VARECMODEL 1900
MICRO 4-WIRE
TRANSMITTER

(BSAP to Mark / Space Protocol (MSP))

APPLICATION NOTES

1. Tables

1.1 Table of Contents

1. TABLES III

1.1 TABLE OF CONTENTS III
1.2 TABLE OF FIGURES VI

2. INTRODUCTION 1

2.1 CONFIGURATION 1
2.2 MSP CHARACTERISTICS 2
2.2.1 MSP Bit Timing 3
2.2.1 3
2.2.2 MSP Message Timing 4
2.2.2 4
2.2.3 MSP Message Structure 4
2.2.3 4
2.2.3.1 MSP Level Response Message 6
2.2.3.1 6
2.2.3.2 MSP Temperature Response Message 8
2.2.3.2 8
2.2.3.3 MSP Level Data Format 10
2.2.3.3 10

3. FUNCTIONAL DESIGN 16

3.1 DESIGN GOALS 16
3.1 16
3.2 DESIGN OVERVIEW 16
3.2 16
3.3 BSAP INTERFACE PROCESS 17
3.3.1 Initialize Parameters 17
3.3.1 17
3.3.2 Write Initial Values 19
3.3.2 19
3.3.3 Read Current Values 19
3.3.3 19
3.3.3.1 Pulse Timing Errors 20
3.3.3.1 20
3.3.3.2 Communication Errors 21
3.3.3.2 21
3.3.3.3 Level Data Errors 22
3.3.3.3 22

3.3.3.4 Temperature Data Errors 22
3.3.3.4 22
3.4 MSP INTERFACE PROCESS 23
3.4 23
3.4.1 MSP Polling 23
3.4.1 23
3.4.2 MSP Reading Responses 24
3.4.2 24
3.4.2.1 Clocking in Marks and Spaces 24
3.4.2.2 Checking the Response for Errors 24

4. ACCOL PROGRAMMING 26

4. 26

4.1 OVERVIEW 26
4.2 COMMUNICATION 26
4.2.1 Ports 26
4.2.1 26
4.2.2 Baud Rate 26
4.2.2 26
4.2.3 Slave Address 26
4.2.3 26
4.3 SLAVE MODULES 26
4.3 26
4.3.1 Common Terminals 27
4.3.1.1 Remote (1 to 127) 27
4.3.1.1 27
4.3.1.2 Index (0) 27
4.3.1.2 27
4.3.1.3 Status 1 27
4.3.1.3 27
4.3.1.4 Status 2 27
4.3.1.4 27
4.3.2 Initialize Parameters—Slave Module 60 28
4.3.2 28
4.3.2.1 Point (60) 28
4.3.2.1 28
4.3.2.2 Mode (0) 28
4.3.2.2 28
4.3.2.3 Intype (?) 28
4.3.2.3 28
4.3.2.4 Outtype (0) 28
4.3.2.4 28
4.3.2.5 Inlist (?) 28

4.3.2.5 28
4.3.2.6 Outlist (1 to 255) 28
4.3.2.6 28
4.3.3 Write Initial Values—Slave Module 61 29
4.3.3 29
4.3.3.1 Point (61) 30
4.3.3.1 30
4.3.3.2 Mode (0) 30
4.3.3.2 30
4.3.3.3 Intype (?) 30
4.3.3.3 30
4.3.3.4 Outtype (0) 30
4.3.3.4 30
4.3.3.5 Inlist (?) 30
4.3.3.5 30
4.3.3.6 Outlist (1 to 255) 30
4.3.3.6 30
4.3.4 Read Current Values—Slave Module 62 31
4.3.4 31
4.3.4.1 Point (62) 31
4.3.4.1 31
4.3.4.2 Mode (1) 32
4.3.4.2 32
4.3.4.3 Intype (0) 32
4.3.4.3 32
4.3.4.4 Outtype (?) 32
4.3.4.4 32
4.3.4.5 Inlist (1 to 255) 32
4.3.4.5 32
4.3.4.6 Outlist (?) 32
4.3.4.6 32
4.3.5 Status 2 Values 32
4.3.5 32

Table of Tables

TABLE 1-MSP POLL MESSAGE 5
TABLE 2-LEVEL RESPONSE MESSAGE 6
TABLE 3-MSP TEMPERATURE RESPONSE BITS 8
TABLE 4-EXAMPLE OF TEMPERATURE DATA 9
TABLE 5-EXAMPLE OF REFLECTED BINARY GRAY CODE 10
TABLE 6-INITIALIZATION PARAMETER DATA 18
TABLE 7-INITIAL VALUE DATA 19

TABLE 8-QUALITY STATUS 20
TABLE 9-PULSE TIMING ERRORS 21
TABLE 10-COMMUNICATION ERRORS 21
TABLE 11-LEVEL DATA ERRORS 22
TABLE 12-LEVEL RANGES 22
TABLE 13-TEMPERATURE DATA ERRORS 22
TABLE 14-TEMPERATURE SENSOR RANGES 23
TABLE 15-MASTER MODULE TERMINALS 27
TABLE 16-PARAMETER INITIALIZATION LIST 29
TABLE 17-WRITE INITIAL VALUES LIST 31
TABLE 18-STATUS 2 ERROR CODES 33

1.2 Table of Figures

FIGURE 1-VAREC 1900 TRANSMITTER CONFIGURATION 1
FIGURE 2-MARK/SPACE BIT TIMING 3
FIGURE 3-MSP MESSAGE TIMING 4
FIGURE 4-MSP POLL MESSAGE 5
FIGURE 5-LEVEL RESPONSE MESSAGE 7
FIGURE 6-MSP TEMPERATURE RESPONSE BITS 9
FIGURE 7-LEVEL DATA FORMATS 11
FIGURE 8-DECIMAL FEET CODING CHART 12
FIGURE 9-FRACTIONAL FEET CODING CHART 13
FIGURE 10-METRIC CODING CHART 14

2. Introduction

This document is intended to serve as a User's Guide to the BSAP to Varec Model 1900 Mark/Space Protocol Converter (the Mark/Space Protocol will be referred to in this document as **MSP**). The protocol converter will be referred to in this document as the **MO33**.

This document assumes familiarity with the Varec Model 1900 Micro 4-Wire Transmitter and its related terminology, as well as user access to related hardware reference manuals.

2.1 Configuration

Figure 1 illustrates the configuration required to interface with the Varec Model 1900 Micro 4-Wire Transmitter (referred to in the remainder of the document as Model 1900):

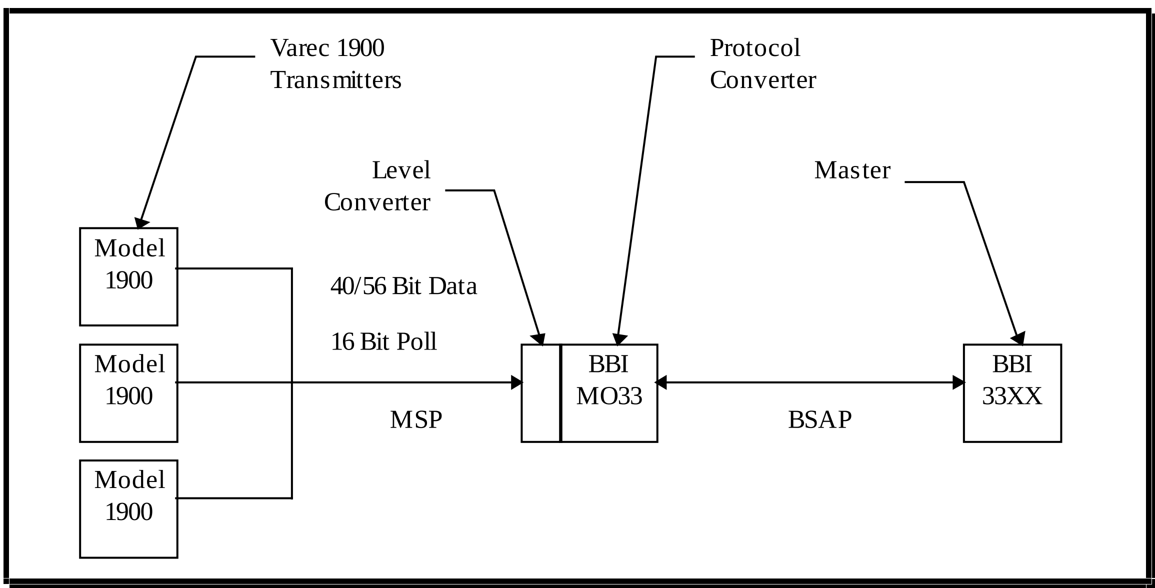


Figure 1-Varec 1900 Transmitter Configuration

The interface between the Model 1900 transmitters and the MO33 consists of 4-wires identified as B+, B-, Mark and Space. A **Level Converter** board is required to <u>convert the 48 VDC Model 1900 levels to the TTL levels</u> available on the MO33 card.

The Operations Manual for the Model 1900 states that the maximum voltage drop limits will be met by limiting:

- the number of transmitters in each "area" to 30 or less,
- the maximum wiring distance to 10,000 feet or less,
- and the maximum resistance on each of the four lines (B+, B-, Mark and Space) to 50 ohms or less.

With these considerations in mind, the MO33's firmware is sized to communicate with a **maximum** of 30 transmitters.

The MO33 polls each of the transmitters with a 16 bit message that contains the transmitter's unique identifier. Upon being polled, the transmitter responds with a 40 (level only) or 56 (level and temperature) bit message containing the data.

2.2 MSP Characteristics

All communication between the MO33 and the Model 1900 transmitters is accomplished over a common pair of Mark/Space pulse code lines. A pulse is sensed as a low (line drops from +48 VDC to approximately 0 VDC) to indicate the presence of either a Mark or Space. If the Mark line is pulsed low, it is designated a Mark and considered a logic "1." If the Space line is pulsed low, it is designated a Space and considered a logic "0." If both the Mark and Space lines are simultaneously low, it is an error or fault condition.

2.2.1 MSP Bit Timing

Two basic data rates are supported over the Mark/Space interface—low speed and high speed. The poll and response timing is illustrated in Figure 2.

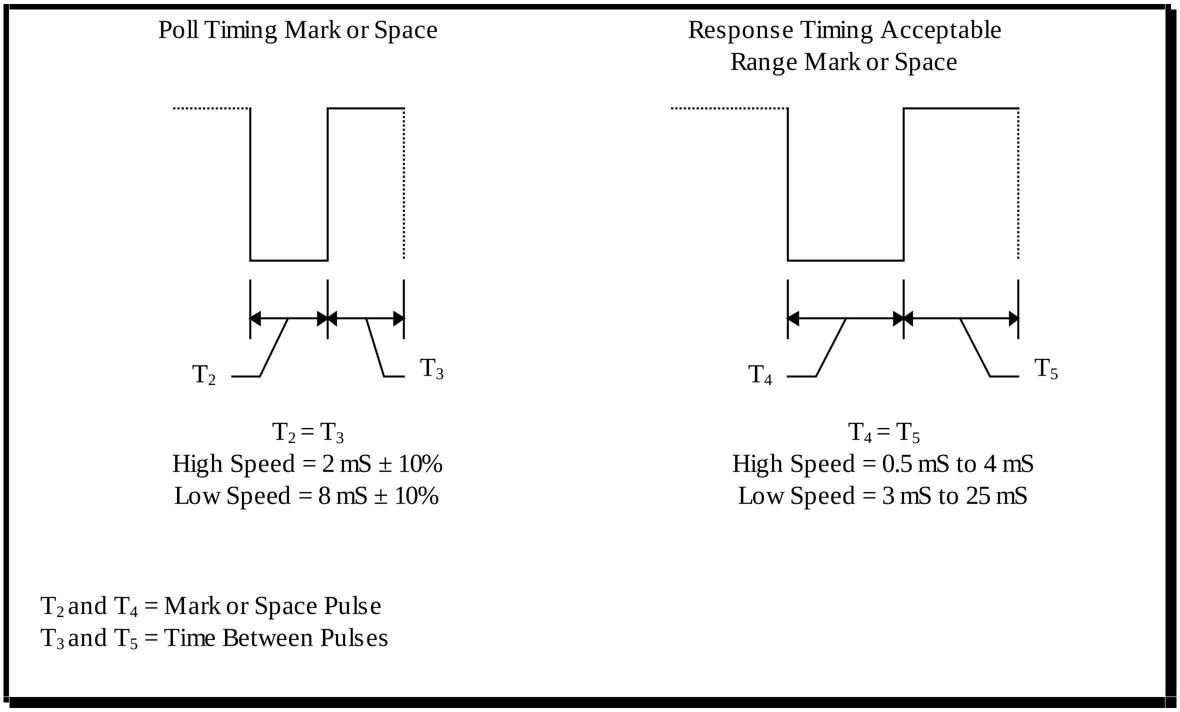


Figure 2-Mark / Space Bit Timing

2.2.2 MSP Message Timing

Overall message timing is illustrated in Figure 3. Note that the delay time between a poll and response is illustrated as time T_1 and the required delay time between a response and the next poll is illustrated as time T_6.

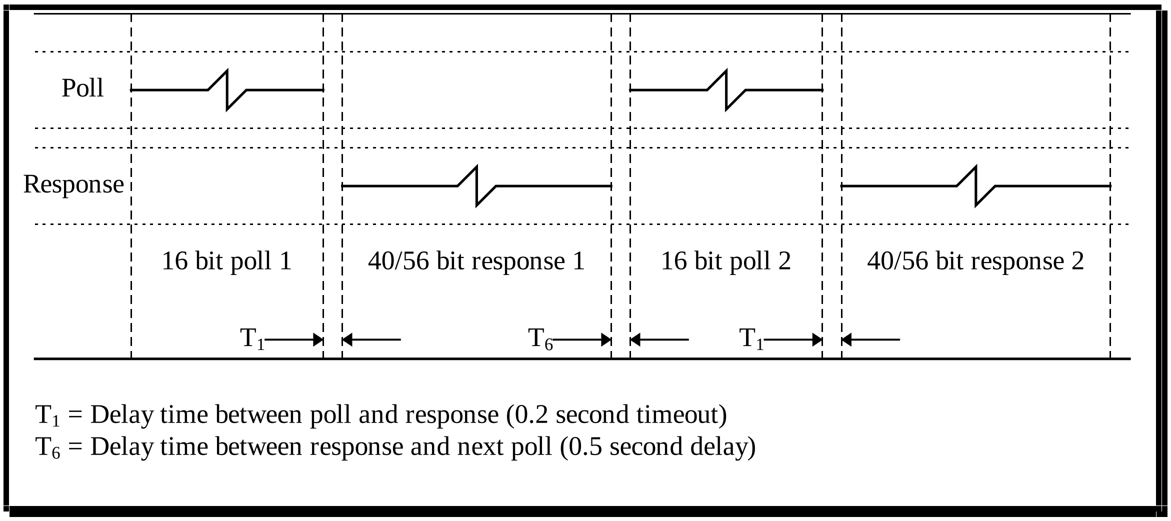


Figure 3-MSP Message Timing

T_1 is not specified due to variations in hardware, however the response time is typically a few hundred milliseconds. The MO33 will allow 0.5 seconds for a response before declaring a time-out. T_6 is the delay before repolling the device and is typically a few hundred milliseconds. With multiple devices on the drop, this should not be a concern. The MO33 will allow 0.5 seconds for this value.

2.2.3 MSP Message Structure

Data transfer between the Model 1900 and the MO33 is a poll/response protocol. The MO33 places a 16 bit poll message on the data lines. This poll message contains the ID of the device from which a response is desired. If the ID in the poll matches the ID of the Model 1900, it responds with a 40 or 56 bit response message.

MSP Poll Message

Table 1 illustrates the poll message as a 16 bit data structure:

Table 1-MSP Poll Message

Bit(s)	*Interpretation*
1	start bit (mark)
2	unused
3	raise command (for 6500 servo gauge)
4	reset command (for 6500 servo gauge)
5 - 8	most significant ID bits (ID x 100)
9 - 12	next most significant ID bits (ID x 10)
13 - 16	least significant ID bits (ID x 1)

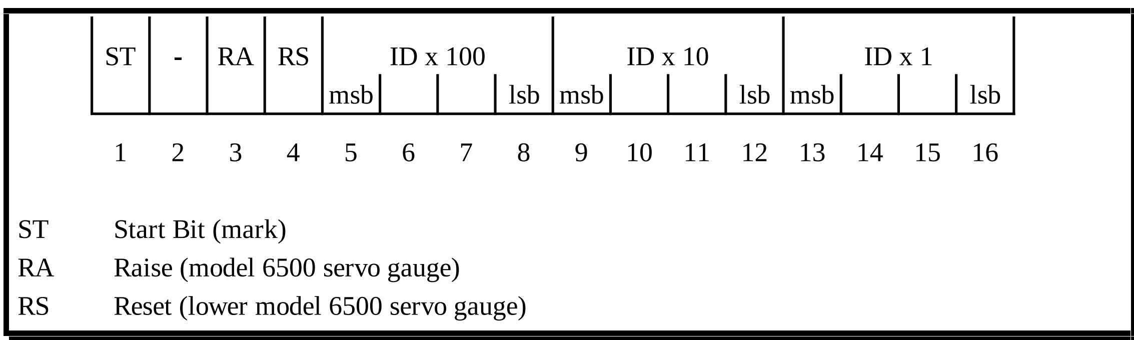


Figure 4-MSP Poll Message

> ***NOTE***: *The Raise and Reset bits are used for the Model 6500 servo gauge. This version of the MO33 firmware does not support this device.*

In Figure 4 the ID bits (5 through 16) represent the address of the Model 1900's. Each Model 1900 is typically in a standby mode while waiting to be polled. After receiving a poll message containing its address, the device "wakes up" and takes a measurement, which is then returned in one of the following response message types. The address consists of three binary coded digits (BCD 000 to 999).

2.2.3.1 MSP Level Response Message

The response message from the Model 1900 can be either 40 or 56 bits long, depending on whether temperature reporting is configured. The basic format of the 40 bit level message is illustrated in Table 2:

Table 2-Level Response Message

Bit(s)	*Interpretation*
1	start bit (mark)
2,3	unused
4-7	most significant ID bits (ID x 100)
8-11	next most significant ID bits (ID x 10)
12-15	least significant ID bits (ID x 1)
16	first data bit (always a space for English[1])
17-37	level data
38	alarm 1 (optional external alarm)
39	alarm 0 (optional external alarm)
40	parity

Bits 4 through 15 contain the BCD address of the device that was polled. The level data can be encoded as decimal feet (15 bits), fractional feet (15 bits) or metric (18 bits). There are 21 bits in the message for level data (bits 17 - 37) of which all unused bits are spaces. Level data is represented as a *Reflected Gray Code*, which is explained in *Section 2.2.3.3*.

[1] According to Varec engineers, this bit is the first data bit for metric level data

NOTE: Bit 17 is interpreted as the first level bit for English data and bit 16 is interpreted as the first level bit for metric data.

ST	--		ID x 100				ID x 10				ID x 1		
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

SP	Level Data	AL1	AL0	PAR
16	17 - 37	38	39	40

ST	Start bit (mark)
ID	Model 4100 address
AL1	External alarm bit 1
AL0	External alarm bit 0
PAR	Parity bit

Figure 5-Level Response Message

The response message provides several opportunities for error detection:

1. **ID** The ID bits should match the polling address
2. **Level** Range checked against the range of the encoder being used.
3. **Parity** The parity bit will be a mark when the sum of the mark pulses in bits 1 through 39 are even. Note that the total number of mark pulses including parity will be odd if there is no error.

2.2.3.2 MSP Temperature Response Message

Bits 1 through 39 of the temperature response message are identical to the level response, i.e. the temperature response also contains level data.

Table 3-MSP Temperature Response Bits

Bit(s)	*Interpretation*
1	start bit (mark)
2,3	unused
4-7	most significant ID bits (ID x 100)
8-11	next most significant ID bits (ID x
12-15	least significant ID bits (ID x 1)
16	first data bit (always a space for
17-37	level data
38	alarm 1 (optional external alarm)
39	alarm 0 (optional external alarm)
40	+100 temp bit
41	sign is positive when in mark state
42	+200 temp bit
43	+400 temp bit
44-47	Temp x 10
48-51	Temp x 1
52-55	Temp x 0.1
56	parity

Referring to Table 3, bits 40 through 55 are unique to the temperature response message. The parity bit has the same meaning as it does in the level response message with the exception of the message size.

Unscaled temperature is contained in bits 44 through 55. Temperature can be measured in degrees Celsius or Fahrenheit. The +100, +200 and +400 bits add to the unscaled temperature when in the mark state.

[2] According to Varec engineers, this bit is the first data bit for metric level data

As an example, a temperature of +256.9 degrees would have bits 40 through 55 as follows:

Table 4-Example of Temperature Data

Bits:	*40*	*41*	*42*	*43*	*44-47*	*48-51*	*52-55*
Data	0	1	1	0	0101	0110	1001
Tem		+	20		5	6	9

Figure 6 is a graphical representation of the temperature response message.

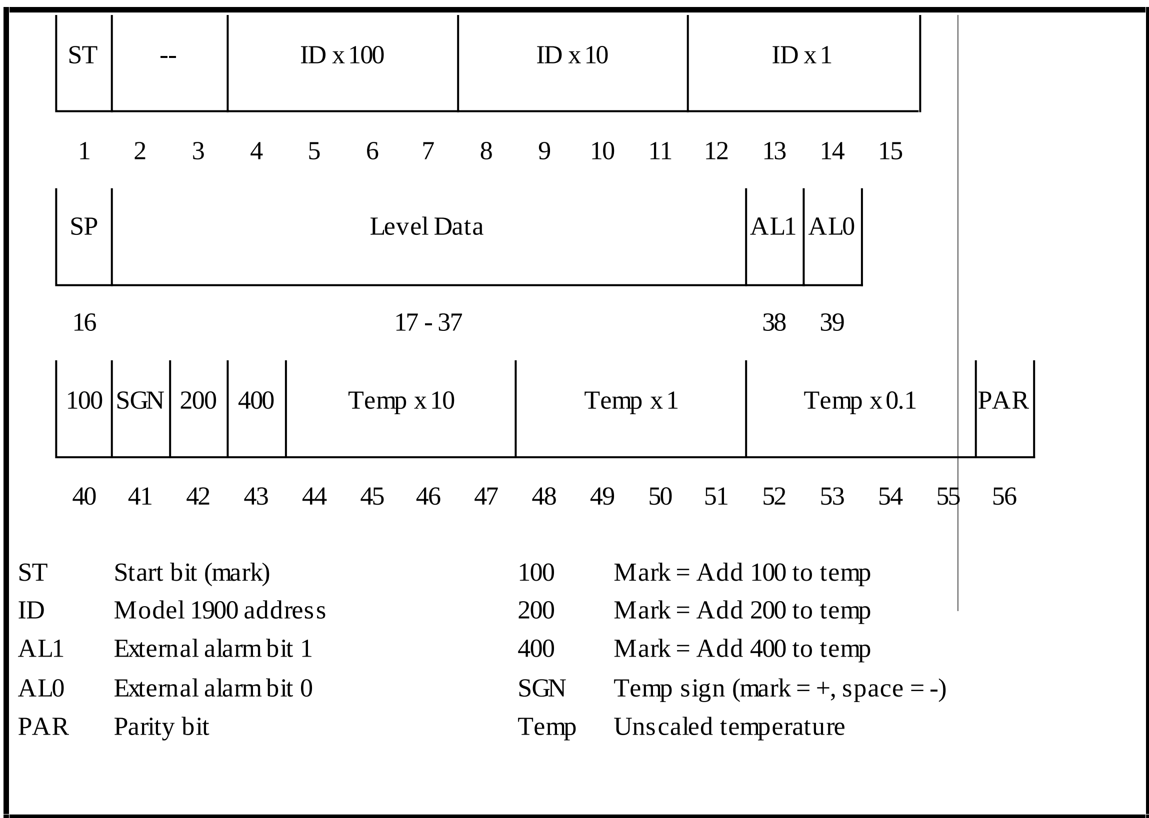


Figure 6-MSP Temperature Response Bits

Error detection in the temperature response message is identical to that of the level response message described in Section 2.2.3.1, with an additional range check added for the temperature data.

2.2.3.3 MSP Level Data Format

Level data in both the level and temperature response messages is encoded in bits 16 through 37. This is contrary to the Varec documentation which illustrates (as in Figure 6) level data as being in bits 17 through 37. Varec engineers have explained that bit 16 is always a space for English transmissions and is the first data bit for metric transmissions. Figure 7 illustrates the bit positions for English and metric formats inside the level data field.

Each of the digits within the level data is encoded using a “reflected binary Gray code.” A characteristic of the Gray code is that for every increment / decrement in level, only one bit change occurs. The absolute meaning of a Gray code digit is dependent on the preceding digit. If the preceding digit (more significant digit) is an even number (where zero is considered even), then the current digit is encoded using a standard Gray code. If the preceding digit is an odd number, the current digit is encoded with the *reflection* of a standard Gray code. Table 5 illustrates how only one bit changes between successive level increments:

Table 5-Example of Reflected Binary Gray Code

29.9	011	100	0000	000**1**
29.9	01**1**	100	0000	000**0**
30.0	01**0**	100	0000	000**0**
30.0	010	100	0000	000**1**

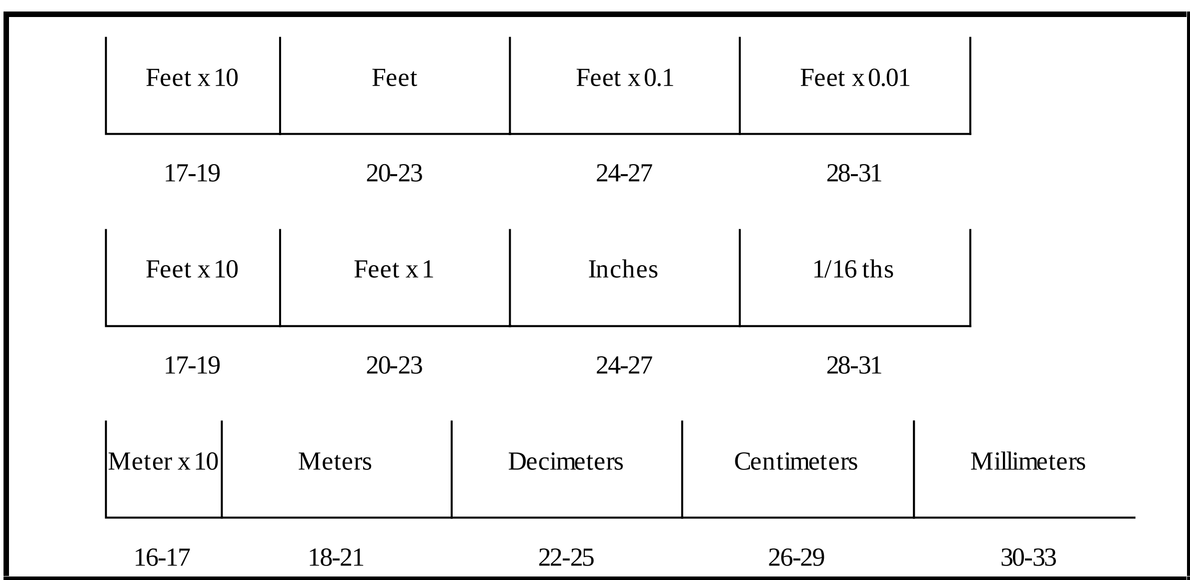
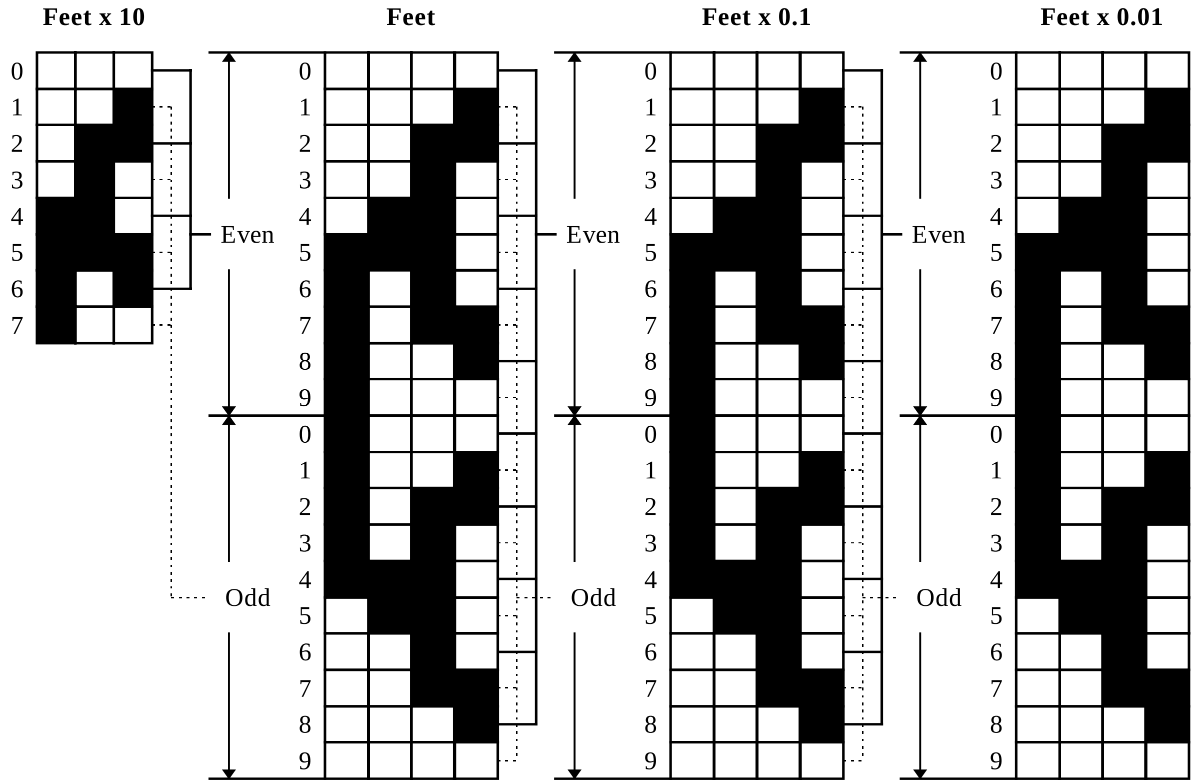


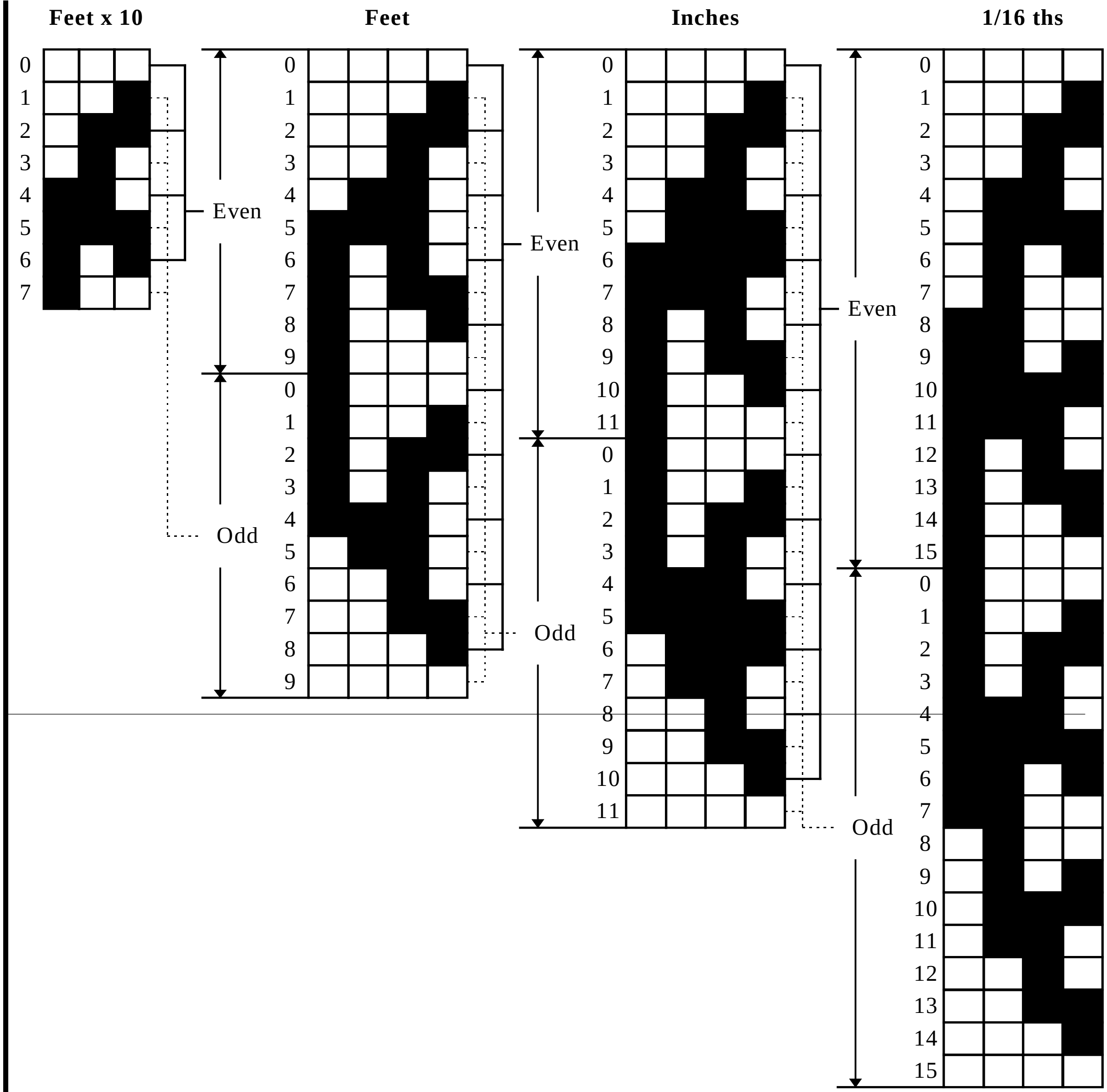
Figure 7-Level Data Formats



The use of ODD or EVEN code depends upon the ODD or EVEN condition of the preceding significant digit.

Shaded (black) area represents Mark (1). Unshaded (white) ares represents Space (0).

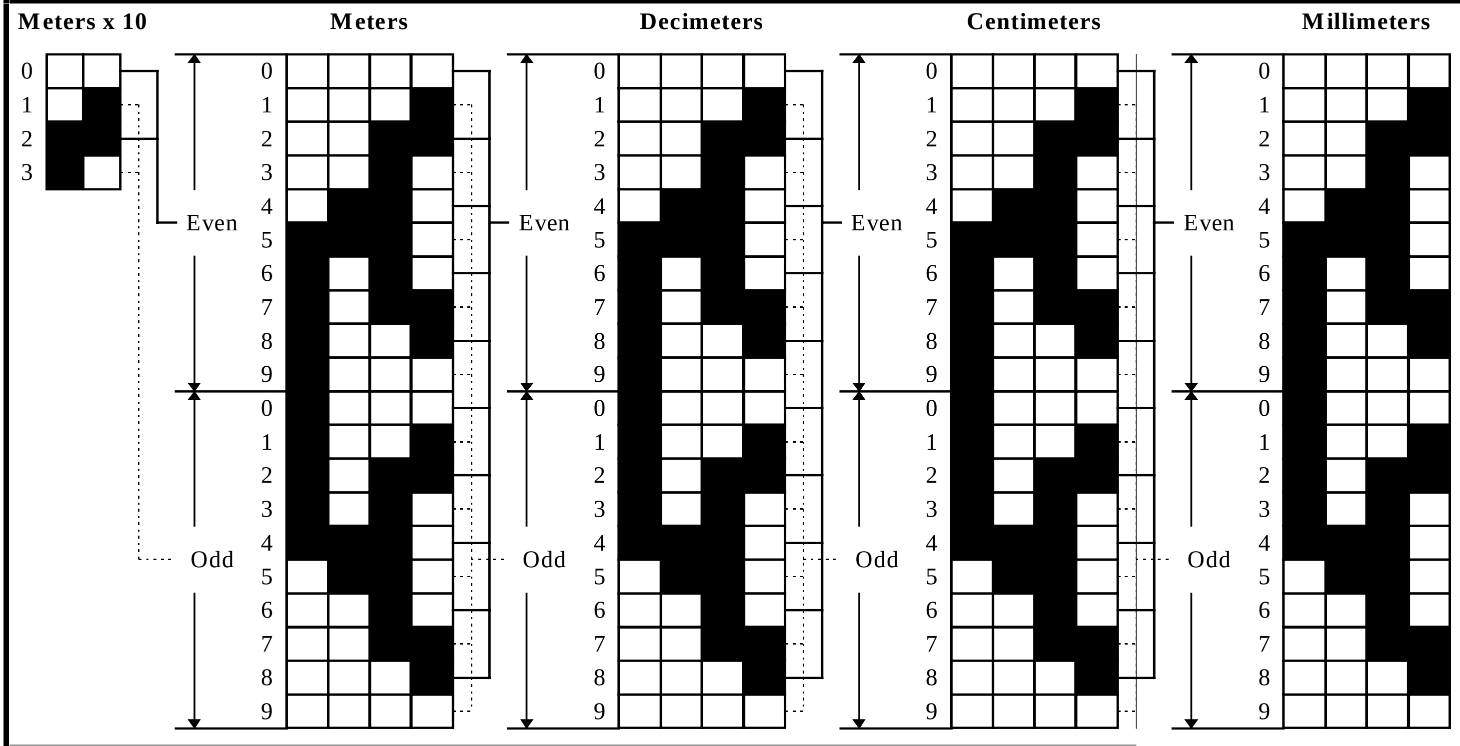
Figure 8-Decimal Feet Coding Chart



The use of ODD or EVEN code depends upon the ODD or EVEN condition of the preceding significant digit.

Shaded (black) area represents Mark (1). Unshaded (white) area represents Space (0).

Figure 9-Fractional Feet Coding Chart



The use of ODD or EVEN code depends upon the ODD or EVEN condition of the preceding significant digit.

Shaded (black) area represents Mark (1). Unshaded (white) area represents Space (0).

Figure 10-Metric Coding Chart

Figure 8, Figure 9 and Figure 10 graphically illustrate how the reflected Gray code works. Note the differences in how the individual digits are coded depending on whether the format is decimal feet, fractional feet or metric.

These figures also illustrate where the reflected Gray code gets its name—the odd half is a mirror image (reflection) of the even half.

> ***NOTE****: If you are also using the Varec documentation, the Fractional Feet Coding Chart (Figure 4-8) in this document has a minor error. In the inches digit on the even 5 row, instead of coding to 1111, the chart should read 0111.*

3. Functional Design

3.1 Design Goals

Ease of programming to the ACCOL programmer was the primary design goal. This has been accomplished by designing the interface to the controller to support BSAP. A BSAP interface enables the ACCOL programmer to use Master modules to communicate with the Model 1900's.

The Model 1900 supports several options:

Encoder: Fractional feet, decimal feet or metric.

Gauge type: Support for 6500 Servo Gauge.

Sensor: Platinum or Copper

Temperature: Fahrenheit[3], centigrade or disabled.

Data rate: High or low speed.

The second design goal was to supply firmware for the MO33 card that supports as many of these options as possible. Where it is not possible to support an option, the interface for that option will be provided with a "Feature Not Implemented Status" returned.

The 6500 Servo Gauge *is not supported* in the first implementation. According to Varec, this servo is rare and is not described in the documentation available.

3.2 Design Overview

Firmware for the MO33 consists of two asynchronous processes:

1. BSAP Interface Process
2. MSP Interface Process

that share a common data area. The BSAP process transmits the data contained in the common data area any time it is polled. Independently, the MSP process continuously polls the Model 1900's and records the returned data into the common data area.

[3] Fahrenheit is not available while using a metric encoder.

The common data area consists of two buffers:

1. **Initialization structure**—encoder, number of transmitters, sensor, temperature, gauge and data rate.
2. **Data buffer**—array of structures containing: Model 1900 address, level data, temperature data, external alarms and status. The array will be sized to support a maximum of 30 Model 1900's.

Separating the data in this manner assumes that all Model 1900's on a single drop use the same encoder type, temperature type, gauge type and data rate. While this requirement places a limitation on what devices can share a drop, it is not unreasonable to expect all devices on a common drop to be the same. At the same time, by separating the data this way, the ACCOL programmer's life should be a little easier since this information will not have to be setup for each device separately (see Section 3.1).

3.3 BSAP Interface Process

The BSAP Interface will appear to the controller as a slave device. Communications will be at a fixed 9600 Baud. The interface will respond to three requests:

1. Initialize Parameters
2. Write Initial Values
3. Read Current Values

The following sections will describe each of these requests in detail.

3.3.1 Initialize Parameters

After a power loss at the MO33, the <u>only</u> command that the MO33 will accept is to Initialize Parameters. If any command other than the Initialize Parameters command is sent, it is ignored by the MO33 and an error is returned in STATUS 2[4] indicating that the MO33 has not been initialized.

[4] STATUS 2 is a terminal in the ACCOL Master Module that is used to communicate with the MO33.

The Initialize Parameters command can be sent at any time, but it wipes out any configuration that may have been set in the MO33 previously and restarts the setup sequence. Parameters to be initialized with this command are illustrated in Table 6:

Table 6-Initialization Parameter Data

Parameter	*Type*	*Description*
Encoder	Analog	0) Fractional feet, 1) decimal feet or 2) metric.
Gauge Type	Analog	Initially not implemented[5] (0 = not used)
Transmitters	Analog	Number of transmitters on this drop (30 maximum)
Metric	Logical	Centigrade if TRUE, else Fahrenheit[6]
Platinum	Logical	Platinum if TRUE, else Copper temperature sensor
Temperature	Logical	Enabled if TRUE
Low Rate	Logical	Low if TRUE, else High speed polling of Model 1900

If an error is reported while initializing parameters, the MO33 will remain in the power-up state waiting to be initialized. While in the power-up state, Model 1900 transmitters are <u>not</u> being polled.

[5] May be implemented later to support 6500 Servo Gauge

[6] Will generate an error in STATUS 2 if metric encoder is selected.

3.3.2 Write Initial Values

After the Initialize Parameters (Section 3.3.1) step has successfully been completed, the only command that the MO33 will accept is to Write Initial Values. The MO33 firmware is sized to communicate with a maximum of 30 Model 1900's. The data to be defined for each Model 1900 is illustrated in Table 7:

Table 7-Initial Value Data

Initial Value	*Type*	*Description*
Address	Analog	Model 1900 polling address
Level	Analog	Initial tank level
Temperature	Analog	Initial tank temperature (ignored if not used)
Quality	Analog	See Table 8-Quality Status
Alarm 0	Logical	State of external alarm switch 0
Alarm 1	Logical	State of external alarm switch 1

Until this step is successfully completed, the MO33 will not begin polling the Model 1900s. An error condition is generated (reported in STATUS 2) if more initial values are sent than were defined in Section 3.3.1, i.e. if Transmitters were defined to be 5 when downloading parameters, the MO33 would expect to receive 5 blocks of initial values (20 analogs and 10 logicals). In the event of an error, the MO33 remains in a state waiting for Initial Values.

3.3.3 Read Current Values

After successfully completing the Write Initial Values (Section 3.3.2) step, the MO33 will accept a Read Current Values command. This command will cause the MO33 to transmit all of the information stored for each of the Model 1900's that it is polling. The data returned is the same as that illustrated in Table 7.

Each digit of the quality analog contains different error conditions that are described in Table 8 (with 0 indicating no error):

Table 8-Quality Status

Digit	*Description*	*1*	*2*	*3*	*4*
1 (LSD)	Pulse Timing	low	high	duplicate	
2	Communication	time out	message length	id	parity
3	Level	conversion	range		
4	Temperature	not available	range		

Priority of error conditions in Table 8 are from top to bottom (least significant to most significant digit) with the top row having highest priority and from left to right (smaller error numbers will mask higher error numbers). The logic behind this is:

> *If pulse timing fails, then communications fails. If communication fails then level and temperature data are invalid.*
>
> *If the message length is incorrect, then the ID contained in the message and parity do not matter.*

Data in the buffer does not change when polling a Model 1900 causes an error.

3.3.3.1 Pulse Timing Errors

This is a critical error. Timing of the pulses (Figure 2) is the basis of all communication. A pulse timing error would ordinarily indicate a noisy line or a fault condition. This is not likely to occur as a result of setting the Model 1900 to the wrong data rate, since the Model 1900 would not be able to recognize its address.

This error is reported in digit 1 (least significant digit) of the status analog. The error conditions are illustrated in Table 9:

Table 9-Pulse Timing Errors

Code	*Name*	*Description*
1	Low	Pulse timing is too short
2	High	Pulse timing is too long
3	Duplicate	Mark and Space lines were both active at the same time

3.3.3.2 Communication Errors

All communication errors indicate that the data is suspect. This error is reported in digit 2 of the status analog. Each error condition is described in Table 10 below:

Table 10-Communication Errors

Code	*Name*	*Description*
1	Time out	No response from the polled device.
2	Message length	An invalid message length was received (40 or 56 bits only).
3	ID	The response ID did not match the polled ID.
4	Parity	A parity error was received.

3.3.3.3 Level Data Errors

Level errors are reported in digit 3 of the status analog and provide information about the level data itself. Table 11 describes each of the level data errors:

Table 11-Level Data Errors

Code	*Name*	*Description*
1	Conversion	Failure during conversion from Gray code
2	Range	Value is outside of encoder's range

Range errors are based on the encoder type as described in the following:

Table 12-Level Ranges

Encoder	*Low Range*	*High Range*
English	0	80 feet
Metric	0	20 meters

3.3.3.4 Temperature Data Errors

Temperature errors are reported in digit 4 of the status analog and provide information about the temperature data itself. Table 13 describes each of these errors:

Table 13-Temperature Data Errors

Code	*Name*	*Description*
1	Not available	Model 1900 is not setup for temperature. This error occurs only if temperature is enabled (Section 3.3.1)
2	Range	Value is outside of sensor's range

Note that in error 1's description, it is possible to read temperature even if temperature is disabled. The MO33 will receive the longer temperature response message (Section 2.2.3.2) even if temperature has been disabled. The main benefit of disabling temperature is to prevent error 1 from occurring on every read. On the other hand, if one or two transmitters are setup for temperature, it is still possible to read those transmitters without generating errors on the rest.

The temperature range check is based on the following:

Table 14-Temperature Sensor Ranges

Sensor Type	*Low Range*	*High Range*
100 Ohm Copper	-100 °F	+400 °F
DIN Platinum	-234 °F	+502 °F

3.4 MSP Interface Process

This is the process that is responsible for generating mark/space polling messages and reading the corresponding mark/space responses from the Model 1900's. This process remains in an idle condition until the BSAP process indicates that a successful download of initial values (Section 3.3.2) from the master controller has been completed.

The download of initial values includes the address of up to 30 transmitters to be polled. Polling proceeds in a circular fashion starting with the first address in the list. After the last address is polled, the procedure begins again with the first address.

3.4.1 MSP Polling

Polling is accomplished by setting up the poll message (Figure 4) with the appropriate transmitter address and then using one of the 68HC11 timer interrupts on the MO33 to sequentially clock out each bit of the message. The interrupt time would be based on the data rate downloaded from the master controller (Section 3.3.1). This procedure is started by enabling the interrupt and then forcing the first interrupt to occur.

Each bit would require two interrupts, one to clock the mark or space pulse and the other to clock the delay between pulses (Figure 2). During the pulse half of this procedure, if the message bit is set, then the mark output line will be forced low, otherwise the space output line is forced low. On the next interrupt, both lines are returned to their normal high state. After each of the poll message bits have been clocked out in this manner, the transmit interrupt is disabled and the receive interrupt is enabled. At this time the MSP Process begins the Read Response Phase described in Section 3.4.2.

3.4.2 MSP Reading Responses

After each poll, the MO33 prepares to clock in the response data by responding to interrupts on the leading and trailing edges of the Mark and Space input lines. Prior to receiving the first bit a time-out timer is set to interrupt in the event that a response is not received within 0.5 seconds.

3.4.2.1 Clocking in Marks and Spaces

After receiving an interrupt, the program checks to see if the time since the previous interrupt is within the timing specification for the data rate that it is running at (Figure 2). If the timing is out of specification, a timing error is set in the status analog associated with the address being polled (Table 9).

If the timing is within specification, the Mark and Space lines are checked for their level. Checking the Mark / Space level is based on whether or not a bit or the delay between bits is being clocked in. If a bit is being clocked in, one and only one of the lines should be active. If the time between bits is being clocked in, both of the lines should be inactive. If either of these conditions fail, a duplicate error is set in the status analog associated with the address being polled (Table 9).

If there are no timing and no duplication errors, the appropriate bit is recorded in the receive buffer. Also at this time a bit count is incremented and if the bit is a mark, a mark count is incremented.

Whether or not an error is generated, the time-out interrupt is reset to 0.5 seconds. At the end of the message (either 40 or 56 bits) the time-out timer will time-out. This method allows for a mixture of short and long message types. At this point the end of message has been received.

The time-out interrupt sets an end of message flag and disables any further time-out interrupts..

3.4.2.2 Checking the Response for Errors

While in the response mode, the main routine is polling the end of message flag. When the end of message is detected, the next level of error checking is performed.

The first step is to check the status analog for the associated transmitter to see if a bit timing error occurred. If so, the MSP process immediately returns to the Polling Phase (Section 3.4.1) and begins polling the next address in the list or, if at the end of the list, the first address in the list.

Communications errors are the next to be tested for. If any of the errors are positive, the error is recorded in the status analog associated with the transmitter address being polled and no further error checking is performed. At this point the MSP Process returns to the Polling Phase (Section 3.4.1) and begins polling the next address in the list or, if at the end of the list, the first address in the list.

If the bit count is 0, a time out error is placed in the status analog associated with the transmitter address being polled. Otherwise, the bit count is compared with 40 or 56 to determining what type of message was received. If the bit count is not equal to 40 or 56, a message length error is set. If the transmitted address is not equal to the address polled, an ID error is set. Finally, if no other communication error is detected, the mark count is tested to be an odd number. If it is not an odd number, a parity error is set.

After determining that there were no communication errors, the level data is converted from Gray code to floating point via a table lookup and a BCD conversion. If the table lookup or the BCD conversion fail, a conversion error is declared. After the conversion, the level data is checked to verify that the value is within the range of the encoder being used. If the data is not within a valid range (Table 12), a range error is set. If no errors are detected, the data is stored in the data block associated with the transmitter that was polled.

Temperature errors are tested even if level errors occur. If temperature is enabled by the parameter download from the master controller (Section 3.3.1), a not available error is set if the transmitter responded with a 40 bit message (level only response). If a 56 bit message is received, a range test is performed on the temperature according to Table 14. If no errors are detected, the temperature is converted from BCD to floating point and stored in the data block associated with the transmitter that was polled.

The MSP process immediately returns to the Polling Phase (Section 3.4.1) and begins polling the next address in the list or, if at the end of the list, the first address in the list.

4. ACCOL Programming

4.1 Overview

Since the MO33 is programmed to communicate via BSAP, the ACCOL programmer can use the Master Module to communicate with the MO33. The MO33 handles all of the details concerned with polling the Model 1900 transmitters. Data received from polling is stored in a MO33 data buffer. Once the MO33 has been initialized (Section 4.3.2) and the initial values have been set (Section 4.3.3), the ACCOL programmer can read the current values at any time (Section 4.3.4).

To the ACCOL programmer, the MO33 appears as a slave controller with three slave modules running (Section 4.3).

4.2 Communication

4.2.1 Ports

Physically the MO33 plugs into the 33XX controller like a modem card. Depending on which board it is plugged into, communication is via Port B or Port D. The port that is used must be configured as a Master mode port. Refer to AIC Manual for details.

4.2.2 Baud Rate

The MO33 is hard wired to communicate at 9600 baud.

4.2.3 Slave Address

The MO33 will respond to <u>any</u> BSAP slave address (1 to 127).

4.3 Slave Modules

There are three slave modules programmed into the MO33 firmware. The first is used to initialize parameters that affect all of the Model 1900 transmitters (data rate, encoder type, etc.). The second is used to initialize values for each transmitter (including its polling address). Finally, the third module is used to read the current data values.

Information is always named from the orientation of the 33XX (the same as for the RTU):

> *e.g. an RTU writes parameters to the MO33;*
>
> *e.g. an RTU writes values to the MO33, the 33XX reads them from the MO33.*

To exchange information, the user must program ACCOL to issue commands (to MO33 Slave Modules) via calls to Master Modules with different terminal values.

Common terminals (whose values or meanings are always the same) are discussed in Section 4.3.1, while those that vary with each target Slave Module are discussed in the sections specific to each Slave Module.

The Master Module terminals are listed in Table 15:

Table 15-Master Module Terminals

Name	*Type*
REMOTE	common
POINT	individual
MODE	individual
INTYPE	individual
OUTTYPE	individual
INDEX	common
INLIST	individual
OUTLIST	individual
STATUS 1	common
STATUS 2	common

The user should also refer to the *ACCOL II Reference Manual (User Manual D4044)* for more information on Master and Slave Modules.

4.3.1 Common Terminals

The common terminals are those Master Module terminals whose values or meanings are the same for each of the MO33 Slave Modules.

4.3.1.1 Remote (1 to 127)

The MO33 will respond to any BSAP slave node address.

4.3.1.2 Index (0)

An entire signal list is always used.

4.3.1.3 Status 1

"Done" signal—as described in *ACCOL II Reference.*

4.3.1.4 Status 2

Result status signal—as described in *ACCOL II Reference*. In addition to the possible values for the STATUS 2 terminal listed in the *ACCOL II Reference* and generated by ACCOL, the Slave Modules in the MO33 may return any of the values listed in Section 4.3.5.

4.3.2 Initialize Parameters—Slave Module 60

After a power loss at the MO33, the only command that the MO33 will accept is to Initialize Parameters (until this is completed, the MO33 remains idle.)

If any command other than the Initialize Parameters command is sent, it is ignored by the MO33 and error -60 (awaiting Initialize Parameters) is returned in STATUS 2 (Section 4.3.5).

The Initialize Parameters command can be sent at any time, but doing so wipes out any configuration that may have been set in the MO33 previously and restarts the setup sequence.

If the attempt to Initialize Parameters fails (i.e. an error is reported in STATUS 2), the MO33 may be left in the same state as if it had never had its configuration set since startup or power loss.

The following sections describe each of the Master Module terminals required to communicate with this module.

4.3.2.1 Point (60)

This contains the slave module number—60—for the Initialize Parameters Module.

4.3.2.2 Mode (0)

This contains the mode of communications for this module. For the Initialize Parameters Module, the mode is Send—0.

4.3.2.3 Intype (?)

The Intype terminal is not used and is ignored by this module.

4.3.2.4 Outtype (0)

Indicates the type of data structure on the OUTLIST terminal. For this module Outtype will always be Signal List—0.

4.3.2.5 Inlist (?)

The Inlist terminal is not used and is ignored by this module.

4.3.2.6 Outlist (1 to 255)

The Outlist contains the Parameter Initialization list number. This list must be formatted as illustrated in Table 16.

Table 16-Parameter Initialization List

No.	Parameter	Type	Description	Values
1	Encoder	Analog	0) Fractional feet, 1) decimal feet or 2) metric.	0 to 2
2	Gauge Type	Analog	Initially not implemented[7] (0 = not used)	0
3	Transmitters	Analog	Number of transmitters on this drop	1 to 30
4	Metric	Logical	Centigrade if TRUE, else Fahrenheit[8]	True or False
5	Platinum	Logical	Platinum if TRUE, else Copper temperature sensor	True or False
6	Temperature	Logical	Enabled if TRUE	True or False
7	Low Rate	Logical	Low if TRUE, else High speed polling of Model 1900	True or False

If the value for any parameter is outside the indicated range or is non-integral—-50 (invalid signal value) is returned in STATUS 2.

4.3.3 Write Initial Values—Slave Module 61

After the Initialize Parameters step has completed successfully, the only[9] command that the MO33 will accept is to Write Initial Values. Until this is completed, the MO33 continues to wait. If anything other than the Write Initial Values command is sent, it is ignored by the MO33 and error -61 (awaiting Write Initial Values) is returned in STATUS 2.

Write Initial Values commands can only be sent right after the Initialize Parameters command. At any other time a Write Initial Values command is ignored by the MO33 and error -9 (Slave is off line) is returned in STATUS 2.

[7] May be implemented later to support 6500 Servo Gauge

[8] Will generate an error (-50) in STATUS 2 if metric encoder is selected.

[9] The MO33 will also accept an Initialize Parameters (Section 4.3.2) command during this step.

The following sections describe each of the Master Module terminals required to communicate with this module.

4.3.3.1 Point (61)

This contains the slave module number—61—for the Write Initial Values Module.

4.3.3.2 Mode (0)

This contains the mode of communications for this module. For the Write Initial Values Module, the mode is Send—0.

4.3.3.3 Intype (?)

The Intype terminal is not used and is ignored by this module.

4.3.3.4 Outtype (0)

Indicates the type of data structure on the OUTLIST terminal. For this module Outtype will always be Signal List—0.

4.3.3.5 Inlist (?)

The Inlist terminal is not used and is ignored by this module.

4.3.3.6 Outlist (1 to 255)

The Outlist contains the Write Initial Values list number. This list must be formatted as illustrated in Table 17.

Table 17-Write Initial Values List

No.	*Name*	*Type*	*Description*	*Values*
(N-1)*6+1	Address	Analog	Model 1900 polling address	1 to 999
(N-1)*6+2	Level	Analog	Initial tank level	Table 12
(N-1)*6+3	Temperature	Analog	Initial tank temperature[10]	Table 14
(N-1)*6+4	Quality	Analog	Quality Status	Table 8
(N-1)*6+5	Alarm 0	Logical	State of external alarm switch 0	True or False
(N-1)*6+6	Alarm 1	Logical	State of external alarm switch 1	True or False

Where N is the number of transmitters defined in the Initialize Parameters Section 4.3.2. If there are fewer Initial Values than the number of transmitters would require, error -6 (invalid slave input structure) is returned in STATUS 2. Extra initial values are reported as error -11 (slave input structure overflow).

If the value for any parameter is outside the indicated range (or is non-integral for values other than level and temperature)— -50 (invalid signal value) is returned in STATUS 2.

4.3.4 Read Current Values—Slave Module 62

This command is only available after the Write Initial Values command has completed successfully. A status error of -60 indicates that the MO33 has not been initialized (or has had a power failure, Section 4.3.2). A status error of -61 would indicate that the MO33 is waiting to have the initial values written (Section 4.3.3).

Assuming these prior initialization steps have been completed, the Read Current Values step may be called at any time. The results are returned as described in Section 4.3.4.5.

4.3.4.1 Point (62)

This contains the slave module number—62—for the Read Current Values Module.

[10] No error checking is performed on this value if temperature was disabled in the Initialize Parameters Section 4.3.2.

4.3.4.2 Mode (1)

This contains the mode of communications for this module. For the Read Current Values Module, the mode is Poll—1.

4.3.4.3 Intype (0)

Indicates the type of data structure on the INLIST terminal. For this module Intype will always be Signal List—0.

4.3.4.4 Outtype (?)

The Outtype terminal is not used and is ignored by this module.

4.3.4.5 Inlist (1 to 255)

The Inlist contains the Read Current Values list number. This list is formatted the same (and can actually be the same list) as the Output List for Write Initial Values illustrated in Table 17.

4.3.4.6 Outlist (?)

The Outlist terminal is not used and is ignored by this module.

4.3.5 Status 2 Values

Table 18 *(reproduced from the ACCOL II Reference manual, with additions)* lists the possible error conditions that can be detected by the Master and Slave Modules. *Positive* number codes indicate warnings or informational messages, while *negative* number codes indicate errors which inhibit proper execution of the modules. Along with each message, a letter indication notes whether the error is detected by a module routine (R), or a module task (T), and whether it is reported to the Master Module (M) or the Slave (S), or both. Whenever an error is detected by any component, the task or module will set the STATUS 2 terminal of the module with the error code.

Bold entries represent errors that are directly written by the MO33 firmware.

Table 18-Status 2 Error Codes

Code	*Detect*	*Report*	*Description*
2	T	M,S	Signal was control inhibited
1	T	M	Slave has more data to send
***0**	**R,T**	**M,S**	**Successful completion. Slave Module execution was completed in the MO33 without error.**
***-1**	**R**	**M**	**Invalid mode (must be 0 or 1). The wrong MODE value was supplied to the MO33 Slave Module.**
-2	R	M	Invalid slave node number
-3	R	M	Invalid output list number
-4	R	M	Invalid master input structure number
-5	R	M	Invalid master output structure
-6	R	M,S	Invalid slave input structure. A signal in the signal list or data array is required but absent, or is not of the expected type.
-7	R	M,S	Invalid slave output structure number
***-8**	**R,T**	**M**	**Invalid slave point. There is no such Slave Module in the MO33.**
***-9**	**T**	**M**	**Slave is off line. As discussed in Section 4.3.3, some Slave Modules can only be accessed as part of the configuration setup sequence. An attempt to issue a Master Module command to access such a Slave Module at the wrong time (other than during its step in configuration) results in this error.**
-10	T	M	Master input structure overflow

***-11**	**T**	**M, S**	**Slave input structure overflow. Number of transmitters were defined to be larger that 30 in during Parameter Initialization (Section 4.3.2) or more initial values were defined during Write Initial Values (Section 4.3.3) than were defined in Parameter Initialization.**
***-12**	**T**	**M, S**	**Both input and output slave errors. The signal list to be read from the MO33 contains too many signals or values.**
-13	T	M, S	Implicit type conversion attempted
-14	T	M	Communications error (Request Send Error)
-15	T	M	Communications error (Time Out On Response)
-16	T	M, S	Communications Failure (Slave Send)
-17	R	M	Invalid type, must be 0, 1, 2 or 3
-18	R	M	Master has a zero length I/O list
*** -19**	**T**	**M, S**	**Slave has a zero length I/O list. This error indicates that during Initialize Parameters (Section 4.3.2), zero transmitters were defined.**
-20	R	M	Peer request block is unwired
-21	T	M, S	Signal array could not be updated
-22	T	M, S	Signal array could not be updated because it was write protected, or a constant, or overflow attempted
-23	R	M	Node routing table not yet initiated
*** -50**	**(T)**	**(M)**	**A signal in the signal list or data array contains a value which is unusable for the purpose implied by its position in the list or array.**
*** -60**			**Awaiting Initialize Parameters (Section 4.3.2) which could also indicate a power failure has occurred.**
*** -61**			**Awaiting Write Initial Values (Section 4.3.3)**

Return to Application Notes Menu

Return to the List of Manuals