VSI-S Usage Examples

Draft Revision 1.2
3 January 2002

Table of Contents
1. Introduction
2. Usage Examples
   2.1 Setup and Record
   2.2 Setup and Playback
   2.3 Media Copy
1. Introduction
Presented in this document are several typical examples of VSI-S ‘conversations’ for a typical hypothetical VSI-S compatible system. DTS responses are indented for clarity.

2. Usage Examples

2.1 Setup and Record
Setup a DIM to record 8 bit streams at an effective sample rate of 16 Msamples/sec/bit-stream; set the DOT clock; start recording.

```plaintext
reset = system;
!reset = 0;
status?
!status? 0 : 0x0;
1PPS_source = alt1pps;
!1PPS_source = 0;
CLOCK_frq = 32 : 16;
!CLOCK_frq = 0;
DOT_set = 2002y182d16h32m30s;
!DOT_set = 0;

(about a second later…..)

DOT?;
!DOT? 0 : 1 : 2002y182d16h32m31.175s;
BS_mask = 0xff;
!BS_mask = 0;
BS_mask?
!BS_mask? 0 : 0xff;
receive = on;
!receive = 0;
status?;
!status? 0 : 0x80;

(sometime later…end of media)
status?;
!status? 0 : 0xa0;
receive?;
!receive? 0 : off;
```

reset system
reset successful
query system status
OK
specify 1-pps tick from alternate input
OK
Specify clock freq as 32 MHz; sample rate 16 MHz
OK
Enable DOT clock set on next ALT1PPS tick
OK
DOT running; current DOT clock reading
specify bit streams 0-7 as active
OK
query bit-stream mask
OK
start recording
OK
get status
recording
get status
recording stopped (due to hitting end of media)
amatically set to ‘off’ at end-of-media
### 2.2 Setup and playback

Setup DOM to playback the data recorded in Example 2.1. Reproduce the 8 recorded bit streams to DOM output bit-streams 8-15, respectively, at 8 Msamples/sec/bit-stream with 32 MHz DPSCLOCK; set the ROT clock, start playback.

```plaintext
status?
  !status? 0 : 0x0;
query system status
DPSCLOCK_source = dpsclock : 32;
  32 MHz DPSCLOCK
!DPSCLOCK = 0;
OK
RCLOCK_frq = 8;
  Reproduced data at 8 Mbps/bit-stream
!RCLOCK_frq = 0;
OK
DPS1PPS_source = dps1pps;
  Set tick source
!DPS1PPS_source = 0;
OK
ROT_set = 2002y182d16h32m35s;
  Enable ROT clock set on next DPS1PPS tick
!ROT_set = 0;
OK
(a bout a second later.....)

!ROT_set? 0 : 1 : 2002y182d16h32m36.875s; ROT running; current ROT clock reading
```

```plaintext
re-map input bit-streams 0-7 to
!crossbar = 0;
output bit-streams 8-15, respectively
OK
transmit = on;
start playback
!transmit = 0;
OK
status?;
get status
!status? 0 : 0x100;
playback pending (i.e. sync’ing)
(a few seconds later.....)

!status? 0 : 0x200;
playback active
status?;
get RCLOCK information
RCLOCK_frq?;
Retrieve current RCLOCK freq, plus original
!RCLOCK_frq? 0 : 8 : 8 : 16 : 0xff;
DIM BSIR (16) and original DIM bit-mask (0xff)

(sometime later...end of media)
get status
status?;
playback stopped (due to hitting end of media)
!status? 0 : 0x300;
transmit?;
automatically set to ‘off’ at end-of-media
!transmit? 0 : off;
query status
status?;
status sticks until next transmit command
!status? 0 : 0x300;
(either ‘on’ or ‘off’)
transmit = off;
OK
!transmit = 0;
status?;
!status? 0 : 0x0;
idle
```
2.3 Media copy

Copy from a DOM to a DIM using PDATA/QDATA to automatically set the DOT clock in the DIM. Assume various DOM/DIM clocks and clock ratios are already properly set. DOM and DIM commands are shown separately since they may be separate units.

DOM:

QDATA_cntl = 0x2;                  \hspace{1cm} Causes QDATA to issue a ‘DOT_set’ command at every ROT1PPS tick, with the time adjusted forward by one second for proper setting of the DOT clock in the DIM.

!QDATA_cntl = 0;                   \hspace{1cm} OK

transmit = on;                     \hspace{1cm} Start DOM playback

!transmit = 0;                     \hspace{1cm} OK

DIM:

PDATA_cntl = 0x10;                 \hspace{1cm} Enable DIM to execute DOT_set commands arriving via PDATA

!PDATA_cntl = 0;                   \hspace{1cm} OK

receive = on;                      \hspace{1cm} Start DIM record

!receive = 0;                      \hspace{1cm} OK