VLBA Sensitivity Upgrade MEMO 19

MARK 5 MEMO 62

Mark5C−, A Proposed Mark5C Emulator

National Radio Astronomy Observatory & MIT Haystack Observatory

January 9, 2007

Abstract

At the U.S. VLBI meeting at Arecibo Observatory (Nov 29-30, 2007), the idea of a full emulator for Mark5C was discussed. In general the idea was very well received. In this memo the proposed emulator is described, the advantages of spending effort on this, and mention a possible extension to it that may have considerable strategic value.

1   Introduction

Mark5C will be a VLBI recorder system that significantly deviates from the philosphy of its ancestors. Mark5C accepts data over a single 10 Gigabit ethernet port rather than from a VSI-H or similar data parallel connector. Most VLBI data recorders have been restricted to record 2^n bitstreams, each being a particular digit in a binary word, and each such word written in strict time order [1]. The Mark5C will, on the other hand, allow recording of an arbitrary number of channels by forming data packets, each containing data from only one channel. Each packet will consist of about $\sim 10^5$ samples. This new recording system will offer much greater flexibility in the data it can record. Further the computer industry standard data port and packetized data format will allow for natural processing by the software correlators that are expected to dominate VLBI processing within the next several years.

The Mark5C is to be developed with a rapid schedule in order to meet the needs of the VLBA sensitivity upgrade and desires of other stakeholders. Rapid development is possible because the required new hardware is restricted to a new daughter board that plugs into the Amazon Sreamstor card. It is likely that the digital backend being developed in parallel will benefit from end-to-end testing before Mark5C is ready. For this and other reasons enumerated below the construction of a Mark5C emulator, operating well below the Mark5C bandwidth specification (at least initially), is thought to be extremely valuable. It is this emulator, here dubbed Mark5C− (*Mark 5 C minus*), that will be the focus of this memo. There is a strong possibility that the functionality described here will eventually be considered a sub-mode of Mark5C rather than an entity of its own. This Mark5C− project is a software-only project that is expected to require a negative amount of net effort in software, though requires significant Mark5C software investment to be made earlier than otherwise.

2   What composes Mark5C−?

Mark5C− is a full implementation of Mark5C on existing hardware. This is possible because the ethernet ports on PC motherboards are hardware compatible with the data input port on the Mark5C. On the Mark5C, the datapath from the data input port to the hard disk arrays includes a daughter board that attaches directly to modern (i.e., *Amazon*) streamstor cards (see Figure 1). In the case of Mark5C−, the data enters a standard Ethernet network interface card (NIC), crosses the PCI bus, into system memory, crosses the PCI bus again to end up on the Streamstor card, in exactly the reverse path indended for data playback (see Figure 2). With current generation hardware (1 Gigabit ethernet, PCI-X system bus) the data rate may be limited considerably compared to that of the Mark5C system. The Mark5C− architecture is, however, well positioned to take advantage of the natural maturing of the PC market. With a single 1 Gigabit ethernet port, it is expected that ~ 800 Mbps might be achievable. Using two 1 Gigabit ethernet ports, the rate could be doubled, but the bus speed of the PC motherboard will possibly be an issue.

---

[1] Note that Mark III typically recorded 14 or 28 tracks. Though not a power of two, this offered no flexibility of the form promised by Mark5C.

Mark5C− will be controlled by the same command set as Mark5C (See Mark5 memo **61**). Additionally, disk modules written by Mark5C− will be indistinguishable from those recorded with a true Mark5C recorder. These two facts together ensure that no software change to online telescope control code or correlator playback code will be needed upon migration to a Mark5C system.

2.1 Hardware

Starting from a Mark5 unit *of any generation*, no additional hardware is required to build a Mark5C−. In order to attach a Digital Back-End with a 10 Gigabit data port to a 1 Gigabit interface, even when employing severe data rate throttling (i.e., operating below 1 Gbps), an ethernet switch with 1 and 10 Gigabit ports will be required. This switch will eventually be needed in almost any conceivable systems that will eventually use Mark5C. The purchase of the switch could be eliminated by installing a 10 Gigabit ethernet port on the PC motherboard of the Mark5 unit, with the likely advantage of allowing higher data rates, perhaps 2 Gbps or more. A full 4 Gbps or faster Mark5C− will require that the Streamstor card be attached to the motherboard with a PCI-express bus; this will also be required for playback at these rates.
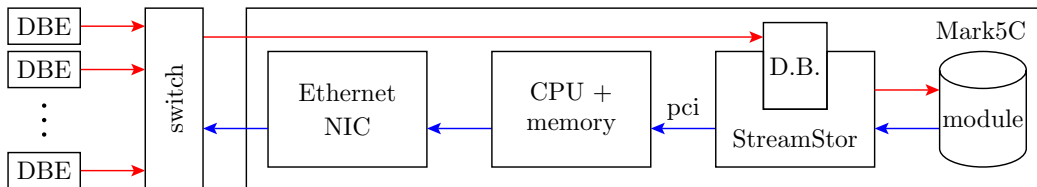

Figure 1: *The Mark5C data path.* Arrows show the path taken by VLBI data in recording (red lines pointing to the right) and in playback (blue lines pointing left). Note that on record, the high bandwidth data flows straight from the network switch to the StreamStor card daughter board (D.B.) and does not incur PCI bus limitations. The source of data on record is assumed to be one or more Digital Back Ends (DBEs); on playback it is assumed that the attached switch is part of a software correlator.
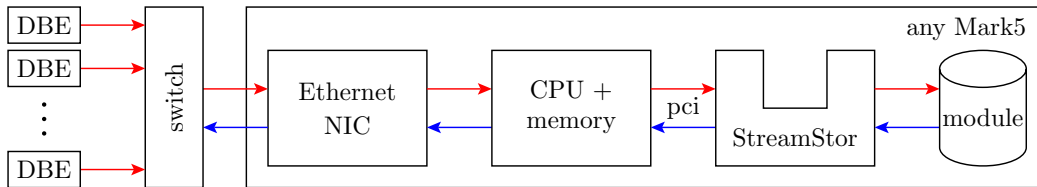

Figure 2: *The Mark5C− data path.* Here the record and playback data paths are symmetric; the standard Ethernet NIC carries data both directions. Note that any generation of Mark5 can be used.

2.2 Software

Software to support Mark5C− will be heavily based on that of Mark5C. If designed properly, the two programs (if separate) would share the entire overall program structure and would use different functions to implement some of the command set. The `record` function will likely differ the most between the two systems and will represent the largest Mark5C− specific code. This bit of code will be very similar to that of the `net2disk` function of Mark5B so code implementing this already exists in some form. Additionally, a Mark5C packet receiver test program already being planned is to contain a partial implementation of this; the Mark5C− software could be made to serve all the functionality of this already planned software and perhaps Mark5C− could be used instead for the packet receiver's intended purpose.

3 Why?

The reasons to implement Mark5C− are manyfold; this list is likely incomplete as additional uses for it will undoubtedly come up.

1. Allows testing of full data path using new Digital Back Ends, including fringe tests, before Mark5C can be delivered.

2. Mark5C− is a reference implementation of the Mark5C specification that can be used to aid Mark5C development.

3. Mitigates risk of a delay in the Mark5C hardware development.

4. Forces Mark5C software development to happen earlier.

5. Allows real-time software development cycle to begin earlier.

6. Can be generalized to explore new recording technologies (see below).

7. Combines two development projects (Mark5C test suite, and Mark5C control software) with the likely result of reducing the total software effort.

4 Extensions to Mark5C−

There is a cost to bringing the volume of data onto the CPU main memory, namely a likely reduction in data throughput, but by doing so, many new options become available.

A very natural extension to Mark5C− is to target a RAID array rather than a StreamStor disk module for storage. The philosophy behind Mark5C makes losing small amounts of data much less harmful than for previous generations of recorders. This reduction in absolute real-time may shift the technology of choice to less expensive, more standard (COTS) components.

Some recording modes may require software inspection of packets before deciding whether to record the data or not. A Mark5C− system could be extended to allow data selection. This is perhaps a simple system on which to test burst mode recording. In this case the target storage medium could be either a RAID array, a StreamStor disk module, or an eVLBI network socket.

It is thought that the pure Mark5C− described in preceding sections will not require change to the proposed (in progress) command set for Mark5C, except perhaps the addition of a new sub-mode switch. The extensions described in this section may require a richer set of commands.