VLBI setups for correlation using DBBC backends, FILA10G formatters and VDIF data format

Matteo Stagni

June 7, 2016

IRA 495/16

Referee: S. Buttaccio, M.Nanni

Abstract

A description of the current setups used in VLBI introducing hardware (DBBCs and Fila10G) and software (VDIF data format) changes. This paper provides the correlator point of view in changes applied to VEX files and in data formats when dealing with VLBI experiments of a variety of possible Astronomical and Geodetic setups.

Contents

1 Introduction

Since the introduction of software correlation in our institute in 2013, there has been a series of difficulties in interpreting both the hardware and software VLBI setups. The primary cause could be addressed to the newly introduced backends (DBBC) and formatters (FILA10G) at the Italian antennas (Medicina - Noto and Sardinia Radio Telescope), because this change affected many facets of the pipeline production of VLBI data, including the production of schedules through SCHED software and the way to interpret these changes in a software correlator like DiFX.

Even though one of the main purposes of the new backends is to introduce a higher degree of flexibility in the setups, like introducing higher data rates of observing, there has been a dramatic impact on the failure rate of observations, since the scheduling software SCHED and SKED were not able to cope with these rapid changes and as a result the observing schedules produced were (and still are) describing setups that do not exist anymore and are completely overridden by the present hardware. Moreover these changes require to match the way software correlators interpret data, thus incurring in minor and major non-documented software tweaks both on the production of schedule files and the way a correlator processes the data produced by the antennas.

In this paper the aim is to document in the clearest way the most common pitfalls the author has encountered during correlation issues and the solutions adopted once the setups were interpreted correctly.

In describing all the known setups at present the goal is to establish some major guidelines and best practices at least for all VLBI operations in Italy.

2 MARK5B

The Mark5B header format is the *de-facto* present standard when dealing with VLBI data. It is supported by a number of recorders and correlators, though its dimension was not designed with network capabilities in mind. Its biggest downside lays in its frame length, 10000 bytes which cannot be supported by standard network appliances which are limited to Jumbo frames up to a size of 9000 bytes.

Looking at the composition of the header we can find a useful sync word at the beginning (*ABADDEED*), followed by the frame number within second, which depends on the sampling rate decided for the observation.

There is a sensitive bit that follows the frame number section in Figure 1 marked as T which signals the presence of the *Test Vector Generator*, random noise produced by the formatter to test the hardware. The presence of this bit has proved difficult to manage in recent times because most correlators have been programmed to discard such data,

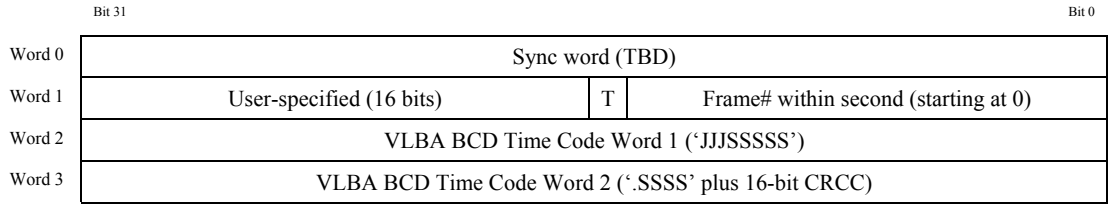

Figure 1: Disk Frame Header format [9]

whereas it is occasionally included into the frame number section like when the sampling rate of the VLBI experiment rises over 2 Gbit/s.

The last two words of the Mark5B header are composed by a 4 bytes time code where the first three numbers represent a shortened Modified Julian Date day (*JJJ*) and the last five numbers the second of the day starting from 0 at midnight (*SSSSS*). The following word contains the fractional part of the second (*.SSSS*)[1]and 2 bytes of CRCC (usually marked as zeros).

2.1 MARK5B data format on the network

In order to transmit Mark5B data packets on a TCP/IP network in UDP mode, a Fila10G board splits the 10000 bytes Mark5B frame into two 5008 UDP packets, preceded by a Packet Serial Number (*PSN*). The serial number is identical for the two packets, though only the first one contains the header. It is usually the task of the recorder program to discard single packets or in case of a successful transmission, to join them together and write the reconstructed frame to disk.

This design complicates the analysis of the header, in this case the sync word in the Mark5B header helps to determine whether the packet contains a valid header or not.

[1]When using a Fila10G formatter, in anticipation of the VDIF design, the fractional part of the second is absent, usually marked as zeros. It is thought to be sufficient to use the frame counter to determine the fractional part of the second. This is not that case when the formatter is a Mark5B recorder, which fulfills the complete standard and marks the fractional part of the second according to the connected PPS (Pulse Per Second) from a Maser clock

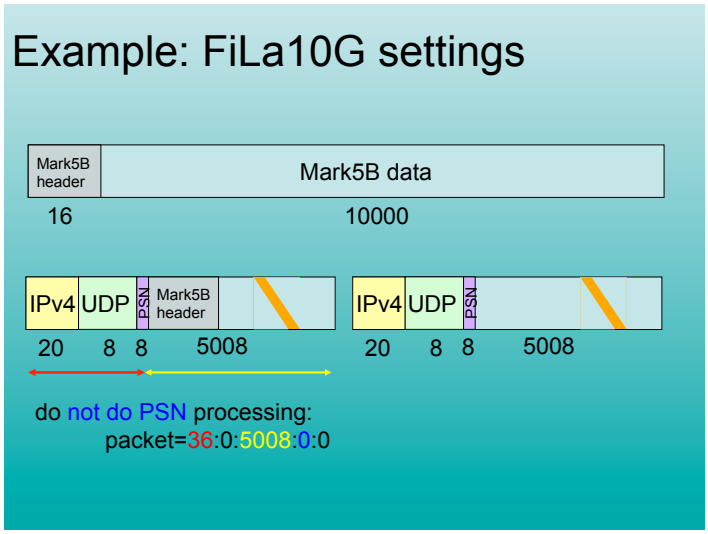

Figure 2: Mark5B network data packets as produced by a Fila10G formatter board

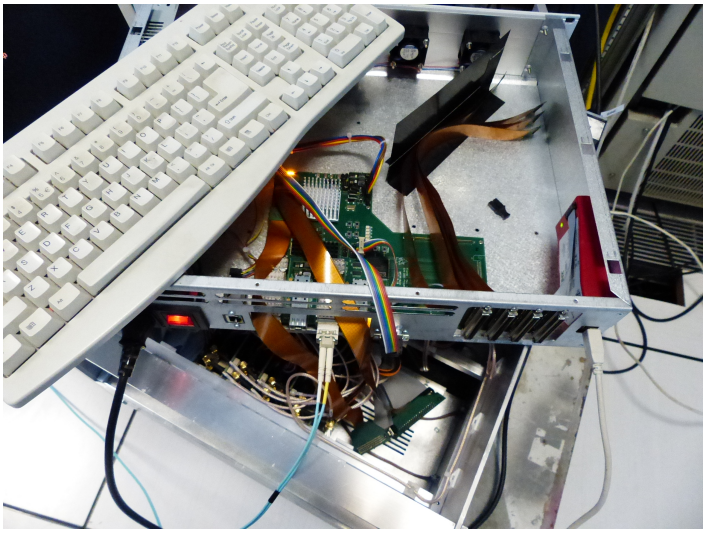

*Figure 3: A Fila10G (**Fi**rst - **la**st) external board*

3 VDIF

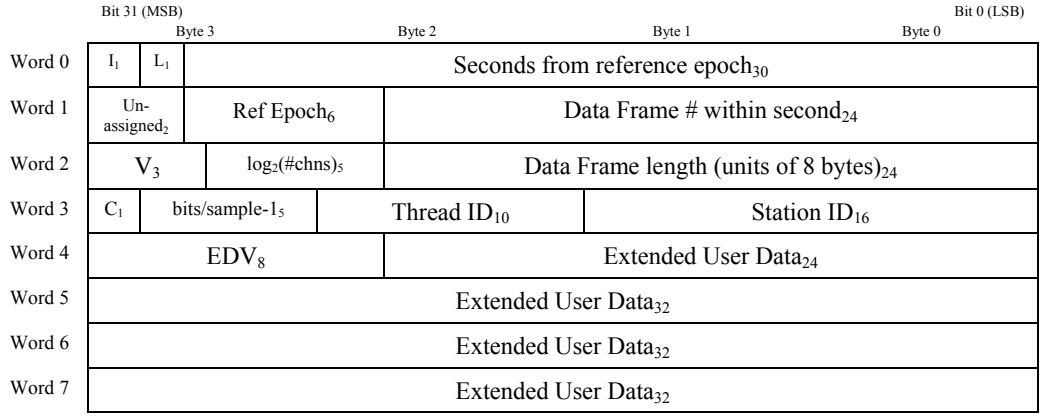

Figure 4: VDIF Data Frame Header format; subscripts are field lengths in bits; byte numbers indicate relative byte address within 32-bit word in little endian format [5]

The VDIF data format has been designed to overcome all the problems previously mentioned. Every packet that is sent to the network, or reordered, contains an header and in Word 2 is defined the data frame length which one would sensibly limit to fit into a Jumbo frame.

Timestamp inside the header has seen an evident change from the Mark5B format. Now there is a 6 bit Reference Epoch counter in Word 1 (*Ref Epoch*) that is incremented every six months starting from year 2000 (0), so the seconds now do not reset at midnight every day, but instead begin from the reference epoch considered. Seconds can be found in Word 0 and even if this design may not be thought as straightforward as the previous one, it prevents leap second problems.

A data frame number is still present to mark the sampling rate of the experiment and determine the fractional part of the second. In this case more room has been allowed (3 bytes) to fulfill the requirements of higher rates.

3.1 VTP - VDIF transport protocol

In addition to this design there are 8 bytes that come before the VDIF header, similar to the Mark5B PSN previously mentioned. When using a Fila10G formatter in recent firmare implementations the VTP 'pre-header' is a counter that starts the moment the Fila10G begins sending data. This may be helpful in case of missing packets or packets that come in wrong ordering due to the stateless UDP transfer protocol.

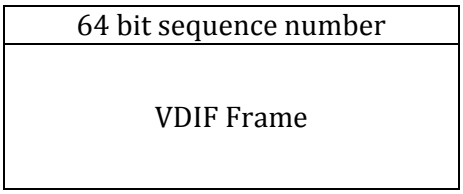

Figure 5: VTP/UDP packet structure [6]

4 VEX

The 'VEX-file' format (VEX = 'VLBI Experiment') has been invented to prescribe a complete description of a VLBI experiment, including scheduling, data-taking and correlation. This includes all setup and configuration information, as well as the schedule of observations. VEX is designed to be independent of any particular VLBI data-acquisition system or correlator, and is expandible to accommodate new equipment, recording and correlation modes. Every attempt has been made to consider the requirements and concerns of both the astronomy and geodetic VLBI communities in the construction of the VEX format [1].

VLBI data formats at present are multiplexed. This is an artifact from the time when data were recorded onto magnetic tapes. The tape was divided into tracks and each sub-band was split over one or more tracks. In principle the way these sub-bands are divided over the tracks is arbitrary and the exact mapping is recorded in the VEX file. The concept of a track is converted to a disk based format by packing all tracks into a stream of input words. The size of each data word in bits is equal to the number of tracks, with a logical one-to-one mapping of bit location inside a data word to a track number. Currently only the VDIF format [5] offers the possibility to record each sub-band in a separate data frame and therefore avoid the need for corner turning, although VDIF supports multiplexing as well [4].

The present major limitation of the VEX file is based on its skeleton that was designed when VLBA backends were dominant and tapes were the common means of recording data at high rates. In fact since 2009 a committee was established in the VLBI community to overcome the intrinsic limitations of VEX files in respect to modern DBEs (Digital Back Ends) and move to VEX2 file format [3]. The current transition status to VEX2 is unknown, so to accomodate new definitions of hardware and setups most correlation softwares have established hacks that overcome these information gaps.

The following sections explain in detail how these hacks work when a DBBC backend is used and either Mark5B or VDIF data format is chosen.

5 DBBC correlation setups

The DBBC backend setups are shaped after what is described as the VSI (VLBI Standard Interface) that organizes how the data is sampled and decoded (playback) at the correlator [8]. The principle of the VSI ordering of samples could be described by the following "algorithm":

```
first Upper Side Band channels
     increasing BBC number
then Lower Side Band channels
     increasing BBC number
```

which is somehow 'broken' and 'fixed' by SCHED software when describing a VLBA setup in a VEX file (see Table 1). On the contrary the situation described in the 'algorithm' has to be strictly applied when VDIF data output is chosen for the experiment (see Table 2), as there are less means to correct the setup. The meaning of 'breaking' and 'fixing' the 'algorithm' is based on the fact that setups are described and linked together in multiple VEX block keywords. To understand a VLBI setup in VEX one must refer to the $FREQ, $BBC, $IF and $TRACK blocks, being $FREQ the most descriptive of all blocks and providing the right connections between all of them.

The most common hack to fit a backend that diverges from how the VLBA backend behaves is to change the $TRACK block in the VEX file, though when VDIF data got into use this has become a pitfall because data could not be described as MARK5B format anymore. To avoid the issue other means of describing the correct setups have been put in place, essentially by re-programming software correlators.

The subdivision of the sky frequency into channels for the DBBC is carried out in two different so-called 'personalities' - analogue to digital conversion of the DBBC which are DDC (Digital Down Conversion to Base Band of Independent Channels) and PFB (Multi Equispaced Channel Conversion to Base Band Polyphase Filter Bank) [7]. The latter has only been recently tested in VLBI to produce a better bandwidth sampling using overlapping channels. There is a forthcoming 'personality in the future DBBC2010 that is called DSC (Direct Sampling Conversion) where all the bandwidth of the receiver will not be sub-divided into channels and will require further adjustments when preparing a VEX file.

The commonly used DDC 'personality' in VLBI can support up to 4 IF inputs on the DBBC that take in 512 MHz each, namely 0-512 MHz (A), 512-1024 MHz (B), 1024-1536 MHZ (C) and 1536-2048 (D). The 4 IF can be sub-divided into up to 16 BBC channels from 1 to 16MHz or from 2 to 32 MHz, depending on the DBBC firmware. This setup flexibility has proven difficult to be rapidly implemented in SCHED software to produce a correct VEX file.

The setup that the DBBC advertises are GEO, ASTRO, ASTRO2, WASTRO, TEST, LBA, 8bit. These setups or so-called 'VSI modes' are a way to code the correct mapping of channels and bbcs. However, given the flexibility of the backend, these could be easily overridden. The problem that arises in this case is the impossibility to correctly match the information provided by a VEX file, unless asking each station to provide the setup in use.

5.1 DBBC + Mark5B data format

When a DBBC is connected to either a FILA10G formatter set in Mark5B data format or directly through a VSI cable to a Mark5B recorder and formatter, according to the setup chosen the $FREQ block of the VEX produced by SCHED software will look something like what is shown in Table 1. As mentioned before this breaks the VSI 'algorithm' standard because it is how a VLBA backend would organize the band conversion and channels sub-division. However in this case the 'hack' used to make ends meet is to re-organize the ordering in a correct manner by re-writing the legacy $TRACKS block that describes how bits were written on tapes - now on disks.

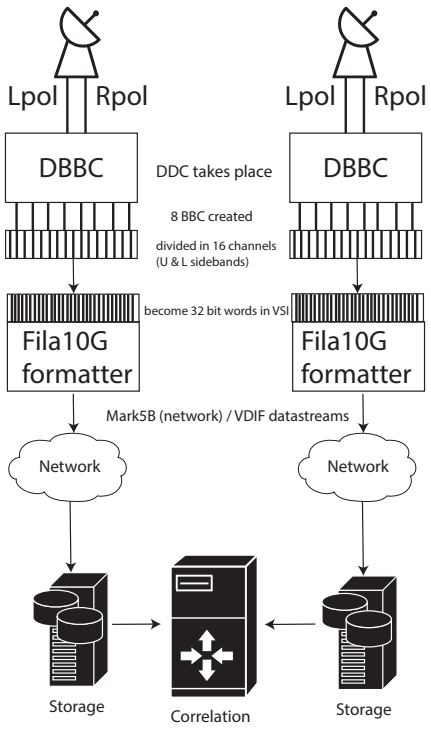

Figure 6: Data sampling pipeline from antennas through DBBCs and Fila10G formatters to correlator. This provides a behavior example of DDC ASTRO/ASTRO2/WASTRO DBBC personalities.

ASTRO MARK5B				ASTRO2 MARK5B				ASTRO3 MARK5B				GEO MARK5B			
CH	**Sideband**	**BBC**	**Pol**	**CH**	**Sideband**	**BBC**	**Pol**	**CH**	**Sideband**	**BBC**	**Pol**	**CH**	**Sideband**	**BBC**	**Pol**
1	L	BBC1	R	1	L	BBC1	R	1	L	BBC1	R	1	U	BBC1	R
2	L	BBC5	L	2	L	BBC9	L	2	L	BBC9	L	2	L	BBC1	R
3	U	BBC1	R	3	U	BBC1	R	3	U	BBC1	R	3	U	BBC2	R
4	U	BBC5	L	4	U	BBC9	L	4	U	BBC9	L	4	U	BBC3	R
5	L	BBC2	R	5	L	BBC2	R	5	L	BBC3	R	5	U	BBC4	R
6	L	BBC6	L	6	L	BBC10	L	6	L	BBC11	L	6	U	BBC5	R
7	U	BBC2	R	7	U	BBC2	R	7	U	BBC3	R	7	U	BBC6	R
8	U	BBC6	L	8	U	BBC10	L	8	U	BBC11	L	8	U	BBC7	R
9	L	BBC3	R	9	L	BBC3	R	9	L	BBC5	R	9	U	BBC8	R
10	L	BBC7	L	10	L	BBC11	L	10	L	BBC13	L	10	L	BBC8	R
11	U	BBC3	R	11	U	BBC3	R	11	U	BBC5	R	11	U	BBC9	R
12	U	BBC7	L	12	U	BBC11	L	12	U	BBC13	L	12	U	BBC10	R
13	L	BBC4	R	13	L	BBC4	R	13	L	BBC7	R	13	U	BBC11	R
14	L	BBC8	L	14	L	BBC12	L	14	L	BBC17	L	14	U	BBC12	R
15	U	BBC4	R	15	U	BBC4	R	15	U	BBC7	R	15	U	BBC13	R
16	U	BBC8	L	16	U	BBC12	L	16	U	BBC15	L	16	U	BBC14	R

Table 1: Channels, BBCs, Sidebands and Polarization assignments when using the several ASTRO and the GEO setups on a DBBC when **Mark5B data format** *is selected*

```
$TRACKS

* ASTRO
def MARK5B.16Ch2bit1to1;
* mode =  1    stations =Sr:Mc
*    firmware_type = DBBC_DDC;
*    format = MARK5B, and fan-out = 1
*    mode requires  32.00Mb/s/tr; stations using disks
     track_frame_format = MARK5B;
     fanout_def =   : &CH01 : sign : 1: 18;
     fanout_def =   : &CH01 :  mag : 1: 19;
     fanout_def =   : &CH02 : sign : 1: 26;
     fanout_def =   : &CH02 :  mag : 1: 27;
     fanout_def =   : &CH03 : sign : 1:  2;
     fanout_def =   : &CH03 :  mag : 1:  3;
     fanout_def =   : &CH04 : sign : 1: 10;
     fanout_def =   : &CH04 :  mag : 1: 11;
     fanout_def =   : &CH05 : sign : 1: 20;
     fanout_def =   : &CH05 :  mag : 1: 21;
     fanout_def =   : &CH06 : sign : 1: 28;
     fanout_def =   : &CH06 :  mag : 1: 29;
     fanout_def =   : &CH07 : sign : 1:  4;
     fanout_def =   : &CH07 :  mag : 1:  5;
     fanout_def =   : &CH08 : sign : 1: 12;
     fanout_def =   : &CH08 :  mag : 1: 13;
     fanout_def =   : &CH09 : sign : 1: 22;
     fanout_def =   : &CH09 :  mag : 1: 23;
     fanout_def =   : &CH10 : sign : 1: 30;
     fanout_def =   : &CH10 :  mag : 1: 31;
     fanout_def =   : &CH11 : sign : 1:  6;
     fanout_def =   : &CH11 :  mag : 1:  7;
     fanout_def =   : &CH12 : sign : 1: 14;
     fanout_def =   : &CH12 :  mag : 1: 15;
     fanout_def =   : &CH13 : sign : 1: 24;
     fanout_def =   : &CH13 :  mag : 1: 25;
     fanout_def =   : &CH14 : sign : 1: 32;
     fanout_def =   : &CH14 :  mag : 1: 33;
     fanout_def =   : &CH15 : sign : 1:  8;
     fanout_def =   : &CH15 :  mag : 1:  9;
     fanout_def =   : &CH16 : sign : 1: 16;
     fanout_def =   : &CH16 :  mag : 1: 17;
enddef;
*================================================
* GEO
  def Mk341_2f_1b-SX02;
    fanout_def = A : &CH01 : sign : 1 : 02 : 04;
    fanout_def = A : &CH02 : sign : 1 : 06 : 08;
    fanout_def = A : &CH03 : sign : 1 : 10 : 12;
    fanout_def = A : &CH04 : sign : 1 : 14 : 16;
    fanout_def = A : &CH05 : sign : 1 : 18 : 20;
    fanout_def = A : &CH06 : sign : 1 : 22 : 24;
    fanout_def = A : &CH07 : sign : 1 : 26 : 28;
    fanout_def = A : &CH08 : sign : 1 : 30 : 32;
    fanout_def = A : &CH09 : sign : 1 : 03 : 05;
    fanout_def = A : &CH10 : sign : 1 : 07 : 09;
    fanout_def = A : &CH11 : sign : 1 : 11 : 13;
    fanout_def = A : &CH12 : sign : 1 : 15 : 17;
    fanout_def = A : &CH13 : sign : 1 : 19 : 21;
    fanout_def = A : &CH14 : sign : 1 : 23 : 25;
    fanout_def = A : &CH15 : sign : 1 : 27 : 29;
    fanout_def = A : &CH16 : sign : 1 : 31 : 33;
  enddef;
```

The examples of $TRACKS provided encompasses the ASTRO and GEO setups. It is worthwhile noting that whenever the LO (Local Oscillator) frequency is above the observed sky frequency an inversion between the Lower Side Band and Upper Side Band occurs in the $TRACKS fanout_def bits. The odd numbering of the bits starting from 2 depends on the first 2 bits used by the tape controllers.

5.2 DBBC + VDIF data format

In case of VDIF data being selected on a FILA10G formatter, the $FREQ block of VEX must match the description provided in Table 2 when using DiFX software correlator [2]. In fact the hack that DiFX uses to supplement the lack of information provided for VDIF data by SCHED is to only process what is written in the $FREQ block and do not bother about what is written in the $TRACKS block. The SFXC [4] software correlator instead relies on newly introduced blocks that bridge a gap between VEX and VEX2, namely $BITSTREAMS and $THREADS that do the re-ordering of the incorrect information provided by the $FREQ section in VLBA style.

ASTRO VDIF				ASTRO2 VDIF				ASTRO3 VDIF				GEO VDIF			
CH	**Sideband**	**BBC**	**Pol**	**CH**	**Sideband**	**BBC**	**Pol**	**CH**	**Sideband**	**BBC**	**Pol**	**CH**	**Sideband**	**BBC**	**Pol**
1	U	BBC1	R	1	U	BBC1	R	1	U	BBC1	R	1	U	BBC1	R
2	U	BBC2	R	2	U	BBC2	R	2	U	BBC3	R	2	U	BBC2	R
3	U	BBC3	R	3	U	BBC3	R	3	U	BBC5	R	3	U	BBC3	R
4	U	BBC4	R	4	U	BBC4	R	4	U	BBC7	R	4	U	BBC4	R
5	U	BBC5	L	5	U	BBC9	L	5	U	BBC9	L	5	U	BBC5	R
6	U	BBC6	L	6	U	BBC10	L	6	U	BBC11	L	6	U	BBC6	R
7	U	BBC7	L	7	U	BBC11	L	7	U	BBC13	L	7	U	BBC7	R
8	U	BBC8	L	8	U	BBC12	L	8	U	BBC15	L	8	U	BBC8	R
9	L	BBC1	R	9	L	BBC1	R	9	L	BBC1	R	9	L	BBC1	R
10	L	BBC2	R	10	L	BBC2	R	10	L	BBC3	R	10	L	BBC8	R
11	L	BBC3	R	11	L	BBC3	R	11	L	BBC5	R	11	U	BBC9	R
12	L	BBC4	R	12	L	BBC4	R	12	L	BBC7	R	12	U	BBC10	R
13	L	BBC5	L	13	L	BBC9	L	13	L	BBC9	L	13	U	BBC11	R
14	L	BBC6	L	14	L	BBC10	L	14	L	BBC11	L	14	U	BBC12	R
15	L	BBC7	L	15	L	BBC11	L	15	L	BBC13	L	15	U	BBC13	R
16	L	BBC8	L	16	L	BBC12	L	16	L	BBC15	L	16	U	BBC14	R

Table 2: Channels, BBCs, Sidebands and Polarization assignments when using the several ASTRO and the GEO setups on a DBBC when **VDIF data format** *is selected*

```
*================================================
* DEFINITION OF VDIF TRACK BLOCK USING DIFX
*================================================
$TRACKS
def VDIF;
     track_frame_format = VDIF/8032/2;
enddef;

*================================================
* DEFINITION OF VDIF THREAD BLOCK USING SFXC
*================================================
$THREADS;
*** single-thread set-up
*** FORMAT: last field = total bit-rate over all threads
*** THREAD: thrd.ID : backend # : recorder # : data-rate of thread : N_chan : N_bit : : : bytes/packet ;
def VDIF512;
 format = VDIF : : 512;
 thread =  0 : 1 : 1 : 512 : 16 : 2 : : : 8000;
 channel = &CH01 : 0 :  8;
 channel = &CH02 : 0 : 12;
 channel = &CH03 : 0 :  0;
 channel = &CH04 : 0 :  4;
 channel = &CH05 : 0 :  9;
 channel = &CH06 : 0 : 13;
 channel = &CH07 : 0 :  1;
 channel = &CH08 : 0 :  5;
 channel = &CH09 : 0 : 10;
 channel = &CH10 : 0 : 14;
 channel = &CH11 : 0 :  2;
 channel = &CH12 : 0 :  6;
 channel = &CH13 : 0 : 11;
 channel = &CH14 : 0 : 15;
 channel = &CH15 : 0 :  3;
 channel = &CH16 : 0 :  7;
enddef;
```

The examples provided of the VEX blocks describe what is needed by either DiFX or SFXC to process VDIF data in a correct manner. DiFX requires only a definition of the payload of the data excluding the header (8000) and the number of bits per sample (2). SFXC instead uses a more articulated block $THREADS to reproduce what was implemented to change the bits order in case of Mark5B data format in the $TRACKS block. This is what actually could be required in case the future experiment will split the bandwidth to be correlated across several correlation facilities.

6 Conclusions

The setups described in this paper are the product of several information sources including email conversations, Haystack memos and HatLab papers. The effort infused is to better document and pack together all available information in a more robust way.

VEX2 could be a solution to all the aforementioned problems, though a partial implementation exists at Jive for the SFXC software correlator, the effort to switch to VEX2 in SCHED seems far from being implemented.

The increasing bandwidth in observations imposes dire challenges in networks and storage solutions.

The hope for the future is that during these times of transition in interferometrical observations that are paving the way for future systems such as VGOS for Geodesy and SKA for Astronomy, a better harmonization and coordination of the data production pipeline is seriously taken into account. In fact 'traditional' VLBI experiments could provide many reliable solutions, testbeds and useful baselines during the time of transition to such systems.

References

[1] Whitney Alan, Lonsdale Colin, Himwich Ed, Vandenberg Nancy, van Langevelde Huib, Mujunen Ari, and Walker Craig. Vex file definition/example. Technical Report 1.5b1, NASA/GSFC/NVI, http://www.vlbi.org/vex/docs/vex%20definition%2015b1.pdf, January 2002.

[2] A. T. Deller, S. J. Tingay, M. Bailes, and C. West. Difx: A software correlator for very long baseline interferometry using multiprocessor computing environments. *Publications of the Astronomical Society of the Pacific*, 119(853):318, 2007.

[3] Himwich Ed. Vex definition. Technical Report 1.9996, NRAO, https://safe.nrao.edu/wiki/bin/view/VLBA/Vex2doc, February 2015.

[4] A. Keimpema, M. M. Kettenis, S. V. Pogrebenko, R. M. Campbell, G. Cimó, D. A. Duev, B. Eldering, N. Kruithof, H. J. van Langevelde, D. Marchal, G. Molera Calvés, H. Ozdemir, Z. Paragi, Y. Pidopryhora, A. Szomoru, and J. Yang. The sfxc software correlator for very long baseline interferometry: algorithms and implementation. *Experimental Astronomy*, 39(2):259–279, 2015.

[5] Chris Phillips, Alan Whitney, Mamoru Sekido, and Mark Kettenis. Vlbi data interchange format (vdif) specification. Technical Report 1.1.1, MIT Haystack Observatory and JIVE and CSIRO/ATNF and NICT, http://www.vlbi.org/vdif/, June 2009.

[6] Chris Phillips, Alan Whitney, Mamoru Sekido, and Mark Kettenis. Vtp: Vdif transport protocol. Technical Report 1.0.0, http://www.vlbi.org/, October 2013.

[7] Gino Tuccari. http://www.evga.org/teaching/ivs_school_2016/l03.pdf.

[8] Alan Whitney. http://www.haystack.mit.edu/tech/vlbi/mark5/mark5_memos/039.pdf.

[9] Alan Whitney and Roger Cappallo. Mark5 memo 019. Technical Report 19, MIT, HAYSTACK OBSERVATORY, http://www.haystack.mit.edu/, 2004.