Proposed

GOES HDR
Binary Protocol Specification

12/08/2006

Microcom Design, Inc.
10948 Beaver Dam Road, Suite C
Hunt Valley, MD 21030

Table of Contents

List of Figures

List of Tables

1 Introduction

Since its inception, the GOES HDR system has left open the possibility for binary message transmissions to be utilized. Currently only ASCII and Pseudo-Binary (which uses a subset of the ASCII character set) formats are in use. The reason for this is that a binary standard for such communications has been left “To Be Determined” in the GOES HDR Transmitter Certification Standards. This document details a proposed binary protocol specification that, if adopted, will provide a mechanism to rapidly transition to binary messages, significantly reducing the message lengths.

The binary protocol detailed in this document defines both the basic binary message structure, as well as several specific binary message formats. The structure portion of the protocol defines the individual fields that are used/required in a binary message. The format portion of the protocol specifically addresses the data fields in a binary message. In addition to providing for a free format binary data, this protocol defines three specific message formats that will allow the existing message formats (ASCII and Pseudo-Binary) to be compacted at the transmitter and de-compacted at the receiver (a.k.a. the demodulator). These compaction schemes will reduce the time and cost for GOES users to transition to binary since the data delivered to their information processing systems will be in the same format currently being utilized.

2 Binary Message Structure

The binary protocol utilizes a packet structure with variable length (i.e. user selectable) packets. The packet structure is shown in Figure 1. As shown, each packet consists of a one byte length field, the data packet, and a 16-bit CRC field.

Packet Length 1 Byte	Data Bytes	16-Bit CRC

Figure 1: General Packet Structure

The CRC is a hash function that produces a checksum, which allows verification of the packet data. The 16-bit CRC-CCITT, which has a polynomial generator of $x^{16} + x^{12} + x^5 + 1$ (truncated polynomial 0x1021), must be implemented. The CRC is generated from both the Packet Length and the Packet Data bytes.

To facilitate both long and short binary messages, the binary message structure can be in either one of the two formats shown in Figure 2 and Figure 3.

The message structure shown in Figure 2 is for short binary messages where only a single packet is required.

Carrier 0.5s 0.25s	Clock 1-0-1 1=180	FSS 15 Bits	GOES ID 4 Bytes	Flag Byte 8 bits	Packet Length 1 Byte	Data	16-Bit CRC	Encoder Flush (Variable)

Figure 2: Binary Message Structure Single Packet

The message structure shown in Figure 3 is utilized for longer binary messages where multiple packets are required to completely send the data. In addition to the individual packet length fields, a total message length field is sent immediately following the GOES flag byte. This message length is the total number of bytes that will be transmitted following this field, including all the bytes in each packet (length byte + data bytes + 2-byte CRC) and the two encoder flush bytes.

Carrier 0.5s 0.25s	Clock 1-0-1 1=180	FSS 15 Bits	GOES ID 4 Bytes	Flag Byte 8 bits	Message Length 2 Bytes	Data Packets | Length | Data | CRC |	Encoder Flush 2 Bytes

Figure 3: Binary Message Structure Multiple Packets

This message length is a two byte field with the length represented as a 14-bit value. The seven most significant bits are sent in the first byte and the least significant seven bits are sent in the second byte. Both bytes include an odd parity bit.

$P_O L_{13} L_{12} L_{11} L_{10} L_9 L_8 L_7$ MS Byte	$P_O L_6 L_5 L_4 L_3 L_2 L_1 L_0$ LS Byte

Figure 4: Message Length Structure

Note that for both formats, the Carrier, Clock, FSS, and GOES ID fields are identical to the non-binary message structure.

The GOES HDR flag byte immediately follows the GOES ID (as in the standard message format), but is extended by this specification as defined in Table 1. Note that the bit numbering convention is the same as in the Certification Standard, and the previously defined bits are unchanged. However, the binary protocol makes use of some of the previously "spare, undefined" bits and also implements some of the bits that were reserved for "Possible Future Enhancement", but were never fully specified. Only the modified bits are defined herein.

Table 1: GOES HDR Flag Byte

Bit(s)	Name	Description
1 LSB	Multiple Packets B_{MP}	0 = Message includes only one packet. 1 = Message includes multiple one packet. Only used for Binary and Compacted Messages.
2	UTC Time Sync B_{TS}	0 = No UTC Time Sync since last transmission. 1 = UTC Time Sync since last transmission. Only used for Self-Timed Messages.
3	Compaction B_{CM}	0 = No Compaction in Use. 1 = ASCII/Pseudo Binary Compaction Active. (Binary application to be determined.)
4	Coding Type B_{CM}	0 = ASCII Numeric Compaction. 1 = ASCII Alphanumeric Compaction. (Binary and Pseudo-Binary use to be determined.)
5	Spare	TBD – Send as 0
7/6	Message Type B_{MT}	00 = Reserved. 01 = ASCII 10 = Binary 11 = Pseudo-Binary
8 MSB	Parity P_O	Odd Parity

Bit 1 is the Multiple Packet flag (B_{MP}) and is used to define whether the multiple packet (B_{MP}=1) structure of Figure 3 is used, or the single packet (B_{MP}=) structure of Figure 2 is in use. Note that B_{MP} is only active when the message type flags define a Binary message type (B_{MT}=10), or when Compaction is active (B_{CM}=1) for Pseudo-Binary or ASCII messages. When this bit is not applicable, it must be sent as a 0.

Bit 3 is the Compaction flag (B_{CM}). When ASCII or Pseudo-Binary messages are sent with a compaction algorithm, this flag must be true (B_{CM}=1); otherwise, it must be 0 (indicating a standard ASCII or Pseudo-Binary message). Note that when this flag is true, the B_{MP} bit is active. The use of this flag for a Binary message type (B_{MT}=10) is TBD and should be set to 0 as the default.

Bit 4 is the Coding Type flag (B_{CT}). For ASCII compacted messages (B_{MT}=01 and B_{CM}=1), one of two compaction algorithms can be used as defined in Sections 3.3 and 3.4. This flag bit defines which algorithm is in use for the message; B_{CT}=0 indicates Numeric ASCII Compaction, and B_{CT}=1 indicates Alphanumeric ASCII Compaction.

2.1 Length Versus EOT

Note that for both binary message structures shown in Figure 2 and Figure 3, there is not an EOT sequence specified. The required length fields eliminate the need for an EOT.

2.2 Encoder Flush

As shown in Figure 2, the Encoder Flush field for the single packet message is of variable length. At a minimum, two “encoder” flush bytes are required. These two flush bytes are required to flush the transmitter’s encoder. Flush bytes are all zeroes prior to scrambling.

However, depending on the length of the packet, additional flush bytes may be required in the single packet case to ensure the Packet Length field is fully flushed from the Trellis decoder at the demodulator. In order to be sure the Packet Length is flushed from the decoder, a minimum of 16 bytes (including the flush bytes) must be encoded, scrambled and modulated following this field. As such, if the total bytes including the data in the packet and the two byte CRC is equal to or greater than 14, then no additional flush bytes are required. Otherwise, additional all zero bytes must be presented to the scrambler/encoder to bring the total number of transmitted bytes after the packet length up to 16.

Alternately, instead of transmitting additional flush bytes, the packet size may be increased to 12 bytes and null data bytes appended to the packet data to bring the total number of packet bytes, excluding the packet length, to 14 (packet size of 12 bytes plus 2 CRC bytes). In which case, only the minimum requirement of two zero bytes need to presented to the scrambler/encoder to bring the total number of bytes after the packet length to 16 and flush the encoder.

For the multiple packet case (shown in Figure 3), only two Encoder Flush bytes should be required. While the total number of bytes following the Message Length field must also be 16 bytes to ensure this field is fully flushed from the decoder, there is no reason to utilize the multiple packet structure with messages of such a small size.

3 Binary Message Formats

In addition to defining the binary message structure, this specification details one free format binary message and three compacted binary message formats. The compacted formats are based on existing ASCII and Pseudo-Binary messages.

A compacted message starts with the data in ASCII or Pseudo-Binary (PB) format; the transmitter compacts the data and transmits it in binary, then the receiving equipment de-compacts it back to its original ASCII or PB format. Note that while the information begins and ends in ASCII or PB, these messages are still considered a binary message since the compacted bytes can take on any 8-bit value.

3.1 Binary Message Format

The binary message format does not impose any restrictions on the actual message data. Data bytes may take on any 8-bit value.

The single packet binary message structure beginning with the Flag Byte is shown in Figure 5. The actual flag byte value is also shown; the bit defined as X is the UTC Time Sync bit (B_{TS}), which is unrelated to message structure. The bit designated as P is the odd parity bit for the flag byte.

Flag Byte P10000X0	Length L_B	Data (L_B + 1) Bytes	16-Bit CRC	Encoder Flush

Figure 5: Binary Message Structure Single Packet

Multiple packet binary messages are transmitted with the following structure:

Flag Byte P10000X1	Message Length	Data Packets | L_B | Data | CRC |	Encoder Flush

Figure 6: Binary Message Structure Multiple Packets

Within each Data Packet, the Packet Length (L_B) is one less than the number of data bytes in the packet. In other words, the Packet Length does not include the two CRC bytes nor does it include the Packet Length byte itself. As such, the maximum number of data bytes in a packet is 256, and the minimum is 1.

3.2 Compacted Pseudo-Binary Message Format

The Compacted Pseudo-Binary Message is used to send Pseudo-Binary in a compressed format.

Pseudo-Binary (PB) bytes are ASCII characters in the format shown in Figure 7. The Compacted Pseudo-Binary message strips out the two most significant bits since these bits do not carry any information.

$$P_O \; 1 \; B_5 \; B_4 \; B_3 \; B_2 \; B_1 \; B_0$$

Figure 7: Pseudo-Binary Bit Map

The resultant six information bits from every byte are compacted together to form the binary information to be transmitted. Every four PB bytes is compacted into three binary bytes as shown in Figure 8.

$P_O1a_5a_4a_3a_2a_1a_0$ $P_O1b_5b_4b_3b_2b_1b_0$ $P_O1c_5c_4c_3c_2c_1c_0$ $P_O1d_5d_4d_3d_2d_1d_0$

$a_5a_4a_3a_2a_1a_0b_5b_4$ $b_3b_2b_1b_0c_5c_4c_3c_2$ $c_1c_0d_5d_4d_3d_2d_1d_0$

Figure 8: Pseudo-Binary Compaction

Once compacted, the resulting binary bytes are packetized and transmitted. The message structures for Compacted Pseudo-Binary messages are shown Figure 9 and Figure 10. Note that in both cases the flag byte defines a Message Type of Pseudo-Binary (B_{MT}=11), but the compaction flag is true (B_{CM}=1).

Flag Byte	Length	Data	16-Bit	Encoder
P11001X0	L_C	Integer[3*(L_C+1)/4]+1	CRC	Flush

Figure 9: Compacted Pseudo-Binary Message Structure Single Packet

Flag Byte	Message	Data Packets	Encoder
P11001X1	Length	| L_C | Data | CRC |	Flush

Figure 10: Compacted Pseudo-Binary Message Structure Multiple Packets

Within each Data Packet, the Packet Length (L_C) is one less than the number of Pseudo-Binary characters to be de-compacted. The character count is used instead of the number of binary bytes in order to determine if the last six bits in the compacted three bytes carries meaningful data.

The actual byte length of the packet is given by the equation shown below; the Integer[x] function returns just the integer portion of x.

$$\text{Integer}[3*(L_C+1)/4)]+1$$

Since the maximum number of characters is 256, the maximum byte length for a Pseudo-Binary Compacted packet is 192. Therefore, this compaction scheme can yield nearly a 25% reduction in message length. Note also that it is not necessary to always send the binary data in multiples of three bytes.

3.3 Numeric ASCII Compaction Format

The Numeric ASCII Compacted Message format is used to compact ASCII messages that only consist of numeric ASCII characters (digit, polarity signs, decimal point, etc.). This format translates the ASCII characters shown in Table 2 to a 4-bit code. Every two of these four bit codes are then compacted into a single byte.

Table 2: Numeric ASCII Character Set

Char	Code	Char	Code	Char	Code	Char	Code
0	0000	4	0100	8	1000	. (dp)	1100
1	0001	5	0101	9	1001	+	1101
2	0010	6	0110	space	1010	-	1110
3	0011	7	0111	,	1011	/	1111

The Numeric Compaction algorithm also supports the special character translation to two 4-bit codes shown in Table 3. Since the sequences shown in the Numeric Chars column are not encountered when representing numeric values, these special sequences further enhance the usability of this compaction scheme. After de-compaction to the numeric character set using Table 2, the demodulator must also look for the sequences shown in Table 3, and translate these sequences back to the corresponding ASCII character(s).

Table 3: Numeric ASCII Special Character Translation

ASCII Char(s)	Numeric Chars	Double Code
cr/lf	++	1101 1101
#	+-	1101 1110
=	-+	1110 1101
:	..	1100 1100
E	--	1110 1110

Once compacted, the resulting binary bytes are packetized and transmitted. The message structures for Compacted Numeric ASCII messages are shown Figure 11 and Figure 12. Note that in both cases the flag byte defines a Message Type of ASCII (B_{MT}=01), but the compaction flag is true (B_{CM}=1).

Flag Byte P01001X0	Length L_C	Data Round(L_C/2) Bytes	16-Bit CRC	Encoder Flush

Figure 11: Compacted Numeric ASCII Message Structure Single Packet

Flag Byte P01001X1	Message Length	Data Packets | L_C | Data | CRC |	Encoder Flush

Figure 12: Compacted Numeric ASCII Message Structure Multiple Packets

Within each Data Packet, the Packet Length (L_C) is one less than the number of 4-bit codes to be de-compacted. The code count is used instead of the number of binary

bytes in order to determine if the last four bits in the compacted data carries meaningful data. Note that the special translation sequences count as two 4-bit sequences even if the resulting ASCII equivalent is a single character. Since the maximum number of 4-bit codes is 256, the maximum number of bytes that can be included in a Compacted Numeric ASCII packet is 128, resulting in nearly a 50% compression ratio.

3.4 Alphanumeric ASCII Compaction

The Alphanumeric ASCII Compacted Message format is used to compact ASCII messages that consist of the subset of the ASCII character shown in Table 4. The first column of Table 4 is the numeric characters from Table 2. However, these characters are now encoded as a 5-bit binary value with the most significant bit being 0. An additional 31 characters (including the uppercase letters) is defined using a 6-bit code that has the most significant bit set to 1. This variable size code set provides a total of 47 characters; the six bit all ones code is not assigned (n/a).

Table 4: Alphanumeric ASCII Character Set

Char	Code	Char	Code	Char	Code
0	00000	A	100000	Q	110000
1	00001	B	100001	R	110001
2	00010	C	100010	S	110010
3	00011	D	100011	T	110011
4	00100	E	100100	U	110100
5	00101	F	100101	V	110101
6	00110	G	100110	W	110110
7	00111	H	100111	X	110111
8	01000	I	101000	Y	111000
9	01001	J	101001	Z	111001
space	01010	K	101010	cr/lf	111010
,	01011	L	101011	#	111011
.	01100	M	101100	=	111100
+	01101	N	101101	:	111101
-	01110	O	101110	;	111110
/	01111	P	101111	n/a	111111

During the translation process, the 5 or 6-bit codes are continuously packed together to form bytes. The compacted data is then packetized using the message structures

shown in Figure 13 and Figure 14, and then transmitted similar to the other compaction schemes.

Flag Byte P01011X0	Length L_B	Data L_B Bytes	16-Bit CRC	Encoder Flush

Figure 13: Compacted Alphanumeric ASCII Message Structure Single Packet

Flag Byte P01011X1	Message Length	Data Packets | L_B | Data | CRC |	Encoder Flush

Figure 14: Compacted Alphanumeric ASCII Message Structure Multiple Packets

As the data is received, the codes are de-compacted and reverse translated. The de-compaction algorithm first requires the examination of the next un-compacted bit to determine how many total bits to extract (0 = 5 bits and 1 =6 bits).

Within each Data Packet, the Packet Length (L_B) is one less than the number of data bytes in the packet. The maximum number of data bytes in a packet is 256. To ensure any trailing bits are not erroneously decoded, unused bits must be filled with ones (1). Since the six bit all ones code in Table 4 has been reserved and there is not an all ones 5-bit code, the trailing bits of 1's can be readily discarded.

Assuming all characters are equally likely to be represented, the average number of bits per character is 5.66. For a maximum packet length, this would yield 361 characters in a 256 byte message (8*256/5.66). However, if the data to be compacted is totally numeric, as many as 409 ASCII characters can be represented in a single packet (8*256/5 = 409). In other words, the compression ratio is between 29% and 37% depending on the data structure.