DAMS-NT Network Interface Specification
Version 8.2

April 7, 2020

Table of Contents

List of Tables

1 Overview and Background

This document defines a standard network interface for a demodulator system. Previous versions of this standard were defined by NESDIS (National Environmental Satellite Data Information Service) for use with NOAA's legacy computer system. In NESDIS terminology, the demodulator system is called a Data Acquisition and Monitoring System or DAMS. The latest generation of DAMS units made use of DSP "new technology", hence the name "DAMS-NT".

Since its publication, the DAMS-NT specification has been viewed favorably by several U.S. Government Agencies that make use of the GOES DCS. By standardizing the demodulator interface, an agency is free to freely mix demodulator and computer systems from different vendors.

In 2002, the National Interagency Fire Center (NIFC) contracted to enhance the LRGS (Local Readout Ground System) platform to support the DAMS-NT specification. This was the first operational DAMS-NT implementation. Several districts of the U.S. Army Corps of Engineers are now using the LRGS platform to receive data from DAMS-NT units provided by different manufacturers.

The DAMS-NT interface specification allows a computer system to pull DCP messages in real-time from a demodulator system over a network. Each demodulator system can multiplex a large group of demodulators (up to one thousand).

The specification covers four separate interfaces:

1. DCP Message Interface: provides a mechanism for a computer to retrieve DCP messages in real-time.
2. Event Interface: provides a mechanism for the DAMS-NT to report unsolicited changes in status (i.e. "events"). Each event is assigned a priority. This allows computer system to offer a display of alert messages to the user.
3. Real-Time Status Interface: Provides a mechanism for the computer to poll the DAMS-NT for its current status.
4. Configuration Interface: Provides a mechanism for the computer to remotely control some of the operational parameters from the DAMS-NT. In particular, the computer can control demodulator-channel assignments.

All of the interfaces are TCP socket interfaces. In most cases data is transmitted in ASCII. Section 1.2 provides details on connections and disconnections. The remaining sections provide details on each functional interface.

1.1 RFC 2119

RFC (Request for Comments) 2119 specifies keywords recommended for use in writing an ICD document.. The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in IETF RFC 2119. This ICD can be found at:

http://www.ietf.org/rfc/rfc2119.txt?number=2119

1.2 BNF Notation

This document uses BNF (Backus Naur Form) to define the syntax of messages sent between client and server. The following conventions are used:

Notation	***Meaning***
`::=`	Is defined as
`'literal'`	A literal string is enclosed in quotation marks
`nonterminal`	Non-terminal symbols are not enclosed in quotation marks.
`one | two`	Pipe symbol means 'or'. This rule means "one or two".
`{ rule }`	Curly brackets mean zero or more repetitions of the enclosed rule.
`[ optional ]`	Rules in square brackets are optional.
`DIGIT`	Any ASCII digit 0 through 9
`LETTER`	Any ASCII upper or lower case letter
`CRLF`	ASCII Carriage Return followed by Line Feed
`SP`	ASCII Space Character
`STRING`	Any sequence of printable ASCII characters except CRLF. May contain space or tab characters.
`OCTET_STRING`	Any sequence of 8-bit binary octet values. May be ASCII or binary.
`( group of symbols )`	Parentheses used for grouping within rules.
`# comment`	Characters after an un-quoted pound sign are comments.

Table 1-1: BNF Conventions Used in this Document.

All four interfaces are designed in such a way to be easily tested with Telnet:

- All requests are composed of ASCII characters and terminated with CRLF.
- For all services except the DCP Message Interface, responses are also composed of ASCII characters.

Some requests and responses contain a time stamp. All time stamps MUST be in UTC and SHALL be formatted as follows:

```
time ::= YYDDDHHMMSS
```

- 'YY' is the last two digits of the year.
- 'DDD' is the Julian day of the year (January 1 == day 1)
- 'HHMMSS' is the UTC hour, minute, and second of the day.

Integers are made up of at least one digit:

```
integer ::= DIGIT { DIGIT }
```

Hex numbers are represented by <hexstring>:

```
hexstring ::= hexdigit { hexdigit }
hexdigit :: DIGIT |
            'a' | 'b' | 'c' | 'd' | 'e' | 'f' |
            'A' | 'B' | 'C' | 'D' | 'E' | 'F' |
```

1.3 Document History

This ICD has gone through several revisions. The following subsections provide a list of changes in reverse chronological order.

This document was originally prepared by ILEX Engineering, Inc. under a contract with NOAA/.

The modifications made to Version 8.0 to produce 8.1 were made by Ernest Dreyer of the United States Geological Survey.

The modifications made to Version 8.1 to produce 8.2 were made by Brett H. Betsill of Microcom Design, Inc. at the direction of NOAA/NESDIS under the DCS Sustaining Engineering contract.

1.3.1 Differences between Version 8.1 and 8.2

1. Added “List of Tables”.
2. For use with non-demodulator receive systems, added provision to only support the DCP Message Interface socket (See section 2).
3. Made the Real-Time Status and Configuration Interfaces optional.
4. Added optional Extended Statistics (extendedstats) option (See section 3). An additional flag was defined in the 'errorFlags' value of the message header to indicate that the optional Extended Statistics times that may appear after the header or carrier times are in fact present. (See section 3.2).
5. For use with non-demodulator receive systems, added provision for alternate Slot Number use (See section 3.1)
6. Added provision for Missed Message Blocks (See section 3.4). As a consequence of this addition, the DCP Message start pattern must not be set to “MM”/CR/LF (See section 3.5), and is recommended to remain “SM/CR/LF”.
7. Added startPattern note in Configuration Interface (See section 6).
8. Modified assign command in Configuration Interface to address Certification Standard 2 (CS2) requirements. Modified Section 6, and added/enhanced sections 6.1 through 6.4.

9. Replaced “Mapping 1200 Baud Channel Numbers to Frequencies” section with “6.3.1 CS1/DUAL 1200 Baud Channel Numbers”.

1.3.2 Differences between Version 8.0 and 8.1

1. An additional flag was defined in the 'errorFlags' value of the message header to indicate that the optional times that may appear after the header are in fact present. (See section 3.2).

2. Specified that the 'startPattern', the 4-byte value that indicates where the data begins after a header, must not appear anywhere in the section reserved for vendor-specific information. (See Table 3-3 and section 3.3)

3. Removed the requirement to replace “prohibited” characters, as defined by a table in Version 8.0, with a '$' character. There are no longer any prohibited characters.

1.3.3 Differences between Version 7 and Version 8

1. The table in Section 2 defining the socket-level interface ports was simplified. Since ports are required to be configurable, there is no need to define conventions for east and west satellites.
2. Language about specific systems have been replaced by the more generic term “computer system”.
3. Section 5 addition of the EquipId status command. This allows a computer system to discover the manufacturer and model number of the DAMS-NT equipment.
4. Addition of feature whereby DAMS-NT can include carrier start/drop times with millisecond resolution for every DCP message. This feature is enabled by a new configuration command defined in Table 6-10. The format of the time fields is defined in the BNF in section 3.
5. Addition of vendor-specific data on the message socket. See section 3.3.
6. A new configuration command “paritycheck” controls the behavior of DAMS-NT for erroneous and “prohibited” characters. Section has been added defining the prohibited characters. Table 6-10 defines the “paritycheck” command. Section 3 defines the DAMS-NT behavior when parity checking is enabled or disabled.

1.3.4 Differences between Version 6 and Version 7

The following substantive modifications have been made to this specification from Version 6 to Version 7:

1. The table in Section 2 defining the socket-level interface ports has been modified to accommodate GOES-East and GOES-West satellites.
2. Some minor changes have been made to Table 2, to clarify the startTime and errorFlags definitions in the DCP message header.

3. A new requirement has been added to Section 3, to replace ASCII or pseudo-binary message bytes with a '$' symbol, when a parity error or "prohibited" character is detected by the DAMS.
4. Minor changes have been made to clarify the meaning of error / message bits in the message header.

1.3.5 Differences between Version 5 and Version 6

The following substantive modifications have been made to this specification from Version 5 to Version 6:

1. Section 4, table 3, priority 0 removed. If there is no event, the server MUST respond with the NONE response, as shown in the BNF.
2. Section 3, table 2, offset 32 (errorFlags). Vague wording in the table description removed. Refer to the new section 3.2 for an exact description of the errorFlag values.
3. The 'GetFaults' operation has been added to the Real Time Status interface. This is now included in the BNF in section 5. Section 5.2 has been added to describe the details of this operation.

1.3.6 Differences between Version 4 and Version 5

The following substantive modifications have been made to this specification from Version 4 to Version 5:

1. The protocol is now specified in BNF. Some of the old tables have been retained for clarity.
2. Event Interface 'Poll' message now includes the letter 'P'. In the previous version, a poll was a simple linefeed character, meaning that a poll was a subset of a max-priority command, increasing the possibility of confusion between client and server.
3. All command and response lines are now terminated with an ASCII CRLF sequence rather than a simple Linefeed.
4. The message interface is now defined as a one-way real-time message stream over a socket. The previous mechanism for client-polling has been removed.

2 Socket-Level Interface

The DAMS-NT SHOULD provide servers that listen for connections on four TCP ports. Table 2-2 shows default port assignments for each interface. The DAMS-NT MUST be capable of configuring these port numbers through an external mechanism. Computer systems accessing the DAMS-NT MUST provide a mechanism to specify non-default ports.

Interface	***TCP Port***
DCP Message Interface	17010
Event Interface	17011
Real-Time Status Interface	17012
Configuration Interface	17013

Table 2-2: Default Port Assignments.

Each interface MUST be capable of supporting multiple simultaneous client connections. A common scenario would be for one computer system to run real-time software to retrieve data. Another computer might connect to the same DAMS-NT to view the message flow for trouble-shooting.

A new bi-directional application connection (socket) is be created by the server when a client connects to one of the above ports. The server SHOULD close this bi-directional socket when it detects an I/O error indicating that the client has disconnected.

The servers SHOULD be implemented in a multi-threaded fashion so that each connection is independent. A client that is slow to read data MUST NOT affect other clients.

The interfaces are designed so that they can be easily tested with available tools such as Telnet.

For use of the DAMS-NT Network Interface on non-demodulator receivers, e.g. an HRIT receiver, it is permissible to only support the DCP Message Interface socket.

At a minimum, the interface MUST support the DCP Message Interface. For a demodulator based interface, the system SHALL also support the Event Interface. The Real-Time Status and Configuration Interfaces are options; however, if these interfaces are implemented, they MUST conform to this specification.

3 DCP Message Interface

The DCP Message Interface provides a mechanism for multiple clients to receive DCP messages from the DAMS-NT unit.

Multiple independent client connections MUST be supported by the DAMS-NT. The DAMS-NT MUST maintain a context for each client such that each client is guaranteed to get the entire message stream. The DAMS-NT SHOULD buffer approximately one hour's worth of DCP messages to accommodate slow clients.

The DCP Message Interface is a one-way stream of DCP messages. After a client establishes a connection, the server sends new messages as they become available. The first message sent to a client must be complete.

The server continually sends responses to the client as follows:

```
response ::= none | dcpmsg
none ::= 'NONE' CRLF
dcpmsg ::= header data CRLF [carriertimes] [extendedstats]
header ::= #(55 or 68 character sequence as defined below)
data ::= OCTET_STRING

carriertimes ::= carrierstart SP carrierdrop CRLF
carrierstart ::= YYDDDHHMMSSmmm
carrierdrop ::= YYMMDDHHMMSSmmm

extendedstats ::= slvl SP phns SP gdph SP freq SP type [SP armf] CRLF
slvl ::= { DIGIT } DIGIT '.' DIGIT
phns ::= { DIGIT } DIGIT '.' DIGIT
gdph ::= { DIGIT { DIGIT } } DIGIT '.' DIGIT
freq ::= SIGN { DIGIT { DIGIT } } DIGIT '.' DIGIT
SIGN ::= '+' | '-'
type ::= '0' | '1' | '2'
armf ::= HEXCHR HEXCHR
```

When the DAMS-NT acquires a new DCP message it MUST send a 'dcpmsg' to each currently-connected client.

If the DAMS-NT has no new DCP messages, and it has been more than 10 seconds since the last message sent to a client, the server MUST send a 'none' message. The client is then guaranteed that it will receive timely responses, and can thus detect the difference between a dead socket and the case where no new data is available.

Note that DCP message data can (and frequently do) contain embedded and terminating CRLF patterns. The CRLF that terminates the response MUST be in addition to all message octets.

A new feature in this specification version 8 allows for the DAMS-NT to include carrier start/drop times immediately after the message data. This feature is enabled or disabled with the "carriertime" configuration command defined in Table 6-10. The time format is the same as the message start time in the header, but with 3 added digits for milliseconds. That is: YYDDDHHMMSSmmm.

If enabled, carrier times MUST immediately follow the CRLF that terminates the message-proper. It MUST be in the exact format shown. That is, carrier-start followed by a single space, followed by carrier-drop followed by CRLF.

Offset	*Name*	*Length*	*Format*	*Description*
0	startPattern	4	ASCII byte values	The start pattern can be provided by a host Client via the configuration command. The default start pattern is the.ASCII Characters 'SM' followed by CR & LF. See additional notes in Section 3.5 below.
4	slotNum	3	digits – zero filled	See discussion below on slot numbers and channels
7	channel	3	digits – zero filled	DCS channel message was received from.
10	spacecraft	1	Character	'E' or 'W': Other values may be implemented in the future.
11	Baud	4	Digits – zero filled	'0100', '0300', '1200'
15	startTime	11	YYDDDHHMMSS	UTC Time of message start (i.e. frame synch)
26	signalStrength	2	digits	Signal strength in dB
28	freqOffset	2	+/- 1 digit	Sign character followed by 1 digit. In units of 50Hz
30	modulationIndex	1	'N', 'H', or 'L'	N=normal, H=high, L=low
31	dataQuality	1	'N', 'F', or 'P'	N=normal, F=fair, P=poor
32	errorFlags	2	Character	2 Characters representing error and message flags. See Section 3.2 below.
34	origAddress	8	Hex Characters	Original DCP Address Received from Platform
42	dcpAddress	8	Hex Characters	BCH corrected address. This field is always set, even if there were no errors.
50	length	5	Digits- zero filled	Number of message bytes to follow

Table 3-3: 55-Character DCP Message Header Format.

A new feature in this specification (Version 8.2) allows for the DAMS-NT to include extended message statistics immediately after the message data, and following the "carriertimes" field if it is included. If the extended message stats are appended to the message, bit 5 (0x20) of the "errorFlags" field MUST be set to 1; otherwise this bit MUST be cleared to 0. The "extendedstats" field MUST include the following five fields, in order, and space separated:

- Signal Level ("slvl"): Signal Strength in dB. This is the same parameter as the "signalStrength" field in the header, but with a resolution of 0.1 dB.
- Phase Noise ("phns"): Phase Noise of the received message in degrees RMS to a resolution of 0.1 degrees.
- Good Phase Percentage ("gdph"): A percentage score with a resolution of 0.1% computed as 100 times the "good phase" symbols divided by the total symbols. For a HDR message, a "good phase" symbol is one that is received within ±8.4376º of the octal phase points. For 100 bps messages, "good phase" symbol is one that is received within

±15º of the ±60º phase points. The "dataQuality" character in the header is determined from this percentage as follows:

	N	F	P
HDR:	GP >= 85%	> GP >= 70%	> GP
100:	GP >= 65%	> GP >= 55%	> GP

- Frequency (freq): Frequency offset from channel center to 0.1 Hz resolution.
- Message Type (type): 0 for 100 bps, 1 for Version 1 (aka CS1) HDR transmission, or 2 for Version 2 (aka CS2) HDR transmission.
- ARM Flags (armf): Hexadecimal representation of Abnormal Received Message (ARM) flags for message as defined below. This is the only optional field in the Extended Stats. If thi information is not known to the DAMS-NT Server it MUST be omitted and the CRLF must immediately follow the type field.

Bit Value	***Meaning***
0x01	Address Corrected
0x02	Bad Address – Not Correctable
0x04	Invalid Address – Not in PDT
0x08	PDT Incomplete
0x10	Timing Error – Outside Window
0x20	Unexpected Message
0x40	Wrong Channel
0x80	Reserved for Future

Table 3-4: ARM Flag Bit Defines

Note that with the possible exception of 'startPattern' All header fields are in ASCII with no parity bits. The normal operating mode for DAMS-NT will be to do NO parity checking and NO character substitution. That is, message data MUST be transmitted exactly as received, with parity bits (if applicable) intact.

Version 8 of this spec provides a new configuration command "paritycheck" to enable/disable parity checking. On ASCII messages, if and only if parity checking is enabled, the DAMS-NT shall do the following:

- Replace characters that fail the parity check with a '$'.

Note that for HDR platforms, the first byte of data transmitted by the platform immediately following the DCP address is a special flag value. This byte MUST show up as the first byte of message data and MUST be included in the 'length' field of the header.

3.1 Slot Numbers and Channels

A DAMS-NT demodulator unit supports multiple 'slots'. Each slot refers to a single demodulator.

The configuration interface (described in section 6) provides a mechanism for an external computer system to control the channel, spacecraft, and baud-rate assignments for each slot.

Slots are numbered from 0 … 999. A DAMS-NT unit MAY have non-contiguous ranges of slots.

For trouble-shooting and load-testing scenarios, the DAMS-NT MUST support the assignment of more than one slot to the same channel.

For use of the DAMS-NT DCP Message Interface on non-demodulator receivers, e.g. and HRIT receiver, the slot number SHOULD be filled with either '000' or some other relevant numerical value and specified in the unit's User Manual. In the case of an HRIT receiver, it is suggested that the Slot Number field be filled with the Virtual Channel the DCS files is received on.

3.2 Error / Message Flags

The 'errorFlags' value in the message header contains two hex digits of message-specific bits and/or error flags. This is capable of representing a combination of 8 binary values, where a bit set to one means the corresponding attribute is true. Table 3-5 contains the defined error flag values that MUST be implemented by the DAMS.

Bit Value	*Meaning*
0x01	Message contains parity errors (for ASCII or pseudo-Binary messages only)
0x02	Binary message (default = ASCII)
0x04	Binary message with bit errors (reserved for future use).
0x08	Loss of lock termination (i.e., no EOT)
0x10	Message contains additional message times
0x20	Message contains extended quality statistics

Table 3-5: Error / Message Flag Values

3.3 Vendor-Specific Additional Data

This extension is added for the DAMS-NT specification version 8.

DAMS-NT manufacturers MAY add additional data after a DCP message as defined above and before the next start pattern. This allows a manufacturer to include additional status information about the message demodulation process, etc.

For obvious reasons, this "extension" data must NOT contain in any place the 4-byte 'startPattern' (see Table 3-3) that is used to indicate the start of DCP message data.

The format of the "extension" data is to be determined by the manufacturer and MAY be proprietary. However, extension data is intended for increased visibility into the operation of the demodulator system, not for interpretation of message data. The manufacturer MUST NOT place information necessary to interpret DCP messages into the extension area.

Computer systems receiving data from the DAMS-NT should be programmed to ignore any unrecognized extension data. That is, any data after the CRLF sequence that terminates a message, but before the start-pattern for the next.

The "EquipId" status command (see section 5) may be used to determine the DAMS-NT manufacturer and model number. A computer system can use this command to determine the expected format for any vendor-specific additional data.

3.4 Missed Message Block

For use of the DAMS-NT Network Interface on non-demodulator receivers, e.g. an HRIT receiver, or for a DRGS that has a database capability to generate Missing Message notifications, this extension to identify a missed expected message is added for the DAMS-NT specification version 8.2.

For a Missed Message Block, only a header field is supplied, I.e. there is not equivalent data filed since no message was actually received. Further, while the general structure of the block is similar to the DCP Message Header, several key differences should be noted.

- The start pattern is defined as "MM" followed by a CR and LF to distinguish it from the default start pattern, "SM"/CR/LF. used in the DCP Message Header.
- Start time is replaced with the beginning of the DCP window and an End Window has been added. Both include the sub-seconds similar to the Carrier Times.
- All message performance parameters have been removed.
- No message length field is included.

Offset	*Name*	*Length*	*Format*	*Description*
0	startPattern	4	ASCII byte values	ASCII Characters 'MM' followed by CR & LF
4	slotNum	3	digits – zero filled	See discussion in 3.1on slot numbers.
7	channel	3	digits – zero filled	DCS channel message was expected on**.**
10	spacecraft	1	Character	Spacecraft message was expected to come through (e.g. 'E' or 'W').
11	baud	4	Digits – zero filled	Expected Baud as defined by NOAA PDT database. ('0100', '0300', '1200').
15	windowTime	14	YYDDDHHMMSSZZZ	UTC Time of DCP window start
29	windowEnd	14	YYDDDHHMMSSZZZ	UTC Time of DCP window end
43	dcpAddress	8	Hex Characters	DCP Address.

Table 3-6: 51-Character DCP Missed Message Block Format.

3.5 Start Pattern

While the start pattern defined in Table 3-3 for standard DCP messages can be user defined, it is recommended that it remain as the default, "SM"/CR/LF (0x53, 0x4D, 0x0D, 0x0A) for the following two reasons:

- To provide a consist approach to ensuring the start pattern does not exist in the Vendor Specific data as specified in Section 3.4.
- To reliably distinguish it from the Missed Message block start pattern ("MM"/CR/LF) defined Section 3.4.

If the DAMS-NT Server does not support a definable startPattern, the start pattern standard DCP messages for MUST be (0x53, 0x4D, 0x0D, 0x0A).

4 Event Interface

The event interface is used to transmit unsolicited status changes from DAMS-NT to clients in the form of ASCII text messages.

After establishing a connection, the client sends one of two request types to the server:

```
request ::= poll | MaxPriority
poll ::= ( 'P' | 'p' ) CRLF
MaxPriority ::= DIGIT CRLF
```

The MaxPriority request is used by the client to tell the server the 'verbosity' of events it wishes to receive.

The DAMS-NT manufacturer is free to determine what constitutes an event, and to assign priorities to each event type. However, the general indications defined in Table 4-7 SHOULD be observed. In particular, priority numbers 5 and higher are to be considered debugging messages used only for tracing and trouble-shooting. Priority numbers 4 and lower are operational messages that should be brought to the attention of the system users during normal operations.

Priority	***Meaning***
1	(Highest Priority) Indicates catastrophic failure that renders this DAMS-NT unit unusable.
2	ERROR – Internally-detected error that may render one or more channels unusable.
3	WARNING – Correctable or transient anomalies
4	INFORMATIONAL – Noteworthy events that are not necessarily anomalies
5	DEBUG Level 1 (least voluminous)
6	DEBUG Level 2
7	DEBUG Level 3
8	DEBUG Level 4
9	DEBUG Level 5 (most voluminous, e.g. trace)

Table 4-7: Priority Levels for Event Messages.

For example, if the client requests a "maximum priority" of 4, then the server SHOULD NOT send any events with a priority number greater than this. This will be common for clients that only want to see operational (rather than debug) events.

After receiving a MaxPriority request, the server MUST respond by echoing the new maximum priority value:

```
MaxPriorityResponse ::= DIGIT CRLF
```

The ‘poll’ request is used by the client to retrieve the next event message. The server MUST maintain a queue of a reasonable size to accommodate slow clients. Upon receiving a ‘poll’ the server responds as follows:

```
PollResponse ::= none | event
none ::= ‘NONE’ CRLF
event ::= priority SP time SP [ eventnum SP] text CRLF
priority ::= DIGIT
eventnum :: {DIGIT}

# time ::= YYDDDHHMMSS UTC time to seconds resolution
text ::= STRING # ASCII no more than 80 chars in length
```

If there are no new events with an appropriate priority (i.e. events not already delivered to this client), the server MUST send the ‘none’ response. Upon receiving this response, the client SHOULD wait a brief period (e.g. 100 milliseconds) before polling again.

If there is a new event with an appropriate priority, the server MUST send it to the client in the format shown.

The Event Server MUST support multiple independent client connections. The DAMS MUST maintain a context for each client such that each client is guaranteed to have access to the entire event stream. Each client may set a different max priority and thus see different subsets of the event stream.

If the Event Server does not understand the query (i.e. it is not a MaxPriority or a Poll command), it MUST respond with an error:

```
error ::= ‘ERROR’ [ STRING ] CRLF
```

The OPTIONAL STRING component of the response may provide additional information about the nature of the error, but the length of the entire response SHOULD NOT exceed 80 characters.

5 Real-Time Status Interface

The Real-Time Status Interface provides a mechanism for a client to poll the DAMS for its current status.

```
request ::= BusyBits | CurrentTime | LastMsgTime | GetFaults |
                EquipId
BusyBits ::= ( 'B' | 'b' ) CRLF
CurrentTime ::= ( 'T' | 't' ) CRLF
LastMsgTime ::= ( 'L' | 'l' ) CRLF
GetFaults ::= ( 'F' | 'f' ) CRLF
EquipId :: = ('E' | 'e' ) CRLF
```

If the Real-Time Status Interface is implemented, the DAMS supplier MUST support the above-defined operations. The DAMS supplier MAY supplement this interface by defining additional request types. If the DAMS supplier chooses to do so, it MUST follow the following conventions for single and multi-line responses.

The response may be single line or multi-line:

```
response ::= SingleResponse | MultResponse | ErrorResponse
SingleResponse ::= STRING CRLF
MultResponse ::= { STRING CRLF } "OK" CRLF
ErrorResponse ::= 'ERROR' sp STRING CRLF
```

Single-line responses MUST be a single ASCII line of text followed by CRLF.

For multi-line responses (e.g. busy-bits), each line MUST be terminated by CRLF. After all response lines have been sent, a single line containing "OK" followed by CRLF shall be sent.

Table 5-8 describes the status commands that the server is REQUIRED to implement.

Command Character	***Meaning***	***Response***
B	BusyBits	Multiple lines, each containing a start-slot followed by a string of hex characters, representing one bit per slot (see below)
T	CurrentTime	Current internal time in the following format: YYDDDHHMMSS
L	LastMsgTime	End time for the last received message: YYDDDHHMMSS
F	GetFaults	Single line containing abbreviations of faults that are currently asserted by the DAMS, or the string "NONE" if now faults are currently asserted.
E	EquipId	Single line containing manufacturer name and model number of the DAMS-NT equipment.

Table 5-8: Commands for Status Interface

5.1 The BusyBits Query

The client has a need to monitor the busy/idle status of each slot in real time. The response to the busy-bits query is a hex representation of a binary value that contains one bit per slot.

The response to the BusyBits query is a multi-line response. Each line will be formatted as follows:

```
BusyBitsResponse ::= { Range } "OK" CRLF
Range ::= slotnum [ SP ] ':' [ SP ] hexstring
slotnum ::= integer      # Slot number in range 0...996
```

As shown, the response can contain multiple lines. Each line contains the busy values for a contiguous range of slots. After all lines, a single line containing "OK" followed by CRLF MUST be sent.

The *slotnum* indicates the first slot number represented by the bits in *hexstring.* The value of *slotnum* must be in the range 0…996 and MUST be an integer multiple of 4.

Following *slotnum* and the colon-delimiter is a string of hex digits. Each digit represents the busy status for 4 demodulator slots. The first hex digit is for slots *slotnum* through *slotnum*+3. The second hex digit is for slots *slotnum*+4 through *slotnum*+7, etc.

Each hex digit represents a 4-bit binary value (i.e. a 'nibble'). The 4 bits contains busy indication for 4 demodulator slots. Within a 4-bit nibble, the LSB is the first slot ant eh MSB is the last slot. Table 5-9 shows an example for the digit numbers and masks for the first 5 slots, assuming that 'slotnum' = 0.

Slot #	Hex Digit #	Mask
0	0	0x01
1	0	0x02
2	0	0x04
3	0	0x08
4	1	0x01
Etc.		

Table 5-9: Busy Bits example where start-slot = 0.

A channel is considered busy from the time carrier is detected until the DAMS detects that all message data has been received.

5.2 The GetFaults Query

A client can query the DAMS-NT for any faults that are currently asserted. A fault SHOULD only be asserted if one or more of the currently assigned slots is inoperable, or if the DAMS as a whole is inoperable. Hence the faults provide the client with a way of determining if the DAMS is currently usable or not; and if it is not, they provide terse abbreviations as a diagnostic aid.

```
GetFaultsResponse ::= ( Faults | "NONE" ) CRLF
Faults ::= WORD { SP WORD }
WORD ::= LETTER { LETTER | DIGIT | '-' }
```

The response contains space-delimited words. Each word MUST begin with a letter and may contain only letters, digits and hyphens.

Each word SHOULD be an abbreviation representing a specific fault. Fault abbreviations may be defined by the DAMS vendor but MUST be documented, along with trouble-shooting procedures to be followed when a given fault is asserted.

If the DAMS if operating normally and no faults are currently asserted, the DAMS MUST respond with the word "NONE" followed by CRLF.

6 Configuration Interface

The Configuration Interface provides a mechanism for a client to control the DAMS global configuration.

```
request ::= startpattern + assign | dump | carriertime | paritycheck
startpattern ::= 'startpattern' hexstring CRLF
assign ::= 'assign' slotnum chan [baud mode [manu]] CRLF
chan ::= DIGIT { DIGIT }
baud ::= ( '100' | '300' | '1200' [ | 'AUTO1' | 'AUTO2' ] )
mode ::= ( 'CS1' | 'CS2' [ | 'DUAL' ] )
manu ::= manufacturer specific configuration settings
dump ::= 'dump' CRLF
carriertime ::= 'carriertime' ( 'on' | 'off')
paritycheck ::= 'paritycheck' ( 'on' | 'off')
```

If the Configuration Interface is implemented, the DAMS supplier MUST support the above-defined operations. The DAMS supplier MAY supplement this interface by defining additional request types. If the DAMS supplier chooses to do so, it MUST follow the following conventions for single and multi-line responses.

The response may be single line or multi-line:

```
response ::= SingleResponse | MultResponse | ErrorResponse
SingleResponse ::= STRING CRLF
MultResponse ::= { STRING CRLF } "OK" CRLF
ErrorResponse ::= 'ERROR' SP STRING CRLF
```

The startpattern command gives the DAMS the 4-byte value that is used by the Message Server to delimit the start of a new DCP message. The command contains an 8-character hex string. The 8 hex digits represent a 4-byte binary value. After receiving this value, the DAMS DCP Message Interface MUST use this value to delimit the start of all new messages.

NOTE: The startpattern definition only applies to standard DCP messages as defined in Table 3-3. The Missed Message startPattern is fixed and is defined in Table 3-6.

If the command is not successful, the response should start with the word "ERROR" followed by a description of the problem (all on one line), followed by CR/LF. The total length of the response line MUST be 80 characters or less.

Table 6-10 contains the commands that a DAMS-NT supplier SHOULD implement. When a DAMS-NT receives that it does not support, it MUST respond with an error.

Command	*Meaning*
startpattern *8-hex-digits*	Sets the 4-byte start pattern used by the DCP message interface.
assign *slot chan baud mode manu*	This command makes a slot assignment. Slot and Channel are numeric, and further defined in Sections 6.1 and 6.3. Baud is one of the keywords as specified above and defined in Section 6.4. Mode is one of the keywords as specified above and defined in Section 6.2. A special case of assigning a slot to channel 0 means to clear the slot assignment (i.e. disable this demodulator). When the channel is 0, the baud, mode, and manu fields MUST be omitted. Following the mode specification keyword, additional manufacturer specific configuration settings may be included, but are not required. If the DAMS-NT unit does recognize any of the information included after the mode setting, the DAMS-NT MUST respond with an error.
Dump	This command causes the DAMS to echo its complete current configuration back to the client, formatted as a series of configuration commands. A line with the word "OK" indicates the end of the configuration dump.
carriertime (on | off)	This command causes the DAMS-NT to either include or exclude the extended carrier start & stop times as defined in section 3). "on" means to include carrier times, "off" means to exclude them. If carrier times are not supported, the DAMS-NT MUST respond with an error.
paritycheck (on | off)	This command causes the DAMS-NT to check parity on ASCII message bytes. Characters that fail the parity check are to be replaced by '$'. It also causes the DAMS-NT to replace "prohibited characters" with '$'.

Table 6-10: Required Commands for Configuration Interface.

When "assign" commands are successfully executed, the "OK" response should be returned immediately. The action SHOULD be performed by the DAMS asynchronously. This will enable the client to send many assign commands back-to-back without waiting for each one to take effect.

```
OK_RESPONSE ::= "OK" CRLF
```

If problems are subsequently encountered in executing an "assign" command, the DAMS SHOULD report the problem via the events interface.

When the DAMS unit is powered-up, it SHOULD revert to the last known configuration that was in effect.

NOTE: Prior revisions of this protocol included a *spacecraft* (sc) field in the assign command. This field has been removed in this revision as the satellite a DCP message is received via cannot be programmatically specified. The determination of the GOES spacecraft that the message passed through is based on which satellite the receiving antenna is pointed at. DAMS-NT systems should provide an independent mechanism to make this determination.

6.1 Demodulator Slots

Demodulator slots are numbered from 0 to 999. A DAMS unit MAY support non-contiguous ranges of slots. For example, a unit may have slots 0…31, and 128…195, but may be missing 32…127.

The "dump" command MUST list all slots supported by the DAMS, even if no assignment is currently made to that slot. Slots with no assignment are indicated by a channel value of 0.

6.2 Mode Designations

The demodulator slot mode is used to specify the operational characteristics of the assignment as defined below:

CS1	The slot is to be configured for the legacy Certification Standard 1 operation (see Sections 6.3 and 6.4 for channel and baud rate options).
CS2	The slot is to be configured for Certification Standard 2 operation (see Sections 6.3 and 6.4 for channel and baud rate options).
DUAL	The slot is to be configured for Dual CS1 and CS2 operation. Since DUAL mode operation allows both CS1 and CS2 operation, the channel and baud rate options must conform to the subset of the allowed options that are compatible with each mode (see Sections 6.3 and 6.4).

To be V8.2 compliant, the DAMS-NT MUST support CS1, and CS2 operation. Support for DUAL operation is optional. If the DAMS-NT does not support DUAL operation, the DAMS-NT MUST respond with an error if DUAL mode operation is requested.

6.3 Channel Numbers

Channel numbers are numeric values. With the adoption of CS2, the channel capacity of the GOES DCS was essentially doubled. As such, the valid channel numbers are a function of the Mode or certification setting.

Further, CS1 1200 baud channels utilized a different numbering designation that specified different operation frequencies. For CS2, 1200 baud channels use the new channel numbering convention that does align with CS2 300 baud numbers and frequencies.

Provided below are the allowed channel numbers based on both mode and baud options. Note that for DUAL mode operation, the allowed

Mode	***Baud***	***Allowed Channel Numbers***
CS1	100	1, 2, 3, … 266
CS1	300	1, 2, 3, … 266
CS1	1200	1, 4, 5, 8, 9, 10 … 129, 130, 131, 132, 133 (see Section 6.3.1)
CS1	AUTO1	1, 2, 3, … 266
CS1	AUTO2	1, 2, 3, … 266
CS2	300	1, 2, 3, … 266, 301, 302, 303, … 566
CS2	1200	3, 6, 9, … 264, 301, 304, 307, … 565 (see Section 6.3.2)
DUAL	300	1, 2, 3, … 266
DUAL	1200	1, 4, 5, 8, 9, 10 … 129, 130, 131, 132, 133 (see Section 6.3.1)

Table 6-11: Allowable Channel Number by Mode and Baud

6.3.1 CS1/DUAL 1200 Baud Channel Numbers

For legacy reasons, the original CS1 1200 baud channel numbers followed two different conventions. DCP transmitters utilized an alternate or 'A' channel numbering scheme with channel numbers from 1A to 133A. However, the legacy reception equipment could not accept these designations, and the closest CS1 300 baud channel number was utilized that would maintain an odd/even approach.

This was done since CS1 1200 baud channels were twice as wide as the CS1 100 or 300 baud channels. Specifically, CS1 100 and 300 baud channels are 1.5 kHz wide, and CS1 1200 baud channels are 3 kHz wide with the center frequency being halfway in-between the center frequencies for two adjacent 1.5 kHz channels.

Earlier versions of this protocol specified the use of the 100/300 channel designations instead of the alternate 1200 designation, and this convention is carried forward in this revision for CS1 and DUAL mode operation..

Use the following algorithm to translate from an alternate 1200 channel designation to the equivalent 300 channel designation:

```
If ('A' channel number (C_A) is odd)
   The equivalent channel number (C) is C = 2*C_A - 1
Else
   The equivalent channel number (C) is C = 2*C_A
```

Examples:

Alternate 1200 baud channel # 91A is equivalent channel 181 (181 = (2 * 91) - 1).

Alternate 1200 baud channel # 92A is equivalent channel 184 (184 = 2 * 92).

Use the following algorithm to translate from the equivalent 300 channel designation to alternate or 'A' 1200 channel designation:

```
If (Equivalent channel number (C) is odd)
   The 'A' channel number (C_A) is C_A = (C + 1) / 2
Else
   The 'A' channel number (C_A) is C_A = C / 2
```

Examples:

Equivalent 1200 baud channel # 181 is channel 91A (91 = (181 + 1) / 2).

Equivalent 1200 baud channel # 184 is channel 92A (92 = 184 / 2).

NOTE: The above algorithm does not address invalid equivalent 1200 channel designations; i.e. channels 2, 3, 6, 7, … 262, 263, 266 are not valid equivalent CS1 1200 channel designation and do not correlate to a proper 'A' channel.

6.3.2 CS2 300/1200 Baud Channel Numbers

CS2 1200 channel numbers align with the frequency equivalent CS2 300 channel numbers. However, not all CS2 300 channel numbers are allowed to be used for a 1200 CS2 channel.

To accommodate the doubling of to the channel capacity from CS1, the original 1.5 kHz 100/300 CS1 channel bandwidths were cut in half to form the 750 Hz CS2 300 channels. Further, to avoid having to change the numbering of the existing channel frequencies, new channel designation were interspersed between the legacy channel centers.

CS1 specified 1,500 Hz channels designated 1 through 266. CS2 added channels 301 through 566. The center frequency of channel 301 is exactly half way between the original centers of channel 1 and channel 2, and so on.

CS2 1200 channel bandwidth requirements require three (3) CS2 300 channels; i.e. CS2 1200 baud channels require 2,250 Hz of bandwidth. As such, only every third channel may be a CS2 1200 channel. The first set of 750 Hz channels in the CS2 mapping is 1, 301, and 2; accordingly the first allowable CS2 1200 channel is 301. The next set of three 750 Hz channels is 302, 3, and 303 and is CS2 1200 channel 3.

Continuing in this fashion, the allowable CS2 1200 channels become 3, 6, 9, … 264, and 301, 304, 307, … 565.

The following algorithm can be used to determine if the channel number correlates to an allowable CS2 1200 channel:

```
If (Channel number < 300)
   Valid channel number must be evenly divisible by 3
Else
   Subtract 1 and the resulting value must be evenly divisible by 3
```

6.4 Baud Rate Specifications

While the basic DCP baud rates are 100, 300, or 1200, the baud rate specification for the assign command is defined as a keyword to allow for the special auto baud detection options. The baud rate option also impacts the allowable channel number s

All DAMS-NT systems MUST support the three standard options of 100, 300, and 1200.

Baud selections of AUTO1 and AUTO2 are optional.

AUTO1 specifies an automatic baud detection for either 100 or CS1 300 operation, and excludes the reception of CS2 messages.

AUTO2 specifies an automatic baud detection for either 100 or CS1 300 operation, but also includes the reception of CS2 300 messages.

At present there is no automatic baud detection defined for 300 and 1200 messages.

As shown in Section 6.3, the baud rate in conjunction with the mode specification determines the allowable channel number. Additionally, the mode specification determines the allowable baud rate specifications as defined below, and as summarized in Table 6-11: .

CS1	Allows baud rate specifications of 100, 300, 1200, AUTO1 and AUTO2.
CS2	Allows baud rate specifications of 300 and 1200 only.
DUAL	Allows baud rate specifications of 300 and 1200 only.