Atacama Large Millimeter / submillimeter Array

APP Tests on Absolute Timing

ALMA-05.11.61.03-0001-A-REP

2014-12-01

Prepared by:	Organization Role	Date and Signature
G. Crew	MIT	
Approved by:	Organization Role	Date and Signature
Authorized by:	Organization Role	Date and Signature

Change Record

Version	Date	Affected Section(s)	Author	Reason/Comments
A	2014-12-01	All	G. Crew	First Issue

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Purpose

This document describes tests conducted to establish the relative and absolute timing of the APP modifications to the ALMA system.

1.2 Scope

This document is (as of version A) limited to a specific test performed on October 18, 2014 during the software verification mission. https://ictwiki.alma.cl/twiki/bin/view/Control/AppMissionChile102014. Additional testing of this type is expected in future campaigns and will be collected in further revisions of this document.

1.3 Reference Documents

The following documents contain additional information, are referenced in this document, and should be consulted for further, more detailed information.

Table 1.1: *Reference Documents*

Reference	Document Title	Document ID
[RD1]	APP Update to Corr/Control Design	ALMA-05.11.61.01-0001-A-DSN
[RD2]	APP H Maser Procedures	ALMA-05.11.21.02-0001-A-PRO

1.4 Acronyms

ACS	ALMA Common Software
ALMA	Atacama Large Millimeter/submillimeter Array
APP	ALMA Phasing Project
FPGA	Field Programmable Gate Array
GPS	Global Positioning System
MIT	Massachusetts Institute of Technology
PIC	Phasing Interface Card
TE	Timing Event

Chapter 2

Analysis of Oct 18, 2014 Timing Experiment

The PICs receive two special timing pulses at a cadence of once per second. A counter of clock cycles between each pulse and the internal 1PPS signal is maintained by each PIC and reported as status. One of these signals comes from the “correlator” GPS unit (so called, to distinguish it from the ALMA GPS unit to which the Master Clock is synchronized.

During the October mission we examined these signals and adjusted the 1PPS coming from the Hydrogen Maser to verify which signal was which.

2.1 PIC Signals

A number of commands are necessary to program `ipython` to properly retrieve the PIC status.

```
# do this outside of an observation!!!

import Acspy.Clients.SimpleClient
import PolarizationTypeMod, BasebandNameMod
import Correlator

pic = client.getComponentNonSticky("CORR/PIC_CONTROLLER")
pids = [id1X, id2X, id3X, id4X, id1Y, id2Y, id3Y, id4Y ]
availableBasebands = [
    BasebandNameMod.BB_1, BasebandNameMod.BB_2,
    BasebandNameMod.BB_3, BasebandNameMod.BB_4]
idList = []
for bb in availableBasebands:
    idList.append(Correlator.PICId(bb, PolarizationTypeMod.BOTH))
    pic.checkPICStatus(idList, False)
for p in pids:
    print pic.getPICStatus(p).maserOffset, pic.getPICStatus(p).gpsOffset
```

When run, the output looks like this:

```
    # maser   gps   offset v 1PPS
    124999957 3434
    124999956 3433
    124999959 3436
    124999955 3432
    124999956 3432
    124999954 3431
    124999958 3435
    124999953 3430
```

and it may be repeated as desired

```
    # maser    gps   offset v 1PPS
    124999957 3434
    124999956 3433
    124999959 3436
    124999955 3432
    124999956 3432
    124999955 3431
    124999958 3435
    124999953 3430
```

but note that in normal operation, it is unlikely that any of the ticks will move by as much as a clock cycle (8 ns) in a short time.

At this time, the (temporary) monitoring of the Hydrogen Maser tick relative to the "Correlator" GPS unit was showing:

```
2014-10-20T18:20:23 1413829223.02  +9.99972147099777E-001
```

I.e., the Maser and GPS are nearly synchronous, separated by 27.9 us (1.0 - +9.99972147099777E-001 = 2.79E-008); *e.g.* $3432 * 8 = 27456$.

As described in Chapter 2 of [RD2], it is possible to remotely adjust the 1PPS signal that the Hydrogen Maser produces. The `ADJ1PPS` modifies a register that holds an offset (in nanoseconds) to the currently operating pulse:

```
crc-03 gcrew:gcrew 135> nc -u -p 14000 10.197.48.50 14000
GETID;
$GETID;Hard=T4S.711.oct12.b.37;Soft=T4S-SW-100-a-15;
RDFS;
$RDFS;=1420405750.294469;
RDREG=ADJ1PPS;
$RDREG=ADJ1PPS;=;
ADJ1PPS=337497800;
$ADJ1PPS=337497800;
```

I.e., the offset register is empty, and we can set it to a large value (0.337497800 seconds). Thereafter the external measurement shifts correspondingly:

```
2014-10-20T18:23:32 1413829412.01  +6.62474356064621E-001
2014-10-20T18:23:33 1413829413.01  +6.62474356020676E-001
2014-10-20T18:23:34 1413829414.01  +6.62474356030441E-001
```

(*I.e.*, $0.66247435606462 + 0.337497800 = 0.99997215606462$). This change is reflected in the PIC status:

```
for p in pids:
    print pic.getPICStatus(p).maserOffset, pic.getPICStatus(p).gpsOffset
    # maser    gps   offset v 1PPS
    124999956 42190659
    124999955 42190658
    124999958 42190661
    124999954 42190657
    124999955 42190657
    124999953 42190656
    124999957 42190660
    124999952 42190655
```

(*I.e.*, $42190655 * 8 = 337525240$ ns). Finally, restoring the Maser tick to its original state:

```
ADJ1PPS=000000000;
$ADJ1PPS=000000000;
```

and confirmed externally,

```
2014-10-20T18:40:11 1413830411.01  +9.99972145654465E-001
2014-10-20T18:40:13 1413830413.01  +9.99972145630051E-001
2014-10-20T18:40:15 1413830415.01  +9.99972145791183E-001
```

we find the same of the PIC signals:

```
for p in pids:
    print pic.getPICStatus(p).maserOffset, pic.getPICStatus(p).gpsOffset
    # maser   gps   offset v 1PPS
    124999956 3434
    124999955 3433
    124999958 3436
    124999954 3432
    124999955 3432
    124999953 3431
    124999957 3435
    124999952 3430
```

Since we *are* adjusting the Maser 1PPS, and we *are not* adjusting the Correlator GPS, and the first column is unchanged throughout, and the second column mirrors the change...it is rather clear that either the cables are swapped, or that the signals are mis-routed in the PIC FPGA design, or something equivalent. (The same offset discrepancies are seen in all PICs, so we are in any case looking at a systematic issue.)

Swapping the cables was the easiest solution, and that was done after the mission ended, but that the swap occurred is not in dispute, and the signals are both present. It can be confirmed at a later date.

2.2 Master Clock

From the preceeding section (Section 2.1) we note that the other signal, is that of the (unchanged) GPS unit, which is independent of the GPS unit which is used to synchronize the Master Clock after a full reset of the control system. Since there are 125 million ticks per second, we see that the PIC typically sees an offset of 44 ticks, or about 352 ns.

To understand this, we examine the system logs. A convenient way to convert time systems is using `ipython` with a simple command fragment:

```
import Acspy.Common.TimeHelper
import Acspy.Common.EpochHelper
import acstime

# e.g.
gps = 136329495596620000L
eh = Acspy.Common.EpochHelper.EpochHelper(gps)
fmt = "%Y-%m-%dT%H:%M:%S." + ("%06d" % eh.microSecond())
eh.toString(acstime.TSArray, fmt, 0, 0) + ' (converted)'
```

In the following fragments from one log file `log2014-10-18T18:23:19.938-2014-10-18T18:32:33.639-AOS.xml` we insert such conversions to allow us to correlate the log comments with the (presumably higher precision data reported). For readability, some line breaks have been inserted:

```
2014-10-18T18:25:59.648 Resetting the TE handler command fifo.
2014-10-18T18:25:59.662000 (converted)
2014-10-18T18:25:59.714 GPS time = 136329495596620000[100ns].
2014-10-18T18:25:59.726000 (converted)
2014-10-18T18:25:59.775 GPS time = 136329495597260000[100ns].
2014-10-18T18:25:59.777 The estimated time shift will be
```

```
                        (CRD - GPS clock) = 0.0000000000s.
2014-10-18T18:26:00.032000 (converted)
2014-10-18T18:26:00.085 GPS time = 136329495600320000[100ns].
2014-10-18T18:26:05.761 The phase of the TE signal has been realigned
                        with the 1pps signal of the GPS clock. Please
                        expect the LORR to report a condition about
                        the TE phase. The LORR needs to resync itself
                        with the new TE phase. The reset time of the CRD
                        has been set to 136329495660000000[100ns]
                        (target time system).
2014-10-18T18:26:06.000000 (converted) RESET TIME OF CRD
2014-10-18T18:26:06.326000 (converted)
2014-10-18T18:26:06.386 GPS time = 136329495663260000[100ns].
2014-10-18T18:26:06.398000 (converted)
2014-10-18T18:26:06.452 GPS time = 136329495663980000[100ns].
2014-10-18T18:26:12.000000 (converted) T0 for the array
2014-10-18T18:26:12.048 teHandler time stamp = 136329495720000000[100ns].
2014-10-18T18:26:14.665 The MasterClock has successfully set the CRD reset
                        time to 136329495660000000[100ns] and T0 for the
                        ArrayTime to T0 = 136329495720000000[100ns]. From
                        now on all time information on this computer will
                        be target time system.
```

From this one should note that ACS (and the underlying kernel timekeeping resolution) is only 100 ns. Unfortunately, the reported results are only presented to the nearest microsecond. This is not inconsistent with the 400 ns timing error observed in Section 2.1. Note however, that two independent GPS units should generally agree to a much better resolution than that. (The "correlator" GPS unit reports a RMS error of 33 ns.)

On the other hand, the PIC timing is driven by the TE which is aligned with the 1PPS, but generated separately. So, there are a number of places where an error such as this might be expected to creep in. It would take more testing to resolve this, but as long as it is small (*i.e.* of this order of magnitude) we do not really care.

2.3 Summary

The timing reported by the system is consistent with expections to within a few hundred nanoseconds. Normal "fringe search" windows are typically a few microseconds, so this is adequate.