Atacama Large Millimeter / submillimeter Array

APP Mark6/OFLS/PIC Acceptance Report

ALMA-05.11.50.03-0001-A-REP

2014-11-30

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	2014-11-30	All	G. Crew	First Issue

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Purpose

This document describes the tests performed at the ALMA OSF during (software) engineering time to establish the correct operation of all hardware components (PAS/Acceptance).

1.2 Scope

This document is limited to specific tests and discussions that relate directly to the hardware performance and requirements verification. The test conducted for software purposes and as (engineering time) preparation for eventual commissioning are not discussed here.

There are extensive notes on the software verification missions from the three campaigns conducted so far to be found in the ALMA twiki:

R10.6 https://ictwiki.alma.cl/twiki/bin/view/Control/AppMissionChile012014

R2014.2 https://ictwiki.alma.cl/twiki/bin/view/Control/AppMissionChile072014

R2014.4 https://ictwiki.alma.cl/twiki/bin/view/Control/AppMissionChile102014

The tests described in this document were carried out (or repeated) during the October 15-28, 2014 campaign.

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 Mark6 Recorder Test Procedures	ALMA-05.11.50.02-0001-A-PRO
[RD3]	APP Tests on Absolute Timing	ALMA-05.11.61.03-0001-A-PRO
[RD4]	ICD between APP and ALMA Computing	ALMA-05.11.00.49-0004-B-ICD

1.4 Acronyms

ACS ALMA Common Software

ALMA Atacama Large Millimeter/submillimeter Array

APP ALMA Phasing Project

CAN Controller Area Network

CASA Common Astronomy Software Applications

CCL Control Command Language

CPU Central Processing Unit

DiFX Distributed FX (Software Correlator)

DRX Digital Receiver

DTX Digital Transmitter

GPS Global Positioning System

IP Internet Protocol

LTA Long Term Accumulator

MIT Massachusetts Institute of Technology

NIC Network Interface Card

NTP Network Time Protocol

OSF Operations Support Facility

OFLS Optical Fiber Link System

PAS Preliminary Acceptance on Site

PIC Phasing Interface Card

PPS Pulse Per Second

PSN Packet Serial Number

TE Timing Event

TFB Tunable Filter Bank

UDP Universal Datagram Protocol

VEX VLBI EXperiment file

VDIF VLBI Data Interchange Format

VLBI Very Long Baseline Interferometry

VOM VLBI Observing Mode

WVR Water Vapor Radiometer

Chapter 2

End to End Software Tests

As described in the introduction, the VOM was exercised from the high-level software many times in order to verify various aspects of that software. A number of those tests serve to validate the hardware installation for accepatance purposes. We shall describe these in some detail in this section.

Note that full analysis of some of this data is still on-going (*i.e.* the parts that relate more importantly to the software verification and potential use to support commissioning). Again, here we merely focus on the aspects of the tests which support hardware acceptance.

2.1 Single Dish VLBI

This test is the simplest—it asks all four quadrants to perform essentially the same processing.

2.1.1 Test Command

This test ([RD2], Section 7.5, case A) was executed as follows:

```
VLBITestObs.py -b 3 -N 4 -d 10 -i 1 -Q 4 -T \
    --app-comp="DV04,DA51" -R DA52 -s 1924-292 \
    --app-clone-bb=3 --app-single-ref="DA52,DA52,DA52,DA52" --app-show-ss
Archived to uid://A002/X909624/X163
```

This is a band-3 setup, with the same frequency programming (what would normally be used by BB_3 for all four quadrants. Four 10-sec scans were requested (with 1-sec integrations), and the only antenna (of the 11 antennas assigned to the array), DA52 was retained in the phased array. (All other antennas become part of the comparison array.) The source is a bright quasar (1924-292).

Because there is only one receiver, but all quadrants are programmed to the same frequency, the signal from the receiver is digitized separately by four processing chains at the antenna. Thus all channels share mostly the same receiver noise, but the filters and gains are different. Then the 3 bits per sample sent from the DTX units are sent to the station units at the front of the correlator (DRX), after which all correlators proceed in parallel to correlate (essentially) the same input signal. At the back end, the PICs send the 8 data streams to the recorders for subsequent VLBI analysis, which is performed *in situ* on the recorders using DiFX.

Because there is no physical distance between the "four" antennas (from a VLBI perspective), this is a zero-baseline test. Even so, the ALMA system is managing the delays to the antenna, and DiFX is dutifully taking into account the (zero) delays between antennas.

2.1.2 General Correlation Details

For correlation purposes, we need to name each data stream as if it were an antenna (DiFX doesn't currently handle as much data as ALMA produces in a more normal way). We use the following assignments throughout:

```
Quadrant 1 (BB_1) Polarization X --> Station Ax
Quadrant 1 (BB_1) Polarization Y --> Station By
Quadrant 2 (BB_2) Polarization X --> Station Cx
Quadrant 2 (BB_2) Polarization Y --> Station Dy
Quadrant 3 (BB_3) Polarization X --> Station Ex
Quadrant 3 (BB_3) Polarization Y --> Station Fy
Quadrant 4 (BB_4) Polarization X --> Station Gx
Quadrant 4 (BB_4) Polarization Y --> Station Hy
```

After correlation, the post-processing tool `fourfit` drops one letter and refers to the stations as `A` through `H`. As a final confusion, the polarization labels change from X/Y to R/L, since VLBI is normally conducted with circular polarizations. For our purposes, this is of no consequence and we just label them all R indifferently.

A final (minor) annoyance is that plotting 32 channels is at present difficult (hard to see what is going on, among other things), so we typically only use half of the channels. Again, for our purposes, this is fine. `fourfit` typically produces one plot per station pair

2.1.3 Fringe Plot Description

As an example, the output of one non-trivial correlation is shown in Figure 2.1. In this presentation, we have cropped out quite a bit of detail which is not particularly relevant (especially since there is "phase cal" or geometry delay between the stations). The full plot set for this experiment is to be found on the mission page, https://ictwiki.alma.cl/twiki/pub/Control/AppMissionChile102014/awQjA-2005-1924-292.pdf; this test and plots are briefly described in the section `Day 2014-10-20, DiFX`.

In this presentation, the "stations" are listed in the upper right, and the columns of numbers (green labels) are the most interesting data—data such as SNR, Amp(litude) and Phase, Delays (single-band, SBD, and multiband, MBD) and Fringe rate. The Amplitude is the normal correlation amplitude, but it is multiplied by 10000 for some historical reason. Experiment names (awQjA) and scan numbers are randomly assigned; the "Exper #" is a sequential processing counter that is used for only for tracking processing.

The large plot shows the multiband delay (blue) and delay rate (red). The smaller green plot is the singleband delay. The processing in `fourfit` is to average over the individual channels to obtain an average cross-correlation delay peak within the channel, and then to look at the channel-channel phase slope to produce the so-called multiband delay. The Fourier transform of the singleband delay is the averaged cross power spectrum (Xpower Spectrum) which shows amplitude (blue) and phase (red).

Finally, the 17 narrow panels show the time series of data in each of the channels—amplitude (blue) and phase (red)—together with an average. In this particular case, you will notice a "glitch" in amplitude within the first sample (4-sec); and corresponding phase ramp thereafter. The gltch appears to be an artifact of the ALMA delay server which affects the first correlator scan, and the subsequent phase ramp results from an incorrect determination of the fringe rate as a result. Ultimately this leads to a loss of correlation amplitude.

By hand-tuning the processing in `fourfit`, we can drop this initial segment of data and see the result shown in Figure 2.2. As with the previous example, the full set of plots is available on the mission page, https://ictwiki.alma.cl/twiki/pub/Control/AppMissionChile102014/awQjA-2005-1924-292b.pdf. You should note that the delay-rate curve is now less noisy and symmetric, and that the correlation amplitude has climbed approximately 10%.

One can also note a slight variation in the amplitudes from one channel (TFB) to the next. Also, the slight ripple in the cross power spectrum is definitely introduced by the tunable filter banks (TFBs). The shape mirrors that of the filter design (R. Lacasse, private communication) shown in Figure 2.3. On the other hand, we note that the signals are only 90% correlated, and appear to have a relative phase of 130 degrees. The exactly cause of this is not known; but once again, we appeal to the separate digitizers and tunable-filter settings, and argue that there are many places for small errors to accumulate, and that ALMA wasn't perhaps designed for this test.

Similar results are obtained for the other 16 channels (b d f h j l n p r t v x z B D F, instead of a c e g i k m o q s u w y A C E), and the other quadrants (like polarization). To show that not all

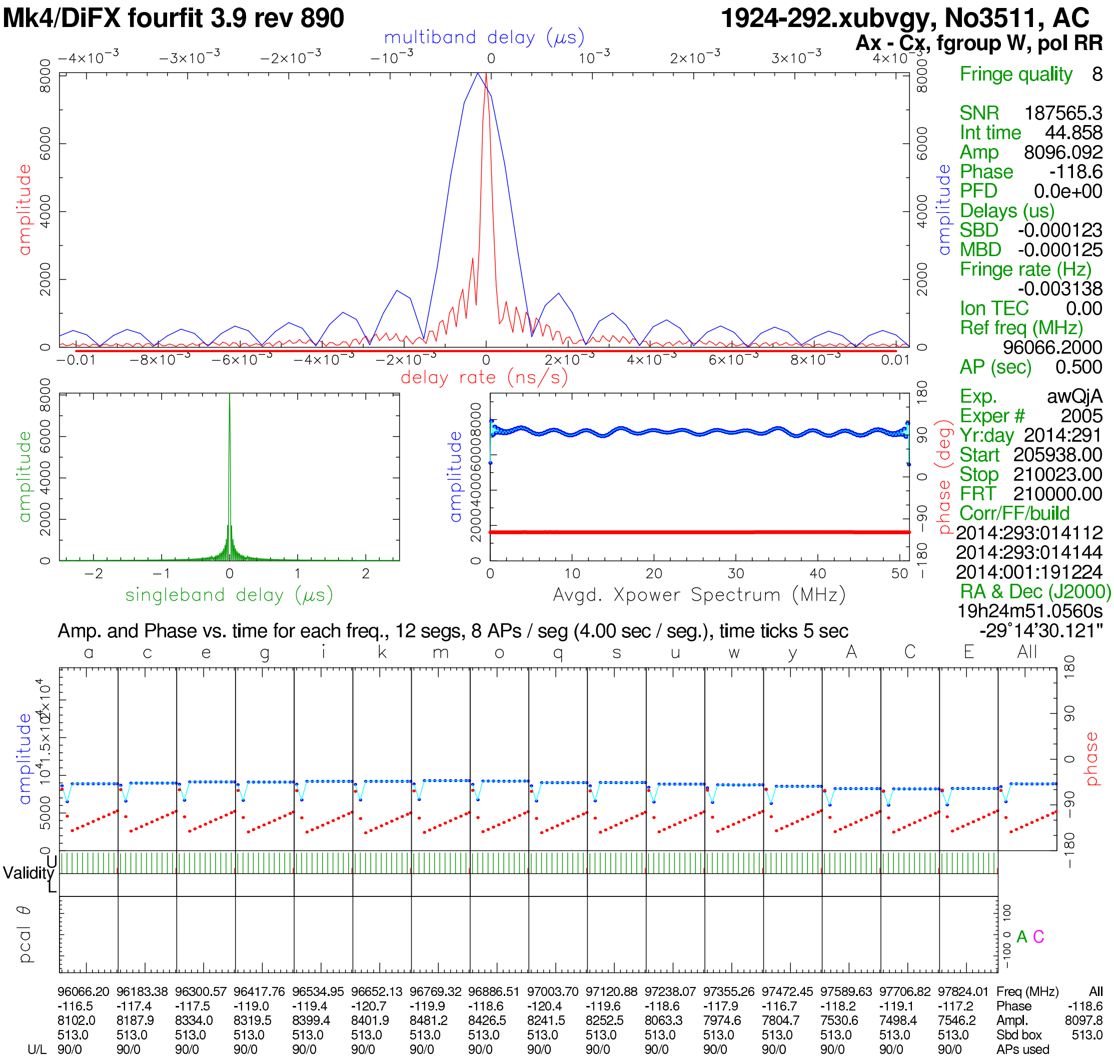

Figure 2.1: *A(Q1X) v C(Q2X), full scan*

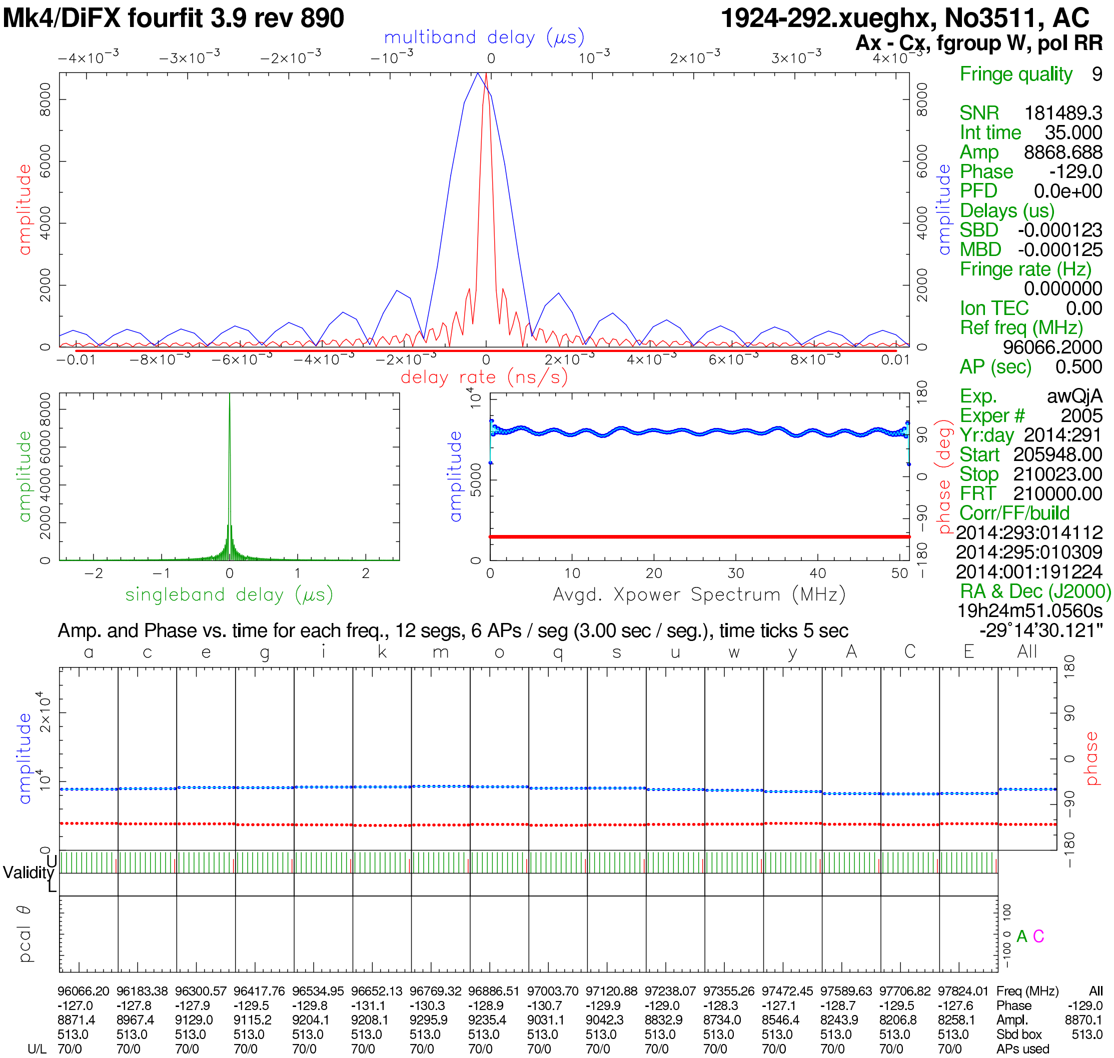

Figure 2.2: *A(Q1X) v C(Q2X), partial scan*

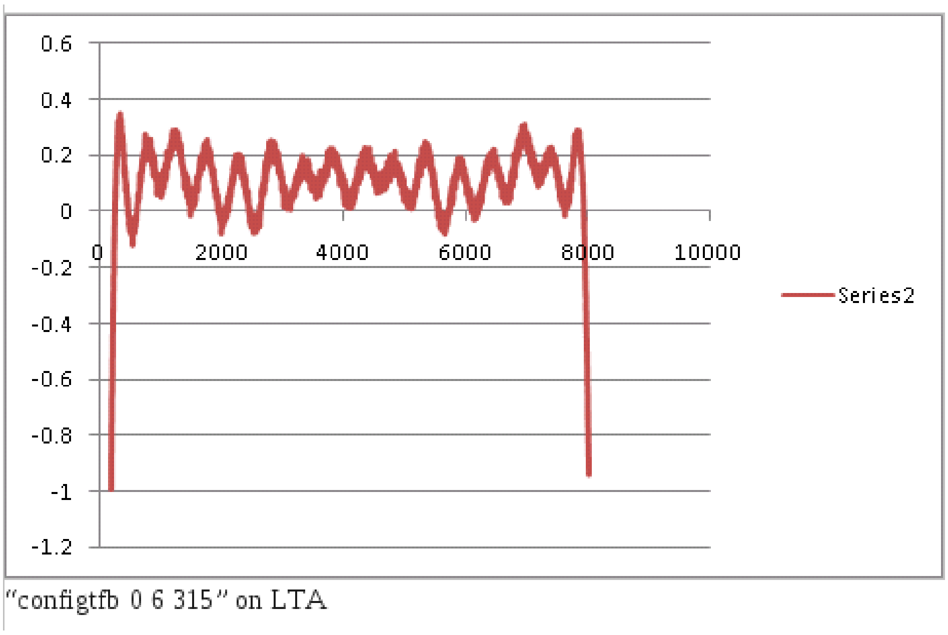

Figure 2.3: *TFB shape measurement; log (not linear) scale*

these plots are identical, see Figure 2.4 which shows one disconnected data cable (corresponding to frequency "i") in the plot. It's cabling peer, "j" is likewise disconnected, but the exact cause of the problem has not been fully resolved. Note also that the correlation amplitude is correspondingly lower for the missing channel (about 83%).

For completeness, we show a cross-polarization case in Figure 2.5. Note here that the amplitude is at the 1% level and that there is now structure in that as one moves across the frequency channels. This is qualitatively consistent with what is observed from a CASA view of the ALMA data; but again, there has not been time for a detailed analysis.

2.1.4 Some Important Results

To be clear, we summarize some of the results evident in Figures 2.1, 2.2, 2.4, and 2.5, as well as the 68 other plots not shown here.

1. All like signals correlate at the 90% level, with various relative phases; this should be closer to 100%.
2. Polarization leakage appears at the 1% level, consistent with expectations.
3. The delays between quadrants are essentially zero ($\sim$ 100 ps).
4. The VLBI system (PICs, OFLS and recorders) collected 64 Gbps from 4 quadrants in correct data (VDIF) such that it could be properly correlated.
5. This test demonstrates that ALMA could be used now for single-dish VLBI.

2.1.5 A Note on Correlation

All of the data described in this report was taken with the normal ALMA TFB setup for a 2-GHz continuum band. This overlaps the 62.5 MHz channels by a factor of 15/16 (58.59375 MHz) to account for the falling response at the edges of the TFB bands. When fully stitched together this amounts to $32 \times 62.5 \times 15/16$ or 1.875 MHz of bandwith (which is similar to the bandwidth limiting Nyquist filters on the digitizers at the antennas).

The astute reader may have noticed that the cross power spectra only contain 51.2 MHz of bandwidth. Because the channels overlap, and because of some numerical details in how we use

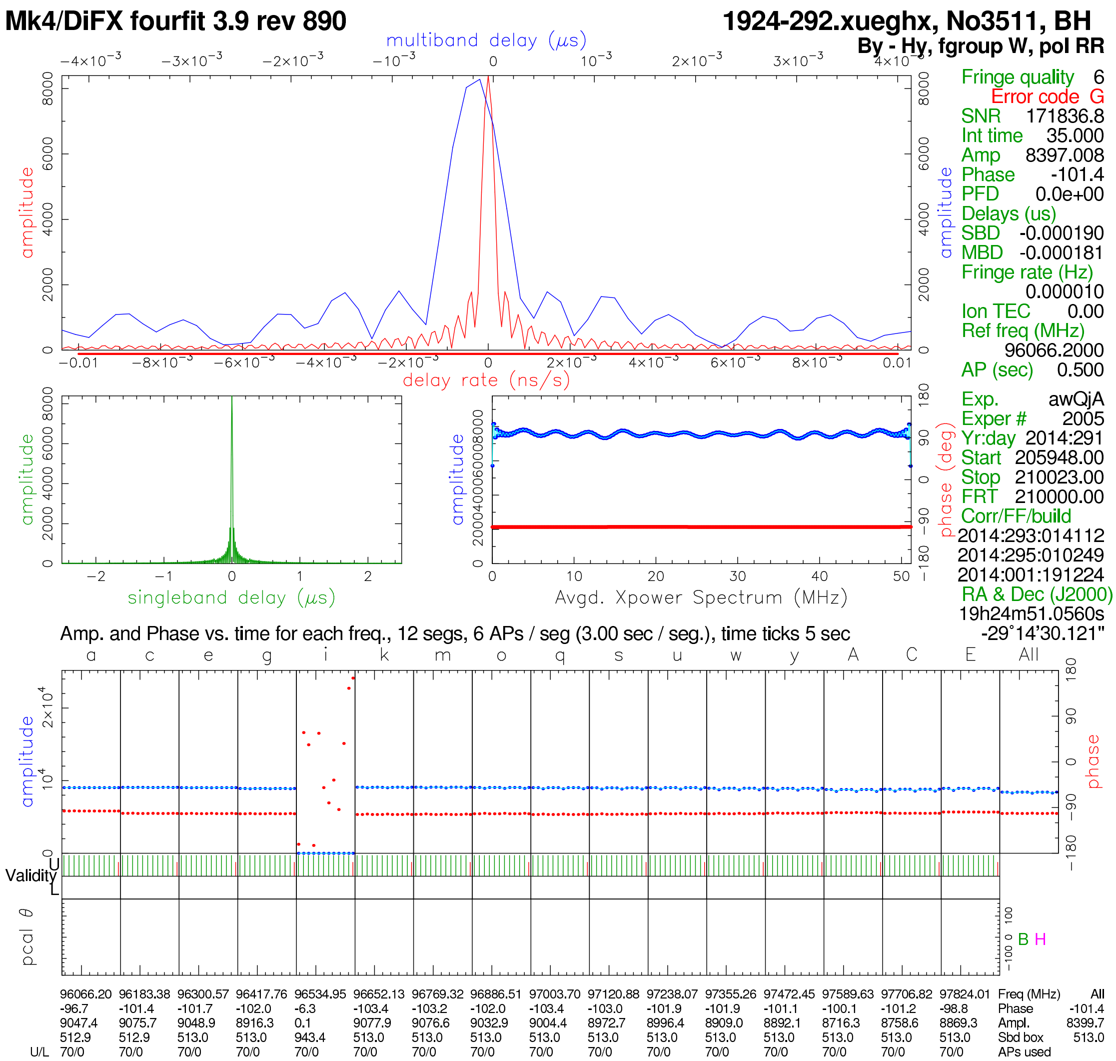

Figure 2.4: *B(Q1Y) v H(Q4Y), partial scan*

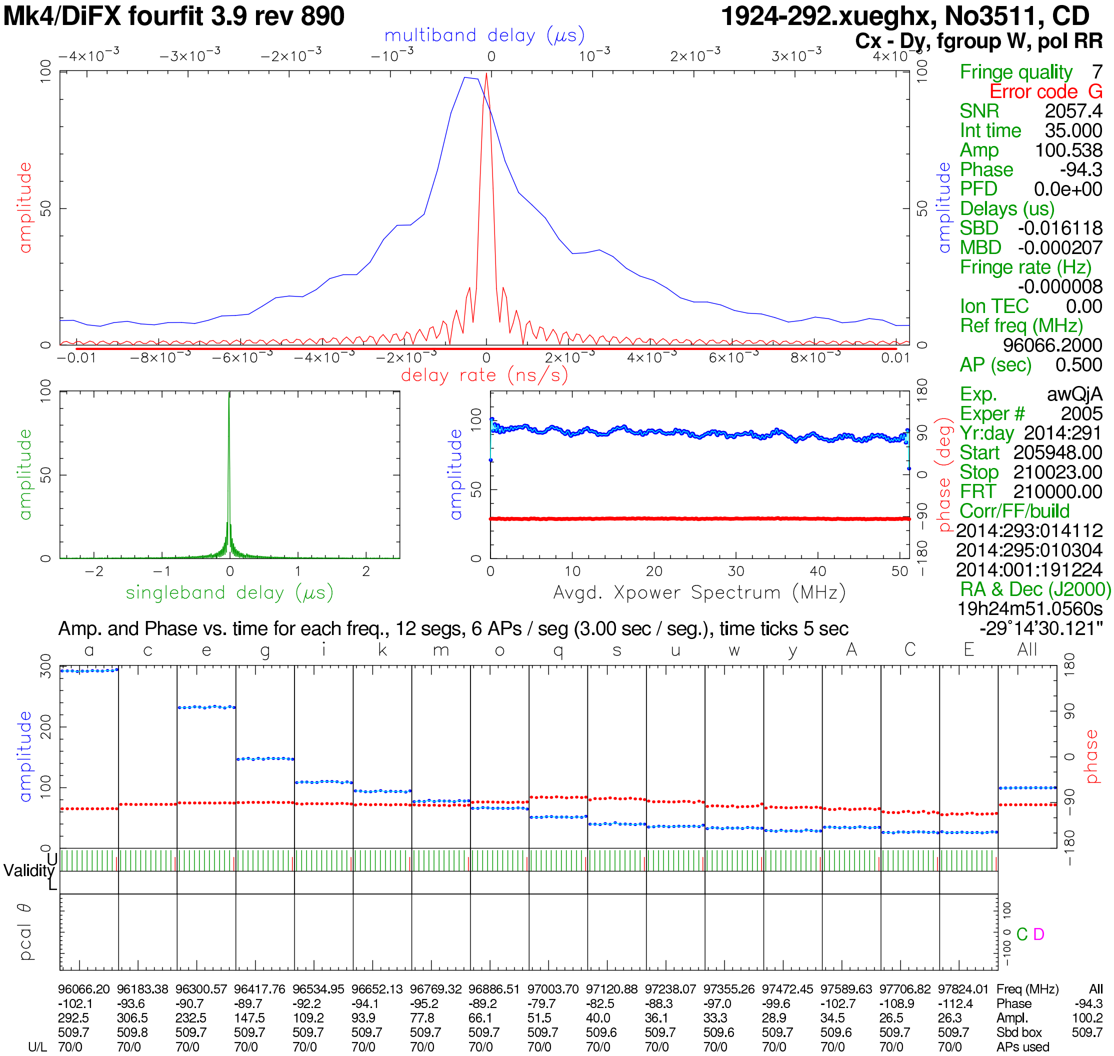

Figure 2.5: *C(Q2X) v D(Q2Y), partial scan*

this data in the DiFX correlation, it is necessary to trim the channels a bit further from their "useful" 58.6 MHz down to 51.2 MHz, as shown in Figure 2.6. This is properly accounted for in the

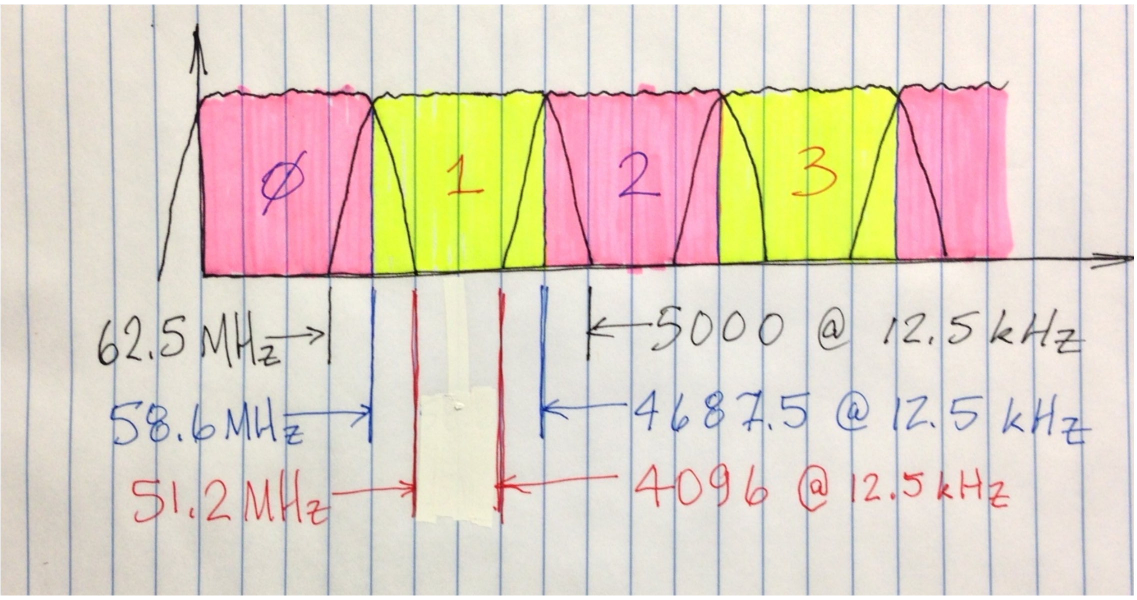

Figure 2.6: *Sketch of TFB channels during correlation*

processing, and does not appreciably affect the results.

(The specific numerical detail is that the underlying sampling is made at a non-integral power-of-two frequency, yet the VLBI processing software is expecting a power-of-two sample-points in the spectrum. By trimming the bands, we can provide 4096 (2^{12}) sample points to `fourfit` to analyze the data.)

These issues are not relevant to a more VLBI-like frequency setup (non-overlapping channels). In any case, $32 \times 62.5 \times 4$ MHz (with 2 bits/sample, Nyquist sampled) of nominal bandwidth (8 GHz) in two polarizations are "recorded" for VLBI correlation ($8 \times 2 \times 2 \times 2$ for the grand total of 64 Gbps).

2.2 Four Phased Quadrants

Normally, all quadrants would have phased arrays. This test, similar to the above runs a phase-array in all four quadrants (but with the same frequency settings).

2.2.1 Test Command

This test ([RD2], Section 7.5, case B) was executed as follows:

```
VLBITestObs.py -b 3 -N 4 -d 10 -i 1 -Q 4 -T \
    --app-comp="DV04,DA51" -R DA52 -s 1924-292 \
    --app-clone-bb=3 --app-single-ref="KEEP,KEEP,KEEP,KEEP" --app-show-ss \
    --app-faker 40,2,8
Archived to uid://A002/X909624/X170
```

Two antennas (DV04, DA51) are set aside for comparision purposes; otherwise this command is similar to the one described in Section 2.1.1. This time, however, there are 4 phasing loops active.

The phased array contained 11 antennas, DA42, DA45, DA46, DA49, DA52, DA61, DA63, DV06, DV15, DV22, and DV25.

2.2.2 Some Test Details

Because the phasing loop was active (and because there currently is some lack of understanding of what the phasing loop is doing at some times), we shall not dwell on the those details, but merely point out some features of this test.

At the end of every scan, the observation script arranges for a scan check (*i.e.* did the data get properly recorded). For this test, we show a sample, noting that this is quite similar to what is produced in all the other cases:

```
2014-10-18T21:04:35.982 Finished doAppScanSequence(136329590520000000) (23.982)
2014-10-18T21:04:35.982 BB_1-Scan:0:0:1234:190:HxDU_Aa_No0175:recorded:2014y291d21h03m18s:46:2:0;
2014-10-18T21:04:35.983 BB_1-PolX:0:0:1234:190:HxDU_Aa_No0175:2:-:OK:vdif:2014y291d21h03m18s:46.000:92.368:16.064:0;
2014-10-18T21:04:35.983 BB_1-PolY:0:0:1234:190:HxDU_Aa_No0175:2:-:OK:vdif:2014y291d21h03m18s:46.000:92.368:16.064:0;
2014-10-18T21:04:35.983 BB_2-Scan:0:0:1234:69:HxDU_Aa_No0175:recorded:2014y291d21h03m18s:46:2:0;
2014-10-18T21:04:35.983 BB_2-PolX:0:0:1234:69:HxDU_Aa_No0175:2:-:OK:vdif:2014y291d21h03m18s:46.000:92.368:16.064:0;
2014-10-18T21:04:35.983 BB_2-PolY:0:0:1234:69:HxDU_Aa_No0175:2:-:OK:vdif:2014y291d21h03m18s:46.000:92.368:16.064:0;
2014-10-18T21:04:35.983 BB_3-Scan:0:0:1234:9:HxDU_Aa_No0175:recorded:2014y291d21h03m18s:46:2:0;
2014-10-18T21:04:35.984 BB_3-PolX:0:0:1234:9:HxDU_Aa_No0175:2:-:OK:vdif:2014y291d21h03m18s:46.000:92.368:16.064:0;
2014-10-18T21:04:35.989 BB_3-PolY:0:0:1234:9:HxDU_Aa_No0175:2:-:OK:vdif:2014y291d21h03m18s:46.000:92.368:16.064:0;
2014-10-18T21:04:35.989 BB_4-Scan:0:0:1234:187:HxDU_Aa_No0175:recorded:2014y291d21h03m18s:46:2:0;
2014-10-18T21:04:35.990 BB_4-PolX:0:140:1234:187:HxDU_Aa_No0175:2:-:OK:vdif:2014y291d21h03m18s:46.000:92.368:16.064:0;
2014-10-18T21:04:35.990 BB_4-PolY:0:140:1234:187:HxDU_Aa_No0175:2:-:OK:vdif:2014y291d21h03m18s:46.000:92.368:16.064:0;
2014-10-18T21:04:35.990 VOM finished 1924-292
```

For each quadrant, the output indicates the time the scan starts (note that all timestamps are identical), the length of the scan (46 seconds), the number of GBs consumed (92GB in this case) and the data rate used for that calculation (16.064 Gbps).

This is to be compared with the intent logged by the script at the outset:

```
2014-10-18T21:02:18.499 Scan No0175 at 2014291210318 (46s) on 1924-292 (136329589980000000)
2014-10-18T21:02:18.500 Generated fake schedule for HxDU
2014-10-18T21:02:24.598 PIC(polz:4) VDIF-8 station 4161 and 2^5 ch
2014-10-18T21:02:24.847 Delivered recorder schedule.
```

I.e., the VLBI scan was scheduled to start at 2014291210318 (YYYYDOYHHMMSS) and run for 46 seconds, and that's exactly what was recorded.

From the 72 plots (including autocorrelations) produced from this data reduction, we show a representative pair, Figures 2.7 and 2.8. The full set of plots is available on the mission page, https://ictwiki.alma.cl/twiki/pub/Control/AppMissionChile102014/aHxDU-2006-1924-292.pdf As discussed, the phasing loop was doing different things in the different quadrants (which we haven't fully analyzed here). The salient point, however, is that the quadrants start out with unphased arrays, and (glitch aside) correlate somewhat in later scans (at different levels).

2.2.3 Some Important Results

We summarize some of the important take-away messages from this test:

1. The phasing loop may operate independently in different quadrants
2. The correlation amplitudes range from $\sim 10\%$ to 50%.
3. The delays between quadrants are negligible
4. The VLBI scans are run precisely as scheduled.

2.3 First 32-Channel Fringe

The following test was actually the first one carried out. It had previously been performed during the July mission, but with only 16 channels. Here we present the 32-channel version.

2.3.1 Test Command

This test ([RD2], Section 7.5, case C) was executed with the following command:

```
VLBITestObs.py -b 3 -N 4 -d 10 -i 1 -Q 4 -T \
    --app-comp="DV04,DA51" -R DA52 -s 1924-292 \
    --app-clone-bb=3 --app-single-ref="KEEP,DA52,DV22,DA49" --app-show-ss
Archived to uid://A002/X909624/X156
```

It differs from the previous test principally in that only one of the quadrants (BB_1) will have a phased array. The other quadrants will have two separate antennas.

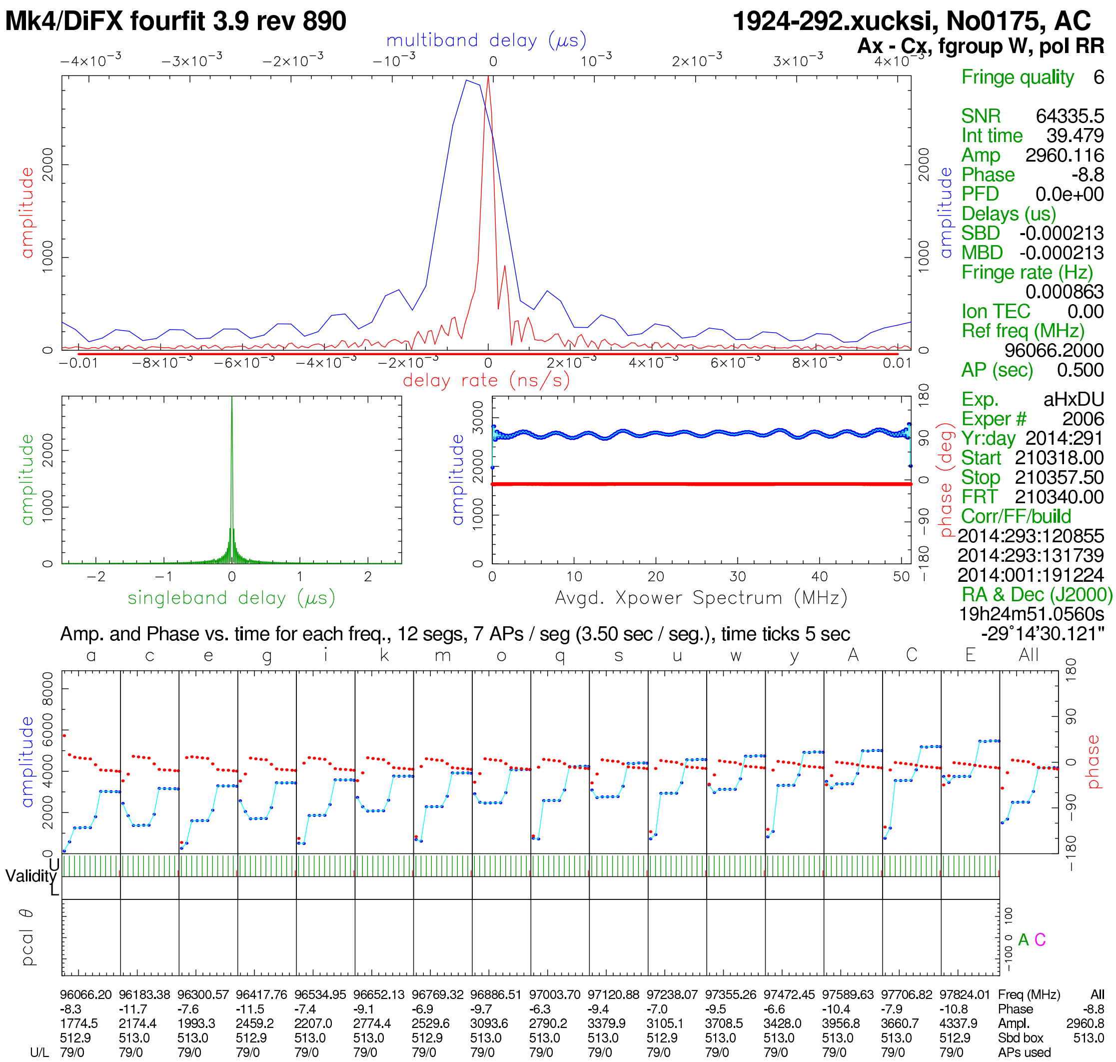

Figure 2.7: *A(Q1X) v C(Q2X), full scan*

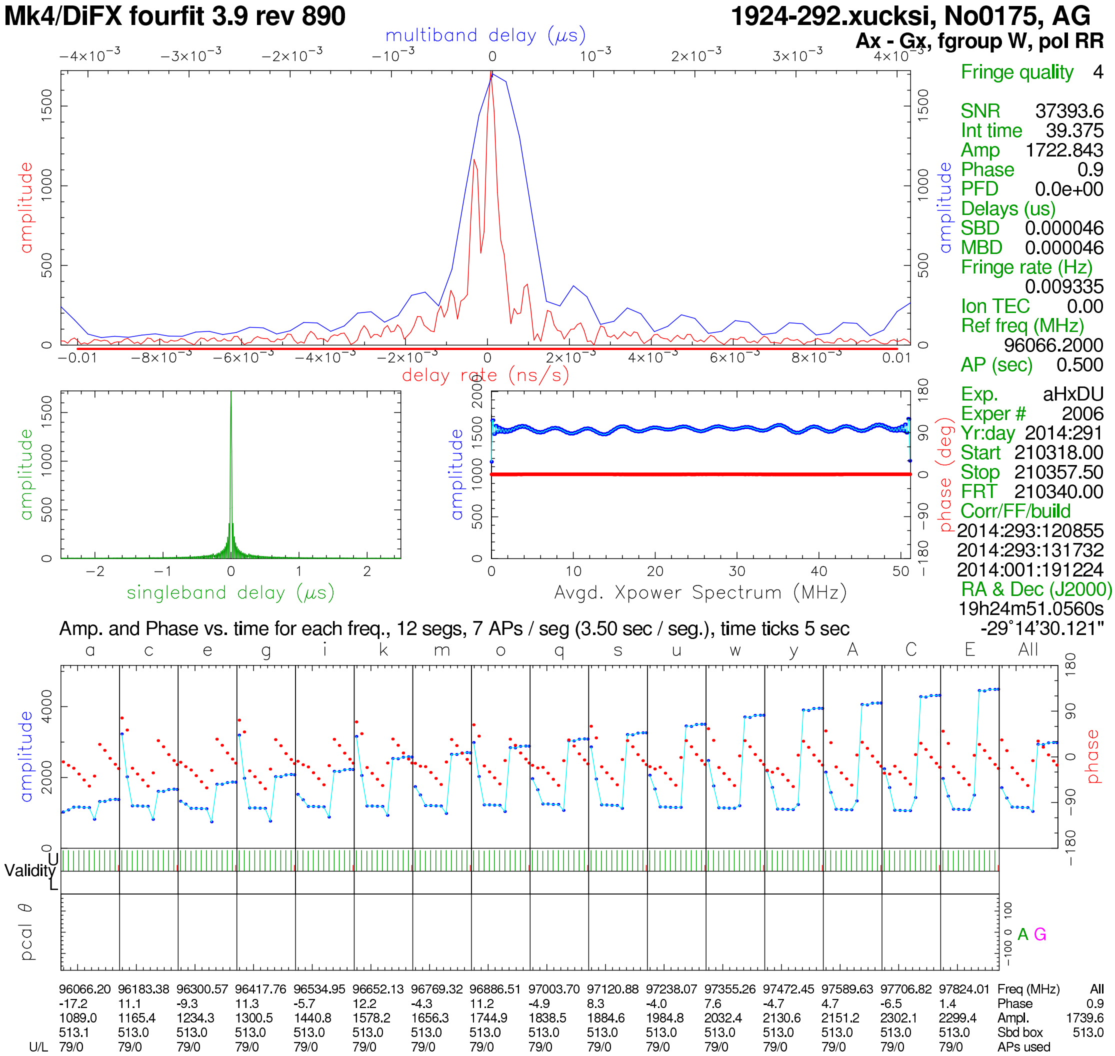

Figure 2.8: *A(Q1X) v G(Q4Y), partial scan*

2.3.2 Some Test Details

The point of this test is to allow quantitative comparison of the data taken for VLBI with what ALMA normally provides to CASA. That level of analysis is beyond what we require here which is to point out some features of the hardware which we verify.

An interesting plot from this correlation is Figure 2.9. It is very similar to Figure 2.1, except now the correlation amplitude is about 23%. This is roughly consistent with the correlation between the sum of 11 antennas and one of its constituents. It is slightly lower for the same reasons that apply to the test in Section 2.1.

As with the preceeding tests, the full set of plots is available from the mission page, `https://ictwiki.alma.cl/twiki/pub/Control/AppMissionChile102014/ao01f-2003-1924-292.pdf`

2.3.3 Some Important Results

We summarize some of the important take-away messages from this test:

- the correlation of the phased sum with the reference antenna is consistent with $1/sqrt(11)$ (since the correlation is dominated by receiver noise).

2.4 Fewer than 32 Channels

2.4.1 Test Command—Sixteen Channels

```
VLBITestObs.py -b 3 -N 3 -d 10 -i 1 -Q 4 -T \
    --app-show-ss -R DA52 -s 1337-129 --app-faker 40,2,8 --app-clone-bb=3 \
    -c 4 --app-single-ref="DA52,DA52,DA52,DA52"
Archived to uid://A002/X91023f/X9b
```

As with the other tests, the full set of plots for 16 channels is available on the mission page, `https://ictwiki.alma.cl/twiki/pub/Control/AppMissionChile102014/aDbm0-2008-1337-129.pdf`.

2.4.2 Test Command—Eight Channels

```
VLBITestObs.py -b 3 -N 3 -d 10 -i 1 -Q 4 -T \
    --app-show-ss -R DA52 -s 1337-129 --app-faker 40,2,8 --app-clone-bb=3 \
    -c 3 --app-single-ref="DA52,DA52,DA52,DA52"
Archived to uid://A002/X91023f/Xa5
```

As with the other tests, the full set of plots for eight channels is available `https://ictwiki.alma.cl/twiki/pub/Control/AppMissionChile102014/aR9sf-2021-1337-129.pdf`.

2.4.3 Some Representative Plots

Two representative correlations are shown in Figures 2.10 and 2.11. These differ from Figure 2.1 mostly in the number of channels and are otherwise not interesting.

2.4.4 Less than Eight Channels

The system was exercised to produce four, two and one channels. The data for four channels was taken, but not yet analyzed. The recorders were not properly programmed for two and one channels (the packet size changes, and the recorders need alternate programming for this.)

2.4.5 Some Important Results

- The system works as expected for 32, 16, 8, 4, 2 and 1 channel.

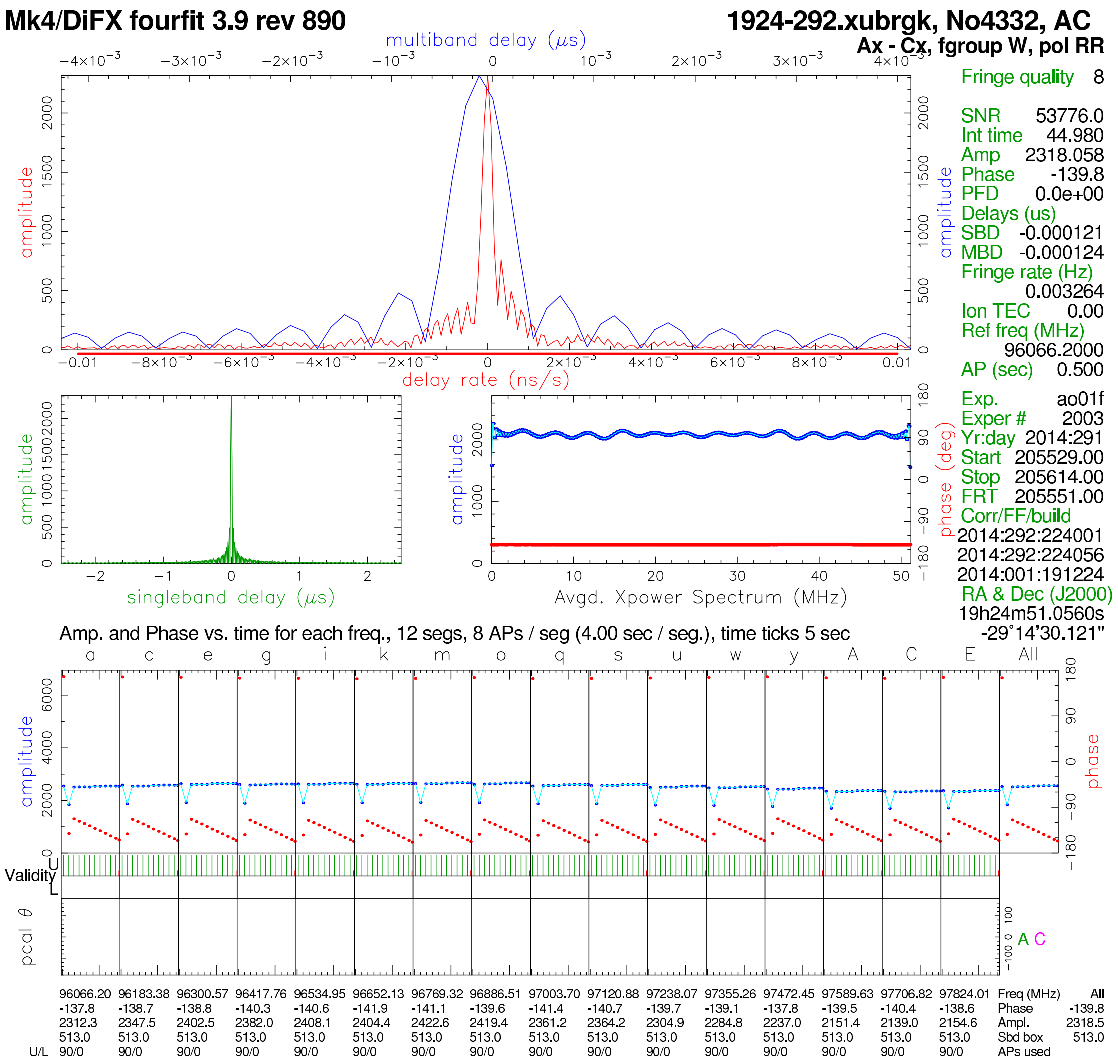

Figure 2.9: *A(Q1X) v C(Q2X), full scan*

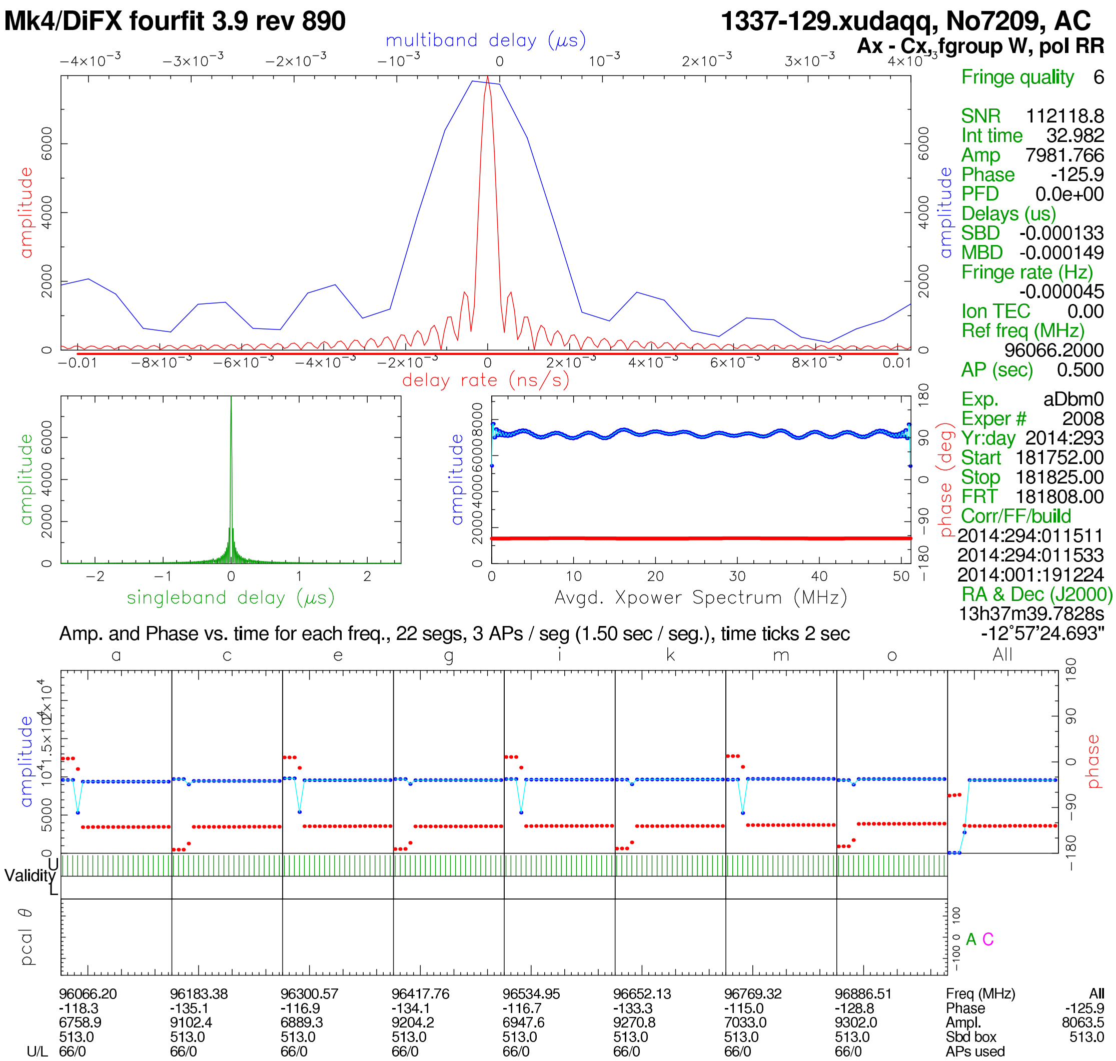

Figure 2.10: A(Q1X) v C(Q2X), full scan, 16 channels (8 shown)

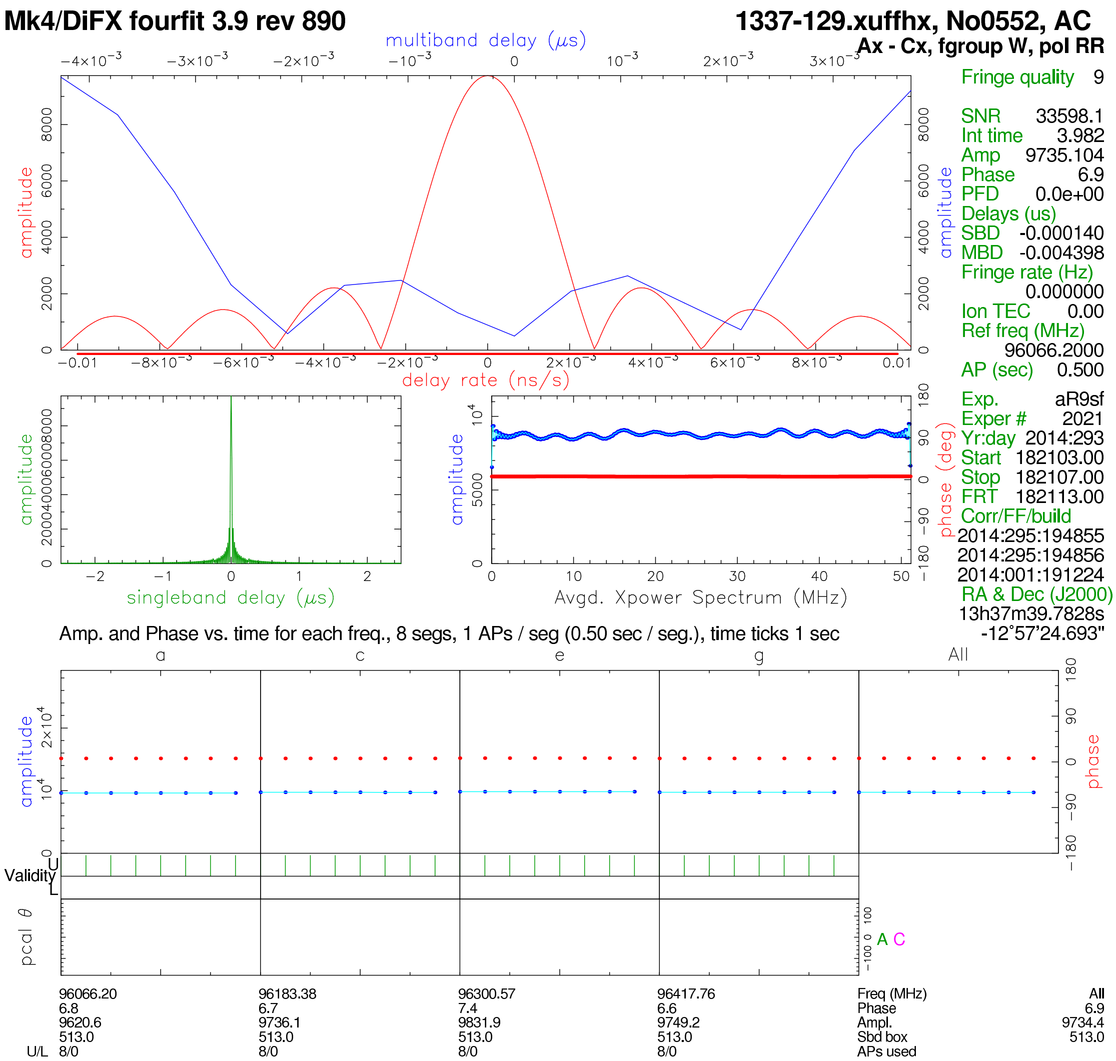

Figure 2.11: *A(Q1X) v C(Q2X), full scan, 8 channels (4 shown)*

Chapter 3

Combined Mark6/OFLS Performance

The ALMA configuration for using the recorders is somewhat different than was used in development. During the mission we identified and fixed a configuration issue with the recorders. Briefly the interrupts associated with network traffic were not being properly distributed among the CPUs, leading to an unexplained, higher-than-expected packet loss rate. The problem has been resolved in release 1.2 of the software.

However, as it turned out, the fix was patched prior to the last day of testing, so it was possible to record all the data from October 25, 2014 and produce an estimate of the data loss rate through the entire system. This test consisted of `grep`ing for `filled` in the `dplane` logs as described in [RD2], section 7.3.3.

For a count of packets, a good estimate is provided by:

```
ls -lhtr /mnt/disks/[12]/?/data | grep G.Oct.25 |\
awk '{s+=$5}END{printf "%d\n", s*1024*1024*1024 / 8032}'
ls -lhtr /mnt/disks/[34]/?/data | grep G.Oct.25 |\
awk '{s+=$5}END{printf "%d\n", s*1024*1024*1024 / 8032}'
```

i.e. , list the recorded file fragments by stream grouping in human mode (`G` is GiB or 1024^3B) and convert to a packet count. A simple division produces an error rate:

```
Recorder   Input     Fill   Total        Error Rate
---------- --------- ----   ----------   -----------
Mark6-4005 stream-0:  909 / 1153122787 = 0.000000788
Mark6-4005 stream-1: 1220 / 1153229733 = 0.000001058
Mark6-4006 stream-0:   62 / 1152989104 = 0.000000054
Mark6-4006 stream-1: 1712 / 1153109418 = 0.000001485
Mark6-4007 stream-0:  574 / 1153042577 = 0.000000498
Mark6-4007 stream-1: 1885 / 1153296574 = 0.000001634
Mark6-4008 stream-0: 1420 / 1153096050 = 0.000001231
Mark6-4008 stream-1: 1587 / 1153082682 = 0.000001376
```

which is $O(10^{-6})$ (*i.e.* one part per million) for all PIC/OFLS/Mark6 recorder combinations.

Note that the requirement is have an error rate no higher than 5%.

Chapter 4

Timing Verification

This section addresses a number of timing considerations.

4.1 Absolute Timing

Detailed tests of timing are ultimately a commissioning activity. However, in preparation for that, logs were examined in detail to establish that the timing within the APP modifications to ALMA were as expected. These are detailed in [RD3] to which the reader is referred.

However, the principal conclusion was that, indeed, the ALMA master clock appears to be properly synchronized with the GPS time system, and that the timestamps provided in the VDIF headers the PICs deliver to the recorders are within a few hundred ns of correctness.

The absolute timing of the data (*i.e.* how much earlier the receiver signals were digitized prior to transmission to the correlator), is not established at this time. (This is a commissioning activity.)

4.2 Recorder Timing

For operational sanity, it is important that the recorders be time-synchronized with the other machines (it is a requirement for this reason). In practice this is arranged using NTP. The configuration on each of the recorders is to be configured to take the time service from the server which also supports the entire AOS network (10.197.50.101). In addition, the normal Debian default is to access recorders randomly drawn from a pool at reboot; this has not been disabled, as it provides an independent check on timing. Finally, two of the recorders (2 and 3) receive level-3 information from the other two (1 and 4) via the private network. This allows a final check, and an ability for the 4 recorders to coast through external network issues should such arise.

A report to verify this was generated Nov 20, 2014 as follows:

```
oper@Mark6-4005:~/test$ ntpq -pn
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
*10.197.50.101   LOCAL(0)         2 u  123 1024  377    0.508   -2.045   0.933
+200.89.75.197   200.54.149.24    2 u 1049 1024  377   46.780   -4.000   6.943
-200.1.19.4      200.20.186.76    2 u  928 1024  377   60.174   -6.991  12.855
+200.1.22.6      69.36.224.15     2 u  957 1024  375   58.614   -4.073   8.796
+200.1.19.16     200.54.149.24    2 u 1014 1024  377   51.806   -4.656   4.854

oper@Mark6-4006:~/test$ ntpq -pn
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
*10.197.50.101   LOCAL(0)         2 u  634 1024  377    0.497   -1.192   0.661
 192.168.3.5     10.197.50.101    3 u  548 1024  377    3.038    2.529   2.974
 192.168.3.8     10.197.50.101    3 u  718 1024  377    3.077    2.102   1.581
+200.1.22.6      69.36.224.15     2 u  617 1024  375   39.687    6.761   2.345
```

```
+200.1.19.4      200.20.186.76    2 u  763 1024  377   56.430    2.675   6.028

oper@Mark6-4007:~/test$ ntpq -pn
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
*10.197.50.101   LOCAL(0)         2 u  853 1024  377    0.524   -1.905   0.196
 192.168.3.5     10.197.50.101    3 u   99 1024  377    3.705    2.396   2.202
 192.168.3.8     10.197.50.101    3 u  135 1024  377    3.708    1.906   2.499
+200.89.75.197   200.54.149.24    2 u   69 1024  377   31.551    4.601   4.215
 200.1.19.16     .STEP.          16 u  15d 1024    0    0.000    0.000   0.000

oper@Mark6-4008:~/test$ ntpq -pn
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
*10.197.50.101   LOCAL(0)         2 u  427 1024  377    0.578   -2.560   0.318
 10.197.52.51    10.197.50.101    3 u  118 1024  377    2.842    1.880   1.465
+200.89.75.197   200.54.149.24    2 u  149 1024  377   34.691    7.130   9.449
+200.1.19.4      200.20.186.76    2 u   32 1024  377   37.884    4.966   8.306
+200.1.19.16     200.54.149.24    2 u   58 1024  377   35.890    5.852   8.533
```

In these reports, numerical values have been used for network addresses; 192.168.3 is the private recorder network. 10.197.50.101 is the AOS NTP server, and the other 200 addresses are the random choices. In the refid column, `.STEP.` indicates the this particular server (200.1.19.16) was not reachable. This is normal, and the reason why so many servers are used. The delays/offsets/jitters are all in milliseconds. The leading * indicates the server that is the best choice (*i.e.* the local one) with a jitter of at most 1 ms among the recorders, and deduced offsets of less than 3 ms (presumably compensated).

This is nominal, and excellent for operational purposes. It also allows us to verify the timestamps "on the wire" at the few-millisecond level of confidence. This is a very long time in VLBI terms; but it is an important sanity check, as errors in the ALMA system are most likely to be at multiples of the 48-millisecond TE or the every-6-second alignment of the PPS with TE.

Chapter 5

Additional Acceptance Considerations

5.1 Comments on VLBITestObs

While `VLBITestObs.py` is a test script used for testing the capabilities of the software and hardware, it is very similar on some particulars to what will eventually be used for real observations. The main respect in which it is artificial is that it manufacters a VLBI schedule to present to the recorders rather than to extract one from the VEX file that would normally be used.

In this section we make, for convenience a few observations relevant to the requirements and specifications of the APP.

5.1.1 Number of antennas in the phased array

Both this test observing script as well as the underlying software controller (`PhasingController` described in [RD1]) enforce an odd number of antennas in the phased array.

5.1.2 Timing of scans

The `-app-faker` argument to the script (seen above in Section 2.2.1) hints to the automatic "fake" VLBI schedule generator (which the recorders run with) that the setup for a source is to take 40 seconds, that 2 seconds are expected between correlator subscans, and that 8 seconds are to be allowed after the scan before starting the next scan.

Most of the 40 seconds are required by the Control system to repoint the antennas (from an arbitrary previous pointing). This total of 48 seconds is larger than the minimum specified (10s) of the requirements.

5.1.3 Spectral configuration

The receiver selection (band 3 via `-b 3` in the samples above) results in a "default" selection of 2 USB and 2 LSB 2-GHz frequency bands in the test script (which merely relays the request to the normal Control system function calls).

It always commands for mode13 (dual polarization, all 4 stokes products).

All legal ALMA receivers may be specified here (whether commissioned or not). In practice we have tested with bands 3 and 6.

The VOM (which `VLBITestObs.py` drives) generates conventional spectral windows for programming the front end electronics (receivers, *&c.*) for observations. So, its use if the radio IFs is thoroughly conventional.

On the other hand, the recorded data must match that taken at other stations, which typically channelize within the IF to channels of width 2^n MHz which are placed on integral MHz boundaries. Sometimes this is done with frequency offsets to avoid local interference.

Accomodation for this at ALMA is possible using the TFBs which may be tuned at resolutions of 2.000 GHz / 65536 = 30.517578125 kHz. The `VLBITestObs.py` script has options to manufacture a set of equally sized spectral windows at equal spacings in the IF; the Correlator software then automatically adapts these to legal values.

In a VLBI experiment, the frequency specification is made in the VEX file; we have demonstrated the capability to make whatever tunings ALMA is capable of, but it is a commissioning activity to demonstrate this in a VLBI experiment.

5.1.4 Number of quadrants in use

By default the test script programs all four quadrants, but the `-Q` option is available to specify fewer (1, 2 or 4 are the only legal values). Using precisely 3 quadrants is disallowed by the ALMA control system.

5.1.5 Phasing and VLBI Independence

For test purposes, if no other, the test script has explicit options to disable either the phasing system (`-I`) or the VLBI backend (PICs and recorders) (`-V`). Both have been used in testing. For example:

```
VLBITestObs.py -b 6 -N 1 -d 60 -i 1 -Q 4 -T \
    --app-show-ss -R DA61 --app-comp="DA46" -s 1337-129 \
    --app-clone-bb=4 --app-wvr=Abs,Rel,False,False --app-fast=1.5 \
    --app-clone-bb=4 -I
Archived to uid://A002/X91837f/X52
```

is an example of a test of the so-called "Fast" loop (WVR) phasing correction (`Abs(olute)` in quadrant 1, (`Rel(ative)` in quadrant 2, and disabled in quadrants 3 and 4 (`False`).

And another example, with no recorders:

```
VLBITestObs.py -b 3 -N 10 -d 15 -i 1 -Q 85e9,85e9,85e9,85e9 \
    --app-show-ss -R DA61 --app-comp="DA46" -s 3c454.3 \
    --app-disable="BB_1,BB_4" --app-faker 40,2,8 -V
Archived to uid://A002/X91837f/X9e0
```

5.1.6 Logging

The test script uses the normal ACS logging system, as do the underlying components. The observing script can do exactly the same. As discussed in [RD2], section 6.5.4, `ripVLBIlog.py` can be used to extract logs from the main ALMA server.

For example, this test command:

```
VLBITestObs.py -b 3 -N 10 -d 12 -i 1 -Q 4 -T \
    --app-show-ss -R DA61 --app-comp="DA46" \
    --app-clone-bb=4 \
    -s 1337-129,3c279,1337-129,3c279,1337-129,3c279,1337-129,3c279
```

produced the following startup sequence (where we have deleted some of the pure-debugging items and inserted newlines for readability):

```
2014-10-25T17:43:01.468 Constructing source list
2014-10-25T17:43:01.535 Source list constructed
2014-10-25T17:43:02.130 __init__ - Getting component CONTROL/Array004
2014-10-25T17:43:03.070 Adjusting 0.500/1.000 for 4 basebands
2014-10-25T17:43:03.070 Setting dump/integ/subscan time to 0.528/1.056/12.672 s
2014-10-25T17:43:03.070 Rate check1: 2.85 < 62.50 MHz
2014-10-25T17:43:03.070 Rate check2: 5.70 < 62.50 MHz
2014-10-25T17:43:10.792 .py
    --appFast 3.0 --appWVR False,False,False,False --arrayName Array004
    --inttest True --subscanDuration 12.672 --appVomQual 0.66, 0.50, 0.33
    --appSingleRef KEEP,KEEP,KEEP,KEEP --noIntent False --appNumSWs 1
    --appSpecWidth 1920 --noVLBI False --polarization 4
    --source 1337-129,3c279,1337-129,3c279,1337-129,3c279,1337-129,3c279
    --numbasebands 4 --wait 20 --subscanRepeats 10 --specAveFac 1
    --appFaker 75,2,8 --band 3 --appVomSPA 3 --scode Aa --appVomEff 0.0,1.0
    --appRDC True,True,True,True --appSWBW 0.0 --appCloneBB 4
    --appHelp False --appElev False --atmCalPars 2.0,1.5
    --referenceAntenna DA61 --packMode ONE_PER_ANT --dumpt 0.528
```

```
    --integrationDuration 1.056 --appComp DA46 --appShowSS True --nchlog2 5
    ASDM = uid://A002/X91837f/Xfd
2014-10-25T17:43:10.792 Beginning Execution, uid is uid://A002/X91837f/Xfd
2014-10-25T17:43:10.792 makeSpecSpecIT
2014-10-25T17:43:10.793 obsfq: 97937500000.0
2014-10-25T17:43:10.793 freqs:
    [85000000000.0, 86875000000.0, 97000000000.0, 98875000000.0]
2014-10-25T17:43:10.793 bands: ['BB_1', 'BB_2', 'BB_3', 'BB_4']
2014-10-25T17:43:10.794 dump 528.0ms chav 1056.0ms int 1056.0ms 97.9375GHz
2014-10-25T17:43:10.794 Cloned:
    [98875000000.0, 98875000000.0, 98875000000.0, 98875000000.0]
2014-10-25T17:43:10.828 sw-00: 3000.000 MHz 0 1920
2014-10-25T17:43:10.829 sw-00: 3000.000 MHz 0 1920
2014-10-25T17:43:10.835 sw-00: 3000.000 MHz 0 1920
2014-10-25T17:43:10.836 sw-00: 3000.000 MHz 0 1920
2014-10-25T17:43:55.540 Phased array:
    ['DA42', 'DA45', 'DA51', 'DA54', 'DA56', 'DA61', 'DA63',
     'DV04', 'DV08', 'DV09', 'DV11', 'DV12', 'DV15']
2014-10-25T17:43:55.540 Effic. array: ['DA46']
2014-10-25T17:43:55.540 array:
    Control.AppArrayParams(baseband=BB_1, phasedArray=['DA42', 'DA45',
    'DA51', 'DA54', 'DA56', 'DA61', 'DA63', 'DV04', 'DV08', 'DV09',
    'DV11', 'DV12', 'DV15'], efficiencyArray=['DA46'], refAntenna='DA61')
2014-10-25T17:43:55.541 array:
    Control.AppArrayParams(baseband=BB_2, phasedArray=['DA42', 'DA45',
    'DA51', 'DA54', 'DA56', 'DA61', 'DA63', 'DV04', 'DV08', 'DV09',
    'DV11', 'DV12', 'DV15'], efficiencyArray=['DA46'], refAntenna='DA61')
2014-10-25T17:43:55.541 array:
    Control.AppArrayParams(baseband=BB_3, phasedArray=['DA42', 'DA45',
    'DA51', 'DA54', 'DA56', 'DA61', 'DA63', 'DV04', 'DV08', 'DV09',
    'DV11', 'DV12', 'DV15'], efficiencyArray=['DA46'], refAntenna='DA61')
2014-10-25T17:43:55.541 array:
    Control.AppArrayParams(baseband=BB_4, phasedArray=['DA42', 'DA45',
    'DA51', 'DA54', 'DA56', 'DA61', 'DA63', 'DV04', 'DV08', 'DV09',
    'DV11', 'DV12', 'DV15'], efficiencyArray=['DA46'], refAntenna='DA61')
2014-10-25T17:43:56.002 Fast[BB_1]: disabled; True
2014-10-25T17:43:56.002 Fast[BB_2]: disabled; True
2014-10-25T17:43:56.002 Fast[BB_3]: disabled; True
2014-10-25T17:43:56.002 Fast[BB_4]: disabled; True
2014-10-25T17:43:56.010 packMode: ONE_PER_ANT
2014-10-25T17:43:56.019
    Scan No3351 at 2014298174531 (138s) on 1337-129 (136335519310000000)
    Scan No3352 at 2014298174912 (138s) on 3c279 (136335521520000000)
    Scan No3353 at 2014298175253 (138s) on 1337-129 (136335523730000000)
    Scan No3354 at 2014298175634 (138s) on 3c279 (136335525940000000)
    Scan No3355 at 2014298180015 (138s) on 1337-129 (136335528150000000)
    Scan No3356 at 2014298180356 (138s) on 3c279 (136335530360000000)
    Scan No3357 at 2014298180737 (138s) on 1337-129 (136335532570000000)
    Scan No3358 at 2014298181118 (138s) on 3c279 (136335534780000000)
2014-10-25T17:43:56.028 Generated fake schedule for 4n-0
2014-10-25T17:44:00.604 PIC(polz:4) VDIF-8 station 4161 and 2^5 ch
2014-10-25T17:44:00.786 Delivered recorder schedule.
2014-10-25T17:44:00.789 BB_1 schedule is correct.
2014-10-25T17:44:00.789 BB_2 schedule is correct.
2014-10-25T17:44:00.789 BB_3 schedule is correct.
2014-10-25T17:44:00.790 BB_4 schedule is correct.
2014-10-25T17:44:00.790 Done with APP_VLBIConfig
2014-10-25T17:44:00.790 VOM Prep was 59.608 secs
```

That is, it too, approximately one minute to parse all the command options, locate the sources, set up the spectral configuration and finally manufacture a VLBI schedule and deliver it to the recorders. Thereafter, the system points to the source (1337-129 is the first on the list) and prepares to run the first VLBI scan:

```
2014-10-25T17:44:00.790 VOM starting 1337-129 at 2014298174531
2014-10-25T17:44:09.009 LO ref freq is 9.28112e+10, obs freq 9.79375e+10
2014-10-25T17:44:09.009 No ATM Cal since Fast Loop is disabled
2014-10-25T17:44:09.161
    Determining what correlator calibrations need to be done for subscan #1
2014-10-25T17:44:09.178    New  -1755045286   347080056        NONE  True ''
2014-10-25T17:44:09.178 Cache contains 0 calibrations
2014-10-25T17:44:09.178 No match in the cache - doing a new calibration.
2014-10-25T17:44:09.179 Doing 1 calibrations
2014-10-25T17:44:16.033 New calibrations id's are  20
2014-10-25T17:44:16.050 Prep for Scan Sequence complete (15.254 s)
2014-10-25T17:44:16.050 Entering doAppScanSequence(136335519310000000) (74.950)
2014-10-25T17:44:16.214 First scan is expected to start at 17:45:31.008
2014-10-25T17:44:16.215 Sleeping for 44 seconds
2014-10-25T17:45:00.545 All 10 scans are expected to complete at 17:48:23.559
2014-10-25T17:45:00.546
    Waiting for up to 208.013 seconds for the scan sequence to complete
```

These scans are all a bit more than two minutes of recording, so the total wait at the recorders for the end of the recording is 3 minutes. At this point, the scans are checked and a report issued:

```
2014-10-25T17:47:59.003 Scan sequence completed at 17:47:59.003
2014-10-25T17:47:59.003 Executing post-scan check
2014-10-25T17:48:12.655 Post-scan check completed
2014-10-25T17:48:12.656 Finished doAppScanSequence(136335520770000000) (15.656)
2014-10-25T17:48:12.657
    BB_1-Scan:0:0:1234:277:4n-0_Aa_No3351: recorded:2014y298d17h45m31s:138:2:0;
    BB_1-PolX:0:0:1234:277:4n-0_Aa_No3351:2:-:OK:vdif:2014y298d17h45m31s:138.000:277.104:16.064:0;
    BB_1-PolY:0:0:1234:277:4n-0_Aa_No3351:2:-:OK:vdif:2014y298d17h45m31s:138.000:277.104:16.064:0;
    BB_2-Scan:0:0:1234:156:4n-0_Aa_No3351:recorded:2014y298d17h45m31s:138:2:0;
    BB_2-PolX:0:0:1234:156:4n-0_Aa_No3351:2:-:OK:vdif:2014y298d17h45m31s:138.000:277.104:16.064:0;
    BB_2-PolY:0:0:1234:156:4n-0_Aa_No3351:2:-:OK:vdif:2014y298d17h45m31s:138.000:277.104:16.064:0;
    BB_3-Scan:0:0:1234:97:4n-0_Aa_No3351:recorded:2014y298d17h45m31s:138:2:0;
    BB_3-PolX:0:0:1234:97:4n-0_Aa_No3351:2:-:OK:vdif:2014y298d17h45m31s:138.000:277.104:16.064:0;
    BB_3-PolY:0:0:1234:97:4n-0_Aa_No3351:2:-:OK:vdif:2014y298d17h45m31s:138.000:277.104:16.064:0;
    BB_4-Scan:0:0:1234:275:4n-0_Aa_No3351:recorded:2014y298d17h45m31s:138:2:0;
    BB_4-PolX:0:0:1234:275:4n-0_Aa_No3351:2:-:OK:vdif:2014y298d17h45m31s:138.000:277.104:16.064:0;
    BB_4-PolY:0:0:1234:275:4n-0_Aa_No3351:2:-:OK:vdif:2014y298d17h45m31s:138.000:277.104:16.064:0;
2014-10-25T17:48:12.667 VOM finished 1337-129
```

After this it proceeds to the next scan:

```
2014-10-25T17:48:12.668 VOM starting 3c279 at 2014298174912
2014-10-25T17:51:52.957
    BB_1-Scan:0:0:1234:278:4n-0_Aa_No3352:recorded:2014y298d17h49m12s:138:2:0;
    BB_1-PolX:0:0:1234:278:4n-0_Aa_No3352:2:-:OK:vdif:2014y298d17h49m12s:138.000:277.104:16.064:0;
    BB_1-PolY:0:0:1234:278:4n-0_Aa_No3352:2:-:OK:vdif:2014y298d17h49m12s:138.000:277.104:16.064:0;
    BB_2-Scan:0:0:1234:157:4n-0_Aa_No3352:recorded:2014y298d17h49m12s:138:2:0;
    BB_2-PolX:0:0:1234:157:4n-0_Aa_No3352:2:-:OK:vdif:2014y298d17h49m12s:138.000:277.104:16.064:0;
    BB_2-PolY:0:0:1234:157:4n-0_Aa_No3352:2:-:OK:vdif:2014y298d17h49m12s:138.000:277.104:16.064:0;
    BB_3-Scan:0:0:1234:98:4n-0_Aa_No3352:recorded:2014y298d17h49m12s:138:2:0;
    BB_3-PolX:0:0:1234:98:4n-0_Aa_No3352:2:-:OK:vdif:2014y298d17h49m12s:138.000:277.104:16.064:0;
    BB_3-PolY:0:0:1234:98:4n-0_Aa_No3352:2:-:OK:vdif:2014y298d17h49m12s:138.000:277.104:16.064:0;
    BB_4-Scan:0:0:1234:276:4n-0_Aa_No3352:recorded:2014y298d17h49m12s:138:2:0;
    BB_4-PolX:0:0:1234:276:4n-0_Aa_No3352:2:-:OK:vdif:2014y298d17h49m12s:138.000:277.104:16.064:0;
    BB_4-PolY:0:0:1234:276:4n-0_Aa_No3352:2:-:OK:vdif:2014y298d17h49m12s:138.000:277.104:16.064:0;
2014-10-25T17:51:52.965 VOM finished 3c279
```

`et cetera:`

```
2014-10-25T17:51:52.965 VOM starting 1337-129 at 2014298175253
2014-10-25T17:55:34.651
    BB_1-Scan:0:0:1234:279:4n-0_Aa_No3353:recorded:2014y298d17h52m53s:138:2:0;
    BB_1-PolX:0:0:1234:279:4n-0_Aa_No3353:2:-:OK:vdif:2014y298d17h52m53s:138.000:277.104:16.064:0;
    BB_1-PolY:0:0:1234:279:4n-0_Aa_No3353:2:-:OK:vdif:2014y298d17h52m53s:138.000:277.104:16.064:0;
    ...
2014-10-25T17:55:34.658 VOM finished 1337-129

2014-10-25T17:55:34.659 VOM starting 3c279 at 2014298175634
2014-10-25T17:59:15.698
    BB_1-Scan:0:0:1234:280:4n-0_Aa_No3354:recorded:2014y298d17h56m34s:138:2:0;
    BB_1-PolX:0:0:1234:280:4n-0_Aa_No3354:2:-:OK:vdif:2014y298d17h56m34s:138.000:277.104:16.064:0;
    BB_1-PolY:0:0:1234:280:4n-0_Aa_No3354:2:-:OK:vdif:2014y298d17h56m34s:138.000:277.104:16.064:0;
    ...
2014-10-25T17:59:15.707 VOM finished 3c279

2014-10-25T17:59:15.708 VOM starting 1337-129 at 2014298180015
2014-10-25T18:02:56.704
    BB_1-Scan:0:0:1234:281:4n-0_Aa_No3355:recorded:2014y298d18h00m15s:138:2:0;
    BB_1-PolX:0:0:1234:281:4n-0_Aa_No3355:2:-:OK:vdif:2014y298d18h00m15s:138.000:277.104:16.064:0;
    BB_1-PolY:0:0:1234:281:4n-0_Aa_No3355:2:-:OK:vdif:2014y298d18h00m15s:138.000:277.104:16.064:0;
    ...
2014-10-25T18:02:56.711 VOM finished 1337-129

2014-10-25T18:02:56.711 VOM starting 3c279 at 2014298180356
2014-10-25T18:06:37.806
    BB_1-Scan:0:0:1234:282:4n-0_Aa_No3356:recorded:2014y298d18h03m56s:138:2:0;
    BB_1-PolX:0:0:1234:282:4n-0_Aa_No3356:2:-:OK:vdif:2014y298d18h03m56s:138.000:277.104:16.064:0;
    BB_1-PolY:0:0:1234:282:4n-0_Aa_No3356:2:-:OK:vdif:2014y298d18h03m56s:138.000:277.104:16.064:0;
    ...
2014-10-25T18:06:37.814 VOM finished 3c279
```

and so forth....

These logs are similar to those usually delivered to the VLBI correlators as standard practice.

As a minor comment, the control system software allows for the recorder to (eventually) provide separate information on the X and Y polarization data streams (`PolX` and `PolY`), but at present the recorder merely reports on both (twice). This could be revised in the future.

5.1.7 Protocols

The `VLBITestObs.py` script has options sufficient to exercise every protocol (highest level to lowest level) required by the APP, with the exception of the routine monitoring of housekeeping which is implemented through the monitoring system.

Monitor Points

All the monitor points are available from `http://monitordata.osf.alma.cl` as planned in [RD1] and mentioned in [RD4]. They are specified from the Control software device generation spreadsheets which are found in any of the software release trees (after release 2014.2):

```
CONTROL/Device/HardwareDevice/VLBIOFLS/idl/VLBIOFLS_spreadsheet.xml
CONTROL/Device/HardwareDevice/VLBIOFLS/idl/VLBIRecorder_spreadsheet.xml
```

from which the lowest level protocols are (automatically) generated. Templates for the TMCDB are also generated from those. These were all tested during the July software verification mission (`https://ictwiki.alma.cl/twiki/bin/view/Control/AppMissionChile072014`). Some additional, more convenient operations are built upon these (*e.g.* the `STATUS()` command of the CCL interface).

Computing ICD Protocols

With regard to the non-monitor (*i.e.* control) protocols specified in [RD4], we review here which ones are involved in the test sessions described in the previous sections. The following CAN protocols are invoked to configure the LTA for forming the array sum when `VLBITestObs` calls the `setAppParameters()` method of the newly created VOM:

```
DOWNLOAD_ANTENNA_SUM_MASK
SET_ANALOG_SUM_MODE
SET_CIC_SUM_INPUT_SWITCH
```

In addition,

```
DOWNLOAD_SUM_SCALING_DATA
GET_SUM_STATUS
```

are called automatically to configure the sum for the number of antennas in the array. The recorders can verify that this has all happened when they receive packets AND the statistics of the bits are distributed appropriately for the sum of antennas. For the test in Section 2.2, all four quadrants are phased, and these statistics are reported in general in the scan check status (Section 2.2.2); specifically, the "OK" on the status line. In more detail, from the scan check (after the test) of one specific fragment (on Recorder1):

```
00000:Report for /mnt/disks/1/0/data/HxDU_Aa_No0175.vdif (8192.5ms) 2014y291d21h03m18s
00000:check 19857 pass + 0 fail | 0 timing / 19857 tested
00000:/mnt/disks/1/0/data/HxDU_Aa_No0175.vdif (c) 29@9493398+012450..9493443+124999
00000:SGv2 2874173204B 8032B/Pkt 1245P/b 288b 357840P 295.526ms
00000:wb:9999848,0,4216808B 287(357315)+0x0+0(0)+1x525 b(P)
00000:sg:810541611D0003EC >124999P/s :0:2869956396 20|8|0:8032| s16737 5#
00000:stats:635424000 samples [00 01 10 11] 19857 pkts
00000:stats:     107992339      209665935      209705787      108059939
00000:stats:      0.169953%      0.329962%      0.330025%      0.170060%
```

The last 2 lines refer to 600 million samples where the four states are distributed as 17%, 33%, 33%, and 17% exactly as expected. Similiar numbers obtain for the rest of the fragments and for the other 3 recorders as well.

When the PICs are to be used (*i.e.* when the `VLBIController` is active),

```
SET_PIC_CONTROL
GET_PIC_Status
```

are used to power-on the PICs and to prepare them for use, and

```
DOWNLOAD_VDIF_HEADER
GET_DOWNLOAD_VDIF_HEADER_STATUS
APPLY_VDIF_HEADER
GET_APPLY_VDIF_HEADER_STATUS
```

are used via the sequence described in [RD4], Section 3.2.2.2.5 to supply the PICs with the VDIF header which will be correct at a specific TE and to apply it at that TE.

Verification of this is perhaps best shown by observing that without the Control system operational, there is no way (via the Engineering port commands) to simultaneously set all of the PICs and their data to the proper time. Thus, in those tests, we can use the packet sniffing tool `vv` to note the PSN values on the various packets arriving at the network devices and compare them to the system clock. *E.g.* from a test session on September 10 (one packet per data stream):

```
root@Mark6-4005:/home/oper/test# for e in {2..5} ; do echo -n $e ; ../bin/vv -i
eth$e ; done
eth2: PSN = 2518211: 2014-09-12 14:15:20 UTC (pkt < now by -20.242441)
eth3: PSN = 2600C9B: 2014-09-12 14:15:06 UTC (pkt < now by -34.342226)
eth4: PSN = 251F721: 2014-09-12 14:15:20 UTC (pkt < now by -20.482220)
eth5: PSN = 26081CA: 2014-09-12 14:15:07 UTC (pkt < now by -34.082254)
```

At this time, 2 of the quadrants were connected to the same recorder; eth2 and eth4 correspond to the two polarizations of one, and eth3 and eth5 are the two for the other. The times were (manually) set and are approximately 14s different, and 20 and 34 seconds off the correct time. By contrast, during a session operated from the Control system, on any port receiving data, one sees, *e.g.* ,

```
root@Mark6-4005:/home/oper/test# ../bin/vv -i eth3 -n2
PSN = AAF6663: 2014-09-10 20:30:55 UTC (pkt < now by -0.002467)
PSN = AAF6664: 2014-09-10 20:30:55 UTC (pkt < now by -0.002582)
root@Mark6-4008:/home/oper/test# ../bin/vv -i eth5 -n2
PSN = AB87A7B: 2014-09-10 20:30:59 UTC (pkt < now by -0.002460)
PSN = AB87A7C: 2014-09-10 20:30:59 UTC (pkt < now by -0.002510)
```

and similarly for all the other devices and recorders. (The precise delay varies.)

The remaining protocol:

```
APPLY_TFB_PHASES
```

existed before the APP project was started. We merely use it to adjust the antenna phases in the slow/fast loops of the phasing system (when the `PhasingController` is active. That we are able to use it is clearly shown from the plots in Figures 5.1 and 5.2 from one VLBI scan of the October 25, 2014 session (not one of the above). The data is in a format developed by the TelCal phasing team for reducing and analyzing performance. Shown are the visibilities received by TelCal in (correlator) subscans 1 and 3 (red) compared with what the slow phase solution correction should produce (blue) for both polarizations. Initially (scan 1, Figure 5.1) the visibilities are randomly distributed with approximately equal amplitudes—*i.e.* on a circle. After commanding the TFBs using the `APPLY_TFB_PHASES` protocol, the red crescent in Figure 5.2 is obtained. If the phasing system were working perfectly, we would be getting something closer to the blue spots—the vertical spread (rotation through several dozen degrees) is some other de-phasing agent which has yet to be properly identified. Scan 2 is not shown, as there are frequently "glitches" in the `DelayServer` which scamble the phases for one scan.

The efficiency of the red crescent is on the order of 90%, which is perhaps acceptable, but we should be doing better.

5.2 Recorder Requirements

In this section we comment on a few specific requirements that the Recorders meet.

5.2.1 Recorder Ethernet Devices

The recorder network interface cards (NICs) each support 2 (nominal) 10 GbE ports. The link-layer (hardware) protocol is designed to transfer 10 Gbps; the actual hardware clocking is faster than 10 GHz to allow for error-correction protocols implemented in hardware. (See, *e.g.*, `http://en.wikipedia.org/wiki/10-gigabit_Ethernet`.)

In the ALMA configuration, two of them are used to support the traffic from two polarizations via the corresponding PICs in the correlator. Each of these provides (at most) an 8 Gbps data stream for an agregate of 16 Gbps into each of the 4 recorders. At reboot, the full device information is as follows:

```
eth3      Link encap:Ethernet  HWaddr 00:60:dd:44:b3:eb
          inet addr:172.16.3.4  Bcast:172.16.3.255  Mask:255.255.255.0
          UP BROADCAST MULTICAST  MTU:9000  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:83

eth5      Link encap:Ethernet  HWaddr 00:60:dd:44:b3:69
          inet addr:172.16.5.4  Bcast:172.16.5.255  Mask:255.255.255.0
          UP BROADCAST MULTICAST  MTU:9000  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
```

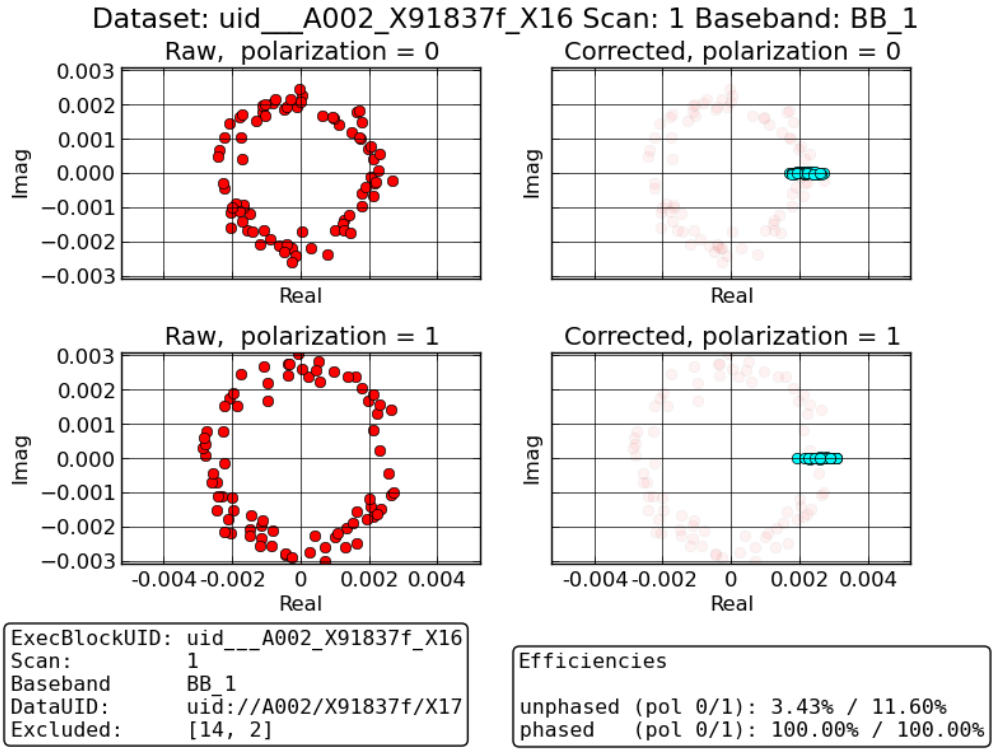

Figure 5.1: *Scan 1*

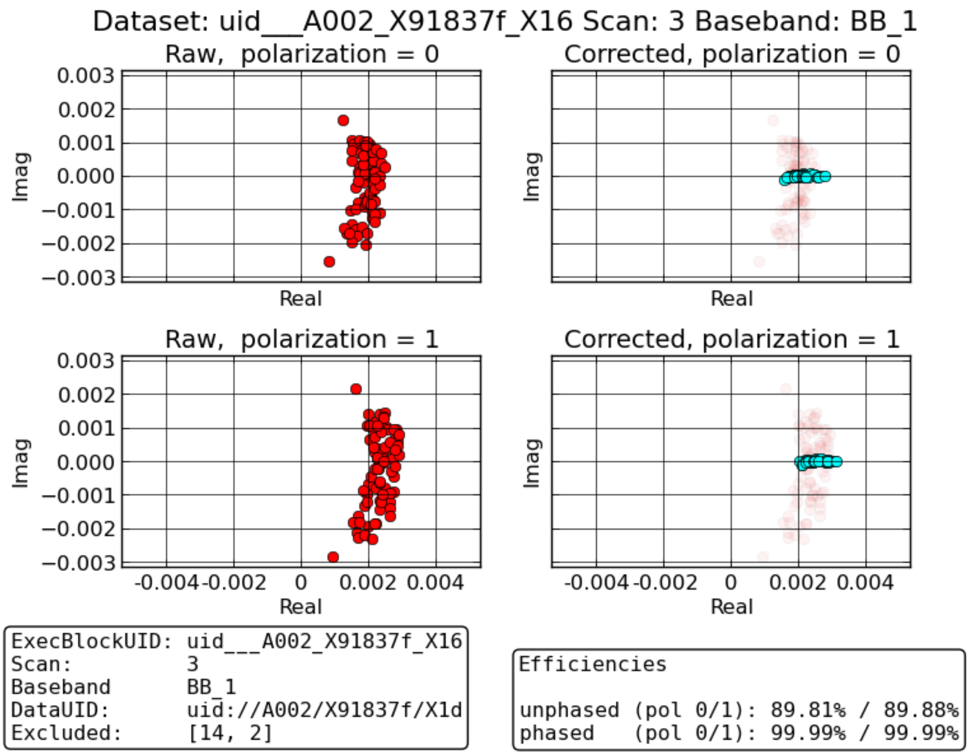

Figure 5.2: *Scan 3*

```
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:90
```

The `MTU:9000` indicates that so-called "jumbo" frames are set—*I.e.* that the ethernet mtu is set to larger than normal. In this case, 9000B per packet are allowed. The VDIF payload is only 8032B, but there is an ethernet packet header as well as UDP and IP headers (a few dozen more bytes).

With 4 recorders, there is an agregate of 80 Gbps of data (one-way) available, but we use only 8/10 of it—64 Gbps.

5.2.2 Modules

In this section we present some information regarding the modules that is citable for requirements verification.

Media

The Mark6 disk modules are based on the modules for the Mark5 recorder series. They share the same form factor, but differ in the internal electronics (eSATA backplane and power connectors). An assembled module is shown in Figure 5.3.

The modules contain 8 disks which need not be of the same type, but which conventionally are. Modules in current use at ALMA were built from Western Digital 3TB disks and Seagate 4TB disks. The Western Digital 3TB disks are somewhat obsolete at this point. We are currently investigating 6TB disks.

Here are 3 current quotes for 4 modules (32 disks/recorder):

```
Description
    Mft Part#     Reseller/Item#        Cost per    Qty     Total Cost
HGST 6TB UltraStar He6 SATA 6Gbps 3.5 HDD
    0F20572       PCNation/17580048       $647.98    32      $20735.36

Seagate 6TB 7200RPM  3.5 HDD
    ST6000NM0024  GovConnect/17008228     $487.29    32     $15,593.28

WD 4TB Black SATA 6Gbps 3.5 Internal HDD
    WD4003FZEX    GovConnect/16331568     $245.86    32      $7,867.52
```

Shipping

The Mark6 modules were designed with shipping in mind. Figure 5.4 shows a module in the single-module packaging that has been used for the past decade. The pink and black objects are hard foam with cut-outs for the module.

5.2.3 Power

As it turns out, 2 of the Mark6 recorders (host and expansion) are plugged into the same power strip, see Figure 5.5, making a measurement of 4.9A in normal 220V operation for an estimated 2 kW for 4 recorders.

Figure 5.3: *Mark6 Module*

Figure 5.4: *Mark6 Module Shipping*

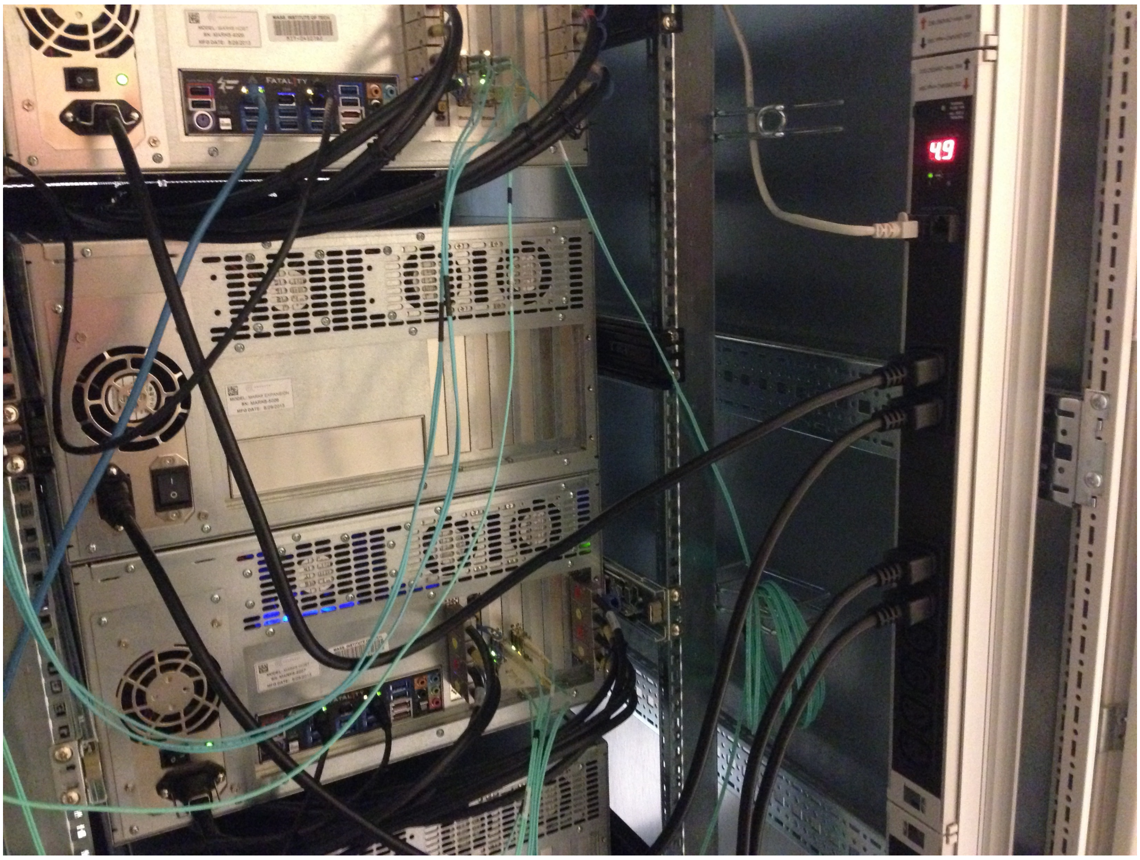

Figure 5.5: *Mark6 power measurement in operation*