GOES HDR Binary Data Message Format Modification Proposals

Signal Engineering, Inc.

March 23, 2006 Revision 1.2

Document Change History

3-23-2006 Changes from Rev1.1

1. GOES International channel 31-bit EOT pattern description clarified.
2. Option 4 (Use existing GOES HDR Binary Data scheme) added.

Overview

In GOES Higher Data Rate (HDR) 300 and 1200 bit per second messages containing ASCII or Pseudo Binary data, the end of the message is marked by the 8-bit ASCII EOT character (value 04). This method works because the rules of the GOES system prohibit transmitters from including an EOT character as part of the data in the message. If an EOT character were to be included in the data portion of the message, a GOES receiver would terminate reception of the message prematurely when the false EOT was received.

The current GOES HDR specification defines Binary data messages as those with the 2-bit data format field in the 8-bit Flagword set to "Binary" and terminated by the same 4-byte pattern used as the EOT pattern for GOES International channel transmissions. The GOES International channel 31-bit EOT pattern in binary format is:
0 0 1 0 0 0 0 0 1 0 1 1 1 0 1 1 0 1 0 1 0 0 1 1 1 1 0 0 0 1 1

(This is equivalent to 20BB53C6 hexadecimal if the 31 bits of the EOT pattern are left-justified in a 32-bit field with the least significant bit of the 32-bit field set to 0.)

The problem with this method is that the message data is binary and may therefore contain any sequence of data values, including the 31-bit EOT pattern for HDR Binary data messages. If the sender's data happens to include the 31-bit HDR Binary data EOT pattern, then the receiver will terminate reception of the message prematurely when the false EOT pattern is received.

GOES DCP data is organized as 8-bit bytes, with the least significant bit of each byte transmitted first and the MSBit transmitted last. If the user data being transmitted by the DCP was binary data and contained the 4-byte sequence 04 DD CA 63 hexadecimal, then the stream of data bits received by the GOES receiver would match the 31-bit GOES International channel EOT pattern. This is because the data byte value 04 hexadecimal is transmitted LSBit first and is received by the GOES receiver as the binary pattern: 0 0 1 0 0 0 0 0, where the leftmost bit is received first. In a similar way, the data byte value DD hexadecimal is received as binary 1 0 1 1 1 0 1 1, CA hexadecimal is received as 0 1 0 1 0 0 1 1, and 63 hexadecimal is received as 1 1 0 0 0 1 1 0. Since the GOES receiver is looking for the 31-bit binary sequence of the GOES International channel EOT pattern, this 4-byte binary sequence will cause the receiver to detect that EOT pattern and terminate the message, even if the message is actually longer and the DCP continues to transmit.

Option 1: Byte Stuffing

One solution would be to modify the original GOES HDR Binary data message format using a technique known as "byte stuffing" that is used in data communication to send data that contains control characters. In this modified HDR Binary format, each time the GOES HDR transmitter encounters the HDR Binary data 4-byte EOT pattern (04DDCA63 hex) in the message data, it inserts an ASCII Data Link Escape (DLE) character (value 10 hex) into the sequence of data bytes following the first three bytes of the 4-byte EOT pattern. This prevents the receiver from encountering the 4-byte EOT pattern until the true EOT pattern is received at the end of the message. The GOES HDR receiver must discard any inserted DLE characters by discarding the character following

each occurrence of three consecutive bytes that contain the first three bytes of the 4-byte EOT pattern.

The transmitter must also insert a DLE character following the first 3 bytes of each occurrence of the 4-byte pattern 04DDCA10 hex. This ensures that the receiver does not discard the byte with value 10 hex in a 4-byte sequence of actual data bytes that happens to contain the value 04DDCA10.

The advantage of this modified HDR binary data format is that it requires the minimum change to the existing GOES HDR specification and is relatively simple to implement. The disadvantage is that it requires that an extra byte of data be added to the message each time the binary data contains the HDR Binary data EOT pattern. This makes the actual length of the transmitted message variable and dependent on the content of the user's binary data.

If the user sends a very long message containing random binary data values, then this method may add a significant amount of extra data. If no precautions were taken, then in the worst case (which is extremely unlikely but still possible), so much extra data would be added that the transmitter could exceed the length of its assigned transmission time slot. In order to guarantee that a binary data message sent using byte stuffing does not exceed its assigned transmission slot, the user would have to assume the worst case scenario: that one extra byte of data will be sent for every four bytes of the user's actual data. The user would then limit the amount of actual binary data in each message so that the combined length of the actual data and the extra data bytes assumed to be added by byte stuffing would not exceed the transmission time slot.

Option 2: Byte Stuffing plus Error Detection Code

One other disadvantage to the simple byte stuffing scheme that was not mentioned above is that there is no mechanism for the HDR receiver to detect errors in the binary data (since binary data does not contain a parity bit like ASCII and Pseudo Binary format data bytes). It would useful to add an error detection code field such as a CRC to the end of the HDR binary data message. This error detection field would cover just the binary data so that the HDR receiver could detect if any errors were present in the binary data.

The error detection code appended to the binary data in the message could be a 2-byte CRC. The 16-bit CRC-CCITT is widely used and can be implemented using a look-up table in a way that requires very little processing power to generate the CRC in the transmitter and to check it in the receiver. If the HDR receiver supported the DAMS-NT interface, it would set the Data Quality metric to "Poor" if the CRC indicated that errors were present in the binary data. The 2-byte CRC field would be appended to the user's binary data before the byte stuffing process was performed.

Option 3: Using a Length Field and Error Detection Codes

To send binary data without using an EOT pattern requires that a Length field, containing the number of binary data bytes in the message, be added to the HDR Binary data message format. The Length field would be located after the Flagword, but before the binary data bytes in the GOES HDR Binary data message. After the HDR receiver recognized from the Flagword that the message was a Binary data message, it would use the contents of the Length field to know when the message had been completely received.

The Length field would be followed by an error correction code field that would protect both the Length field and the HDR Flagword. In this way the HDR receiver can determine if the control fields used to recognize an HDR Binary data message (the Flagword) and determine how long the message is (the Length field) are valid. If errors are detected in the Flagword and/or Length fields and the errors are not correctable, then the HDR receiver would revert to demodulating the message as if it were a GOES HDR ASCII or Pseudo Binary format message. The error correction code field for the Flagword/Length field is necessary because now there is no EOT pattern and the content of the Length field is the only thing that tells the HDR receiver when the end of the Binary message has been reached.

In addition to the new Length field and error correction code field for the Length and Flagword, it would useful to add an error detection code field such as a CRC to the end of the HDR binary data message. This error detection field would cover just the binary data so that the HDR receiver could detect if any errors were present in the binary data. Since GOES HDR Binary data does not have a parity bit in each 8-bit byte as the GOES HDR ASCII and Pseudo Binary data formats do, something like this would be needed for the HDR receiver to be able to detect if any errors were present in the binary data or not.

Ideally the new GOES HDR Binary data message format should be a scheme that is simple to implement and does not require a large amount of processing power in the HDR transmitter or receiver. A reasonable approach would be to use a 14-bit Length field. This would support the maximum message length allowed in the current and proposed GOES HDR standards. Together with the least significant 7 bits of the HDR Flagword field (i.e. everything except the odd parity in the MSBit of the Flagword), this gives 21 bits of information that must be protected by an error correction code field.

A 10-bit BCH error correction code could be used to protect the 21 bits in the Flagword and Length fields. This could detect any errors and correct up to 2 bit errors in the 21-bit field. This would give a total length of 32 bits for the 8-bit Flagword, 14-bit Length field, and 10-bit BCH code field. The binary message data would follow these 32 bits. Since the existing GOES BCH-coded Platform ID is also a 31-bit field containing 21 bits of information and a 10-bit BCH code field, the same BCH coding algorithm can be used for the Flagword and Length fields of the new HDR Binary data format. Using the existing 31/21 bit GOES BCH code algorithm to protect the HDR Binary Flagword/Length field would allow transmitters and receivers to use existing GOES BCH encoding and decoding software routines.

The error detection code appended to the binary data in the message could be a 2-byte CRC. The 16-bit CRC-CCITT is widely used and can be implemented using a look-up table in a way that requires very little processing power to generate the CRC in the transmitter and to check it in the receiver. If the HDR receiver supported the DAMS-NT interface, it would set the Data Quality metric to "Poor" if the CRC indicated that errors were present in the binary data.

A final consideration is choosing a new HDR Binary data format that would minimize compatibility issues with existing GOES HDR receivers that don't support the new Binary message format. As a safeguard against existing GOES HDR receivers continuing to demodulate after the end of a new-format HDR Binary data message, the last byte of the new HDR Binary message format (just after the 2-byte CRC field) should be the 8-bit EOT pattern currently used for GOES HDR ASCII and Pseudo Binary format messages. This would ensure that older GOES HDR receivers would stop demodulating at the end of a new-format HDR Binary data message, even if none of the Length, Flagword/Length BCH code, binary data, or CRC fields happened to contain an occurrence of the 8-bit EOT pattern.

In this new scheme, the GOES HDR receiver knows that the last byte of the message has been received when the number of binary data bytes specified in the Length field and the two CRC bytes and the final 8-bit EOT byte have all been received.

This new HDR Binary data scheme would require always transmitting six more bytes in each message than in the current GOES HDR messages. It would have the advantages that it would not require sending a data-dependent amount of extra data to prevent false detection of an EOT pattern and would detect if errors were present in the message data.

Option 4: Use the existing GOES HDR Binary Data scheme as-is

The simplest option of all in terms of the changes required to the existing GOES DCS system to support binary data in GOES HDR messages would be for the DCP transmitters and the GOES receivers to implement the binary data scheme exactly as described in the current GOES HDR specification. If a GOES DCP transmitted binary data, then it would set the 2-bit data format field in the 8-bit Flagword field to indicate Binary data and terminate the message with the same 31-bit EOT pattern as used for GOES International channel transmissions. The DCP would not modify the user's binary data in any way.

When a GOES receiver received a GOES HDR message with the Flagword indicating Binary data, it would terminate the message only when it had received the 31-bit GOES HDR Binary EOT pattern, not when it had received just the 8-bit EOT pattern used for GOES ASCII and Pseudo Binary format data messages.

In this option, each user would responsible for ensuring that his data did not contain a 4-byte sequence of binary data that a GOES HDR receiver would interpret as the 31-bit EOT pattern. The user could do this by implementing a byte-stuffing scheme as described in Option 1 (or by some other data encoding method) that would be used to pre-process the binary data before it was given to the DCP transmitter. The user would then post-process each binary data message received by the GOES DCS system to recover his

binary data. This is the method that the EUMETSAT organization uses for the transmission of binary data on the METEOSAT Second Generation system: all binary data values are allowed for each 8-bit data byte transmitted, but it is up to the user to ensure that the receiver does not prematurely declare end of message because of the data content.

If the user also wanted error detection or error correction coding, then the user would add it to the binary data before giving the message to the DCP transmitter and check for or correct errors after receiving the DCP message from the DCS system.