# VLBI Standard Interface Specification

*Alan Whitney*

The software part of the VLBI Standard Interface specification, dubbed VSI-S, has been approved and is now complete. VSI-S joins the hardware part of the specification, called VSI-H and published in the IVS 2000 Annual Report, to form a complete interface specification for VLBI data systems. The completion of the VSI-S specification is the next step designed to allow heterogeneous VLBI digital-data systems to be interchangeably connect to both data acquisition and data processing (correlator) systems in a "plug-compatible" fashion.

From the beginning, the VSI effort has sought to separate the particular recording technology used, whether it be magnetic tapes or magnetic/optical disks, for example, from the specification of the interfaces. This allows the designers of VLBI data systems to incorporate the latest technology into their designs while still maintaining interface compatibility with other VSI-compatible systems.

The fledgling VSI concept leading to the development of the VSI specifications was first proposed at the time of the GEMSTONE meeting in Tokyo in January 1999 and was discussed by a small interested group at that meeting. Support was then sought and received from both IVS, representing primarily the geodetic VLBI community, and the Global VLBI Working Group (GVWG), primarily representing the astronomy community, to create a VSI Technology Coordination Group (VSI-TCG) comprised of experts representing all of the major world institutions involved in the development of VLBI equipment.

The membership of the VSI-S committee has a large overlap with the VSI-H committee, but also includes several software experts from several institutions to help guide critical aspects of the VSI-S specification. The members of the VSI-S committee are Wayne Cannon (York University/Crestech, Canada), Brent Carlson (DRAO, Canada), Dick Ferris (ATNF, Australia), Dave Graham (MPI, Germany), Ed Himwich (NASA/NVI, U.S.), Nori Kawaguchi (NAO, Japan), Tetsuro Kondo (CRL, Japan), Ari Mujunen (Metsahovi, Finland), Sergei Pogrebenko (JIVE, Netherlands), Misha Popov (ASC, Russia), Jon Romney (NRAO, U.S.), Ralph Spencer (Jodrell, England) and Alan Whitney (Haystack, U.S., Chair). In addition, there was participation and contribution by many other members of these institutions, which is gratefully acknowledged and appreciated.

Both the VSI-H and VSI-S specifications are intended as starting points from which to progress, and will be extended and amended as needed. Already, several institutions have developed or are developing VSI-compatible data systems. In addition, the VSI specification has been adopted in whole or in part by some other communities, particularly the high-energy physics community in Japan, and is being considered by others. The VSI-H specification was also honored in 2001 to receive a "Radio Day" award from the Japanese Ministry of Public Management, Home Affairs, Posts, and Telecommunications.

New challenges are now arising for VSI. With the surprisingly rapid development global e-VLBI, there is a strong need to develop interoperability standards for this new data-transfer technique. As a result, efforts are now beginning to establish an e-VLBI standard interface, dubbed VSI-E, to assure the smoothest global transition to global e-VLBI data interchange. VSI-H and VSI-S form a broad and useful foundation from which the VSI-E specification will arise, and much of the same group of international VLBI experts will be involved. We hope to report success in this new VSI-E endeavor in the near future.

Please visit the VSI websites at http://www2.crl.go.jp/ka/radioastro/tdc/ivs/vsi/ and http://web.haystack.edu/vsi/index.html for complete VSI information and updates.

# VLBI Standard Software Interface Specification – VSI-S

Revision 1.0
13 February 2003

## Table of Contents



## Tables

## 0. Prologue

The establishment of the hardware portion of the VLBI Standard Interface Revision 1.0, dubbed VSI-H, in July 2000 was the first major step in establishing an international standard for interfaces to VLBI data systems. Since that time, several institutions have developed or are developing equipment to conform to the VSI-H specification.

The fledgling VSI concept leading to the development of the VSI specifications was first proposed at the time of the GEMSTONE meeting in Tokyo in January 1999 and was discussed by a small interested group at that meeting. Support was then sought and received from both IVS, representing primarily the geodetic VLBI community, and the Global VLBI Working Group (GVWG), primarily representing the astronomy community, to create a VSI Technology Coordination Group (VSI-TCG) comprised of experts representing all of the major world institutions involved in the development of VLBI equipment.

Early in the discussions it became clear that the development of a unified VSI hardware and software specification was a very major task and that trying to do them together was very difficult. At the suggestion of Wayne Cannon, it was decided to break the task into two efforts, concentrating on the hardware aspects first (VSI-H) and the software aspects second (VSI-S). The membership of the VSI-S committee has a large overlap with the VSI-H committee, but also includes several software experts from several institutions to help guide critical aspects of the VSI-S specification. The members of the VSI-S committee are Wayne Cannon (York University/Crestech, Canada), Brent Carlson (DRAO, Canada), Dick Ferris (ATNF, Australia), Dave Graham (MPI, Germany), Ed Himwich (NASA/NVI, U.S.), Nori Kawaguchi (NAO, Japan), Tetsuro Kondo (CRL, Japan), Ari Mujunen (Metsahovi, Finland), Sergei Pogrebenko (JIVE, Netherlands), Misha Popov (ASC, Russia), Jon Romney (NRAO, U.S.), Ralph Spencer (Jodrell, England) and Alan Whitney (Haystack, U.S., Chair).

Following the pattern of the development of VSI-H specification, VSI-S was shaped by intensive e-mail discussions, plus face-to-face meetings of opportunity in association with other scheduled meetings around the world. The discussions were always lively, sometimes frustrating, but always enlightening and pursued in a spirit of international cooperation. The final result, VSI-S Revision 1.0, is the result of many hours of work by many people to create the best specification possible. Revision 1.0 of both VSI-H and VSI-S are intended to be starting points, with the expectation they will evolve and modify to meet the developing needs of the global VLBI community; indeed, VSI-H Revision 1.1 is now close at hand.

The VSI-S specification represents the work of many individuals at many institutions, including members of the VSI-S committee and many others, and the Chair gratefully acknowledges and thanks all who have contributed. It is only with the participation of all of you that the result has been so successful.

Alan Whitney
MIT Haystack Observatory
13 February 2003

## 1. Introduction

The VSI-S specification is the software counterpart of the VSI-H specification [Reference 1], which specifies the standardized hardware interfaces of a VLBI Data Transmission System (DTS). Systems adhering to both the VSI-H and VSI-S specifications should be interchangeable with minimal effort at both data-acquisition and data-processing sites.

Reference 1 should be reviewed with respect to terms and acronyms used in this specification. An additional glossary specific to VSI-S is contained in Section 10 of this document.

## 2. Intent of the VSI-S Specification

The goal of VSI-S is to specify a robust, reliable communications protocol to control a VSI-H-compliant DTS. In this regard, VSI-S must address three issues:

1. Specify a communications structure and protocol.
2. Specify a generalized command and response syntax model to be used by the DTS.
3. Specify a base set of commands to configure and operate a generic DTS adhering to the VSI-H specification.

All systems implementing VSI-S are expected to adhere to the specifications in this document to the maximum degree possible.

## 3. Communications Model

### **3.1** *DTS Communication ports*

The VSI-S communications model specifies four communication ports to/from the DTS apart from the primary high-rate data stream:

1. An Ethernet (formally IEEE 802.3) TCP/IP connection serves as the primary VSI-S control interface to a Controller. The suite of VSI-S base commands is supported at this interface.
2. An RS-232 (formally EIA/TIA-574) serves primarily for debug and diagnostic support not specified as part of VSI-S, but may optionally also serve as a secondary VSI-S control interface to a Controller. As such, the suite of VSI-S commands is optionally supported at this interface.
3. A PDATA serial-data line into the DIM from a DAS. This is strictly a one-way communications link.
4. A QDATA serial-data line from the DOM to a DPS. This is strictly a one-way communication link.

### **3.2** *Control-Interface Communication Model*

Both Ethernet and RS-232 are naturally adaptable to the standard OSI layered communications model with TCP at the transport layer. In addition, RS-232 may support a simple 'RS-232 direct' connect which bypasses most of the standard network layers. The communications structures prescribed for VSI-S are shown in Table 1:

| Protocol Layer | Ethernet | RS-232 TCP (optional) | RS-232 Direct (optional) |
|---|---|---|---|
| **Application** | VSI-S | VSI-S | VSI-S |
| **Transport** | TCP | TCP | NPAD[1] |
| **Network** | IP | IP | - |
| **Data Link** | IEEE802.3 | PPP | - |
| **Physical** | IEEE802.3 | EIA/TIA-574 | EIA/TIA-574 |

Table 1: Control interface communications model

TCP provides a reliable message-delivery mechanism using either Ethernet or RS-232. Messages in each direction are independently sequenced and acknowledged. The services necessary to support the TCP-layered services for either Ethernet or RS-232 are commonly available in virtually all modern operating systems.

'RS-232 Direct' is a basic 'dumb' RS-232 connection with no mechanisms to provide reliable error-free delivery of messages in either direction. A single <CR> character is appended at the end of each message as an end-of-message character. This type of connection is only practical in a local environment with a high-reliability connection between Controller and DTS[2].

## 4. Transport Protocol Rules

### 4.1 *TCP*

The primary control connection to the DTS is TCP through the primary control port (Ethernet). Optionally, TCP may be implemented and used through the RS-232 port as a secondary control connection. In both cases, the same rules apply.

#### 4.1.1 Standard VSI Port Number

The standard DTS TCP port number assigned for the VSI-S control connection is 5653 (decimal), which should be used whenever possible.

#### 4.1.2 Connection Rules

In order to increase system robustness, the following additional rules apply to TCP connections:

1. Only one TCP control connection at a time may be open. Additional connections are allowed on other port numbers for monitoring, setup, debug, etc.
2. Should a new request for a TCP control connection be received while an existing such connection exists, the original connection on that port will be closed and the new request honored.
3. In order to prevent unexpected or unauthorized remote operation, the DTS will provide a local mechanism to disable the TCP control port.

[1] NPAD – Non-Reliable Packet Assembly and Disassembly
[2] Note that the Ethernet control interface can also be used to emulate a 'RS-232 direct' interface through the use of a standard terminal emulator box which connects between Ethernet, acting typically as a telnet server, and an RS-232 device.

**4.2** *RS-232 Direct*

The RS-232 Direct connection is a physical one-to-one Controller-to-DTS link and is not subject to most of the complications of TCP. The RS-232 Direct link is expected to be local and reliable. As a result, no special rules are required.

## 5. Application Protocol Rules

The application protocol defines the rules governing the interactions of the Controller and the DTS in the execution of VSI-S-defined commands and responses. For sake of clarity, these are divided into application protocol, DTS response-time and Controller timeout rules, as well as a discussion of the effect of communication latencies.

**5.1** *Application Protocol*

The communications protocol at the application layer is fundamentally half-duplex, to which the following definitions and rules apply:

1. A transaction comprises the Controller sending a single command message to the DTS, and the DTS replying with a response message to that command.
2. The Controller may not send a new command message until the previous transaction is completed.
3. The maximum length of any single message is 1024 characters.

**5.2** *'Response Time' and 'Response Window'*

The DTS response time is defined as the interval beginning at the instant the DTS has received a full command/query message from the Controller and ending when the DTS response is complete; for RS-232 Direct, the response time interval starts at a <CR> from the Controller and ends when the DTS has sent the <CR> terminating its response.

The maximum allowed response time is called the response window. The suggested response window is <500 msec, with a maximum allowed response window of 1 second[3]. The response window of a particular DTS may be determined with the 'response' query.

The execution of commands by the DTS is governed by the following rules:

1. Every command/query shall receive a response within the response window. If the DTS is too busy to act on a request, it must return the appropriate return code (see Section 6).
2. Some commands initiate or enable actions which cannot be completed within the allotted DTS response window (tape positioning, for example). In these cases, the DTS will return an appropriate response code ('1' return code) within the response window. The Controller should follow up with subsequent queries to learn of the current status and (presumed) subsequent completion of the commanded action (see also Section 5.6). If an error occurs during the attempt to execute the command, the 'status' request will return a bit in its response field indicating an error; a subsequent 'get_error' query will give details of the error.
3. If, following the time that an action has been initiated or enabled but before completion of the action, a countervening command with the same keyword is issued, the countervening command

[3] This is the maximum DTS response time for a 'local' connection. Some remote connections may force the Controller to adjust to prevailing conditions on a case by case basis not covered in the VSI-S specification.

takes precedence (i.e. the earlier command is abandoned). No command queuing with the same keyword is allowed. For example, if a tape is commanded to a particular position, but is subsequently commanded to rewind before completion of the original positioning request, the rewind request takes precedence.

4. In some cases a command may conflict with the current DTS status and be impossible to execute. For example, a command to start recording while a tape is rewinding is a conflicting request and causes the appropriate return code to be issued.
5. A query that cannot be completed within the response window will return an appropriate response ('1' return code). *Identical* followup queries will continue to return the same response code until the query is complete, at which point the return code will be '0' and the requested information returned (unless an error has occurred, in which case the appropriate error code will be returned). Other commands and queries may be interspersed between the original query and subsequent identical queries. In some cases, the values of the requested parameters may have changed after the original query, but before successful completion of the query; in such cases, the control software should, to the best of its ability, always return the *current* state of the system.

**5.3** *Communications Break*

The DTS will sense a communications break when its transport process reports (after retries, etc.) inability to send a response. Similarly, the Controller will sense a break when its transport process reports inability to send a command. It will also sense if its command string falls behind schedule, if the operator tells it so, or if an excessive time passes (suggest three *response-window* periods) without response after a successful command transmission.

On detecting a communications break, the DTS should:

1. Abandon any pending response, but otherwise continue in its present state to the extent possible[4].

On detecting a communications break, the Controller should:

1. Inform the operator as appropriate.
2. Close the old connection, and attempt to open a new one. In order to avoid long delays during opens (including both initial opens and time-out recoveries) when the connection path is questionable, each attempt to open a connection should be preceded by quick ‘ping’ or other confirmation of the link to the DTS to make sure a path is available. Any re-connection attempt should be deferred until path confirmation is successful.

On successful re-connection, the Controller should:

1. Presume that the DTS has continued in its last known state, and not interrupt it before ascertaining the actual situation via status/query commands.
2. Not send any nonidempotent[5] commands/queries until the current value of any affected parameters has been determined.

**5.4** *Time-Critical Commands and ‘Safe Window’*

Two commands, namely ‘DOT_set’ and ‘ROT_set’ (see Sections 9.3 and 9.5), are time critical in that they must be issued *between* DOT/ROT ticks and apply their action at the *next* DOT/ROT 1pps tick; the

[4] Not possible if, for example, end-of-media is encountered during a broken connection.
[5] Nonidempotent commands/queries are those which may result in a different effect when performed more than once.

‘send_QDATA’ command may also fall into this category, depending on the mode in which it is executed. The safe window for the issuance of these commands is defined to start at the instant of a DOT/ROT tick and extend for some period before the next tick. It is suggested that a reasonable safe window is 75% of the tick period[6]. The actual length of the safe window for a particular DTS may be determined with a ‘response’ query.

Two queries, namely ‘DOT?’ and ‘ROT?’ capture the current reading of the DOT/ROT clock, respectively, and report them to the user. Because confirmation of the proper setting of the DOT/ROT clock is critical to the proper operation of a DIM/DOM, special rules apply to these two commands. In particular:

1. *The DOT/ROT clock reading must be captured within 10 msec of the receipt of the DOT?/ROT? query.*
2. *The captured DOT/ROT readings must be returned with a minimum resolution of 10 msec.*

Note that, although the *action* triggered by a DOT/ROT query is time critical with respect to the receipt of command/query by the DTS, the DTS response to the Controller is just like the response to any other command or query, namely that it is required to take place within the normal DTS response window.[7]

**5.5** *Time-Critical Commands and Communication Latency*

When the DTS and Controller are both locally located, it is not normally difficult to guarantee that communication latency times are small compared to the windows allowed by time-critical messages. However, latencies large compared to these windows may be experienced when the Controller is remotely located. In this case, reliable execution of time-critical commands must be guaranteed by some other method.

Though not mandated by VSI-H or VSI-S, one option to mitigate arbitrary communication latencies is for the DTS system to maintain a relatively accurate knowledge (within 10’s of milliseconds) of UT through the use of network time protocol (ntp), a local GPS receiver, or some other means; for purposes of this discussion, we will call this ‘DTS system time’. Then, for example, the setting of the DOT clock may be enabled at a specified DTS system time, in which case the DOT clock is properly set to a specified time on the next 1PPS tick following the specified UT. Conversely, the DOT/ROT query returns may also be tagged with the corresponding DTS system time for robust confirmation of the DOT/ROT clock settings; for most usefulness, the returned DTS system time must have a resolution of at least a few tens of milliseconds. Such a strategy is robust in the face of any arbitrary communication latency and is supported by the VSI-S command set.

**5.6** *‘Initiate/Enable’ Commands*

Some commands initiate or enable actions which may not be completed within the response time. An example of such an action is a tape-positioning request; similarly, the ‘DOT_set’ and ‘ROT_set’ commands enable actions which will usually not be immediately completed. In these cases, the DTS will respond with a response code indicating the action has been initiated/enabled, but not completed. It is the responsibility of the Controller to issue followup queries to determine when the requested action has been completed.

---

[6] ROT clock may be speeded up or slowed down, so tick period is not necessarily an actual second. Media-copying operations at speeds different from real-time may potentially result also in a DOT clock period that is not an actual second.

[7] The most common problem is likely to be setting the DOT/ROT clock to an incorrect integer second value. In most cases, such an error is easily corrected with the ‘DOT_inc’ or ‘ROT_inc’ commands, which will increment the respective clock by a specified number of integer seconds.

Depending on the characteristics of a particular DTS, different subsets of commands may fall into the ‘initiate/enable’ category. This is unavoidable and is likely to require some awareness of DTS differences by the Controller.

## 6. Command, Query and Response Syntax

Any message sent to the DTS Control Port must be one of two types:

1. Command – causes an action which may change the state of the DTS
2. Query – queries some aspect of the state of the DTS; *does not change the state of the DTS*

### **6.1** *Command Syntax*

Commands are of the form

<keyword> = <field> : <field> : …. ;

or <keyword>[<n>] = <field> : <field> : …. ;

where <keyword> is a VSI-S command keyword and the optional '[<n>]' designates Port <n> for port oriented commands. In the absence of a designator the command will apply to all ports. The number of fields may either be fixed or indefinite; fields are separated by colons and terminated with a semi-colon. A field may be of type decimal integer, decimal real, integer hex, character, literal ASCII or a special ‘time’ code (see Section 7.2). White space between tokens in the command line is ignored. VSI-S keywords are listed in Section 9.

### **6.2** *Command-Response Syntax*

Each command recognized by the DTS will elicit a response of the form

!<keyword> = < return code > [:<DTS-specific return> :….] ;

!<keyword>[<n>] = < return code > [:<DTS-specific return> :….] ;

where

<keyword> is a defined VSI-S command keyword (see Section 9 for list)

[<n>] is a port designator (Section 6.1)

An 'all ports' command will elicit the requisite list of designated command-responses.

The 'maximum single message length' (Section 5.1) applies to each element of the list.

<return code> is an ASCII integer as follows:

0 - action successfully completed

1 - action initiated or enabled, but not completed

2 - command not implemented or not relevant to this DTS

3 - syntax error

4 - error encountered during attempt to execute command

5 - currently unable to service request; try again later

6 - inconsistent or conflicting request[8]

[8] For example, it is not possible to record during playback or position unloaded media.

7 - no such keyword

8 - parameter error

<DTS-specific return> - one or more optional fields specific to the particular DTS, following the standard fields defined by VSI-S; fields may be of any type, but should be informative about the details of the action or error.

Notes:

1. White space between tokens in the command-response is ignored.
2. The repetition of the keyword in the response is redundant, but is included for readability of communication logs which may be kept by the Controller or DTS.
3. Many parameters have 'power-on' values; if a parameter is not specified (e.g. non-existent or empty field), it may have a specified default value (which may be 'current value'). Some parameters must always be specified and have no default values; failure to specify these parameters will result in a return code 8 ('parameter error'). See Section 9 for details.

**6.3** *Query and Query-Response Syntax*

Queries are of the form

<keyword>?;

or <keyword>[<n>]?<param1>:<param2>:....;

with a response of the form

!<keyword> ? <return code> : <field> : <field> : …: [<DTS-specific return>];

or !<keyword>[<n>] ? <return code> : <field> : <field> : …: [<DTS-specific return>];

where

[<n>] is an optional port designator (Sections 6.1 & 6.2);

<param1>:<param2>:... is an optional specific list of parameters which overrides the default list for this query;

<return code> is an ASCII integer as follows:

0 - query successfully completed

1 - action initiated or enabled, but not completed

2 - query not implemented or not relevant

3 - syntax error

4 - error encountered during attempt to execute query

5 - currently unable to service request; try again later

6 - inconsistent or conflicting request[9]

7 - no such keyword

[9] For example, it may not be possible to query certain DOM parameters while the DIM is active.

8 - parameter error

9 - indeterminate or undefined state[10]

<DTS-specific return> - one or more optional fields specific to the particular DTS, following the standard fields defined by VSI-S; fields may be of any type, but should be informative about the details of the query.

Notes:

1. White space between tokens in a query-response is ignored.
2. Notes that the query return codes are identical to the command return codes except for query return code 9, which does not have a command return code counterpart.
3. When a command and query use the same keyword, the returned query parameters shall be a mirror of the command parameters (i.e. same order and values), although the query may return additional relevant parameters following the mirrored parameters.
4. Some returned query parameters may be specified as optional, in which case a specific DTS may optionally return such parameters.
5. Special care must taken to avoid queries which cause status conditions to be cleared or quantities to be incremented/decremented within the DTS. Such a condition could cause the response to a repeated query to differ from the original response, resulting in possible confusion.

## 7. Keyword and Field Rules

### **7.1** *Keyword Length*

Individual keywords are limited to 16 characters.

### **7.2** *Field Types*

Each field in a command or return statement may be one of the following six types:

Integer – a simple positive, negative or zero decimal integer (examples: '12', '-25'; of course, no quotes in actual usage).

Real – number with a decimal point and/or possible exponent (examples: '1.12', '-2.23e-6')

Hex – in standard 'C' format (example: '0x4a32'); by definition, bit 0 is LSB.

Character – prescribed character string (examples: 'on', 'off'); limited to 16 characters.

Literal ASCII – arbitrary ASCII string enclosed within single or double quotes (examples: 'This is a literal ASCII string' and "This is also a literal ASCII string"); . Any ASCII character in the range 0x20 to 0x7f is allowed, except that any occurrence of the 'enclosing' quote character must be 'escaped' with a leading backslash (example: 'This string contains both a \' and " character' or "This string contains both a ' and \" character").[11]

Time – following the vex format, time is specified as '..y..d..h..m..s', where the '..' fields represent integer year, day-of-year, hour, minute and real second, respectively. Leading zeroes may be dropped. Examples: '2000y212d19h03m', '2003y91d9h23m13.093s'.

[10] For example, an uninitialized parameter with no default (i.e. CLOCK_frq has no default and must be set).

[11] Allowing either a single-quote or double-quote as the enclosing character permits a literal ASCII string to contain a literal ASCII string itself, which is a potentially useful construct for specifying QDATA (see Section 7.2)

**7.3** *Character Set*

Keywords and fields, except literal ASCII fields, may be composed of any standard printable ASCII characters in the range 0x20 to 0x7f except white space and any of the 10 characters ‘’=:;!”?[] (including the single-quote and double-quote characters) may be used in keywords and fields. Case is *not* significant except in literal ASCII fields.

## 8. PDATA/QDATA Usage

The PDATA/QDATA communications links are both one-way serial-data links that serve to allow periodic information and/or commands to be received by the DIM (PDATA) or transmitted by the DOM (QDATA). Both share the following common characteristics:

1. P/QDATA communications is assumed to be a error-free.
2. Since P/QDATA is a one-way communication channel, there will never be a response to a message received via PDATA or to a message transmitted via QDATA.
3. A message is always terminated by a <CR>. The message plus <CR> is called a <u>packet</u>.
4. All characters until a terminating <CR> are considered to be within a message.

The rules specific to PDATA and to QDATA are detailed in the following sections. Systems not implementing PDATA/QDATA should respond to all related commands/queries as ‘command not implemented’ (see Section 6).

**8.1** *PDATA*

The PDATA serial-data line into the DIM may be used to transmit commands or information to the DIM. PDATA transmissions operate under the following assumptions and rules:

1. The DTS recognizes PDATA messages according to the following set of rules, similar to those regarding packets from the Controller to the DTS:
   a. Packets with a zero-length message are ignored.
   b. The issuance of a ‘PDATA_cntl’ command by the Controller clears and resets the PDATA buffer, so that the next received PDATA character is considered to be the first character of a PDATA message (unless it is an <CR>).
2. According to user option, via the ‘PDATA_cntl’ command, the DTS may take any of the following actions with a received message:
   a. Ignore it.
   b. Cause a flag to be set in a ‘status’ query return-code that indicates the DTS has received and queued one or more PDATA messages. These messages may be retrieved by the Controller, one-by-one, using the ‘get_PDATA’ query. The message will be returned as a literal-ASCII field as received. No checking or parsing of the message is attempted.
   c. Pass the PDATA message through to the transmission media if the DTS is so-capable.
   d. If the PDATA message is a legal and recognized DTS command, execute the command.
3. The DTS should provide a reasonably-sized buffer for collecting PDATA to be retrieved with the ‘get_PDATA’ command (suggest minimum 4096 bytes).

The message field of a PDATA packet may be virtually any printable ASCII character string. The format of the message is important only if the DTS expects a particular format or is enabled to attempt execution of a command via PDATA, in which case the message must be in the normal VSI-S command format.

Probably the most useful command to be sent via PDATA is the 'set_DOT' command to set the DOT clock in a media-copying operation. The control for enabling the execution of the 'DOT_set' command is handled with separate option bits in the 'PDATA_cntl' command. The timing of the 'DOT_set' command via PDATA is subject to the same safe window rules as a 'DOT_set' command issue by the Controller (see Section 5.4).

A 'send_PDATA' command is available in the DIM to allow the user to insert arbitrary strings into the PDATA stream, which are then passed to the DTS transmission media if the DTS is so-capable.

**8.2** *QDATA*

The QDATA serial-data line from the DOM may be used to transmit messages to a DPS, or to the DIM of another DTS for media-copying purposes or data-transmission purposes. QDATA transmissions operate under the following rules:

1. The 'send_QDATA' command allows the user to specify a literal ASCII string to be transmitted via QDATA. The DTS does no checking of the string, but strips the leading and trailing single/double quotes and embeds the result within a packet (i.e. appends a trailing <CR>)
2. A 'send_QDATA' command may prescribe QDATA to be transmitted either on 'next ROT tick', in which case the specified packet will be transmitted immediately following the next ROT tick, or immediately following a ROT tick corresponding to a prescribed ROT clock reading. If a ROT clock time is prescribed, the 'send_QDATA' command must be applied no earlier than 60 seconds preceding the prescribed time, nor no later than the close of the safe window (see Section 5.4) for the prescribed time; otherwise, an error will occur.
3. A 'send_QDATA' command requesting to transmit QDATA on the 'next ROT tick' must be sent to the DTS within the safe window (see Section 5.4) preceding the target ROT tick. Otherwise, the time of the actual QDATA transmission may be ambiguous.
4. For systems which have the capability, PDATA may be passed through to QDATA; in this case, a PDATA packet must be received by the DIM within the safe-window period of the DOM in order for the transmission to have an unambiguous epoch. The packet will be transmitted via QDATA on the *next* ROT tick. *Note that all QDATA packets created in this fashion will be shifted one tick later with respect to the time tag originally assigned in the DIM input data stream*[12].
5. The DOM may enable transmission of a 'DOT_set' command immediately following each ROT1PPS tick, which is useful for media copying. In this case, the transmitted time will be adjusted forward by one second for proper setting of the DOT clock.
6. In any case where data sent by the 'send_QDATA' command may conflict with passed PDATA in accordance with 'QDATA_cntl', the 'send_QDATA' command shall have precedence.

---

[12] Since P/QDATA are normally used for transmitting slowly-changing information (source, recording mode, etc.), this should not normally be a problem. If this 'epoch shift' of P/QDATA is not acceptable, it is the responsibility of the user to take whatever special actions may be required.

7. A returned bit in the response to a general ‘status’ request indicates that a QDATA message has been sent, which may be retrieved for inspection with the ‘get_QDATA’ query. The DTS should provide a reasonably-sized buffer for collecting QDATA to be retrieved with the ‘get_QDATA’ command (suggest minimum 4096 bytes)

# 9. VSI-S Command/Query/Response Base Set

The following tables detail the VSI-S command/query base set, which has been constructed for a ‘generic’ DTS, and which should be adhered to as strictly as possible to ensure maximum interoperability between heterogeneous DTS systems. It is suggested that any required additional DTS-specific fields be added beyond the fields defined in the VSI-S base set (see Sections 6.2 and 6.3). In addition, some new commands/queries may need to be added for specific DTS’s, in which case they should be constructed in a similar fashion to those in the suggested VSI-S base set, but in any case should be minimized. Commands/queries in the VSI-S base set which are not relevant or not implemented for a specific DTS must cause the DTS to return the appropriate <return code> (see Section 6) if they are issued by the Controller.

If new commands/queries are required for specific DTS’s, special care must be taken to avoid commands (especially status requests) which cause status conditions to be cleared or quantities to be incremented/decremented within the DTS. Such a condition could cause the response to a repeated command to differ from the original response, resulting in possible confusion.

Command parameters whose field #’s are in brackets (‘[]’) are optional.
An unspecified parameter in a command will be set to the ‘Default’ value; ‘*CV*’-current value, ‘*None*’-parameter must be specified.
Other abbreviations: ’*NA’*-not applicable; *‘SS*’-system specific.

Query field #’s in brackets (‘[]’) are optionally supplied by the DTS (see Section 6.3).

Note: Query field #1 is always the query <return code> (see Section 6.3) and is not listed in the following tables.

## **9.1** *System Commands*

These commands are ‘system level’ commands not specific to either the DIM or DOM.

| Keyword | Field # | Description | Type | Allowed values | Power-on value | Default | DTS-specific? | Comments |
|---|---|---|---|---|---|---|---|---|
| diagnostic | 1 | Start diagnostic self-test | hex | | *NA* | 0 | Yes | Bits set in field 1 specify test(s) to be performed |
| reset | 1 | Initiate system reset | char | system | *NA* | *None* | No | Full system reset.<br>Specific DTS’s may implement other levels of reset, as required. |

Notes:

1. In the case of RS-232, the DTS must include some method or procedure to set a compatible baud rate with the Controller.
2. Simultaneous use of Ethernet and RS-232 control interface is not encouraged as it can lead to timing conflicts. Any simultaneous use of Ethernet and RS-232 may have to adhere to DTS-specific rules outside the scope of the VSI-S specification.

## **9.2** *System Queries and Responses*

These queries are ‘system level’ queries not specific to either the DIM or DOM.

| Keyword | Returned Field # | Description | Type | DTS-specific? | Comments |
|---|---|---|---|---|---|
| DTS_id | 2 | System type | literal ASCII | Yes | Example response: ‘S2 -REC’ or ‘mark5a’ |
| | 3 | System revision level | literal ASCII | | Example response: ‘2.1’ |
| | 4 | Media type | int | No | 0 – magnetic tape; 1 – magnetic disc; 2 – real-time (non-recording); additional defined as req’d. |
| | 5 | Number of DIM Ports | int>=0 | No | 0 on stand-alone DOM |
| | 6 | Number of DOM Ports | int>=0 | No | 0 on stand-alone DIM |
| | [7] | Serial number | literal ASCII | Yes | Serial number of particular DTS unit |

| | | | | | |
|---|---|---|---|---|---|
| status | 2 | General status query | hex | No | Bit 0 – error message pending; (use 'get_error' command for details of error);<br>cleared by get_error command<br>Bit 1 – one or more PDATA packet received (use 'get_PDATA' command to retrieve);<br>cleared when all packets have been read;<br>relevant only if PDATA_cntl bit mask has 0x01 bit set<br>Bit 2 – QDATA packet sent (use 'get_QDATA' command to retrieve);<br>cleared when all packets have been read<br>Bits 4-3: unused<br>Bit 5 – tvg report pending (use 'get_tvr' command to retrieve);<br>cleared when no tvg reports pending<br>Bits 7-6: 00 – receiving (recording) 'off'<br>01 – receiving pending<br>10 – receiving<br>11 – receiving stopped (end of media, etc)<br>Bits 9-8: 00 – transmit (playback) 'off'<br>01 – transmit pending (sync'ing, etc)<br>10 – transmitting (0x<br>11 – transmitting stopped (end of media, etc)<br>Higher-order bits unused. See also Note 1. |
| | [3] | DTS-specific status word | hex | Yes | |
| diag_status | 2 | Diagnostic status | int | | 0 - inactive; 1 – active |
| | 3 | Diagnostic results | hex | Yes | System-specific diagnostic results |
| get_error | 2 | Get error message | int | Yes | Error number |
| | [3] | ASCII error message | literal ASCII | Yes | ASCII error message associated with error number |
| response | 2 | Response window | int | Yes | DTS response window (max response time), in milliseconds |
| | 3 | Safe window | int | Yes | Length of safe window for DOT_set, ROT_set and send_QDATA commands, in milliseconds (see Sections 5.4 and 8.2) |

Notes:

1. In a multi-port system, note that 'status' bits 1, 2 and 5 will be the logical 'OR' of the individual port flags.

## **9.3** *DIM Setup and Operating Commands*

| Keyword | Field # | Description | Type | Allowed values | Power-on value | Default | DTS-specific ? | Comments |
|---|---|---|---|---|---|---|---|---|
| CLOCK_source | 1 | Port for Reference CLOCK and Ref 1PPS | char | port<n> \| internal | port0 | *CV* | No | 0<= <n> <=99 |
| 1PPS_source | 1 | Specify second-tick sync source | char | ref1pps \| alt1pps | ref1pps | *CV* | No | |
| CLOCK_frq[] | 1 | CLOCK frequency | int | 2\|4\|8\|16\|32[\|64\|128] | *SS* | *CV* | No | Units are MHz. 64 and 128 MHz may be supported by some units. |
| BSIR[] | 1 | Bit-stream information rate | int | 2\|4\|8\|16\|32\|[64\|128] | CLOCK_frq | *CV* | No | Minimum BSIR is 2 MHz; max is CLOCK frequency |
| DOT_set | 1 | Enable DOT clock set | time | time | *NA* | *None* | No | Enable DOT clock to be set to specified time on *next* DOT second tick (unless Field 2 if specified); time must be integral second. |
| | [2] | UT to enable DOT clock setting | time | time | *NA* | *NA* | Yes | Enable DOT-clock setting at specified UT time.<br>Applicable only if DIM has reasonably accurate internal UT clock (see Section 5.5) |
| DOT_inc | 1 | Increment DOT clock | int | | *NA* | *NA* | No | >0 – advance DOT clock by specified number of seconds<br><0 – retard DOT clock by specified number of seconds |
| BS_mask[] | 1 | Bit-stream receive mask | hex | 32-bit hex | 0xffffffff | *CV* | No | Specifies bit-streams to be received (recorded) by DIM;<br>VSI-H allows selection of any 1, 2, 4, 8, 16 or 32 bit streams;<br>LSB corresponds to bit-stream 0. |
| PVALID[] | 1 | PVALID line active/inactive | char | on \| off | off | *CV* | No | If 'on', PVALID signal indicates data validity.<br>PVALID may be ignored by some systems. |
| PDATA_cntl[] | 1 | Set PDATA control mask | hex | 0x00 to 0x20 | 0x0 | *CV* | No | Bit mask:<br>Bit 0 – Pass PDATA packets to media (if supported)<br>Bit 1 – Execute valid commands, except DOT_set<br>Bit 2 – Execute next valid DOT_set command only<br>Bit 3 – Execute every valid DOT_set command |
| send_PDATA[] | 1 | Send PDATA string | literal ASCII | | *NA* | *None* | No | Allows arbitrary strings and auxiliary information to be inserted in the PDATA stream to be passed to the DTS transmission media if the DTS is so-capable. |
| | [2] | Prescribed ROT time at which to send PDATA string | time | | *NA* | *None* | No | If specified, will send specified PDATA following specified ROT tick (see Notes); fractional seconds are ignored.<br>If not specified, will be sent immediately following next ROT tick. |
| tvr[] | 1 | Test-vector receiver reporting period | integer secs | >=0 | 0 | *CV* | No | Sets tvr analysis period to specified number of DOT seconds. Value of 0 stops tvr reporting immediately. |
| | [2] | Number of periodic tvr reports to be created | int | >=1 | 1 | *CV* | No | Default = 1 report |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | [3] | Bit-stream mask | hex | 0x00000001 to 0xffffffff | 0x00000001 | *CV* | Yes | Default is DTS dependent (some systems may only be able to analyze a single bit-stream at a time). |
| | [4] | Analysis mask | hex | 0x1 to 0x3 | 0x3 | *CV* | Yes | LSB – error rate analysis<br>MSB – DC-level analysis<br>Default = 0x3 (full analysis) |
| | [5] | Bit-stream rotation | int | 0-31 | 0 | *CV* | Yes | Rotate full set of 32 bit-streams to the left by specified number of positions before entering tvr. For example, '2' places BS0 in place of BS2 before entering tvr, BS1 in place of BS3, BS30 in place of BS1, etc. This capability is useful for unscrambling misdirected bit-streams, but may not exist in some systems or may be implemented in a different fashion. |
| TVGCTRL_set [] | 1 | Set state of TVGCTRL signal | char | on \| off | off | *CV* | No | May not be implemented in all systems |
| receive | 1 | Start/stop data receive (record) | char | on \| off | off | *CV* | | See also Note 1 in Section 9.4 |
| | [2-n] | Parameters | - | | *SS* | *SS* | Yes | DTS-specific; examples – scan name, tape speed, etc |

## **9.4** *DIM Queries and Responses*

| Keyword | Returned Field # | Description | Type | DTS-specific? | Comments |
|---|---|---|---|---|---|
| CLOCK_source | 2 | Port for Ref CLOCK and Ref 1PPS | char | No | Current value |
| 1PPS_source | 2 | Select second-tick sync source | char | No | Current value |
| CLOCK_frq[] | 2 | CLOCK frequency | int | No | Current value |
| BSIR[] | 2 | Bit-stream information rate | int | No | Current value |
| DOT | 2 | DOT status | int | No | 0 – DOT set enabled; 1 – DOT running |
| | 3 | Current DOT clock reading | time | No | Current DOT clock reading to at least 10 msec resolution.<br>Capture of this parameter must occur within 10 msec of receipt of 'DOT?' query |
| | [4] | UT time of DOT clock reading | time | Yes | Reading of system clock at same instant that DOT clock reading is made.<br>Should be returned with maximum precision available; precision of at least a few 10's of msec is required to be useful.<br>Applicable only if DIM has reasonably accurate internal UT clock (see Section 5.5) |
| BS_mask[] | 2 | Bit-stream receive mask | hex | No | Current value |
| PVALID[] | 2 | PVALID line active/inactive | char | No | Current value |
| PDATA_cntl[] | 2 | PDATA control mask | hex | No | Current value of PDATA control mask |

| | | | | | |
|---|---|---|---|---|---|
| get_PDATA[] | 2 | Number of packets available for retrieval (including this packet) | int | No | Number of PDATA packets available for retrieval by Controller;<br>if =0, no packets available. |
| | 3 | Number packets lost since last get_PDATA | int | No | Number of packets lost due to buffer overflow (oldest packets are always discarded first in buffer overflow situation) |
| | 4 | DOT clock reading | time | No | DOT clock reading when PDATA packet in field 5 was received |
| | 5 | PDATA data | literal ASCII | No | |
| tvr[] | 2 | Test-vector receiver reporting period | integer secs | No | Current value;<br>=0 if tvr reporting disabled. |
| | 3 | Number of periodic tvr reports to be created | int | No | *Remaining number* of tvr reports to be transmitted |
| | 4 | Bit-stream mask | hex | Yes | Current value |
| | 5 | Analysis mask | hex | Yes | Current value |
| | 6 | Bit-stream rotation | int | Yes | Current value |
| get_tvr[] | 2 | Number of tvr reports available for retrieval (including this report) | int | No | =0 if no report is available or tvr reporting is disabled. |
| | 3 | Number of tvr report lost since last get_tvr | int | No | Number of reports lost due to buffer overflow (oldest reports are always discarded first in buffer overflow situation) |
| | 4 | End time of reporting period | time | No | |
| | 5 | Bit-stream number | int | No | |
| | 6 | Analysis period in DOT seconds | int | No | |
| | 7 | Error rate | int | Yes | Interpretation may be DTS-specific |
| | 8 | DC offset | int | Yes | Interpretation may be DTS-specific |
| TVGCTRL_set [] | 2 | TVGCTRL signal status | char | No | May not be implemented in all systems |
| receive | 2 | Receive (record) status | char | No | 'on' or 'off. See Note 1 below. |
| | [3-n] | Parameters | - | Yes | |

Notes:

1. After receive (record) is turned 'on', the user should periodically query general 'status?' for details; if recording stops on its own accord (due to end-of-media, etc.), this will be reflected in the response to the 'status?' query as 'receiving stopped', and a 'receive?' query will show the status as 'off'; a subsequent command to turn receive 'off' or 'on' will reset the relevant bits (7-8) in the 'status?' response.

## **9.5** *DOM Setup and Operating Commands*

| Keyword | Field # | Description | Type | Allowed values | Power-on value | Default | DTS-specific? | Comments |
|---|---|---|---|---|---|---|---|---|
| DPSCLOCK_source | 1 | Select DPSCLOCK source | char | dpsclock \| port<n> \| internal | dpsclock | *CV* | No | If 'dpsclockx' on port<n> is selected, it is the responsibility of user to ensure that QCTRL line on port <n> is set appropriately; selection of dps1pps source is forced to the logical counterpart of selected dpsclock source (dps1pps, port<n> or internal) |
| | 2 | DPSCLOCK freq | int | 2\|4\|8\|16\|<br>32 [\|64 \| 128] | *SS* | *CV* | No | Units are MHz. 64 and 128 MHz may be supported by some units. Selection of 'internal' in field 1 disables setting of this parameter. |
| QCTRL[] | 1 | Set QCTRL reverse-channel control signal | char | on \| off | off | *CV* | No | 'on' enables reverse-channel signalling |
| RCLOCK_frq[] | 1 | RCLOCK frequency | int | 0\|2\|4\|8\|16\|32… | 0 | *CV* | Yes | Sets output bit-stream clock rate within DOM-specific limitations;<br>=0 forces RCLOCK frequency to be same as bit-stream information rate (BSIR) specified to DIM by 'BSIR_set' command.<br>In any case, max value is DPSCLOCK frequency. |
| ROT_set | 1 | Enable ROT clock set | time | time | *NA* | *NA* | No | Enable ROT clock to be set to specified time on *next* DPSCLOCK tick (unless Field 2 if specified); time must be integral second. |
| | [2] | UT to enable ROT clock setting | time | time | *NA* | *None* | Yes | Enable ROT-clock setting at specified UT time.<br>Applicable only if DOM has reasonably accurate internal UT clock (see Section 5.5) |
| ROT_inc | 1 | Increment ROT clock | int | | *NA* | *NA* | | >0 – advance ROT clock by specified number of seconds<br><0 – retard ROT clock by specified number of seconds |
| delay | 1 | Specified data delay wrt ROT clock | int | See Comments | 0 | *CV* | Perhaps | Units are sample periods; >0 indicates data delayed relative to ROT clock. Specified delay is set on *next* ROT1PPS tick.<br>Implemented range should be sufficient to cover +/-0.5*ROT1PPS. |
| portmap[] | 1 | Select DIM port to output | int | | DOM[k]= DIM[k] | *CV* | No | <0: restore default (e.g. DOM[1]=DIM[1], DOM[2]=DIM[2], etc.)<br>>=0: map DIM<n> BSn to this port |
| crossbar[] | 1 | Select bit-stream RBS0 | int | 0-31 | 0 | *CV* | No | Causes specified DIM input bit-stream to be directed to RBS0 |
| | 2 | Select bit-stream RBS1 | int | 0-31 | 1 | *CV* | No | Causes specified DIM input bit-stream to be directed to RBS1 |
| | ….. | | | | | | | |
| | 32 | Select bit-stream RBS31 | int | 0-31 | 31 | *CV* | No | Causes specified DIM input bit-stream to be directed to RBS31 |
| QVALID_cntl[] | 1 | QVALID control mask | hex | 0x0-0x7 | 0x2 | *CV* | No | Bit 0 – set QVALID line to 'valid'<br>Bit 1 – enable DOM to manage QVALID<br>Bit 2 – set QVALID according to PVALID (system dependent)<br>If bits 1 and 2 set, QVALID is AND of DOM/PVALID |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| QDATA_cntl[] | 1 | Set QDATA control mask | hex | 0x00 to 0x0f | 0x0 | *CV* | No | Bit mask:<br>Bit 0 – Pass all media-embedded PDATA packets through to QDATA<br>Bit 1 – Enable transmission of DOT_set command at each ROT1PPS tick (with time adjusted forward by one second for proper setting of DOT clock); useful for media copying |
| send_QDATA[] | 1 | Send string to QDATA | literal ASCII | | *NA* | *None* | No | Causes specified string to be transmitted via QDATA |
| | [2] | Prescribed ROT time at which to send QDATA string | time | | *NA* | *None* | No | If specified, will send QDATA following specified ROT tick (see Notes); fractional seconds are ignored.<br>If not specified, will be sent immediately following next ROT tick. |
| tvg[] | 1 | Test-vector generator on/off | char | on \| off | off | *CV* | No | If 'on', replaces DOM output data with TVG data |
| transmit | 1 | Start/stop data transmit (playback) | char | on \| off | off | *CV* | No | See also Note 1 in Section 9.6 |
| | [2-n] | Parameters | - | | *SS* | *SS* | Yes | |

Notes:

1. A 'send_QDATA' command including a prescribed ROT time must be applied to the DTS no earlier than 60 seconds prior to prescribed time, nor no later than the close of the 'safe window' for the prescribed time; otherwise, an error will occur.

## **9.6** *DOM Queries and Responses*

| Keyword | Returned Field # | Description | Type | DTS-specific? | Comments |
|---|---|---|---|---|---|
| DPSCLOCK_source | 2 | Selected DOM clock source | char | No | Current value |
| | 3 | DPSCLOCK freq | int | No | Current value |
| QCTRL | 2 | QCTRL reverse-channel control | char | No | Current value |
| RCLOCK_frq[] | 2 | RCLOCK frequency | int | No | Value set by 'RCLOCK_frq' command |
| | 3 | Current actual RCLOCK frequency | int | No | Non-zero only if DOM is actively reproducing data. See Notes. |
| BSIR_R[] | 2 | BSIR used by DIM | int | No | |
| BS_mask_R[] | 2 | Bit-stream mask used by DIM | hex | No | |

| | | | | | |
|---|---|---|---|---|---|
| ROT | 2 | ROT status | int | No | 0 – ROT set enabled; 1 – ROT running |
| | 3 | Current ROT clock reading | time | No | Current ROT clock reading to at least 10 msec resolution.<br>Capture of this parameter must occur within 10 msec of receipt of ‘ROT?’ query |
| | 4 | Delay data wrt ROT clock | int | No | Delay value (in sample periods) in place at ROT clock reading in field 3. |
| | [5] | UT time of ROT clock reading | time | Yes | Reading of system clock at same instant that ROT clock reading is made.<br>Should be returned with maximum precision available; precision of at least a few 10’s of msec is required to be useful.<br>Applicable only if DOM has reasonably accurate internal UT clock (see Section 5.5) |
| portmap[] | 2 | DIM to DOM port mapping | int | No | Current value |
| crossbar[] | 2 | Select bit-stream RBS0 | int | No | Current value |
| | 3 | Select bit-stream RBS1 | int | No | Current value |
| | ….. | | | | |
| | 33 | Select bit-stream RBS31 | int | No | Current value |
| QVALID[] | 2 | QVALID line active/inactive | char | No | Current value |
| QVALID_cntl[] | 2 | QVALID control mask | hex | No | Current value |
| QDATA_cntl[] | 2 | QDATA control mask | hex | No | Current value |
| get_QDATA[] | 2 | Number of packets available for retrieval (including this packet) | int | No | Number of QDATA packets for retrieval by Controller;<br>if=0, no packets available. |
| | 3 | Number packets lost since last get_QDATA | int | No | Number of packets lost due to buffer overflow (oldest packets are always discarded first in buffer overflow situation) |
| | 4 | ROT clock reading | time | No | ROT clock reading when QDATA packet in field 5 was sent. Since QDATA data is always buffered until the *next* ROT tick, this will always be an integral second. |
| | 5 | QDATA data | literal ASCII | No | QDATA packet data (with any <SOH>,<C>,<CR> stripped) |
| tvg[] | 2 | Test-vector generator on/off | char | No | Current value |
| transmit | 2 | Data transmit on/off | char | No | ‘on’ or ‘off’. See Note 1 below. |
| | [3-n] | Parameters | - | Yes | |

Notes:

1. After transmit (playback) is turned ‘on’, the user should periodically query general ‘status?’ for details; if playback stops on its own accord (due to end-of-media, etc.), this will be reflected in the response to the ‘status?’ query as ‘transmitting stopped’, and a ‘transmit?’ query will show the status as ‘off’; a subsequent command to turn transmit ‘off’ or ‘on’ will reset the relevant bits (7-8) in the ‘status?’ response.
2. If ‘RCLOCK_frq’ is set to ‘0’, the RCLOCK frequency is forced to be same as BSIR at record time. In this circumstance, the actual RCLOCK rate will be data dependent and can vary from scan to scan; thus, the actual RCLOCK frequency can be determined only when data is actively being reproduced. Similarly, the original BSIR and bit-stream mask are data dependent and can be determined only when data is actively being reproduced.

## 9.7 *Media-Management Commands*

| Keyword | Field # | Description | Type | Allowed values | Power-on value | Default | DTS-specific? | Comments |
|---|---|---|---|---|---|---|---|---|
| media | 1 | Initiate media action | char | load \| unload \| pos \| stop | *NA* | *None* | Yes | Other DTS-specific values may be added as needed; ’media=’ command illegal during active media ‘receive’ or ‘transmit’ |
| | [2-n] | Parameters | - | | *SS* | *SS* | Yes | DTS-specific, as necessary |

Notes:

1. The use of ‘parameters’ in the ‘media=’ command is most likely needed when positioning media, where different types of media may require different parameters. For example, tapes may require linear distance measures, discs may require file names or byte positions, etc.

## 9.8 *Media-Management Queries and Responses*

| Keyword | Returned Field # | Description | Type | DTS-specific? | Comments |
|---|---|---|---|---|---|
| media_status | 2 | Media status | char | Yes | Mag tape: Suggest ‘loading’, ‘unloading’, ’unloaded’, ‘positioning’, ‘stopped’, ‘notready’, ‘active’ (receiving or transmitting);<br>Disc: Suggest ‘ready’, ‘notready’, ‘active’;<br>perhaps other DTS-specific responses |
| | [3-n] | Parameters | - | Yes | If positioning, report current position; other parameters as necessary |
| media_ID | 2 | Media identification | char | Yes | Single permanent media identifier, such as Volume Serial Number (VSN), associated with the current media |
| media_SN | 2-… | Media serial numbers | char | Yes | If relevant: List of media serial numbers |
| media_PN | 2-… | Media part numbers | char | Yes | If relevant: List of media part numbers or model numbers |
| media_size | 2-… | Media capacity (GB) | real | Yes | If relevant: List of media capacities in GB’s. |

## 10. VSI-S Glossary

Reference 1 should be reviewed for all terms and acronyms associated with the VSI-H specification. The following addition terms and acronyms are relevant to VSI-S:

### 10.1 *General*

| | |
|---|---|
| ASCII | American Standard Code for Information Interchange; a commonly-used character set |
| EIA/TIA-574 | Formal specification for 'RS-232' on 9-pin connectors |
| EIA | Electronic Industries Assoc – a trade organization and standards-setting body in electronics and communications technology |
| Ethernet | Alternate name for IEEE 802.3 standard |
| IEEE | Institute of Electrical and Electronic Engineers |
| IP | Internet Protocol |
| NPAD | Non-Reliable Packet Assembly and Disassembly |
| ping | Standard Unix procedure for verifying link connectivity |
| PPP | Point-to-Point Protocol |
| RS-232 | An EIA standard for connecting machines with serial interfaces, also known as RS-232-C or EIA/TIA-574; international counterpart is known as CCITT V.24 |
| TCP | Transmission Control Protocol |
| TIA | Trade organization and standards-setting body in communications technology. See http://www.tiaonline.org/about/overview.cfm |

### 10.2 *VSI-S Specific*

| | |
|---|---|
| Packet (wrt P/QDATA) | ASCII message including terminating <CR> |
| Primary control interface | Ethernet control interface on DTS (see Section 3.1) |
| Secondary control interface | Optional RS-232 control interface on DTS (see Section 3.1) |
| Transaction | Message from the Controller, followed by response from the DTS (see Section 5.1) |
| Response time | Interval beginning at instant DTS has received complete message from DTS and ending when the DTS response is complete (see Section 5.2) |
| Response window | Maximum time allowed by Controller for response from DTS, which includes possible transmission delays between Controller and DTS (see Section 5.2); suggested response window is <500 msec, with a maximum allowed response window of 1 second. |
| Safe window | Allowed window between DOT/ROT ticks for safe issuance of commands to be executed on 'next' DOT/ROT tick (see Sections 5.4 and 5.5) |
| Communications break | Loss of connectivity between Controller and DTS (see Section 5.3) |

## 11. References:

1. 'VLBI Standard Hardware Interface Specification – VSI-H', Rev 1.0, 7 August 2000 (available at http://web.haystack.edu/vsi/index.html and http://www2.crl.go.jp/ka/radioastro/tdc/ivs/vsi/ and published in the International VLBI Service 2000 Annual Report)

**Appendix A: Revision History**

Revision 1.0, 13 February 2003

First issue