



ASCII Slave

Overview

Maple Systems' Silver Plus Series Operator Interface Terminals (Maple OITs) can communicate with an ASCII Host using a simple ASCII slave protocol. When configured with EZware-5000, the OIT is the slave in a single-master, single (or multiple)-slave format. Please refer to the "*Silver Plus Series Installation and Operation Manual*" for information on connecting multiple Maple OITs to a single ASCII Host.

Communications Cable

The Maple OIT can be connected directly to a serial port on the Host.

A list of communications cables offered by Maple Systems, as well as instructions to assist you in assembling your own cable, are available on our website at www.maple-systems.com.

WARNING: If your communications cable is not wired exactly as shown in our cable assembly instructions, damage to the OIT or loss of communications can result.

Description

The ASCII Slave protocol allows a Host to read values from and write values to the OITs internal LW (Local Word) and LB (Local Bit) devices. With this protocol, the Maple OIT is the slave, so the Host is the master and must control both reading from and writing to the OIT. This protocol is a silent protocol and it will not initiate communication except after it receives a command.

Accessible OIT Memory

Register Memory

The following table lists the OIT's register memory ranges that the Host can read/write.

| Register Type | Address Range | Format | PLC Register Description |
|---------------|---------------|--------|-----------------------------------|
| LW | 0 - 8999 | dddd | General Use Local Words |
| LW | 9000 - 9999 | dddd | Reserved Local Words ¹ |

Discrete Memory

The following table lists the OIT's discrete memory ranges that the Host can read/write.

| Register Type | Address Range | Format | PLC Register Description |
|---------------|---------------|------------------|----------------------------------|
| LB | 0 - 8999 | dddd (d=decimal) | General Use Local Bits |
| LB | 9000 - 9999 | dddd (d=decimal) | Reserved Local Bits ¹ |

¹ **NOTE:** These addresses are reserved for use by the OIT, and should only be used for their intended purpose. Refer to the "*Silver Plus Series Installation and Operation Manual*" for a detailed list of these registers.

EZware-5000 Settings

The following table lists the communications settings that must be configured in EZware-5000. These settings can be found in the Edit-Set System Parameters menu under the PLC tab. Please note:

- the **Recommended Settings** column provides the recommended setting based upon default settings most commonly used in ASCII devices.
- the **Options** column lists EZware-5000's options; your controller may not support every option

| Name | Recommended Settings | Options | Important Notes |
|-----------------------------------|----------------------|-----------------------------------|--|
| Name: | ASCII Slave | | Description label |
| HMI or PLC | PLC | | |
| Location | Local | Local, Remote | Select local if PLC directly connected to OIT, remote if PLC connected thru another OIT |
| PLC type: | ASCII Slave | | |
| PLC I/F: | RS232 | RS232, RS485 2W, RS485 4W | Must match the controller port setting |
| Station No.: | 1 | 0 to 255 | The station number assigned to the OIT |
| Settings: COM: | COM1 | COM1-COM3 | Serial port of OIT connected to PLC |
| Settings: Baud Rate: | 9600 | 9600, 19200, 38400, 57600, 115200 | Must match the controller port setting. Use the fastest baud rate supported by controller. |
| Settings: Data bits: | 8 | 7 or 8 | Must match the controller port setting |
| Settings: Stop bits: | 1 | 1 or 2 | Must match the controller port setting |
| Settings: Parity: | Even | Even, Odd, none | Must match the controller port setting |
| Settings: Timeout (sec): | 1.0 | 0.1 to 25.5 | Adjust if longer timeout required |
| Settings: Turn around delay (ms): | 0 | 0 to 1000 | Time between the reception of a command and the transmission of the reply |

| Name | Recommended Settings | Options | Important Notes |
|--|----------------------|----------------|--|
| Settings: Protocol | Robust | Robust, Simple | <p>Robust. The protocol uses the non-printable characters STX9 (02H) and ETX (03H), ACK (06H), and NAK (15H); and includes a 2-byte checksum.</p> <p>Simple. Some Host devices (such as some Motion Controllers) are not capable of generating the non-printable characters, or calculating the checksum. In this mode, the data packets are formed as defined in this document, but do not include the STX, ACK, ETX, NAK or checksum. The packet sent by the host should have a CR (0x0D) at the end of the packet, and the packet sent by the OIT should also have a CR at the end.</p> |
| Settings: Response to Write Commands: | On | On, Off | <p>Sets whether or not the OIT responds to Write commands.</p> <p>Note: If set to 1, the Turn-Around delay setting has no effect.</p> |
| Interval of block pack (words): | 5 | 0 to 512 | See <i>Silver Plus Series Installation and Operation Manual</i> |
| Max. read-command size (words): | 120 | | Not Adjustable |
| Max. write command size (words): | 120 | | Not Adjustable |

Network Support

Wiring

This ASCII slave protocol supports network wiring using RS485 2-Wire(Half Duplex) or 4-Wire(Full Duplex) communications based on the setting of the PLC I/F port.

Addressing

This protocol supports multiple OITs. Each unit individual unit must have a separate unique ID#, the station number. Valid station addresses are from 1 to 255. Station address 0 should not be used as a fixed station ID as this station address may be used for the host to send a broadcast message.

Broadcast Messages

When a slave OIT receives a command with a station address of 0, it will be considered a broadcast message. Broadcast messages shall be processed by the OIT, regardless of the OIT's set station ID number. The OIT will process the command, but it will not issue a reply message, regardless of the setting of the Response to Write Commands setting.

Protocol Description

Command List

| Mnemonic | Command Name | Description |
|-----------------|---------------------|---|
| RD | Batch Read | Reads specific data in a contiguous block |
| WD | Batch Write | Writes specific data in a contiguous block |
| RR | Random Read | Reads data from multiple, non-consecutive registers |
| RW | Random Write | Writes data to multiple, non-consecutive registers |
| RC | Read Coil | Reads the specified coils in a contiguous block |
| WC | Write Coil | Writes the specified coils in a contiguous block |

Command Details

Command requests and replies are issued as a series of ASCII coded characters, except for items specified in between the '<' '>' (greater than, less than brackets). The items that are signified by the <> brackets are hexadecimal, non-printable characters. (I.e. <STX> = hex 0x02 = 02H).

RD (Batch Read)

RD Request:

This command reads up to 99 consecutive 16-bit items from the OIT's 'LW' memory area. The command is always 14 bytes long.

| Byte 1 | Bytes 2,3 | Bytes 4, 5 | Bytes 6-9 | Bytes 10, 11 | Byte 12 | Bytes 13, 14 |
|--------|------------|------------|------------|-----------------|---------|--------------|
| <STX> | Station | RD | Addr. | Number of items | <ETX> | Checksum |
| 1 Byte | 2 Bytes | 2 Bytes | 4 Bytes | 2 Bytes | 1 Byte | 2 Bytes |
| HEX | ASCII-Hex# | ASCII | ASCII-Dec# | ASCII-Dec# | HEX | ASCII-Hex# |

Byte 1: Always STX (1 byte hexadecimal value 0x02)

Bytes 2, 3: The Station Number of the OIT to read (two ASCII characters representing a HEX value)

Bytes 4, 5: The command to execute (two ASCII characters 'RD' = 52H,44H)

Bytes 6-9: This is the starting address to read from. Must be 4 bytes long, (4 ASCII characters representing a decimal value)

Bytes 10, 11: This the number of addresses to read, up to 99. Must be 2 bytes long. (2 ASCII characters representing a decimal value)

Byte 12: Always ETX (1 byte hexadecimal value: 0x03)

Bytes 13, 14: The checksum is the lowest 8 bits (1 byte) of the sum of bytes 2 through 12. This checksum result is entered onto the end of the packet as (2) ASCII characters representing a 1 byte hexadecimal value.

Batch Read (RD) Request Example: Read three words starting from address LW100, from the OIT at station 10 (0AH). This will read addresses LW100 – LW102.

| Byte 1 | Bytes 2,3 | Bytes 4, 5 | Bytes 6-9 | Bytes 10, 11 | Byte 12 | Bytes 13, 14 |
|--------|-----------|------------|-------------|--------------|---------|--------------|
| <STX> | 0A | 'RD' | 0100 | 03 | <ETX> | 2E |
| 02 | 30,41 | 52,44 | 30,31,30,30 | 30,33 | 03 | 32,45 |

The checksum (bytes 13 and 14) is calculated as the lowest 8 bits (2 hex digits) of the sum of the hexadecimal values for bytes 2 – 12. The result is then entered as two ASCII characters that represent the last two hexadecimal digits.

$$30 + 41 + 52 + 44 + 30 + 31 + 30 + 30 + 30 + 33 + 03 = 22E.$$

The lowest 8 bits of the result returns 2E. So the Checksum entered is ASCII '2' & 'E' or 32 & 45.

RD Reply:

The reply length is: $L = (N * 4) + 8$

Where: N = the number of requested registers

If the batch read command is successful, the reply length will be at least 12 bytes, but could be as long as 404 bytes. It consists of the STX, followed by four bytes for each requested device, then the ETX and Checksum.

| Byte 1 | Bytes 2, 3 | Bytes 4,5 | Bytes 6-9 | Bytes 10-13 | Bytes 14-17 | Bytes 18 - (L-7) | Bytes (L-6) - (L-3) | Byte L-2 | Byte L-1, L |
|--------|------------|-----------|-----------|-------------|-------------|--------------------|---------------------|----------|-------------|
| <STX> | Station | CMD | Data 1 | Data 2 | Data 3 | Data 4 – Data (-1) | Data N | <ETX> | Checksum |

The above example returns the following, assuming the OIT contains the following data:

| Address | Data |
|---------|------------------|
| 100 | 75 (4B Hex) |
| 101 | 8047 (1F6F Hex) |
| 102 | 16,321 (3FC1Hex) |

The following is the reply packet sent from the OIT:

| | | | | | | | | | |
|-------|-----|-----|-----|-----|-----|-----|-------|-----|-----|
| <STX> | '0' | 'A' | 'R' | 'D' | '0' | '0' | '4' | 'B' | '1' |
| 02H | 30H | 41H | 52H | 44H | 30H | 30H | 34H | 42H | 31H |
| 'F' | '6' | 'F' | '3' | 'F' | 'C' | '1' | <ETX> | 'C' | '1' |
| 46H | 36H | 48H | 33H | 46H | 43H | 31H | 02H | 43H | 31H |

The data values in each requested register are returned in an ASCII representation of the hexadecimal value. The checksum is calculated on bytes 2 – (L-2).

Reply (ERROR):

In the event of an error, the reply is:

| Byte 1 | Byte 2,3 | Byte 4,5 | Byte 6 |
|--------|----------|----------|-----------------------|
| <NAK> | Station | 'R', 'D' | Err Code [#] |

- See appendix A for a list of error codes.

WD (Batch Write)

WD Request

This command writes up to 99 consecutive 16-bit items to the OIT's LW memory area. The length of the command is:

$$L = (N * 4) + 14$$

Where N = the number of registers to write. The command will be at least 18 bytes long, but can be up to 410 bytes long.

| Byte 1 | Bytes 2, 3 | Bytes 4,5 | Bytes 6-9 | Bytes 10-11 | Bytes 12-15 | Bytes 16-19 | Bytes 20 - (L-7) | Bytes (L-6) - (L-3) | Byte L-2 | Byte L-1, L |
|-----------|---------------|--------------|--------------|-----------------|----------------|----------------|-----------------------|---------------------------|-------------|----------------|
| <STX> | Station | 'WD' | Addr. | No. of items | Data 1 | Data 2 | Data 3 – Data (-1) | Data N | <ETX> | Checksum |

Byte 1: Always <STX> (1 byte hexadecimal value 0x02)

Bytes 2, 3: The Station Number of the OIT to write (2 characters of 2 hex digits)

Bytes 4, 5: The command to execute ('WD' or 57H,44H)

Bytes 4-7: This is the starting address (Must be characters of 4 decimal digits).

Bytes 8, 9: This is the number of registers to write to (Must be characters of 2 decimal digits).

Bytes 10 – (L-3): The data to write. Up to 99 items, each data item takes four bytes and they are characters of the hexadecimal value.

Byte (L-2): Always <ETX> (1 byte hexadecimal value 0x03).

Bytes L-1, L: Checksum. The checksum is calculated on bytes 2 – (L-2). (Characters of 2 hex digits)

Batch Write (WD) Request Example: Write 3 words starting from address LW201, to the OIT at station 17 (11H). This will write to addresses LW201, LW202, and LW203.

LW201 = 101 (0x65)

LW202 = 575 (0x23F)

LW203 = 1049 (0x419)

| Byte 1 | Bytes 2, 3 | Bytes 4,5 | Bytes 6-9 | Bytes 10-11 | Bytes 12-15 | Bytes 16-19 | Bytes 20 - 23 | Byte 24 | Byte 25, 26 |
|-----------|---------------|--------------|-----------------|----------------|----------------|----------------|------------------|---------|----------------|
| <STX> | Station | WD | 0201 | 03 | 0065 | 023F | 0419 | <ETX> | 9A |
| 02 | 31,31 | 57,44 | 30,32, 30,31 | 30,33 | 30,30,36,35 | 30,32,33,46 | 30,34,31, 39 | 03 | 39,41 |

The checksum (bytes 25 and 26) is calculated as the lowest 8 bits of the sum of the Hex codes for bytes 2 – 24.

$31 + 31 + 57 + 44 + 30 + 32 + 30 + 31 + 30 + 33 + 30 + 30 + 36 + 35 + 30 + 32 + 33 + 46 + 30 + 34 + 31 + 39 + 03 = 49A$.

The lowest 8 bits of the result returns **9A**.

WD Reply

If the command is successful, the reply is:

| Byte 1 | Byte 2,3 | Byte 4,5 |
|--------|----------|----------|
| <ACK> | Station | 'W', 'D' |

In the event of an error, the reply is:

| Byte 1 | Byte 2,3 | Byte 4,5 | Byte 6 |
|--------|----------|----------|----------|
| <NAK> | Station | 'W', 'D' | Err Code |

RR (Random Read)

RR Request

This command reads up to 99 independently-addressed 16-bit items from the OIT's LW memory area. The length of the command is

$$L = (N * 4) + 8$$

Where N = the number of requested devices

The command will be at least 12 bytes long, but can be up to 404 bytes long.

| Byte 1 | Bytes 2,3 | Bytes 4,5 | Bytes 6-9 | Bytes 10-13 | Bytes 14 - (L-7) | Bytes (L-6) - (L-3) | Byte L-2 | Byte L-1, L |
|--------|-----------|-----------|-----------|-------------|-----------------------|---------------------|----------|-------------|
| <STX> | Station | 'RR' | Addr 1 | Addr 2 | Addr 3 – Addr (-1) | Addr N | <ETX> | Checksum |

Byte 1: Always <STX> (hexadecimal value 0x02)

Bytes 2, 3: The Station Number of the OIT to read (2 chars of 2 hex digits)

Bytes 4, 5: The command to execute ('RR' or 52H, 52H)

Bytes 6-9: This is the first address from which to retrieve data. (4 chars of decimal value)

Bytes 10, 13: This is the second address from which to retrieve data. (4 chars of decimal value)

Phone: 425/745-3229 · Fax: 425/745-3429 · E-mail: maple@maple-systems.com · URL: www.maple-systems.com

Bytes 14 – (L-7): The remaining addresses from which to retrieve data. Each address must be 4 chars of decimal values.

Byte (L-2): Always ETX (hex value 0x03).

Bytes L-1, L: Checksum, calculated as the lower 8 bits(2 hex digits) of the sum of bytes 2 – (L-2). (2 ASCII characters of hexadecimal result)

RR Reply

If successful, the reply length is

$$L = (N * 4) + 10$$

Where N = the number of requested devices

If successful, the reply length will be at least 12 bytes, but can be up to 406 bytes. It consists of the STX, followed by four bytes for each requested device, then the ETX and Checksum.

| Byte 1 | Bytes 2,3 | Bytes 4,5 | Bytes 6-9 | Bytes 10-13 | Bytes 14-17 | Bytes 15 - (L-7) | Bytes (L-6) - (L-3) | Byte L-2 | Byte L-1, L |
|--------|-----------|-----------|-----------|-------------|-------------|-----------------------------|---------------------|----------|-------------|
| <STX> | Station | CMD | Data 1 | Data 2 | Data 3 | Data 4 – Data (-1) | Data N | <ETX> | Check-sum |

The values in each requested device are returned in Hex. The checksum is calculated as the lower 8 bits of the sum of bytes 2 – (L-2).

In the event of an error, the reply is:

| Byte 1 | Byte 2,3 | Byte 4,5 | Byte 6 |
|--------|----------|----------|----------|
| <NAK> | Station | ‘R’, ‘R’ | Err Code |

RW (Random Write)

RW Request

This command writes up to 99 independently-addressed 16-bit items to the OIT's LW memory area. The length of the command is: $L = (N * 8) + 8$

Where N = the number of requested devices

The command will be at least 16 bytes long, but can be up to 800 bytes long.

| Byte 1 | Bytes 2,3 | Bytes 4,5 | Bytes 6-9 | Bytes 10-13 | Bytes 14-17 |
|--------|-----------|-----------|-----------|-------------|-------------|
| <STX> | Station | 'RW' | Addr 1 | Data 1 | Addr 2 |

| Bytes 18-21 | Bytes (L-10) - (L-7) | Bytes (L-6) - (L-3) | Byte L-2 | Byte L-1, L |
|-------------|----------------------|---------------------|----------|-------------|
| Data 2 | Addr N | Data N | <ETX> | checksum |

Byte 1: Always <STX> (hex value 0x02)

Bytes 2, 3: The Station Number of the OIT to read (2 chars of hexadecimal value)

Bytes 4, 5: The command to execute ('RW' or 52H,57H)

Bytes 6-9: This is the first address to write data to. (4 chars of decimal value)

Bytes 10-13: This is the data to write to the address specified by the previous 4 bytes. (4 chars of hexadecimal values)

Bytes 14 – (L-3): The remaining addresses and data to write to the OIT. Each address and data item must be 4 bytes long with the address as 4 chars of a decimal value and the data as 4 chars of hexadecimal value.

Byte (L-2): Always <ETX> (hex value 0x03).

Bytes L-1, L: Checksum, calculated as the lower 8 bits of the sum of bytes 2 – (L-2). (2 chars of hex value)

RW Reply

If the command is successful, the reply is:

| Byte 1 | Byte 2,3 | Byte 4,5 |
|--------|----------|----------|
| <ACK> | Station | 'R', 'W' |

In the event of an error, the reply is:

| Byte 1 | Byte 2,3 | Byte 4,5 | Byte 6 |
|--------|----------|----------|----------|
| <NAK> | Station | 'R', 'W' | Err Code |

RC (Read Coils)

RC Request

This command reads up to 99 consecutive coils from the OIT's 'LB' memory area. The command is always 14 bytes long.

| Byte 1 | Bytes 2,3 | Bytes 4,5 | Bytes 6-9 | Bytes 10, 11 | Byte 12 | Bytes 13, 14 |
|--------|-----------|-----------|-----------|--------------|---------|--------------|
| <STX> | Station | 'RC' | Addr. | No. of items | <ETX> | Checksum |

Byte 1: Always <STX> (hex value 0x02)

Bytes 2, 3: The Station Number of the OIT to read (2 chars of hex value)

Bytes 4, 5: The command to execute ('RC' = 52H,43H)

Bytes 6-9: This is the starting address to read from. (4 chars of decimal value)

Bytes 10, 11: This is the number of coils to read, up to 99. (2 chars of decimal value).

Byte 12: Always <ETX> (0x03)

Bytes 13, 14: The checksum is the lowest 8 bits of the sum of bytes 2 through 12.

Example: Read 12 coils starting from address LB100, from the OIT at Station 7. This will read coils LB100 – LB111.

| Byte 1 | Bytes 2,3 | Bytes 4,5 | Bytes 6-9 | Bytes 10, 11 | Byte 12 | Bytes 13, 14 |
|--------|-----------|-----------|-------------|--------------|---------|--------------|
| <STX> | 07 | 'RC' | 0100 | 12 | <ETX> | 22 |
| 02 | 30,37 | 52,43 | 30,31,30,30 | 30,32 | 03 | 32,32 |

The checksum (bytes 13 and 14) is calculated as the lowest 8 bits of the sum of the Hex codes for bytes 2 – 12.

$$30 + 37 + 52 + 43 + 30 + 31 + 30 + 30 + 30 + 32 + 03 = 222.$$

The lowest 8 bits of the result returns 22.

RC Reply

The reply length is: $L = N + 8$

Where N = the number of requested devices

If the command is successful, the reply length will be at least 9 bytes, but could be as long as 107 bytes. It consists of the STX, followed by one byte for each requested device, then the ETX and Checksum.

| Byte 1 | Bytes 2,3 | Bytes 4,5 | Byte 6 | Byte 7 | Byte 8 | Bytes 9 - (L-4) | Byte (L-3) | Byte L-2 | Byte L-1, L |
|--------|-----------|-----------|--------|--------|--------|--------------------|------------|----------|-------------|
| <STX> | Station | RC | Data 1 | Data 2 | Data 3 | Data 4 – Data (-1) | Data N | <ETX> | Checksum |

If the OIT contains the following data:

| | | | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 100 | 101 | 102 | 103 | 104 | 105 | 106 | 107 | 108 | 109 | 110 | 111 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |

Then, the following data is returned:

| | | | | | | | | | | | | | | | |
|-------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| <STX> | '0' | '7' | 'R' | 'C' | '0' | '0' | '1' | '0' | '1' | '0' | '1' | '1' | '0' | '0' | '0' |
| 02H | 30H | 37H | 52H | 43H | 30H | 30H | 31H | 30H | 31H | 30H | 31H | 31H | 30H | 30H | 30H |

| | | | |
|-----|-------|-----|-----|
| '1' | <ETX> | '4' | '4' |
| 31H | 03H | 34H | 34H |

The values in each requested device are returned in an ASCII representation of the hexadecimal result. The checksum is calculated on bytes 2 – (L-2).

In the event of an error, the reply is:

| Byte 1 | Byte 2,3 | Byte 4,5 | Byte 6 |
|--------|----------|----------|----------|
| <NAK> | Station | 'R', 'C' | Err Code |

WC (Write Coils)

WC Request

This command writes up to 99 consecutive coils to the OIT's 'LB' memory area. The length of the command is: $L = N + 14$

Where N = the number of requested devices

The command will be at least 15 bytes long, but can be up to 113 bytes long.

| Byte 1 | Bytes 2,3 | Bytes 4,5 | Bytes 6-9 | Bytes 10-11 | Byte 12 | Byte 13 | Bytes 14 - (L-4) | Byte (L-3) | Byte L-2 | Byte L-1, L |
|--------|-----------|-----------|-----------|--------------|---------|---------|--------------------|------------|----------|-------------|
| <STX> | Station | WC | Addr. | No. of items | Data 1 | Data 2 | Data 3 – Data (-1) | Data N | <ETX> | Check-sum |

Byte 1: Always STX (hex value 0x02)

Bytes 2, 3: The Station Number of the OIT to read (2 chars of hexadecimal value)

Bytes 4, 5: The command to execute ('WC' = 57H, 43H)

Bytes 6-9: This is the starting address to write to. (4 chars of decimal value)

Bytes 10, 11: This the number of addresses to write. (2 chars of decimal value)

Bytes 12 – (L-3): The data to write. Up to 99 items, (1 char of a binary digit [0,1]for each address.

Byte (L-2): Always ETX (hex value 0x03).

Bytes L-1, L: Checksum (2 char of hexadecimal value)

Example: Write 5 bits starting from address LB214 to the OIT at station 12. This will write to addresses LB214 – LB218.

Write the following data:

| | | | | |
|-----|-----|-----|-----|-----|
| 214 | 215 | 216 | 217 | 218 |
| 1 | 1 | 0 | 0 | 1 |

| Byte 1 | Bytes 2,3 | Bytes 4,5 | Bytes 6-9 | Bytes 10-11 | Byte 12 | Byte 13 | Byte 14 | Byte 15 | Byte 16 | Byte 17 | Byte 18,19 |
|--------|-----------|-----------|-------------|-------------|---------|---------|---------|---------|---------|---------|------------|
| STX | 0C | WC | 0214 | 05 | 1 | 1 | 0 | 0 | 1 | ETX | |
| 02 | 30,43 | 57,43 | 30,32,31,34 | 30,35 | 31 | 31 | 30 | 30 | 31 | 03 | |

The checksum (bytes 18 and 19) is calculated as the lowest 8 bits of the sum of the Hex codes for bytes 2 – 17.

$$30 + 43 + 57 + 43 + 30 + 32 + 31 + 34 + 30 + 35 + 31 + 31 + 30 + 30 + 31 + 03 = 32F.$$

The lowest 8 bits of the result returns 2F.

WC Reply

If the command is successful, the reply is:

| Byte 1 | Bytes 2,3 | Bytes 4,5 |
|--------|-----------|-----------|
| <ACK> | Station | 'W', 'C' |

In the event of an error, the reply is:

| Byte 1 | Bytes 2,3 | Bytes 4,5 | Byte 6 |
|--------|-----------|-----------|-----------|
| <NAK> | Station | 'W', 'C' | Err. Code |

Appendix A - Error Codes

The following table lists the error conditions, and the Error Codes returned for those errors.

| Code | Description |
|-------------|--|
| 06H | Invalid Checksum |
| 10H | Unknown Command |
| 11H | Data Length Error – data overflowed receive buffer |
| 12H | Communication Data Error – ETX not found |
| 7AH | Illegal Address |
| 7BH | More than 99 data items were requested |

Appendix B - Programming Example

The following BASIC code is typical of what might be found in a controller that uses ASCII commands for communication.

Please note that your controller may have different commands or syntax than what is shown below. Some of the variable names may be keywords in your controller, and will have to be changed for your application.

Using the RD command

This sample shows a simple example of how to program for ASCII communications with the OIT. The first routine shows how to construct the string to send the 'RD' command to the OIT. The second routine shows how to process the data returned from the OIT.

' RD Example - send command to the OIT

```
Dim AS$
Dim I
Dim CS
Dim STX$
Dim Cmd$
Dim StationID$
Dim Addr$
Dim Size$
Dim ETX$

STX$ = Chr$(2) ' ASCII "STX" character
StationID$ = "0A" ' number 10 in hexadecimal (must be 2 characters)
Cmd$ = "RD" ' RD command
Addr$ = "0100" ' LW word to start reading from (must be 4 characters)
Size$ = "03" ' number of items to read (must be 2 characters)
ETX$ = Chr$(3) ' ASCII "ETX" character

' assemble the packet
A$ = STX$ + StationID$ + Cmd$ + Addr$ + Size$ + ETX$

' note that this command could also be built like this:
' A$ = Chr$(2) + "0ARD010003" + Chr$(3)

' get the checksum
' The checksum is lowest 8 bits of the arithmetic sum of the Ascii codes of the second through the last characters in the string.
For I = 2 To 12 ' there are always 12 bytes for the RD command
  CS = CS + Asc(Mid$(A$, I, 1)) ' first, add up the characters
Next I

CS = CS And 255 ' now get the lowest 8 bits
A$ = A$ + Chr$(CS) ' append the checksum to the packet

' put here the code to send A$ out the port

End
```

' RD Example - process the data received from the OIT

```
Dim A$
Dim CS$
Dim Value$
Dim Values(3)
Dim I, CS1, CS2, DB, C

' put here the code to get the contents
' of the port and store it in A$

' check length
If Len(A$) < 12 ' valid reply will be at least 12 bytes long
  ' take required steps
End If

' check for STX...
If Left$(A$, 1) <> Chr$(2) Then ' second byte should be STX
  ' Take required steps
End If

' check for ETX...
If Mid$(A$, Len(A$) - 3, 1) <> Chr$(3) Then ' third -to-last byte should be ETX
  ' Take required steps
End If

' check the checksum, always the last 2 bytes of the packet
CS$ = Right$(A$, 2) ' checksum as a string
CS1 = Val("&H" & CS) ' checksum as a numeric value

' calculate what the checksum should be
For I = 2 To Len(A$)
  CS2 = CS2 + Asc(Mid$(A$, I, 1))
Next I

CS2 = CS2 And 255 ' get the lowest 8 bits

' compare the 2 checksums
If CS1 = CS2 Then ' they are equal, so parse out the data

  ' get the number of data bytes
  DB = Len(A$) - 8 ' there are 8 overhead bytes in the packet
  C = 1 ' first location in the Values array

  ' get each set of 4 bytes, and extract the values
  For I = 1 To DB Step 4

    ' values are in Hexadecimal format, and start at the 6th byte
    Values(C) = Val("&H" + Mid$(A$, I+5, 4))

    C = C + 1 ' next element in the Values array
  Next I

Else ' checksums are not equal
  ' Take necessary steps
End If
End
```