excellent general format to use when debugging incoming messages. This format allows you to place the entire contents of the input message buffer into V memory in ASCII format. You can then use products such as TISOFT to view the actual message received. See page 72 for more details.

If you encounter a timeout error when attempting to read a message using Format 3003 and you observe the RCV LED flash, your problem is probably related to incorrect message delimiters. To eliminate this possibility, specify no end delimiter. Use the End-of -Message timeout value in the command block instead. If substituting the timeout for an end delimiter fails, then try specifying no start delimiter.

If you are getting format conversion errors and wish to observe both the errors in decoding the message and the corresponding input ASCII string, you can do the following:

- In the Command Block, set the Command Control Flag to continue processing on error (0x0002).
- Add a Format 3003 to the end of Format Table.

When you trigger the command, GAS will attempt to decode each field. If an error is encountered, it will write the error word to the applicable Format Specification and continue to the next one. When all Format Specifications have been processed (including Format 3003), GAS will raise the error bit and halt. Now you use TISoft to see the entire string and each error that GAS encountered while attempting to decode the string. This technique is especially helpful in catching intermittent errors.

NOTE:
Errors in decoding a field can cause subsequent decoding attempts to fail. When making corrections, start with the first format specification in error, correct the problem, and retry the command.

6.3. PLC Logic Problems

Command Block Errors

When you are first setting up the command blocks in V memory, it is easy to enter incorrect data into the command block or to omit data from the command block. Some mistakes, such as an invalid command or port number will be obvious, because the GAS Protocol Manager will return an error when they occur.

Command Slot Errors

Other errors can occur if you enter the wrong V memory address for the Command Block or when you omit entering an address in the command slot If you enter a V memory address which does not contain a valid command block or enter an address of 0, the GAS Protocol Manager will raise the PLC ERR bit as well as the CMD ERR bit. No error code will be written, since there is no valid V memory location in which to write one.

NOTE:
A command block is considered invalid if offset 2 (Connection Number) does not contain hex 4B in the high byte of the word. This signature byte places all connection numbers in the range of 19201-19299 decimal (4B01-4B63)

You might enter a V memory address of a *valid* command block, but it is *not* the command block you want to process. In this case, you will probably not get an error, just the wrong result.

Command Control Errors

One common error in manipulating the Command Control bits is a failure to assert ERR ACK (Error Acknowledge) after GAS has raised the CMD ERR (Command Error bit). If you fail to acknowledge the error, the associated command slot will appear to "lock up" on the port. In actuality, the module software is waiting on the PLC acknowledgment before proceeding. You can tell that the module is not really locked up by observing that the timer value in the module WX1 is incrementing and that the CMD ERR bit is set. In addition, if you have triggered other commands, they will continue to operate (unless an error also occurs on these command slots). See the applicable *Installation and Operation Guide* for details.

Another potential error is failing to observe the timing protocol for coupled mode. You must wait for the module to assert CMD Busy before lowering the CMD Trigger. Failure to observe the timing may cause the command slot to appear to "lock up". Actually the module may have missed the CMD trigger. See the applicable *Installation and Operation Guide* for details.

Errors in manipulating the Command Control bits may cause "multiple triggering", resulting in an error code. Certain commands, such as those to create a connection, must be run only

once. Improperly constructed PLC logic may repeatedly trigger the command, resulting in an error code such as 0x00A7 (duplicate connection). See the module *Installation and Operation Guide* for examples of PLC logic.

6.4. Development and Debugging Tips

Manual Triggering

You can test your command blocks and format specifications independently from the PLC logic by manipulating the command control bits manually. First, place the PLC in program mode so that the PLC logic will not be executing. Then you can manipulate the bits as shown below. If you are not familiar with the 2573 PLC Command Interface, you should read the *Installation and Operation Guide* for the module. Also see the WX/WY Quick Reference in Appendix C, page 174.

The command control bits are located in the 4th module word. For example, if you logged the module in starting at Word 1, then the command control bits will be located in WY4. See the WX/WY Quick Reference at the end of this manual. There are 4 sets of control bits, one set for each command slot.

	Bits 1 -4	Bits 5 - 7	Bits 8 - 11	Bits 12 -16
WY4	Command Control Bits - Slot 1	Command Control Bits - Slot 2	Command Control Bits - Slot 3	Command Control Bits - Slot 4
	Hex 0-F	Hex 0 - F	Hex 0 - F	Hex 0 - F

Within each set of four Command Control bits:

- The first bit is the Error Acknowledge,
- The second bit is the Command Mode bit
- The third bit is the Command Trigger
- The fourth bit Abort Trigger.

So a bit pattern of 0110 will set the command mode bit and the command trigger. Similarly, the bit pattern 1000 will set error acknowledge. Using hexadecimal notation is a convenient way to observe and manipulate these bits, because each hexadecimal digit represents 4 bits. Thus, the bit pattern when command trigger and command mode are high (0110) is represented by hexadecimal 6 (0+4+2+0). Similarly, an error acknowledge bit high (1000) is represented by hexadecimal 8.

Similarly, the command status bits written by the module are located at the second module word (WX2 in this example) in a bit grouping that matches the command control word.

	Bits 1 -4	Bits 5 - 7	Bits 8 - 11	Bits 12 -16
WX2	Command Status Bits - Slot 1	Command Status Bits - Slot 2	Command Status Bits - Slot 3	Command Status Bits - Slot 4
	Hex 0-F	Hex 0 - F	Hex 0 - F	Hex 0 - F

Within each set of four Command Status bits:

- The first bit is the Command Error bit,

- The second bit is the PLC Error bit,
- The third bit is the Command Busy bit,
- The fourth bit is the Abort Busy bit,

Therefore, to set the Command Trigger (using uncoupled mode) only for the first command slot, set WY4 to hex 2000. To set the Command Trigger for the second slot, set the WY4 to hex 0200. To set the Error Acknowledge bit for the third Command Slot, set WY4 to hex 0080.

When you are using PLC logic to control the PLC, it is often useful to display the module WX/WY words, parts of the command blocks, and sections of the format specifications as a TISOFT chart. Following is an example chart.

```
LOCATION  STATUS    LOCATION  STATUS    LOCATION  STATUS
WX1  =    HEX       V100 =    HEX       V2001     =    INTEGER
WX2  =    HEX       V101      =    INTEGER   V2004     =
     HEX
WY3  =    HEX       V102 =    INTEGER   V2005     =    INTEGER

WY4  =    HEX       V103 =    INTEGER   V2011     =    INTEGER

WY5  =    INTEGER   V104 =    INTEGER   V2014     =    HEX
WY6  =    INTEGER                       V2015     =    INTEGER

WY7  =    INTEGER   V200 =    HEX       V3001     =    INTEGER
WY8  =    INTEGER   V201 =    INTEGER   V3004     =    HEX
                    V202 =    INTEGER   V3005     =    INTEGER
                    V203 =    INTEGER   V3011     =    INTEGER
                    V204 =    INTEGER   V3014     =    HEX
                                        V3015     =    REAL
```

The above chart allows you to examine the value of the module status word (WX1), the command status word (WX2), the module control word (WY3) , and the command control word (WY4) in hex. It also displays the command slots (WY5-WY8) as integers. V100 and V200 are assumed to contain the command blocks you are using. This display shows the first four words, including the error word in hex format. V2000 is assumed to contain the input Format Table and V3000 the output Format Table. The display shows the format numbers, error codes in hex, and data word 1. Change the display to match your setup.

To trigger a coupled mode command for command slot 1 (Assuming the module is logged at Word 1-8):

1. Put the address of the Command Block you want to use in WY5.
2. Change the value of WY4 to hex 6000. WX2 should change to hex 2000, indicating that the module is executing the command.
3. After WX2 changes to hex 2000, set the value of WY4 to hex 0000 (clears the trigger), If WX2 eventually goes from hex 2000 to hex 0000, the command was executed successfully. If WX2 changes to hex A000, a command error was encountered. Inspect the command error word (in the command block - offset 0)and clear the error by changing WY4 to hex 8000. Then clear error acknowledge by setting WY4 to hex 0000.
4. Examine the applicable Format Table words for errors and input data.

APPENDIX A. GAS COMMAND ERROR CODES

This Appendix provides a description of the error codes that may be returned to the PLC by the GAS Protocol Manager. Potential solutions to the error conditions are also presented.

When the GAS Protocol Manager detects an error, it sets the PLC ERR bit in the command status word and places a command error code in offset 0 of the command block. See the *CTI 257x PLC Command Interface Reference Manual* for a complete description of error processing.

System Errors

System errors are detected by the module operating environment. System error codes have hex 00 in the high byte of the command error word.

HEX	DEC	DESCRIPTION	SOLUTION
009F	159	INVALID CONNECTION NUMBER The command block contains an invalid connection number.	Correct the parameter in the connection number in the command block.
00A5	165	INVALID SYSTEM COMMAND Connection number 19200 (system) was specified in the command block but the command is not a valid system command.	Change the command block parameter to a valid connection number or the command number to a valid system command.
00A6	166	CONNECTION NOT ACTIVE An attempt was made to send a command to a connection number that has not been created. Note: you can receive this error if you enter the connection number incorrectly.	Check the following for the Create Connection Command: 1) Make sure that the command block contains the correct data, 2) Ensure that the Command Slot points to the proper command block, 3) Ensure that the Command Trigger has been set. 4) Check for reported errors for the Create Connection Command. Ensure that you have entered the correct connection number in the command block.

HEX	DEC	DESCRIPTION	SOLUTION
00A7	167	DUPLICATE CONNECTION NUMBER An attempt was made to start two protocol managers with the same connection number and the same physical port.	Check the Command Blocks for the respective Create Connection commands. Ensure that the Create Connection commands use different connection numbers and different ports. Make sure that another protocol manager was not started via dipswitch with the same connection number. See the product IOG.
00A8	168	DUPLICATE CONNECTION NUMBER An attempt was made to start two protocol managers with the same connection number.	See above.
00A9	169	INVALID PROTOCOL MGR NUMBER An attempt was made to start a protocol manager which does not exist in the list of valid protocol manager numbers.	This error will occur if you are using PLC logic to create a connection to a port and you specify an invalid protocol manager number in the command block. Review the documentation and correct the command block.
00AA	170	INVALID PORT NUMBER An attempt was made to start a protocol manager with an invalid physical port number. Valid physical port numbers for the 2573 are 1,2,3, or 4.	This error will occur if you enter an invalid physical port number in the command block for the Create Connection command. Correct the command block and retry the command.
00AB	171	PORT ALREADY CONNECTED An attempt was made to connect two protocol managers to the same physical serial port. Only one protocol manager can be connected to one physical serial port.	This error is typically caused by one of two circumstances. 1) The PLC logic triggers the same Create Connection command more than once. 2) You have inadvertently used the same physical number more than once in separate Create Connection commands. In case 1, you should examine and correct your PLC logic. In case 2, you should correct the physical port entry in the applicable command block(s).

GAS Protocol Manager Errors

The following errors are detected by the GAS protocol manager. These protocol manager errors always have hex 26 in the high byte.

HEX	DEC	DESCRIPTION	SOLUTION
266E	9838	INVALID V MEMORY ADDRESS The command block contained value of 0 as the starting V memory address.	Correct the value in the command block.
266F	9839	MAXIMUM FORMATS EXCEEDED The command block contained a value for number of format specifications that exceeded the maximum of 24.	Correct the parameter value in the command block
2670	9840	INVALID FORMAT TABLE SIGNATURE The first word of V memory did not contain a valid "signature" value which identifies it as a format specification. Since the format data is suspect, the protocol manager suspended processing and posted this error.	Ensure that the starting word of the Format Table contains the value 19456 (hex 4C00). Ensure that the address for the format specification entered in the command block is correct.
2671	9841	FORMAT ERROR CODE The GAS protocol manager discovered a format specification error. An error code has been written to the Format Error word of one format specification.	Review the format specifications for one containing a non zero Format Error word. Then correct the format as indicated by the Format Error Code. See Appendix B.
2672	9842	MAXLENGTH OUT OF RANGE The command block contained a value for maximum packet length that was less than 1 or greater than 1080 characters.	Correct the parameter value in the command block.
2673	9843	MISSING TABLE END SPECIFICATION After reading the maximum number of Format Specifications defined in the Command Block., GAS was not able to locate an TABLE END Format Specification	Ensure that the referenced Format Table has an TABLE END Format Specifically. If an TABLE END Format Specification exists, GAS is not reading enough format specifications to find it. Ensure that the Value for Maximum Number of Format Specifications in the Command Block is set high enough to read all format specifications.
2674	9844	INVALID DELIMITER COMBINATION You specified a second set of delimiters without specifying a first set **or** You specified two sets of delimiters which did not contain both a start and an end delimiter.	Correct the Command Block entry. If you use two sets of delimiters, both sets must specify a start delimiter and an end delimiter.

HEX	DEC	DESCRIPTION	SOLUTION
2675	9845	BCC COMPARE ERROR The computed Block Check Character(s) did not match the BCC in the message.	This error may be caused by “noise” on the communications line. Retry the command. If the error persists, you probably have specified the BCC incorrectly. Try bypassing the BCC spec to observe whether you actually receive the message correctly.
2676	9846	INVALID COMMAND FLAG You set a command flag that was not accepted by the command. For example, you set the Inhibit Buffer Clear bit with the PARSE MESSAGE Command.	Clear the option bit.
2697	9879	UNKNOWN COMMAND The protocol manager received a command number which is not valid.	Correct the parameter value in the command block.
2698	9880	COULD NOT OPEN PORT The protocol manager could not open the 257x serial port.	Retry the command. If the problem persists, replace the module.
2699	9881	ERROR READING SERIAL PORT The protocol manager detected an error during a read on the 257x serial port.	Retry the command. If the problem persists, replace the module.
269A	9882	ERROR WRITING SERIAL PORT The protocol manager detected an error during a write on the 257x serial port.	Retry the command. If the problem persists, replace the module.
269D	9885	COMMAND TIMEOUT ERROR The protocol manager detected a command timeout error. This means that a complete message (as defined by the start delimiter, end delimiter, and maximum length) was not received in the specified elapsed time	This may indicate the device is busy, in which case you may choose to retry the command. It may also indicate that the device is off-line or is malfunctioning. Check the status of the device. If you encounter timeout errors often, you may wish to increase the value for command timeout in the command block. If you observe that a message was returned from the device (2573 RCV LED flashes), the start delimiter, end delimiter, or maximum length are probably set improperly. Check the command block and correct. See Chapter 5 for troubleshooting hints.
269E	9886	PROTOCOL MANAGER ALREADY BUSY The protocol manager received a command before having finished executing it's current command. The protocol manager will process only one command at a time.	This error can be encountered when you are using two different command slots to trigger a command to the same connection number. This results from a PLC logic error that must be corrected.

HEX	DEC	DESCRIPTION	SOLUTION
26A0	9888	BAUD RATE SELECTION ERROR The protocol manager was passed an invalid baud rate. Valid baud rates are 75, 110, 150, 300, 600, 1200, 2400, 4800, 9600, 19200.	Correct the parameter value in the command block.
26A1	9889	DATA BITS SELECTION ERROR The protocol manager was passed an invalid data bits value. Valid data bit values are 5, 6, 7, 8.	Correct the parameter value in the command block
26A2	9890	PARITY SELECTION ERROR The protocol manager was passed an invalid parity value. Valid parity values are 0-None, 1-Odd, 2-Even.	Correct the parameter value in the command block
26A3	9891	STOP BIT SELECTION ERROR The protocol manager was passed an invalid number of stop bits. Valid stop bit values are 1 and 2.	Correct the parameter value in the command block
26A4	9892	HANDSHAKE SELECTION ERROR The protocol manager was passed an invalid handshake value. Valid values are 0-No Handshake, 1-Software Handshake, 2-Hardware Handshake.	Correct the parameter value in the command block
26AC	9900	ERROR READING V MEMORY The protocol manager detected an error reading V-memory. You have specified a V memory address that causes the protocol manager to try to access PLC V memory that is not present. This can happen if the specified starting V memory address, plus additional memory implied by a length specification, exceeds the V memory boundary.	You must either: Configure the PLC for more V memory. Alter the V memory address specification so that the access is within the PLC range. Alter the corresponding length specification so that the access is within the PLC range.
26AD	9901	ERROR WRITING V MEMORY The protocol manager detected an error writing V memory.	See error code 26AC above.
26B4	9908	PORT INPUT OVERRUN ERROR Characters in the port input buffer have been overwritten. Characters have been lost.	Retry. If the error persists, reduce the baud rate for this port, increase the PLC task codes per scan, or shut down one of the 2573 ports.

HEX	DEC	DESCRIPTION	SOLUTION
269F	9887	INVALID CONNECTION NUMBER The connection number used in the Create Connection command was out of the valid range.	Correct the parameter value in the command block.
26B5	9909	INPUT FRAMING ERROR Some input characters were lost due to a character framing error.	This may result from a “noisy communications environment. Reduce the baud rate or improve the signal quality. If the error persists is usually results from communications parameters (baud rate, data bits, stop bits, etc.) which do not match the plant floor device.
26B6	9910	INPUT PARITY ERROR A character parity error was detected.	See error code 26B5 above.

Protocol Specific Errors

The GAS protocol manager detects the following errors when using protocol specific format specifications.

SeaBUS Error Codes

These errors are associated with the input format specification 6003.

HEX	DEC	DESCRIPTION	SOLUTION
6010	24592	INVALID SEABUS START POSITION The format specification for SEAbus did not have a valid start position. The only valid start position is 1 or -1.	Correct the parameter value in the command block.
6011	24593	INVALID SEABUS LENGTH The format specification for SEAbus did not have a valid length. The only valid lengths are 1 and –1.	Correct the parameter value in the command block.
6012	24594	INVALID SEABUS SYNC CHARACTER The format specification for SEAbus did not have a valid sync character. The only valid sync characters are 0x14 and 0x27.	Correct the parameter value in the command block.
6013	24595	SEABUS RESPONSE SYNC ERROR The SEAbus device response did not have the correct sync character.	Ensure that you have entered the correct synch character in the format specification. For applications where the 2573 is the master, the value should be 0x27.
6014	24596	SEABUS RESPONSE CHECKSUM ERROR The SEAbus device response did not have a valid checksum.	If the problem persists, check your cabling, termination, and any other condition that might cause “noise” on the data line.
6015	24597	SEABUS RESPONSE SIZE ERROR The SEAbus device responded with a packet with an unexpected length.	If the problem persists, ensure that the MAX LENGTH entry in the command block is equal or greater than the expected length

APPENDIX B. FORMAT ERROR CODES

When the GAS protocol manager encounters an error processing a Format Specification, it:
1. Sets the Command Error Bit,
2. Places a format error code in the Format Error Word of the Format Specification, and
3. Places the command error code in the Command Error Word of the Command Block.

When an error occurs, you should first examine the Command Error Word to determine the type of error that has occurred. If the Command Error Code indicates a Format Error has occurred, then examine the Format Table for a non-zero Format Error Word. The Format Error Code should help you determine the cause of the problem. See Chapter 5 for troubleshooting information. GAS does not reset the Format Error Code nor the Command Error Code to zero after the error is acknowledged. This must be done by PLC logic.

NOTE:
To prevent potential confusion, your PLC logic should reset the error codes to zero once you have corrected the error.

HEX	DEC	DESCRIPTION	SOLUTION
0001	1	INVALID FORMAT SPECIFICATION An invalid or unsupported format specification number was discovered.	Correct the format specification number in error. Note: This error may be encountered if there is no END format specification. Ensure that you have an END format specification as the last specification.
0002	2	LENGTH ERROR The specified length exceeds the remaining characters in the input message.	Correct the format specification. Reduce the length or change the start location. Use a variable length specification instead.
0003	3	INVALID START POSITION The format specification did not have a valid start position. Valid start positions are 1 - 1080 and -1.	Correct the format specification.
0004	4	INPUT LENGTH ERROR Non-numeric characters were found within the number of characters designated by the length specification. Therefore they cannot be converted to a PLC numeric format.	Correct the format specification. Decrease the length or change the start location. Consider using the -1 length value (variable length).

HEX	DEC	DESCRIPTION	SOLUTION
0005	5	MAXIMUM LENGTH EXCEEDED The format specification could not add the specified characters to the output buffer because it would exceed the maximum output buffer length of 1080.	Correct the format specification. This could be caused by an incorrect starting position or an incorrect length.
0006	6	INVALID INTEGER LENGTH The format specification (format number 1001) did not have a valid character length specified. Valid values for length are 1-9 and -1	Correct the format specification.
0007	7	INVALID INTEGER SIZE The format specification contained a character length less than the length required to represent the integer value.	Correct the format specification. Increase the value for character length so that the integer can be represented.
0008	8	INVALID CHARACTER LENGTH The format specification for format number 3001 contained and invalid character length. Valid lengths are -1, and 1 - 12.	Correct the format specification.
0009	9	INVALID FILL LENGTH The format specification for format number 4001 did not have a valid fill char length specified. Valid lengths are 1 through 1080.	Correct the format specification.
000A	10	INTEGER START POSITION ERROR The protocol manager cannot process the input using format specification 1001 because the start position specified did not contain an ASCII digit (0-9) or a specified pad character. This may also occur when the field contains *only* pad characters and no decimal digits.	Correct the format specification.
000B	11	INTEGER TOO LARGE The protocol manager cannot process the input using format specification 1001 because a value > 65535 was encountered. This number cannot be represented in 16 bits. You may encounter this error if you are using a length of -1 (variable length) and read more numeric characters than you expect.	Correct the format specification. Consider using a fixed length value instead of -1. If the number you are trying to decode is really larger than 65535, use another format code, such as 1002.

HEX	DEC	DESCRIPTION	SOLUTION
000C	12	INVALID V MEMORY ADDRESS The format specification 3002 or 3003 had a V-memory address of 0, which is invalid.	Correct the format specification. Enter a valid V memory address.
000D	13	INVALID HEX LENGTH The format specification contained a character length that is invalid for ASCII data. Valid lengths are -1 and 1-4.	Correct the format specification. Enter a valid length.
000E	14	INVALID HEX SIZE The format specification contained a length less than the length required to represent the hexadecimal value.	Correct the format specification. Enter a larger value or use the variable length (-1) value.
000F	15	SIGNED INTEGER TOO LARGE The protocol manager cannot process the input using format 2001 because a value > 32767 was encountered. This number cannot be represented with a 16 bit signed integer. You may encounter this error if you are using a length of -1 and there is not a non-digit character after the 9th position.	Correct the format specification. If you are using variable length (-1) you may need to specify a fixed length. If the number is actually larger, you must use a different format specification, such as 2002.
0010	16	SIGNED INTEGER INVALID START The protocol manager cannot process the input using format 2001 because a character other than '+' (0x2B),'-' (0x2D), or space (0x20) was encountered in the start position specified.	Correct the format specification. Change the start position value.
0011	17	SIGNED INTEGER INVALID DIGIT The protocol manager cannot process the input using format 2001 because there was no valid ASCII digit (0-9) or pad character after the '+','-', or space character. You may also encounter this error if the field contains no valid ASCII digits or only pad characters.	Correct the format specification. Check the value you entered for start position.
0012	18	SIGNED INTEGER INVALID LENGTH The format specification did not have a valid ASCII signed integer length specified. The maximum character length is 10. Valid values are 2-10 and -1.	Correct the format specification. Change the length value.
00013	19	INVALID SIGNED INTEGER SIZE The format specification contained a length which was less than the length required to represent the signed integer value.	Correct the format specification.

HEX	DEC	DESCRIPTION	SOLUTION
00014	20	INVALID REAL NUMBER LENGTH The format specification did not have a valid ASCII real length specified. The valid values for input are 1-16 and -1. The valid values for output are 2-16.	Correct the format specification
00015	21	INVALID NUMBER OF DECIMAL CHAR The format specification did not have a valid value for number of digits to the right of the decimal point. Valid values are 0 - 6.	Correct the format specification.
00016	22	INVALID REAL NUMBER SIZE The format specification had a length which was less than the length required to represent the real value.	Correct the format specification
0017	23	INVALID NEGATIVE REAL The format specification 1002 had a real value which was negative. You cannot represent a negative real value with this format	Check the value for Start Position in the format specification and correct if necessary. Use format specification number 2002 instead.
0018	24	REAL NUMBER INVALID START The protocol manager cannot process the input using format 1002 because the start position specified did not contain an ASCII digit(0-9), decimal point (0x2E), or a specified pad character. This may also occur if the field did not contain a valid decimal digit (e.g. a decimal point and/or pad characters were the only characters present.)	Correct the format specification.

HEX	DEC	DESCRIPTION	SOLUTION
0019	25	REAL NUMBER LENGTH ERROR The protocol manager cannot process the input because length of -1 was specified and: 1) you are using format number 1002 and the character after 16th was a digit or a decimal. This character must be a non-digit or non-decimal for the -1 length to work. 2) you are using format number 2002 and the character after 17th was a digit or a decimal. This character must be a non-digit or non-decimal for the -1 length to work or 3) there were more than 6 digits after the decimal point and the value for length forced more characters to be evaluated.	Correct the format specification. Change the length to a fixed number. Reduce the fixed length value.
001A	26	INVALID HEX NUMBER START The protocol manager cannot process the input using format 1003 because the start position specified contained a character which is not a valid hex character (0-9, A-F, or a-f).	Correct the format specification
001B	27	HEX NUMBER TOO LARGE The protocol manager cannot process the input using format 1003 because a length of -1 was specified and the 5th character was a hexadecimal character. This character must be a non-hexadecimal character for the -1 length to work.	Correct the format specification. Use a fixed length in the specification.
001C	28	INVALID LENGTH SPECIFICATION The format specification contained an invalid value for length.	Correct the format specification. See the format table description for valid length values.
001D	29	INVALID SIGNED REAL LENGTH The format specification 2002 did not have a valid ASCII signed real length specified. Output valid lengths are -1, and 3 -17.	Correct the format specification

HEX	DEC	DESCRIPTION	SOLUTION
001E	30	INVALID SIGNED REAL START The protocol manager cannot process the input using format 2002 because the start position specified contained a character which was not a "+" (0x2B), a "-" (0x2D), or a space character (0x20).	Correct the format specification.
001F	31	INVALID LENGTH - FORMAT 3002 For format number 3002: the number of characters specified in the length word were examined without finding the end of the message.	Correct the format specification. For format number 3002, the length word specifies the *maximum* number of characters that will be examined. Increase the length or check your format specification for other errors.
0020	32	INVALID LENGTH - FORMAT 4002 For format number 4002, the number of characters specified in the length word were examined without finding a null terminator, or A null terminator was found in the very first position, before any data characters.	Correct the format specification. For format number 4002, the length word specifies the *maximum* number of characters in V memory that will be examined . Increase the length or check your format specification for other errors. Check the V memory address specified. You may be pointing to the wrong address.
0021	33	INVALID PRE-PROCESSING METHOD The checksum format specification references a pre-processing method which is not supported.	Correct the format specification. Make sure that you have the correct value in the ***high*** byte of Offset 8.
0022	34	INVALID POST-PROCESSING METHOD The checksum format specification references a post-processing method which is not supported.	Correct the format specification. Make sure that you have the correct value in the ***low*** byte of Offset 8.
0023	35	INVALID CHARACTER FORMAT The checksum format specification references a character format which is not supported.	Correct the format specification. Ensure that Offset 5 contains a valid character format.
0025	37	INVALID BCC The calculated BCC did not match the BCC contained in the input message.	This error could result from communications “noise” on the line. Retry. If the problem persists, you probably have specified the BCC incorrectly. Correct and retry or bypass the BCC format specification.
0026	38	INVALID BCC CALC START POSITION The BCC format specification contained an invalid value for the start position for the BCC calculation (offset 6).	Correct the format specification. Valid values are 1-1080.

HEX	DEC	DESCRIPTION	SOLUTION
0027	39	INVALID ANCHOR POSITION The BCC format specification contained an invalid value for end position for the BCC calculation (offset 7).	Correct the format specification. Valid Values are 1 through 1080 and -1 through -1080.
0028	40	INVALID RELATIVE MOVEMENT VALUE The value in for relative movement in offset 3 is outside the valid range. The minimum value is -1080; the maximum is +1080.	Correct the format specification.
0029	41	INVALID ANCHOR POSITION The buffer position format specification contained an anchor position which was the beginning of the previous field but no previous field had been specified.	Correct the format specification.
002A	42	INVALID CHARACTER STRING LENGTH The buffer position format specification contained a value in offset 6 other than 1 through 8.	Correct the format specification.
002B	43	STRING NOT FOUND The nth occurrence of the specified string was not found in the input message buffer.	This error is typically caused by a mistake in the format specification. You should check: 1. How the anchor position was specified. Make sure of the direction and the occurrence number. 2. The length in Offset 6. Ensure that the value matches the number of characters you wish to match. 3. The character string in Offset 7 through 10. 4. The actual format of the input message to ensure that it matches the expected format.
002C	44	INVALID ANCHOR POSITION The entry in Offset 2 contained an invalid Anchor position.	Correct the format specification.
002D	45	SPECIFIED POSITION BEFORE START The format specification attempted to position the data cursor before the beginning of the message.	Correct the format specification. The error can be caused by an incorrect combination of the anchor position (offset 2) and the relative movement (offset 3). You should check both values carefully. With the string search anchor positions, take note that the anchor position is the ***beginning*** of the character string.

HEX	DEC	DESCRIPTION	SOLUTION
002E	46	SPECIFIED POSITION AFTER END The format specification attempted to position the data cursor after the end of the message.	Correct the format specification. The error can be caused by an incorrect combination of the anchor position (offset 2) and the relative movement (offset 3). You should check both values carefully. With the string search anchor positions, take note that the anchor position is the ***beginning*** of the character string.
002F	47	INVALID BCC START POSITION The specified start position would cause the BCC field to be outside the message buffer. The minimum number is 0. The maximum is 1081 less the length of the BCC field.	Correct the format specification. If the BCC field length is 1, then the maximum number is 1080. If the BCC field length is 2, then the maximum number is 1079. If the BCC field length is 4, then the maximum number is 1077.
0030	48	BCC FIELD IN CALCULATION RANGE You specified a BCC start position which would place the BCC field within the range of characters used to calculate the BCC.	Correct the format specification.
0031	49	DATA IS NOT IN HEX ASCII FORMAT You specified a pre-processing option which expected the data characters to represent a number in hexadecimal format. The data contained characters which were not HexASCII (not 0-9, A-F, or a-f).	Correct the format specification.
6005	24581	INVALID MODBUS START POSITION The format specification for the MODBUS did not have a valid start position. The only valid start positions are 1 and -1.	Correct the format specification.
6006	24582	INVALID MODBUS LENGTH The format specification for the Modbus slave did not have a valid length. The only valid length is -1.	Correct the format specification.
6007	24583	INVALID MODBUS ADDRESS The format specification for the Modbus slave did not have a valid address. The valid range is 0-247 (0x00 - 0xF7).	Correct the format specification.
6008	24584	INVALID MODBUS FUNCTION The format specification for the Modbus slave did not have a valid function. The valid range is 0-255 (0x01 - 0xFF).	Correct the format specification.
6009	24585	INVALID MODBUS DATA COUNT The format specification for the Modbus slave had an invalid data byte count specified. The valid byte counts 0 - 1080.	Correct the format specification.

HEX	DEC	DESCRIPTION	SOLUTION
600A	24586	BAD MODBUS RESPONSE ADDRESS The Modbus slave did not respond with the address entered in the input format specification.	You probably entered the wrong address in the input format specification. This address does not match the address entered in the output specification for the corresponding query. Correct the entry. You may have a problem with the slave or multidrop network setup. See the manufacturer's documentation.
600B	24587	BAD MODBUS RESPONSE FUNCTION The Modbus slave did not respond with the same function code (or it's corresponding exception code) as sent in the command.	You probably entered the wrong function code in the input format specification. This should match the function code entered in the output specification for the corresponding query. Correct the entry. This may indicate a problem with the slave device. This error may be intermittent, retry the command. See the manufacturer's documentation.
600C	24588	INVALID MODBUS BCC The Modbus slave response did not have a valid BCC.	Indicates a problem with the slave device or the communications link. This error may be intermittent, retry the command. If the problem persists and the condition of the data cables and the device. Replace the defective component.
600D	24589	MODBUS MEMORY ERROR The reply from the Modbus slave contained more data than can be stored in the V memory reserved in Data Word 4.	Correct the Format Specification Change the value in Data word 4 to increase the amount of memory reserved.

APPENDIX C. REFERENCE MATERIAL

Hexadecimal to ASCII Conversion Table

00	NUL
01	SOH
02	STX
03	ETX
04	EOT
05	ENQ
06	ACK
07	BEL
08	BS
09	HT
0A	LF
0B	VT
0C	FF
0D	CR
0E	SO
0F	SI
10	DLE
11	DC1
12	DC2
13	DC3
14	DC4
15	NAK
16	SYN
17	ETB
18	CAN
19	EM
1A	SUB
1B	ESC
1C	FS
1D	GS
1E	RS
1F	US

20	space
21	!
22	"
23	#
24	$
25	%
26	&
27	'
28	(
29	)
2A	*
2B	+
2C	,
2D	-
2E	.
2F	/
30	0
31	1
32	2
33	3
34	4
35	5
36	6
37	7
38	8
39	9
3A	:
3B	;
3C	<
3D	=
3E	>
3F	?

40	@
41	A
42	B
43	C
44	D
45	E
46	F
47	G
48	H
49	I
4A	J
4B	K
4C	L
4D	M
4E	N
4F	O
50	P
51	Q
52	R
53	S
54	T
55	U
56	V
57	W
58	X
59	Y
5A	Z
5B	[
5C	\
5D	]
5E	^
5F	_

60	`
61	a
62	b
63	c
64	d
65	e
66	f
67	g
68	h
69	i
6A	j
6B	k
6C	l
6D	m
6E	n
6F	o
70	p
71	q
72	r
73	s
74	t
75	u
76	v
77	w
78	x
79	y
7A	z
7B	{
7C	|
7D	}
7E	~
7F	DEL

Command Block Summary

See Chapter 2 for more information on the command blocks.

Create Connection

Offset	Description	Hex Value	Decimal Value
0	Command Error Word	**0000**	**0**
1	Command (Create Connection)	**0001**	**1**
2	Connection Number (19221 - 19299)		
3	Protocol Manager Number	**0026**	**38**
4	Physical Port Number (1,2,3,4)		
5	Port Baud Rate (300, 600, 1200, 2400, 4800, 9600, 19200)		
6	Bits Per Character (7 or 8)		
7	Parity (0=None, 1=Odd, 2= Even)		
8	Stop Bits (1 or 2)		
9	Handshake (0=None, 1=Software, 2=Hardware)		
10-15	Unused - reserved for future use (Set to 0)	**0000**	**0**

Close Connection

Offset	Description	Hex Value	Decimal Value
0	Command Error Word	**0000**	**0**
1	Command Code (Close Connection)	**0002**	**02**
2	Connection Number (19221 - 19299)		
3 - 15	Unused (Set to 0)	**0000**	**0**

Write

Offset	Description	Hex Value	Decimal Value
0	Command Error Word	**0000**	**0**
1	Command Code (Write)	**2601**	**9729**
2	Connection Number (19221 - 19299)	**0**	**0**
3	Command Control Flag		
4	V Memory Address of Format Table - Output		
5	Maximum # of Format Specifications to Read (0 = Default = 6)		
6- 15	Unused (Set to 0)	**0000**	**0**

Read

Offset	Description	Hex Value	Decimal Value
0	Command Error Word	**0000**	**0**
1	Command Code (Read)	**2602**	**9730**
2	Connection Number (19221 - 19299)		
3	Command Control Flag		
4	V Memory Address of Format Table - Input		
5	Maximum # of input Format Specifications to read (0 = default = 6).		
6	Read timeout in seconds (0 = default = 9 seconds).		
7	Input message maximum length		
8	Input message delimiters, first set		
9	Input message delimiters, second set		
10	End-of Message Timeout in 10ths of seconds (0 = unused, 1-9 = 0.1-0.9 seconds)		
10-15	Unused (Set to 0)	**0000**	**0**

Polled Read

Offset	Description	Hex Value	Decimal Value
0	Command Error Word	**0000**	**0**
1	Command Code (Polled Read)	**2603**	**9731**
2	Connection Number (19221-19299)		
3	Command Control Flag		
4	V Memory Address of Format Table - Output		
5	Maximum # output Format Specifications to Read (0 = default = 6)		
6	V Memory Address of Format Table - Input		
7	Maximum # input Format Specifications to Read (0= default = 6)		
8	Command timeout in seconds		
9	Input Message maximum length		
10	Input Message delimiters, first set		
11	Input Message delimiters, second set		
12	End-of-Message Timeout in 10ths of seconds (0 = not used, 1-9 = 0.1-0.9)		
12 -15	Unused (Set to 0)	**0000**	**0**

Build Message

Offset	Description	Hex Value	Decimal Value
0	Command Error Word	**0000**	**0**
1	Command Code (Build Message)	**2604**	**9732**
2	Connection Number (19221 - 19299)		
3	Command Control Flag	0000	0
4	V Memory Address of Format Table - Output		
5	Maximum # of Format Specifications to Read (0 = Default = 6)	0000	0
6	Starting V Memory Address of Output String		
7- 15	Unused (Set to 0)	**0000**	**0**

Parse Message

Offset	**Description**	**Hex Value**	**Decimal Value**
0	Command Error Word	**0000**	**0**
1	Command Code (Parse Message)	**2605**	**9733**
2	Connection Number (19221 - 19299)		
3	Command Control Flag (see below)	0000	0
4	V Memory Address of Format Table - Input		
5	Maximum # of input Format Specifications to read (0 = default = 6).	0000	0
6	Command timeout in seconds (0 = default = 9 seconds).	0000	0
7	Input message maximum length		
8	Input message delimiters, first set		
9	Input message delimiters, second set		
10	Starting V memory address of input string		
11-15	Unused (Set to 0)	**0000**	**0**

WX / WY Quick Reference