VLBI Data Interchange Format - An Overview

Alan Whitney, Mark Kettenis, Chris Phillips, Mamoru Sekido

Abstract

One important outcome of the 7th International e-VLBI Workshop in Shanghai in June 2008 was the creation of a task force to study and recommend a universal VLBI data format that is suitable for both on-the-wire e-VLBI data transfer, as well as direct disk storage. This task force, called the VLBI Data Interchange Format (VDIF) Task Force, is the first part of a two-part effort, the second of which is addressing standardization of VLBI Transmission Protocols (VTP). The formation of the VDIF Task Force was prompted particularly by increased e-VLBI activity and the difficulties encountered when data arrive at a correlator in different formats from various instruments in various parts of the world. The Task Force proposed a streaming packetized data format that may be used for real-time and non-real-time e-VLBI, as well as direct disk storage. The data may contain multiple channels of time-sampled data with an arbitrary number of channels, arbitrary #bits/sample up to 32, 'real' or 'complex' data; data rates in excess of 100 Gbps are supported. Each data packet is completely self-identifying via a short header, and data may be decoded without reference to any external information. The VDIF Release 1.0 specification was ratified at the 8th International e-VLBI Workshop held in Madrid, Spain in June 2009, and is now being supported in several VLBI data systems currently under development. At the same meeting, a VTP Task Force, chaired by Chris Phillips of ATNF, was appointed to address the standardization of VLBI Transmission Protocols, which is an important follow-on to VDIF; the work of is this group is now underway.

1. Introduction

The VLBI Standard Interface (VSI) specifications—developed in the early 2000s and designated VSI-H and VSI-S—are aimed primarily at recording and playback systems, and they specify standards for a hardware/electrical VLBI data interface and a software control interface, respectively. These VSI specifications intentionally *do not address the format of the transported data.*

In recent years, a number of new VLBI data acquisition and capture systems have appeared, along with the increasing need to interchange data on a global scale, including real-time and near real-time transfer via high-speed network, as well as by standard disk-file transfer. These types of data transfers have been increasingly plagued by the lack of an internationally agreed upon data format, often requiring *ad hoc* format conversions that require both programming effort and computing/storage resources. Recognizing this problem, a so-called VSI-E ('E' for 'e-VLBI') specification, based on standard RTP/RTCP network protocol, was first proposed and implemented in 2003-2004, which specified both data formats and data transport mechanisms for real-time e-VLBI data transfer. Although VSI-E was comprehensive, it was never formally ratified by the larger VLBI community. Its adoption was further hampered by its complexity, and it has been largely abandoned.

The VLBI Data Interface Specification (VDIF) has a somewhat different goal from VSI-E, specifying only a *standardized transport-independent VLBI data interchange format that is suitable for all types of VLBI data transfer, including real-time and near-real-time e-VLBI, as well as disk-file storage.* The VDIF specification, unlike VSI-E, explicitly makes no attempt to define an on-the-wire data transport protocol, which is expected to be the subject of a subsequent specification document. The combination of VDIF, along with this follow-on data transport protocol specification will, when completed, essentially constitute a replacement for VSI-E. And although the VDIF specification makes no mention of data transport protocol, it has been developed with an awareness of expected methods of data transport, including network transport using various standard protocols, as well as physical or electronic transport of standard disk files.

2. VDIF Task Force

The 2008 International e-VLBI Workshop, held 14-17 June 2008 in Shanghai, China, included panel and group discussions specifically targeting the subject of creating an international data format standard. Those discussions led to the creation of a small, broadly-based international task force (subsequently known as the VDIF Task Force) to study the problem and make recommendations to the larger VLBI community. The proposed VDIF specification developed by the Task Force was ratified by the community at the 2009 International e-VLBI Workshop held in Madrid, Spain 22-26 June 2009.

3. Basic VDIF Structure

The discussions at the Shanghai meeting supported the concept of a 'framed' data stream format consisting of a stream of "Data Frames", each containing a short self-identifying Data Frame Header, followed by a Data Array (containing the actual samples), as shown in Figure 1. A similar format is already used by several current and proposed disk-based recording systems.

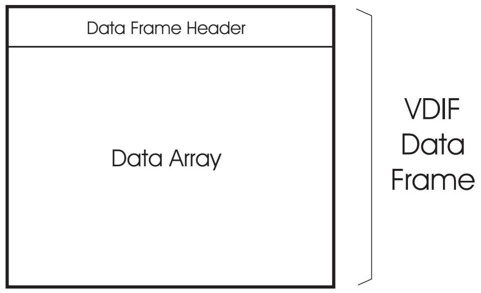

Figure 1. VDIF Data Frame structure showing Data Frame Header and Data Array

Accordingly, the VDIF specification is based upon a basic self-identifying Data Frame, which carries a time segment of time-sampled data from one or more frequency sub-bands. The length of a Data Frame may be chosen by the user to best match the chosen transport protocol; for example, in the case of real-time network transfer, a VDIF Data Frame length would normally be chosen so that exactly one Data Frame is carried by each on-the-wire packet. It is important to emphasize that the VDIF Data Frame is fundamentally transport-protocol independent, so that exactly the same set of Data Frames can represent VLBI data through a network transfer or be stored to a physical disk file.

In some cases, an entire set of sampled frequency sub-bands (or 'channels') may be carried in each Data Frame. In other cases, a single Data Frame may carry data from only a single data sub-band (channel) from among a set of many, in which case a logically parallel set of Data Frames is needed to represent the entire data set. In the VDIF concept, each time-series of Data Frames from the same set of sub-band(s) is known as a 'Data Thread', where each of the Data Frames within the Data Thread is identified by a 'Thread ID' embedded in the Data Frame Header. For actual transmission over a serial-data network, or for storage on a disk file, the set of Data Threads that comprise the data set are merged into a single serial 'Data Stream'. Figure 2 shows schematic example of a Data Stream comprised of three Data Threads. The collection of Data Threads from the beginning to end of a particular observation, typically lasting seconds to minutes, is known as a Data Segment.

In normal usage, it is expected that two types of Data Streams will predominate: 1) a Data Stream consisting of a single Data Thread carrying multi-channel Data Frames or 2) a Data Stream consisting of multiple single-channel Data Threads, though mixing of single-channel and multi-channel Data Threads is not prohibited.

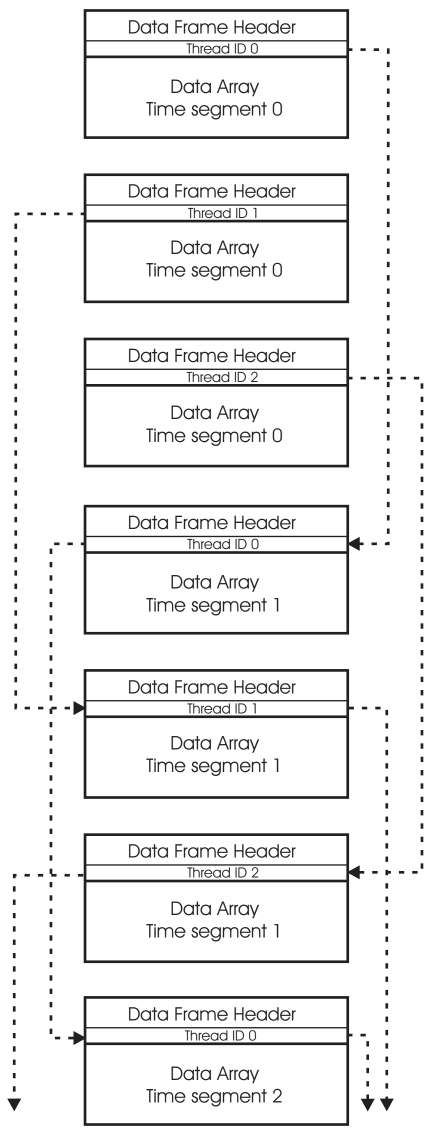

Figure 2. Illustration of Data Threads within a Data Stream

4. VDIF Attributes

The following considerations guided the creation of the VDIF specification:

1. The data in each Data Stream must be decodable using only information embedded within its constituent Data Frames.
2. A Data Thread may be discontinuous in time at the resolution of a Data Frame (e.g., transmitting and capturing Data Frames only during the active part of a pulsar period)
3. Each Data Frame may carry single-bit or multiple-bit samples up to 32 bits/sample.
4. Up to a maximum of 1024 Data Threads, each with a unique Thread ID, may be included in a single Data Stream.
5. A minimum of data manipulation should be necessary to move data between various data transmission techniques (e.g., disk file or real-time transfer).
6. Data rates of up to at least ~100 Gbps should be supported.
7. The data overhead (e.g., embedded auxiliary information required to meet the VDIF requirements) must be as low as practical.
8. Observations over leap seconds and year boundaries must be transparently supported.
9. The VDIF data format must be compatible, in as natural a way as possible, with all expected data transport methods (e.g., network transfer, file transfer, etc.).
10. Some limited amount of auxiliary user-defined data should be allowed in the Data Stream.
11. Within certain defined limits, out-of-time-order data within a Data Thread should be accommodated.

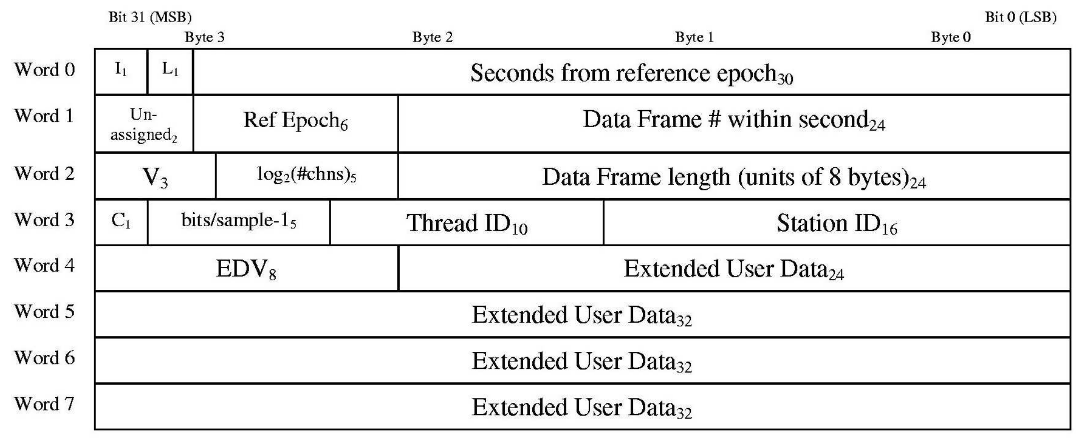

Figure 3. VDIF Data Frame Header format; subscripts are field lengths in bits; byte #s indicate relative byte addresses within a 32-bit word (little endian format)

5. Data Frame Rules

The following rules govern each VDIF Data Frame within a given Data Thread.
Each Data Frame contains a Data Frame Header followed by a Data Array.

1. All Data Frames must have the same Data Frame Header length, Data Array length, #channels, #bit/sample and Station ID.
2. If a Data Frame contains data from multiple channels, the same time-tag must apply across channels.
3. If a Data Frame contains multiple channels, all channels must be sampled with the same number of bits/sample.
4. Each Data Array contains sample data from one or more channels with the format (Section 9) and the encoding (Section 10) specified by VDIF.
5. The Data Frame length (including the Data Frame header) for each Data Thread must meet the following criteria:
    (a) Must be a multiple of 8 bytes (for maximum compatibility with various computer-memory-address schemes and disk-addressing algorithms).
    (b) Must be chosen so that an integer number of complete Data Frames are created in a continuous data flow of exactly one-second duration.
6. Data Frame #0 of each one-second period must contain, as its first sample, the data taken on a second tick of UTC; note that, in the case of time-discontinuous data, Data Frame #0 may not always be present.

These rules are intended to cover both 'on-the-wire' e-VLBI data formats as well as disk-file formats. For 'on-the-wire' real-time e-VLBI, it is expected that each transmitted non-fragmented packet will contain a single VDIF Data Frame as its data payload, in which case the Data Frame length is normally restricted to the range ~64/9000 bytes. These restrictions do not apply to disk-file data format, for which the Data Frame length is limited (by the number of bits available to specify the Data Frame length) to 2^{27} bytes (~134 MBytes).

6. Data Frame Header

Each VDIF Data Frame carries a header as shown in Figure 3, which may be either 16 or 32 bytes in length, depending on whether the Extended User Data words are included.

Details of the fields in the VDIF Data Frame Header are available in the VDIF specification.

7. Byte Ordering

Byte ordering of both the Data Frame Header and the Data Frame is little-endian (Intel x86 order) based on 32-bit words, which is consistent with most existing disk-based systems and software-based correlators.

8. Data Frame Ordering

Data coming from a single data source (e.g., a single dBBC or DBE) will normally be transmitted in strict time order. If directly connected to a local recording device, the recorded data will almost certainly be recorded in exactly the same order. However, Data Frames transmitted through a switch or over a network are not guaranteed to arrive in order.

The VDIF specification does not mandate strict Data Frame ordering within a Data Thread, but a best effort should be made to do so. Some correlation equipment, particularly older types, may be sensitive to Data Frame order, in which case the requirements of Data Frame ordering will be dictated by the correlation equipment. Modern software correlators are generally rather tolerant of minor Data Frame re-ordering of the type that might occur.

9. Data Array Formats

VDIF specifies the format of a Data Array based *solely* on the #channels and #bits/sample specified in the corresponding Data Frame Header. Since these two pieces of information are contained in each Data Frame Header, the samples in each Data Frame may be decoded with no external information.

The number of channels that can be accommodated in a multi-channel Data Array are limited to 2^n, but it is expected that most users will prefer to use single-channel Data Array. The use of multiple single-channel Data Threads both allow the user to transmit an arbitrary number of channels, as well as being a more compatible format for the evolving generation of software correlators.

Any number of bits/sample from 1 to 32 are supported, though the Data Array may contain some pad bits for certain values of bits/sample.

Samples may either be 'real' or may occur in 'complex pairs', such as are sometimes used in standard digital signal processing algorithms.

10. Sample Representation

VDIF-encoded data samples are represented by the desired number of bits in a fixed-point 'offset binary sequence', beginning with all 0's for the most-negative sampled value to all 1's for the most-positive sampled value. For example, 2-bit/sample coding is (in order from most negative to most positive) 00, 01, 10, 11. This coding is compatible with existing Mark 5B, K5 and LBADR disk-based VLBI data systems, though bit-ordering may be different in some cases.

11. Summary

The VDIF specification is one more piece of the on-going effort to achieve Global standardization of VLBI. It will work, however, only if VLBI community members are convinced that VDIF is good both for them and the greater community. Indications to date are that the VDIF is being well-received. The authors, as members of the VDIF Task Force, are, of course, always open to constructive comments and suggestions that may strengthen VDIF further. The full VDIF specification is available at *http://www.vlbi.org*.