Mark6: Design and Status

Roger Cappallo, Chet Ruszczyk, Alan Whitney

Abstract The Mark 6 system is a disk-based data capture and record system, optimized for VLBI. As a follow-on to the successful Mark 5 family, it increases the maximum record rate to 16 Gb/s, using high-performance COTS (Commercial Off-the-Shelf) hardware and open-source software. This paper presents the Mark 6 design, with special emphasis on the software, and its current and future capabilities.

Keywords Mark 6, Data Recording, VLBI

1 Introduction

The Mark 6 data capture and recording system has been developed in response to ever-increasing need for greater sensitivity in VLBI systems. In geodesy, for example, the VGOS (formerly called VLBI2010) system (Niell et al. 2007) is designed around relatively small (12 m), agile antennas, whose decreased gain is compensated for by increased bandwidths (4 GHz). Similarly, in astronomical instruments such as the Event Horizon Telescope (Doeleman 2010), which operates in the (sub) mm wavelength range, amplitudes are affected by extreme atmospheric coherence issues, and wide bandwidths are needed to get sufficient sensitivity over short timescales.

The Mark 6 system (Whitney and Lapsley, 2012) follows closely upon the design of the Mark 5 recording system (Whitney, et al. 2010), with two key improvements: the datarate (and thus recordable bandwidth) is increased by a factor of at least x8, and the design has been changed to be Commercial Off-The-Shelf (COTS) hardware, and entirely open-source software.

2 General Considerations

The goals for the Mark6 can be conveniently placed into two categories, one for essential goals that must be met, and the other of secondary goals, which are desirable, and should be met only if the effort to do so is not too great. All of the essential goals have been achieved, and all of the secondary goals are also expected to be feasible.

Principal goals

- 16 Gb/s sustained record capability
- support all common VLBI formats
- COTS hardware
- 100% open-source software
- relatively inexpensive
- upgradeable to follow Moores Law progress
- smooth user transition from Mark 5
- preserve Mark 5 hardware investments, where possible

Secondary goals

- 32 Gb/s (or more) burst-mode capability
- generalized ethernet packet recorder
- e-VLBI support
- single-step playback as standard Linux files

Roger Cappallo, Chet Ruszczyk, and Alan Whitney
MIT Haystack Observatory, Off Route 40, Westford, MA, USA

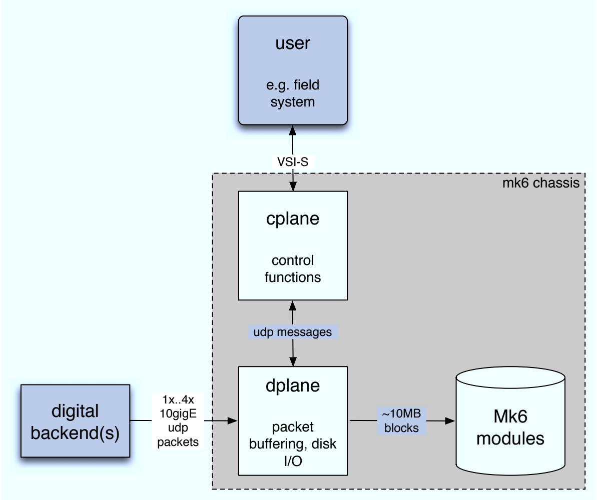

Fig. 1 Top-level block diagram of the Mark 6 software modules, showing the relationship of the two principal software modules, *cplane* and *dplane*.

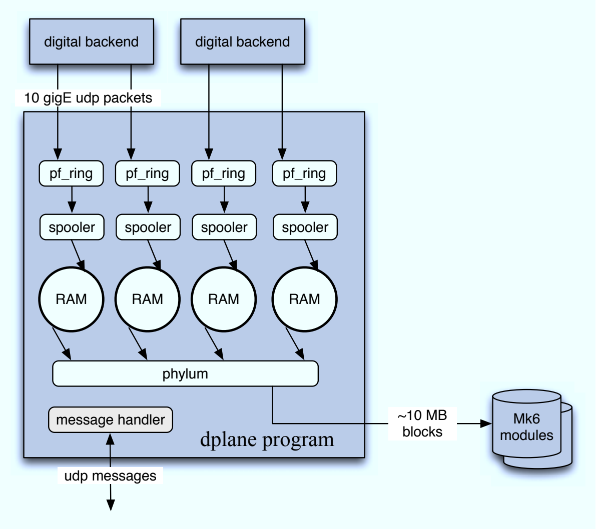

Fig. 2 Block diagram of the *dplane* software module. Each of the 4 parallel datastreams is processed by a separate thread, in a dedicated core for speed.

3 Software Design

The Mark 6 software has been designed using a layered model (see Fig. 1). The interface to the user (or more likely to the user's software) is accomplished by a program called *cplane* (for control plane). The interface to the network and disk hardware is handled by another program, called *dplane* (for data plane). The two programs/planes communicate via UDP messages.

3.1 Control Plane

The control-plane module *cplane* provides an interface to the user, and it implements a number of different high-level monitor and control functions. Due to its relaxed performance demands, *cplane* has been written in Python. The user interface is the standard VSI-S protocol (VSI-S 2003) with command set enhancements specific to the Mark 6 system. The *cplane* program is also responsible for managing the disk modules: it mounts, dismounts, and binds the individual disks as groups comprised of up to 32 disks. Performing and reporting error-checks as well as statuses is another *cplane* responsibility.

During the test and integration phase of the software, *cplane* has been controlled via an XML-based control script called RM6_CC. This script allows simple time-sequencing of scan-based observations. It is expected that a transition will eventually be made to control via standard experiment control software, such as the ”Field System” software from Goddard Space Flight Center (Himwich and Gipson 2011).

3.2 Data Plane

The data-plane module *dplane* handles all tasks specific to the high-speed flow of data. It reads in data from the 10 Gb/s network interface cards, buffers the data in large RAM buffers, and writes it out (if desired) into disk files on multiple disks. The start and end of the data is controlled precisely, by way of inspection of the time stamps within the data packets. Since the performance demands upon *dplane* are very high, the code is written in *C*, and is highly optimized.

The relevant hardware resources of the Mark 6 data pathway include:

- Intel core i7 3930K hex-core hyper-threaded processor
- ASRock Fatal1ty X79 Champion motherboard with 64 GB RAM
- 2 dual-channel 10 Gb/s NIC's
- 1 to 32 SATA hard drives

3.2.1 Architecture

The program is written with multiple threads, dedicating a processor core to each of the 4 input streams (see Fig. 2). The *pf_ring* I/O library (Deri 2004) is used for high-performance buffering of the incoming data packets. SMP affinity is used to spread the interrupt-handling load across different cores. The packets are then spooled into large circular FIFO buffers in RAM. These large ring buffers are given nearly all (e.g. 56 of 64 GB) of the physical memory space, and are locked in, saving only a modest amount of RAM for the OS to use for file caching, etc. The available space is used for 1–4 datastreams, and can be allocated on a dynamic basis if the user so desires. A single disk-writing thread empties these buffers, writing out data blocks to multiple disks.

3.2.2 Scattered File System

In order to be resilient to individual disk failures, and write-speed variations (which can be nearly a factor of 2 between the outer-edge starting tracks on the disks, and the inner-edge ending tracks), we have developed a scattered file system. This system writes blocks ($\approx$ 10 MB in length) to drives based upon which of the drives are ready to accept more data. For this application a RAID data-striping approach would not work so well, as the speed of the ensemble of disks would be limited by the slowest disk in the set.

In order to facilitate reassembly of the data into a continuous stream, a small amount of identifying metadata is prepended to each block. Three methods of data-reassembly may ultimately be used. Currently we use a progam, called *gather*, that efficiently reassembles the scattered data into a single disk file, in the correct time sequence. If data are missing then a fill-pattern is inserted in its place. The disadvantage of using *gather* is obvious – there is an extra copy step in the data pipeline, which is perhaps not an issue if the data need to be copied anyway onto a local storage device at the correlator.

A native-mode reader for the difx correlation software (Deller et al. 2012) is also planned for development, which would allow reassembly on the fly at correlation time. Finally, another flexible approach would be to write a FUSE (File-system in User Space) interface to allow the scattered files to appear as a single, standard Linux file.

There is a option of the program in which only a single file is written out, with no additional metadata inserted. In such a case it is not necessary to perform the extra reassembly step. This mode may be particularly attractive for modules comprised of solid state disks (SSD's), since they could be placed in a high-performance RAID configuration without much jeopardy from a slow disk.

3.2.3 Data Formats

The current software version supports the VDIF format (Whitney, et al. 2009) as well as the Mark5B format, which is converted to VDIF format. In the case of Mark5B input streams, the data are converted to VDIF encoding and a proper VDIF header is generated and prepended to each packet. Words 5 through 8 of the VDIF header then contain the original 4 word Mark5B header. For multiple input streams, each Mark5B stream is assigned to a separate VDIF thread.

3.3 Additonal Features

In order to facilate eventual use of the system as an eVLBI node, the capture of data to ring buffers is managed separately from file writing. The use of a single large FIFO per stream design decouples writing from capturing. This allows the system to keep writing to disks during the antenna slew time, so that the dataflow limitation is that the mean acquisition rate must not exceed the mean disk writing rate. However, an additional constraint is imposed by the finite size of the FIFO buffers. At an input datarate of 16 Gb/s ($\overline{2}$ GB/s) The large RAM is only about 30 seconds deep. This headroom is increased by the continual drain of data to disks; e.g. if 8 Gb/s were being written out, the buffer headroom would increase to 1 minute.

Mark 6 hardware is non-proprietary and the specifications and parts list are openly published. For convenience, but not by necessity, a Conduant chassis and Conduant modules may be used, as they are known to be both reliable and convenient. An upgrade kit, available from Conduant, is offered to facilitate reuse of Mark 5 modules in the Mark 6 system. A dedicated

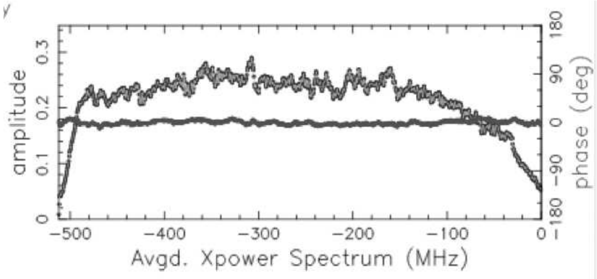

Fig. 3 Cross-power spectrum amplitude and phase, averaged over all four 512 MHz bands.

eVLBI site, though, may find it convenient to use non-Conduant hardware with Mark 6 software, as then there would be no module-interoperability concern.

4 Demonstration Experiment

In June of 2012 we performed a proof-of-concept experiment, which was used to demonstrate that the Mark6 works as intended, and that its combination with the digital backend signal processing performs as expected (see Whitney et al. 2013). A prototype version of the Mark 6 software was used, which captured 16 Gb/s onto 4 x 8 disk modules in RAID mode 0. An aggregate of 4 GHz on the sky was used to observe 3C84 on a Westford, MA – Goddard Space Flight Center baseline (see Fig. 3). The increase in signal-to-noise ratio due to the increased bandwidth was as expected, and no unusual anomalies were detected.

5 Future Plans

The Mark 6 system is expected to be used in an operational setting beginning in the summer of 2013, principally for wideband observations. Work is continuing on the software, with the following list of desired capabilities, placed in very rough priority order:

- performance enhancements and increased diagnostic tools
- native Mark 6 reader for the difx correlation software
- FUSE / Mark 6 file interface
- full support for generalized (i.e. non-VLBI) packet capture
- support for eVLBI via retransmission of the captured ring buffer

References

VSI-S Committee VLBI Standard Software Interface Specification. Web document http://www.vlbi.org/vsi/docs/2003_02_13_vsi-s_final_rev_1.pdf, 2003.

Deri, L. Improving passive packet capture: Beyond device polling. Proc. of SANE. Vol. 9. 2004.

A. Niell, A. Whitney, W. Petrachenko, W. Schlter, N. Vandenberg, H. Hase, Y. Koyama, C. Ma, H. Schuh, and G. Tuccari VLBI2010: A Vision for Future Geodetic VLBI in Dynamic Planet, International Association of Geodesy Symposia Volume **130**, 2007, pages 757–759.

Whitney, A., Kettenis, M., Phillips, C., and Sekido, M. VLBI Data Interchange Format (VDIF). Proceedings of the 8th International e-VLBI Workshop, PoS (EXPReS09), **42**, 2009.

Doeleman, S. Building an event horizon telescope: (sub)mm VLBI in the ALMA era. ”Proceedings of the 10th European VLBI Network Symposium and EVN Users Meeting: VLBI and the new generation of radio arrays. September 20-24, 2010.

A. Whitney, C. Ruszczyk, J. Romney, and K. Owens The Mark 5C VLBI Data System. *International VLBI Service for Geodesy and Astrometry: 2010 General Meeting Proceedings*

Himwich, Ed, and John Gipson. GSFC Technology Development Center Report. *International VLBI Service for Geodesy and Astrometry: Annual Report 2011*

A. Whitney and D. Lapsley. Mark6 Next-Generation VLBI Data System. *International VLBI Service for Geodesy and Astrometry: 2012 General Meeting Proceedings*

Whitney, A.R., Beaudoin, C.J., Cappallo, R.J., Corey, B.E., Crew, G.B., Doeleman, S.S., Lapsley, D.E., Hinton, A.A., McWhirter, S.R., Niell, A.E. Rogers, A.E.E., Ruszczyk, C.A., Smythe, D.L., SooHoo, J., Titus, M., (2013) Demonstration of a 16 Gbps per Station Broadband-RF VLBI System. PASP, **125**, pages 196–203.