VLBA Sensitivity Upgrade Memo # 31

VLBA Station Timing

R. C. Walker, NRAO
December 2, 2010

1. Introduction: The attempts to debug the July 2010 RDBE/Mark5C fringe tests revealed a need for an overview of how timing works with the new hardware at the VLBA stations. The timing responsibilities are spread across several subsystems with different people responsible for each subsystem. All of the necessary pieces are there and apparently working. This memo is an attempt to describe how the station timing is done so that those trying to understand the data produced with the new hardware can understand any impacts that the timing system might have. It is not meant as a specification document or as an attempt to second guess the design. The memo is the result of talking to a number of people including Hichem Ben Frej, Matt Luce, K Scott, James Robnett, Jim Ogle, Steve Durand, Walter Brisken, and Jon Romney.

When VLBI data are recorded, it is imperative to know the exact time, according to some stable station clock, of every sample. Even a one bit slip is a major problem in data analysis. That time has to be specified in a scheme that will not suffer ambiguities over periods of at least a few years because recordings can be aligned to arbitrary offsets. There is no maximum possible delay as there is in a linked system.

In practice, there are two major elements to the time keeping at a VLBI station. One is to maintain time at the microsecond level, typically with the help of one second ticks (1pps). These are derived by counting cycles of reference tones from the fundamental station clock – a highly stable hydrogen maser in the VLBA case. The second element of time keeping is to keep track of the UTC of the current second. Keeping track of seconds sounds like it should be the easier task because the process is a million times slower. In fact, it seems to be the harder task, or at least the one more likely to go wrong. This is related to the fact that there is a secure 1pps at the stations, but not a secure seconds count. This is nothing new. When the VLBA was first built, one second offsets at stations were common until new software tools were built to prevent them.

2. The Original VLBA System: Microsecond level timing is based on the maser, a secure 1pps generator (L109), a station timing module (L108) , and the formatter. A GPS timing receiver is used to sync the 1pps to UTC if needed and to monitor any offsets. The maser generates pure 5 and 100 MHz tones. The 5 MHz tone is used to generate the station 1pps. These parts of the system are not going to change. The 1pps is sent to the old 32 MHz Synthesizer module, where it uses the 5 MHz tone to synchronize the internal 1pps for the formatter with the station 1pps. The data time tags for the old system are based on the time kept in the formatter. Both the maser and the L109 are on the station UPS and also have their own internal batteries. Loss of synchronization of the station 1pps is rare and, I suspect, is generally associated with intentional maintenance activities.

In the old system, UTC is kept by the station computer and is based on the station 1pps. The VLBA formatter is automatically kept in agreement with the computer with no operator intervention. The computer clock can be set by the array operator from a screen with either an input time or by incrementing or decrementing a second. The site computer reads the time from the GPS receiver and

measures any difference with its internal clock. That difference can be displayed and used to help set the computer clock. Alternatively, a program called *vclock* can be run that pings each site to determine the propagation delays to the site, then reads the time and compares with a reference. Many sites can be done at once. Discrepant antennas are obvious and can be adjusted. *vclock* is run at the start of every project. The clock screen, that shows the station time, apparently runs at enhanced priority over other screens. The seconds update at a steady beat and don't skip. Other screens that show time, such as GPS or formatter, do not show steady updates and could not be used directly for setting the time.

I am not aware of other devices, except the Mark5A, that have internal clocks. The ACU does not, for example. The Mark5A contains a Linux computer which gets time from NTP. Other devices respond to the 1pps to align various signals, such as the 80 Hz for the Tcal, but do not need to know the count of seconds. Apparently the ACU just goes where it was last told to go by the control computer. Those commands are updated about 20 times per second.

3. RDBE Microsecond Level Timing: The RDBE uses the station 1pps for microsecond level timing, much like the old VLBA formatter. Like the old formatter, it generates its own internal 1pps ticks and syncs those to the external station tick. For the RDBE with the PFB personality in the FPGA (throughout this document, “FPGA” will mean the FPGA on the ROACH board unless otherwise qualified), this is a 2 stage process. The synthesizer board generates a 1024 MHz signal for the sampler. It divides that by four to derive a 256 MHz signal. From the 256 MHz signal, it generates a 20ms wide 1pps. That 1pps signal is synchronized to the station 1pps (1 μs wide) . The 256 MHz signal is not forced to have zero phase on any particular cycle of the 1024 MHz from which it is derived, so there are possible offsets up to 4ns in 1ns steps after resynchronization of the system (eg power cycle). Also, 1024 MHz is not an integer multiple of 5 MHz so the phase relative to the 5 MHz is not uniquely determined. This adds 5 possible delay offsets of up to about 1 ns to the overall timing uncertainty. The offset between the 20ms 1pps and the station timing 1pps is measured and reported but only in units of cycles of the 256 MHz clock used in the timing board FPGA or 4 ns. This helps determine if the timing has gotten badly out of sync, but could not be used for high resolution corrections for delay variations.

The 20ms 1pps from the timing board is sent to the ROACH FPGA . When using the PFB personality developed at Haystack, that FPGA generates an internal 1pps that is 12ns wide for its own use and for use by the samplers. The FPGA 1pps is based on counting the 256 MHz signal that it gets from the sampler. It is synchronized to the 20ms 1pps from the timing board. The sampler derives the 256 MHz signal from the 1024 MHz signal sent from the synthesizer board. Again, the 256 MHz tone is not forced to a particular cycle of the 1024 MHz and is independent of the 256 MHz in the timing generator, so there is an opportunity for a few times 1ns variations, covering the 4ns period of the 256 MHz tone, in the timing after a power cycle. The DDC personality uses the 1pps from timing board more directly for its timing rather than generate an internal 1pps. In either case, the timing is based on when a particular phase of the sampler's 256 MHz first arrives after the timing board's 1pps, so the timing will be the same for both.

There is no scheme in the FPGA for measuring any offset between the FPGA timing and an outside signal. For the planned DDC scheme, any such offset will be some constant due to propagation delays and latencies plus the 1024 to 256 MHz phase ambiguity. For the PFB which is free running after and initial sync, there is the off chance of a slew if cycles of the 256 MHz are missed. The FPGA 1pps is available on a connector on the RDBE so the offset can be measured using other equipment. Traditionally, a VLBI station's formatter 1pps has been compared with GPS to measure the station clock offset. The external version of the ROACH 1pps will allow that to be continued.

An effort is made to keep track of the time required for the data to propagate through the FPGA so that the time codes ultimately written to the data frame headers refer to the time the signals were sampled. A similar capability will probably be provided for the DDC personality which is still being designed.

An implication of the generation of the 1024 MHz from 5 MHz and the two 256 MHz tones from 1024 MHz (timing generator and sampler) without controlling the phase is that there is a term in any time offset between the time tags on the data and the station clock of up to 9ns that is set randomly each time the system is restarted (power cycle or loss of 5 MHz). Such events are expected to be uncommon, especially if the system gets the planned UPS, so they are very unlikely to happen during any one observation. The 9ns is from the sum of the 1ns due to the phase of the 1024 MHz tone, and the two 4ns uncertainties from the 256 MHz tones as described above. If the FPGA personality is changed without a power or 5 MHz loss, the signals from the timing board and from the sampler do not change. Therefore, the timing is not expected to change, but this should be tested.

An uncertainty at the sub-10ns level in the time tagging relative to the station clock, that does not change during any one experiment, will not affect the vast majority of observations. Thanks to the use of independent frequency/time standards, VLBI observations are tolerant of delay offsets considerably larger than that – typically to a half microsecond or more. In fact, the old system has differences of up to about 30ns between BBCs. If someone ever needs to know the absolute delay through the system, such as for time transfer, there might be a problem. Also if we ever try to do a priori clocks better than 10ns, perhaps for a real-time system, there will be a problem. The delay would need to be redetermined any time the RDBE is reset. The delay is measurable using the pulse cal and could perhaps be measured externally using the 1pps output. If there is a reset during an observation, there will be an impact on how the calibration needs to be done. Delays cannot be carried across the reset unless pulse cal data are used, and the pulse cal generator was not reset. I don't believe that all this will have a significant impact, but it is a effect that needs to be in the institutional memory in case it matters to someone in the future.

4. RDBE Count of Seconds: The ROACH FPGA keeps track of which second is current. When necessary, such as after boot, the PPC can set this time by telling the FPGA the current second. Typically this is done once, then the FPGA's idea of the current second is monitored and compared with the PPC's idea of the current second. The PPC has access to the FPGA 1pps. After a 1pps interrupt, the PPC gets the FPGA time and sends it out by mulitcast. That multicast can be used externally to compare times. The 1pps interrupt could in principle be used to help avoid the errors that might be incurred in comparing or setting times between the PPC and the FPGA in the unlikely event that the comparison or setting is done between the change of seconds on one device and on the other (the PPC is not synced to the station 1pps, so there will be an offset). The probability of such errors would scale with any offset between the PPC internal clock and the seconds transitions indicated by the 1pps in the FPGA. Any such offset is expected to be small in normal operations, probably in the vicinity of a millisecond. There is no battery to maintain the PPC clock through reboots or loss of power, so in such circumstances, the PPC time needs to be reset and there can be larger offsets. The monitoring of the offset between the FPGA and the PPC should catch any error if one happens.

The PPC gets its time from an NTP server. That server is the new Linux station computer. NTP is moderately sophisticated about synchronizing a computer clock, by adjusting its rate, to a server. It takes into account propagation delays so large separations are generally not a problem. The NTP server at each station in turn gets its time from two servers at the EVLA based on GPS and one at the GBT based on a maser. The multiple servers allow the system to have some chance of detecting when one is

giving spurious results. Since there is no battery for the PPC clock, after a boot the clock needs to be reset from scratch. The NTP server is asked to send the current time, which is used to set the PPC clock, probably to 0.1 sec or better. Then NTP goes about its business of adjusting the clock rate to bring the PPC clock into accurate alignment with the server clock. Soon after that initial time request is made, the PPC sets the FPGA on the assumption that the PPC time is already good enough to get the right second.

The Mark5C also gets time from the NTP server which should be adequate for activities like turning recordings on and off.

There is a utility running in the MIB in the VLBA version of the system that monitors the difference in times in the PPCs at the various VLBA sites, taking into account propagation delays. It could potentially look at other sites too. It can raise an alarm if the differences get too high. Since the PPC time is being compared to the FPGA time, this indirectly insures that the FPGA times at the different sites agree. As such, it serves a purpose like *vclock* although in practice it appears to be somewhat more stable.

The above describes various comparisons of the times in the FPGA, the PPC and between sites. At least for now, there is no automatic resetting of times if an offset if found. That requires manual intervention, probably by operations. That could be automated, but any loss of time is likely to be the result of a fairly serious event and it may be a good idea that the operators have to think about what is happening before the FPGA clock is reset.

5. Problems: Around the time of the early fringe tests, a few problems were found that affected the fringe delays or even the ability to get fringes. One deserves a mention because it had an impact on the firmware.

The wiring on the ROACH board inverts the voltage of the 1pps signal from the timing board from what was expected. As a result, some early versions of the FPGA firmware triggered on the wrong edge of the 20ms wide 1pps pulse causing a large clock offset. That offset could be seen in the timing of the power changes due to Tcal injection and in the zero baseline fringe check between the Mark5A and Mark5C done at Haystack based on Los Alamos recordings. Haystack avoided the error by using firmware to invert the tick and that will become standard.

6. Questions and Suggestions: The abilities of the old system to monitor station time relative to the local GPS timing receiver and to poll the stations for time offsets relative to a standard seem like they would be useful in the new system. With the NTP system, they won't strictly be necessary. But they are good sanity checks and in VLBI, where there is no immediate feedback from correlation, sanity checks are useful. While both systems are still there, this function could be covered by a utility, like the new one described earlier, that allows the difference between the concept of time in the PPC or Linux computer and the VME to be measured. It might also be possible to add the new system to *vclock* to allow the sanity check. As long as NTP is setting the times in the new systems, it would seem dangerous to allow operators to use *vclock*, or any other program to adjust those times.

The GPS talks to the old station computer through an RS232 port. That suggests that only one computer can communicate with the GPS. Once we abandon the old VLBA formatters, perhaps the GPS should be moved to the new Linux station computer where it can be used to compare times between the GPS, the station computer, the PPC, and, indirectly, the ROACH FPGA. It will still be necessary to insure good time in the old control computer too, because pointing depends on it. That

will happen if something is making sure that the old and new computers are in sync. In the long run, the new computer will take over control of everything, but that may take years.

7. A comment for future systems: This memo has attracted some attention from persons involved in the design of future systems such as the SKA. The system described above is strongly influenced by the design of the original VLBA system and has to live within the constraints imposed by the modules that will be retained, such as the secure 1pps (L109). With a new design, I would suggest that the equivalent of the secure 1pps, which is on very secure power, also keep track of the count of seconds. Then all other computers and devices at the station that need time should get it from that device. Meanwhile, GPS should be used to synch both the 1pps and the count of seconds when that is necessary, which should be rare. The relation between GPS and that device should be monitored an any discrepancies noted. But GPS is more likely to glitch than the secure system, so any resync should be done carefully by maintenance or operations personnel. The sites have some of the worlds most accurate clocks, and have GPS timing receivers that can deliver time at the 10s of nanoseconds level. Using utilities like vclock and NTP, that are very much less accurate, should not be necessary.