HRIT DCS File Format
Microcom Design Inc.

10948 Beaver Dam Road, Suite C
Hunt Valley, Maryland 21030-2237

Tel: 410.771.1070
Fax: 410.771.0018

1. Introduction

This specification defines the proposed HRIT file format to replace the existing LRIT/HRIT DCS File format.

2. HRIT DCS File Naming

The filename for HRIT DCS messages will as follows:

pH-YYDDDHHMMSS-Q.dcs

where,

- H designates the new HRIT file format
- YYDDDHHMMSS is the file date/time of creation in UTC Julian format
- Q is an ASCII letter (A to Z) used in the event two files are generated at the same time

3. HRIT DCS File Structure

The BNF HRIT DCS file structure is summarized below:

FILE := FILE_HEADER BLOCKS FILE_CRC32

FILE_HEADER := FILE_NAME FILE_SIZE SOURCE TYPE EXP_FIELD HDR_CRC32
FILE_NAME := 32*octet
FILE_SIZE := 8*octets
SOURCE := 4*octets
TYPE := 4*octets (:= “DCSH”, see Section 3.1.4)
EXP_FLD := 12*octets
HDR_CRC32 := 4*octets

BLOCKS := BLK_ID BLK_LNG BLK_DATA BLK_CRC
BLK_ID := 1*octet
BLK_LNG := 2*octets
BLK_DATA := 0..65,530*(octets)
BLK_CRC16 := 2*octets

FILE_CRC32 := 4*octets

DCP_BLOCK := DCP_ID BLK_LNG DCP_HDR DCP_DATA BLK_CRC (see Section 3.3)
DCP_ID := 1*octet (:= 0x01)
BLK_LNG := 2*octets
DCP_HDR := 36*octets (see Section 3.3.1)
DCP_DATA := 0..16,000*(octets)
BLK_CRC16 := 2*octets

10948 Beaver Dam Road, Suite C
Hunt Valley, Maryland 21030-2237
Tel: 410.771.1070
Fax: 410.771.0018

MM_BLOCK := MM_ID BLK_LNG MM_HDR BLK_CRC (see Section 3.4)

MM_ID := 1*octet (:= 0x02)

BLK_LNG := 2*octets

MM_HDR := 24*octets (see Section 3.4.1)

BLK_CRC16 := 2*octets

3.1. File Header

The FILE_HEADER precedes all DCS message and only occurs once. The File Header consists of several fields as summarized below and detailed in the following subsections.

FILE_NAME	Original HRIT filename
FILE_SIZE	File Length in octets/bytes
SOURCE	Site source code for the file (i.e. WCDA or NSOF)
TYPE	Internal DCS file type (not the same as the LRIT/HRIT file code)
EXP_FIELD	Expansion Field 2 – Reserved for Future Use
HDR_CRC32	32-bit Header CRC check

3.1.1. FILE_NAME := 32*octet

The FILE_NAME field is a 32-byte field containing the DCS message filename as defined in Section 4. The filename is left-justified, and the remainder of the field will be filled with space characters.

3.1.2. FILE_SIZE := 8*octets

An 8-character field with the ASCII representation of the total file size in bytes. Left-justified and space filled.

3.1.3. SOURCE := 4*octets

A 4-character field identifier specifying the source of the HRIT DCS data.

Data sourced from Wallops Command and Data Acquisition station will use the code “WCDA”.

Data sourced from NOAA’s Satellite Operations Facility will use the code “NSOF”.

Future site source designations will be defined as necessary.

3.1.4. TYPE := 4*octets (:= “DCSH”)

A 4-character field identifier defining a DCS file type. This field will use the code word “DCSH”.

Note that this type designation is not the same as, and is in addition to, the LRIT/HRIT header file type, which is a byte value of 0x82.

In the existing LRIT/HRIT file format this field is “DCSD”.

3.1.4.1. EXP_FILL := 12*octets

A 12-character text field reserved for future use. This field is space filled.

3.1.4.2. HDR_CRC32 := 4*octets

The HDR_CRC32 is a 32-bit CRC generated per IEEE RFC 1952 on the first 60 bytes of the FILE_HEADER, i.e. the FILE_NAME thru EXP_FIELD sections. The value is stored in the file as binary values in little-endian format.

See Section 4 for code samples on computing the CRC.

10948 Beaver Dam Road, Suite C
Hunt Valley, Maryland 21030-2237
Tel: 410.771.1070
Fax: 410.771.0018

3.2. BLOCKS

Following the file header, the remainder of the file will include one or more data concatenated BLOCKS; the FILE_CRC32 field (see Section 3.4) is appended after the last data block.

All data BLOCKS will have the following general format:

BLK_ID	Block data type identifier
BLK_LNG	Unsigned integer value of block length (total bytes all fields)
BLK_DATA	Block data (0 to 65,530 bytes)
BLK_CRC16	16-bit CRC on all preceding bytes in the block.

To support future enhancements while maintaining backward compatibility, this proposed HRIT file format utilizes a block identifier (BLK_ID) immediately followed by a block length (BLK_LNG) field. When implementing the file processing code, systems should be designed to look at the block identifier to determine the type of data in the block and the appropriate handling. If the BLK_ID is not a value the code recognizes or supports, the code should then use the length field to skip over this data.

This approach allows future data block types to be defined without negatively impacting deployed systems until the code can be updated to support the new block type.

3.2.1. BLK_ID := 1*octet

The BLK_ID is a single byte field identifying the type of data in the block. Presently the only block type defined is the DCS message block (see Section 3.3).

Future block types may be defined as needed or desired

3.2.2. BLK_LNG := 2*octets

The block length (BLK_LNG) field is a 2-octet unsigned integer value designating the total length of the block in octets (aka bytes). Since the value includes the identifier, the length itself, the variable data, and the 16-bit block CRC; this field can have a value between 5 and 65,535.

Note that a block length of 5 indicates no data is present in the block. Further, the maximum size of the data field is 65,530 octets.

The two byte block length value is provided in little-endian order.

3.2.3. BLK_DATA := 0..65,530*(octets)

This is the variable length data field. The actual data in the block depends on the block type.

3.2.4. BLK_CRC16 := 2*(octets)

The BLK_CRC16 is a 16-bit CRC that can be used to validate the block. This 16-bit CRC is identical to the LRIT/HRIT packet CRC defined by the generator polynomial:

$$G(x) = x^{16} + x^{12} + x^5 + 1$$

The CRC is initialized to “all ones” prior to the CRC calculation. All bytes beginning with the block identifier (BLK_ID) up to and including the last byte in the data field is included in the CRC generation. The value is stored in the file as binary values in little-endian format.

See Section 4 for code samples on computing the CRC.

10948 Beaver Dam Road, Suite C
Hunt Valley, Maryland 21030-2237
Tel: 410.771.1070
Fax: 410.771.0018

3.3. DCP Message Blocks

DCP Message blocks are the primary data blocks in the HRIT DCS file.

DCP Message blocks have a block type identifier of 0x01 and have a variable length based on the DCP message header and DCP data. The DCP Message block consists of the following fields

DCP_ID	DCP Message Block ID (:= 0x01)
BLK_LNG	DCP Message Block Length (see Section 3.2.2)
DCP_HDR	DCP Message Header (see Section 3.3.1)
DCP_DATA	DCP Message Data (data as received from DCP or informational message)
BLK_CRC16	16-bit CRC (see Section 3.2.4)

3.3.1. DCP_HDR

The DCP Message Header is a 36-byte field defined by Table 1.

Table 1: DCP Message Header

Field Name	Bytes	Format
Sequence Number	3	Integer Unsigned
Message Flags/Baud	1	Bit Mapped
Message ARM Flag	1	Bit Mapped
Corrected Address	4	Hexadecimal
Carrier Start	7	BCD
Message End	7	BCD
Signal Strength X10	2	Integer Unsigned
Frequency Offset X10	2	Integer Signed
Phase Noise X100	2	Integer Unsigned
Good Phase X2	1	Integer Unsigned
Channel/Spacecraft	2	Integer Unsigned/Bit Mapped
Source Code	2	ASCII Characters
Source Secondary	2	TBD
Total:	36	

10948 Beaver Dam Road, Suite C
Hunt Valley, Maryland 21030-2237
Tel: 410.771.1070
Fax: 410.771.0018

3.3.1.1. Message Flags/Baud

The Message Flags field is a bit-mapped byte defined utilized as follows:

Table 2: DCP Message Flags

B7	B6	B5	B4	B3	B2	B1	B0
					Data Rate: 001=100; 010=300; 011=1200 000=Undefined; 100-111 Reserved		
				0=>CS1 Platform; 1=>CS2 Platform			
			Message Received with Parity Errors (ASCII or Pseudo-Binary)				
		Reserved for Future					
	Reserved for Future						
Reserved for Future							

Unused or reserved bits will be set to 0.

3.3.1.2. Message ARM Flag

The Abnormal Received Message flag byte is defined in Table 3.

Table 3: DCP Message ARM Flags

B7	B6	B5	B4	B3	B2	B1	B0
							Address Corrected
						Bad Address – Not Correctable	
					Invalid Address – Not in PDT		
				PDT Incomplete			
			Timing Error – Outside Window				
		Unexpected Message					
	Wrong Channel						
Reserved for Future							

3.3.1.3. Corrected Address

This Corrected Address is a 4-byte hexadecimal (binary) field providing the BCH correction of the received Platform Address. If the address is received without errors or is uncorrectable, this field will match the Received Address field.

3.3.1.4. Received Address

This is the unmodified 4-byte hexadecimal (binary) Platform Address or ID as received.

3.3.1.5. Carrier Start

The Carrier Start is a 7-byte BCD numeric field providing the carrier start timestamp of the message. The BCD field format is: YYDDDHHMMSSZZZ where

YY	=	last two digits of the year
DDD	=	Julian day of the year
HH	=	Hour
MM	=	Minute
SS	=	Second
ZZZ	=	Sub-second to millisecond resolution

10948 Beaver Dam Road, Suite C
Hunt Valley, Maryland 21030-2237
Tel: 410.771.1070
Fax: 410.771.0018

3.3.1.6. Message End

The Message End field is a 7-byte BCD numeric field providing the message end timestamp. The BCD field format is the same as the Carrier Start field defined in the previous section.

The Message End is the time when the signal energy was no longer detectable.

3.3.1.7. Signal Strength X10

The Signal Strength field is a 2-byte unsigned integer indicating the received message signal level in dBm EIRP. The field value is the signal level multiplied by 10; i.e. 0.1 dB resolution. The two byte value is provided little-endian.

The range of this value requires 10-bits. When processing the field, the upper six bits should be masked off to allow future use of these bits for other purposes; i.e. the data field should be masked with 0x03FF before processing.

Figure 1 shows the Signal Strength binary format. After masking off the unused bits, the resulting integer value from the least-significant 10-bits must be divided by 10.

B_{15}	B_{14}	B_{13}	B_{12}	B_{11}	B_{10}	B_9	B_8	B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0
Reserved for Future						Signal Strength X10									
0	0	0	0	0	0	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Figure 1: Signal Strength Format

3.3.1.8. Frequency Offset X10

The Frequency Offset field is a 2-byte signed integer indicating the frequency offset from the channel center of the received message. The field value is the frequency offset multiplied by 10; i.e. 0.1 Hz resolution. The two byte value is provided little-endian.

The range of this value requires 14-bits including the two's complement sign bit. When processing the field, the upper six bits should be masked off and the sign bit extended to allow future use of these bits for other purposes; i.e. the data field should be masked with 0x3FFF and sign extending before processing.

Figure 2 shows the Frequency Offset binary format. After masking off the unused bits and sign extending as necessary, the resulting integer value from the least-significant 14-bits must be divided by 10.

B_{15}	B_{14}	B_{13}	B_{12}	B_{11}	B_{10}	B_9	B_8	B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0
Reserved for Future						Frequency Offset X10 (2's complement)									
0	0	S	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Figure 2: Frequency Offset Format

3.3.1.9. Phase Noise X100

The Phase Noise field is a 2-byte unsigned integer indicating the phase noise in degrees RMS of the received message. The field value is the phase noise multiplied by 100; i.e. 0.01 degree RMS resolution. The two byte value is provided little-endian.

The range of this value requires 12-bits. When processing the field, the upper four bits should be masked off to allow future use of these bits for other purposes; i.e. the data field should be masked with 0x0FFF before processing.

Figure 3 shows the Phase Noise binary format. After masking off the unused bits, the resulting integer value from the least-significant 12-bits must be divided by 100.

B_{15}	B_{14}	B_{13}	B_{12}	B_{11}	B_{10}	B_9	B_8	B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0
Reserved for Future				Phase Noise X100											
0	0	0	0	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Figure 3: Phase Noise Format

10948 Beaver Dam Road, Suite C
Hunt Valley, Maryland 21030-2237
Tel: 410.771.1070
Fax: 410.771.0018

3.3.1.10. Good Phase X2

The Good Phase is percentage score indicating the quality of the received message. This is a single byte unsigned integer value with a resolution of 0.5%. Messages received with a Good Phase score of 85% or higher are considered "good". A good phase score between 70% and 85% is considered fair, and below 70% is considered poor.

The percentage scores of 70% and 85% roughly correlate to BER estimates of 10^{-4} and 10^{-6}. In other words, a Good Phase score of 85% or higher indicates the BER is 10^{-6} or better, while a score of 75% or lower indicates a BER of 10^{-4} or worse.

The range of this value requires all 8-bits of the byte field. The 8-bit unsigned integer value must be divided by 2.

3.3.1.11. Channel/Spacecraft

The Channel and field provides the DCS channel the messaged was received on along with an indication of which GOES satellite the message was received from as provided by the ground station.

While generally speaking, the satellite should be the GOES spacecraft the platform is assigned to, there are rare instances this will not be the case. For example, there have been rare occasions where one of the GOES satellites has failed, and the corresponding receive system has utilized the other spacecraft to continue operations.

Figure 4 shows the Channel/Spacecraft binary format. The least-significant 10-bits provide the unsigned integer value for the received channel. Presently the DCS channels range from 1-266 and 301-566.

B_{15}	B_{14}	B_{13}	B_{12}	B_{11}	B_{10}	B_9	B_8	B_7	B_6	B_5	B_4	B_3	B_2	B_1
Spacecraft				Reserved		Channel Number								
See Table 4				0	0	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1

Figure 4: Channel/Spacecraft Format

The four most-significant bits provide the spacecraft or satellite (e.g. E, W, etc.) code as defined in Table 4.

Table 4: Spacecraft Codes

Code	Spacecraft
0000	Unknown
0001	GOES-East or 'E'
0010	GOES-West or 'W'
0011	GOES-Central or 'C'
0100	GOES-Test or 'T'
0101-1111: Reserved for Future	

3.3.1.12. Source Code

The Source Code field is a two-character designation of the DRGS system the messages was received on. Presently, the following source codes have been defined:

Table 5: DCS Source Codes

Code	Site Source/System
UP	NOAA WCDA E/W Prime – Wallops Island, VA
UB	NOAA WCDA E/W Backup – Wallops Island, VA
NP	NOAA NSOF E/W Prime – Suitland, MD
NB	NOAA NSOF E/W Backup – Suitland, MD

10948 Beaver Dam Road, Suite C
Hunt Valley, Maryland 21030-2237
Tel: 410.771.1070
Fax: 410.771.0018

Code	Site Source/System
XE	USGS EDDN East – EROS, Sioux Falls, SD
XW	USGS EDDN West – EROS, Sioux Falls, SD
RE	USACE MVR East – Rock Island, IL
RW	USACE MVR West – Rock Island, IL
d1	NIFC West Unit 1 – Boise, ID
d2	NIFC West Unit 2 – Boise, ID
LE	USACE LRD East – Cincinnati, OH
SF	SFWMD East – West Palm Beach, FL
OW	USACE NOW – Omaha, NE

3.3.1.13. Source Secondary

The Secondary Source code is a field that has been suggested as “value-added” source information. Two octets have been reserved for this field, but the specifics of the field are still to be defined.

Initially these fields will be set to two nulls 0x00.

3.3.2. DCP_DATA := 0..*(octet)

The DCP_DATA field is the variable length message data received from the platform or the additional text information for an informational message.

For a DCP message, the data is provided as received; i.e. NO $ character substitution is performed (e.g. for byte with parity errors).

10948 Beaver Dam Road, Suite C
Hunt Valley, Maryland 21030-2237
Tel: 410.771.1070
Fax: 410.771.0018

3.4. Missed Message Block

Block type 0x02 designates a missed DCP message block.

A Missed Message block is similar to a DCP Message block, but with unused Header fields omitted and there is no actual DCP data following the header. The Missed Message block consists of the following fields:

MM_ID	Missed Message Block ID (:= 0x02)
BLK_LNG	Missed Message Block Length (see Section 3.2.2)
MM_HDR	Missed Message Header (see Section 3.4.1)
BLK_CRC16	16-bit CRC (see Section 3.2.4)

3.4.1. MM_HDR

The Missed Message Header is a 24-byte field defined by Table 6.

Table 6: Missed Message Header

Field Name	Bytes	Format
Sequence Number	3	Integer Unsigned
Message Flags/Baud	1	Bit Mapped
Platform Address	4	Hexadecimal
Window Start	7	BCD
Window End	7	BCD
Channel/Spacecraft	2	Integer Unsigned/Bit Mapped
Total:	24	

3.4.1.1. Missed Message Flags/Baud

The Message Flags field is a bit-mapped byte defined utilized as follows:

Table 7: DCP Missed Message Flags

B7	B6	B5	B4	B3	B2	B1	B0
					Data Rate: 001=100; 010=300; 011=1200 000=Undefined; 100-111 Reserved		
				Not Used			
			Reserved for Future				
		Reserved for Future					
	Reserved for Future						
Reserved for Future							

Note that the Data Rate information will be set based on the database definition for the Platform.

3.4.1.2. Platform Address

This Platform Address is a 4-byte hexadecimal (binary) field providing the Platform Address of the Missed Message.

10948 Beaver Dam Road, Suite C
Hunt Valley, Maryland 21030-2237
Tel: 410.771.1070
Fax: 410.771.0018

3.4.1.3. Window Start

The Window Start is a 7-byte BCD numeric field indicating the start of the self-timed window for the expected, but missed, message. The BCD field format is: YYDDDHHMMSSZZZ where

YY	=	last two digits of the year
DDD	=	Julian day of the year
HH	=	Hour
MM	=	Minute
SS	=	Second
ZZZ	=	Sub-second to millisecond resolution

3.4.1.4. Window End

The Window End field is a 7-byte BCD numeric field indicating the end of the self-timed window for the expected message. The BCD field format is the same as the Window Start field defined in the previous section.

3.4.1.5. Channel/Spacecraft

The Channel/Spacecraft field provides the DCS channel and satellite messaged was expected on. This field follows the same format as a standard DCP Message (see Section 3.3.1.11) but is filled in based on the anticipated values from the DCP database.

3.5. FILE_CRC32 := 4*octets

The FILE_CRC32 is a 32-bit CRC generated per IEEE RFC 1952 on all bytes in the file preceding the FILE_CRC32 field. Note that this includes the FILE_HDR and the contained HDR_CRC32 field in it as well. The value should be stored in the file in little-endian format.

See Section 4 for code samples on how to compute the CRC.

10948 Beaver Dam Road, Suite C
Hunt Valley, Maryland 21030-2237
Tel: 410.771.1070
Fax: 410.771.0018

4. CRC Generation

4.1. CRC32 Generation C Code Example

```
/*  Generate Table of CRCs for all bytes for a fast CRC */
unsigned long crc32_table[256];

void create_crc32_table(void) {
  unsigned long c;
    int n, k;

  for (n = 0; n < 256; n++) {
    c = (unsigned long) n;
    for (k = 0; k < 8; k++) {
      if (c & 1)
        c = 0xEDB88320L ^ (c >> 1);
      else
        c = c >> 1;
    }
    crc_table[n] = c;
  }
}

/*  Update a running crc with the bytes buf[0..len-1] and return
    the updated crc. The crc should be initialized to zero. Pre- and
    post-conditioning (one's complement) is performed within this
    function so it shouldn't be done by the caller. */
unsigned long update_crc(unsigned long crc, unsigned char *buf, int len) {
  unsigned long c = crc ^ 0xFFFFFFFFL;
  int n;

  for (n = 0; n < len; n++)
    c = crc_table[(c ^ buf[n]) & 0xFF] ^ (c >> 8);
  return c ^ 0xFFFFFFFFL;
}
```

4.2. CRC16 Generation C Code Example

TBD