VDIF - VLBI Data Interchange Format

VDIF Task Force:
Alan Whitney, MIT (chair)
Mark Kettenis, JIVE
Chris Phillips, ATNF
Mamoru Sekido, NICT

2009.06.25
8th Intl e-VLBI Workshop
Madrid, Spain
Motivation & Execution

• Variety of VLBI data formats used internationally complicates easy international data transfer

• Internationally constituted VDIF Task Force appointed in Shanghai in June 2008 to study problem and create a recommended uniform transport-independent VLBI data-format standard

• Data-transport standard (VTP?) will be addressed separately

• Combination of data-format and data-transport standards will effectively replace proposed VSI-E
Assumptions

• Data are assumed to be one or more time series of uniformly time-sampled data
• Each time series may have its own sample rate, bits/sample and place of origin (i.e. station)
Major VDIF attributes

- Data may be single-channel or multi-channel
- Number of channels can be arbitrary (i.e. not confined to $2^n$)
- Data may be single bit or multi-bit samples
- Data are self-identifying wrt time tag, data source, #bits/sample
- Data can be decoded without external reference
- Data may be discontinuous in time (e.g. pulsar data)
- Data are packetized into Data Frames suitable for on-wire transfer as well as direct disk file storage
- Support data rates up to at least 100Gbps
- Non-VLBI specific; suitable for most any uniformly time-sampled data set
Hierarchical Data Structure

- Aggregate data flow is defined as a Data Stream
- A Data Stream is organized into self-identifying Data Threads
  - Each Data Thread may have its own #channels, sample rate, and bits/sample
- Each Data Thread contains a serial set of Data Frames
- Each Data Frame consists of a Data Frame Header followed by a Data Array
  - Data Array length may be chosen by user
  - Data Array may contain single-channel or multi-channel data
Illustration of multi-thread VDIF Data Stream
Data Frame Rules

• Each Data Frame has 16/32 byte header followed by a Data Array of user-specified length
• Data Frame length for a single Data Thread is fixed for a particular scan
• #Data Frames per second must be an integer
• Data Frame may not span a second boundary
• Data Frame length must be a multiple of 8 bytes
  – For Ethernet transfer, length would normally be chosen to be <~9000 bytes
  – length is allowed to be as long as one second
Data Frame Header Content

- Time (seconds since specified epoch)
- Frame # within second
- Stream ID
- Station ID (2-char ASCII code)
- ‘Data-invalid’ marker
- #channels
- Bits/sample
- ‘Complex’ (‘In-phase/Quadrature’ channels) data marker
- Data Array length
- VDIF version #
- Optional user-defined 16-byte extension
  - Up to 255 unique user-defined formats may be ‘registered’ so that they are easily identified
  - registry to be set up at Haystack VSI web site
## Data Frame Header Format

<table>
<thead>
<tr>
<th>Bit 31 (MSB)</th>
<th>Byte 3</th>
<th>Byte 2</th>
<th>Byte 1</th>
<th>Byte 0</th>
<th>Bit 0 (LSB)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Word 0</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>I₁</td>
<td>L₁</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Seconds from reference epoch₃₀</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Word 1</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Un-assigned₂</td>
<td>Ref Epoch₆</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data Frame # within second₂₄</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Word 2</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>V₃</td>
<td>log₃(#chns)₅</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Data Frame length (units of 8 bytes)₂₄</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Word 3</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>C₁</td>
<td>bits/sample-1₅</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Thread ID₁₀</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Station ID₁₆</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Word 4</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>EDV₈</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Extended User Data₂₄</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Word 5</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Extended User Data₃₂</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Word 6</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Extended User Data₃₂</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Word 7</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Extended User Data₃₂</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Byte order: little-endian
Data Array Format

• Data Array format is based solely on the #chans and #bits/sample (as specified in the corresponding Data Array Header)

• Adherence to the Data Array format specification is necessary to ensure that the data are properly interpreted
Data Frame ordering

• Data Frames from a **single source** will normally be transmitted and received in **strict time order**
• Data Frames transmitted through a switch or over a network are not guaranteed to arrive in order
• VDIF does not mandate strict Data Frame ordering within a Data Thread or among Data Threads, but some correlators (particularly legacy hardware correlators) may require strict ordering
Usage example 1

- Data Stream with multiple single-channel Data Threads (VLBI2010 model)
  - Supports arbitrary # of channels (one Data Thread per channel)
    - Allows better fine-tuning of aggregate data rate for better utilization of e-VLBI transfers
  - Supports 1 to 32 bits/sample (some packing inefficiency for some values of bits/sample)
  - Preferred for new equipment and applications
  - Best compatibility with software correlators
Usage example 2

• Data Stream with one or more multi-channel Data Threads
  – Multiple channels in a single Data Stream
  – Primarily targeted at legacy VLBI data sources
  – Limited to $2^n$ channels ($0 \leq n \leq 31$)
  – Limited to $2^k$ bits/sample ($0 \leq k \leq 5$)
  – Avoids ‘corner turning’ requirement
  – Adaptable to support some older equipment
‘Simple’ VDIF Data Stream

• Each Data Thread within a ‘simple’ VDIF Data Stream must have same:
  - # of channels
  - #bits/sample
  - data type (‘real’ or ‘complex’)
  - #Data Frames/sec
  - Data Frame Header Length
  - Data Array Length

• Expected to be most common usage

• Useful VDIF Format Designator is constructed as
  “<total sample-data rate> - <total #chans> - <#bits/sample> [- <#threads>]”
  e.g. 1024-16-2-1 or 1024-16-2
Note similarity to VLBA mode designation
‘Compound’ VDIF Data Stream

- A ‘compound’ VDIF Data Stream contains two or more intermixed ‘simple’ Data Streams, each of which is called a ‘Data Group’
- Set of numerical Thread IDs within each Data Group must occupy a unique, non-overlapping range
- Useful VDIF Format Designator is constructed as “DataGroup1 Designator> + <DataGroup2 Designator> + …. “
  e.g. 1024-16-2-16+256-8-2
File-naming conventions

- Applies only to data stored in named disk files
- File-name suffix ‘vdif’
- Otherwise, should conform to internationally agreed file-naming convention available at http://www.haystack.mit.edu/tech/vlbi/vsi/index.html
- Example: gre53_ef_scan035_fd=1024-16-2.vdif
  which specifies
  - Experiment: gre53
  - Station: ef
  - Scan name: scan035
  - VDIF Format Designator: 1024-16-2
VDIF Status

- VDIF Draft Release 1.0 has been available for community comment for ~6 months
  - Has been carefully reviewed by several key members of global VLBI community
  - Final ratification hoped for at this meeting
  - Ratification important because it allows FPGA/hardware designers to proceed
The Next Step – VLBI Transport Protocol (VTP)

- VTP is complementary to VDIF for data transported over high-speed networks
- What are the possible characteristics of VTP?
  - Transparently support current and future transport protocols (i.e. TCP, UDP, Tsunami, etc, etc)
  - Multi-cast support?
  - Negotiate (via TCP?) a mutually acceptable transport protocol between data source and data sink
  - Normally will be one VDIF Data Frame per transport packet
  - Define a ‘wrapper’ around each VDIF Data Frame to enhance data accountability
  - Support easy integration into VEX and SNAP command streams
  - Must be simple, easy to implement and easy to use

- Goal is to have draft VTP spec ready in a few months
Generalized 10GigE Data Distribution Concept

dBBC/
DBE/
VDBE

dBBC/
DBE/
VDBE
Thank you