RDBE Calibration Data and DiFX

VLBA Sensitivity Upgrade Memo 44

Walter Brisken & Amy Mioduszewski

19 Oct 2015

1 Introduction

This memo describes the path that calibration data takes within the RDBE-based VLBA and GBT and the WIDAR-based VLA. In this document calibration data consists of system temperatures ($T_{\rm sys}$), antenna gains (as function of pointing and frequency), pulse calibration data (tone phases and amplitudes), round-trip (cable cal) data, weather data and system generated data editing flags. The impatient reader should flip to Figs. 1 & 2 to get the big picture. From there details of interest can hopefully be found in the text.

Some information about each bit of software described here can be found on the VLBA Software Wiki at https://staff.nrao.edu/wiki/bin/view/NM/VlbaSoftware.

2 Historical VLBA calibration data paths

Over the history of the VLBA the standard path for calibration data to join visibility data from the correlator (either the hardware correlator or DiFX) as evolved. These are described below.

2.1 Pre-calibration transfer

In the earliest days of the VLBA the calibration data path was similar to what was typical of other VLBI stations (and this practice continues today at many VLBI stations). Calibration data derived during the observation were extracted from the monitor data archive by a program called `tsm` which creates a file called *project*`cal.vlba` (or its compressed equivalent with a trailing `.gz`). Here *project* is the name of the experiment with session code if applicable (e.g., `bb210a` or `tx001`). This file can be found in `/home/vlbiobs/astronomy/`*mmmyy*`/`*project*`/` where *mmmyy* describes the month of the observation in a format such as `jul13`. This directory is available for web browsing via http://www.vlba.nrao.edu/astro/VOBS/astronomy/*mmmyy*/*project*/. Users would download this file, separate it into parts (often with AIPS task VLOG) for consumption into AIPS through different tasks.

2.2 Calibration transfer

To simplify access to calibration data by users the VLBA hardware correlator began attaching calibration data directly to the IDI-FITS files produced by the correlator. The *project*`cal.vlba` remained available in cases where problems in "calibration transfer" occurred or if manual edits to the calibration data were required. When concatenating these FITS-IDI files in AIPS, a procedure (MERGECAL) is required to ensure the calibration data were not duplicated.

2.3 Pre-RDBE DiFX

The calibration data path of the VLBA DiFX correlator was designed to mimic the calibration transfer convenience introduced by the hardware correlator. In order to generalize the capability and to avoid exposing core DiFX utilities to the complications and possible ambiguities[1] of the ANTAB format, a separate command line program called `vlog` was produced that converts the *project*`cal.vlba` file into four separate files that `difx2fits` looks for when forming FITS-IDI output files. These files are called `tsys`, `pcal`, `weather`, and `flag`. Formats for these files are documented in the `DiFX Reference Manual` (http://cira.ivec.org/dokuwiki/lib/exe/fetch.php/difx/difxuserguide.pdf). The gain tables are generated by selecting relevant entries from files found in a directory pointed to by environment variable `GAIN_CURVE_PATH` . These files are in a format readable by ANTAB. Unlike for system temperature measurements in ANTAB format, there are no concerns about ambiguities in the gain data. In the VLBA deployment, `GAIN_CURVE_PATH` points to a directory (currently `/home/swc/difx/gaincurves/`) which contains symbolic links to gain files maintained by VLBA operations which live in `/home/jansky3/vlbaops/TCAL/` .

Because `difx2fits` is run on whole experiments at a time users do not need to run MERGECAL in this case.

3 Sources of RDBE calibration data

This section describes the source of calibration data for the VLBA wideband observing system (using RDBEs and Mark5C recorders). As noted below, some data from some antennas other than the VLBA make use of the same path.

3.1 System temperature

System temperature values derived from the legacy observing system are not appropriate for use with data collected with the wideband (RDBE) system as bandwidths and bandpass shapes differ and in principle introduced noise could differ.

For both RDBE personalities (PFB and DDC) switched power is extracted within the RDBE firmware. Squared signal level, a quantity proportional to power, is separately accumulated for the two states of the injected switched power ('on' and 'off') for each baseband channel created by the RDBE. In the case of the PFB, all 32 polyphase channels produce switched power even though only 16 polyphase channels get recorded. In the case of the DDC, switched power is measured only within the bandpasses requested by the observer. The RDBE server application collects the integrated switched power every second and multicasts the information using multicast port 20021 and multicast group 239.193.2.6. The switched power records are time-tagged by the server application.

The site control computer (e.g., `pt-cc`) runs a process called `rspr` (the RDBE Switched Power Receiver) which collects switched power from the two RDBEs. `rspr` maintains knowledge of the configuration of the RDBEs by monitoring information multicast by the executor and by periodically polling monitor points within the RDBEs via their MIB interface. Using this state information, each set of switched power data is associated with a sky frequency, polarization, bandwidth and sideband for unambiguous context. Switched power data and the channel context is written to

[1] ANTAB formats don't contain enough information in uncommented fields to uniquely associate $T_{\rm sys}$ data to the correct baseband channels so VLBA-specific comments are currently used as a guide. Further, ANTAB files do not assign timestamps that are unique as the year is not provided.

a text file on the control computer in `/tmp/sp_staging/` in files with systematic names of the form *station_yyyy_ddd_hh_mm_ss*`.switched_power` . A new file is created whenever an experiment starts or stops, or when UTC midnight occurs. Normally between experiments the RDBEs are not configured with frequency information and therefore although switched power data is being generated it is not saved. At the close of each file, it is compressed using the `xz` compression tool. This tool when used in its level 3 compression mode was tested to perform 30% better than `gzip` and about 15% better than `bzip2`. Compression modes greater than 3 resulted in significantly longer execution time with little or negative gain in compression. After compression the file is moved to `/tmp` where `RMDS` (the Remote Monitor Data Server) looks for these files. They are then copied to Socorro and placed in `/home/vlba/metadata/switchedpower` for long term storage. The files on the control computer are moved from `/tmp` to `/tmp/rspr.tmp` where they remain for 10 days to allow for recovery if data transmission from the site fails. Up through this point the handling of switched power data is completely automatic.

Computation of system temperature from switched power files (both those suitable for DiFX and in ANTAB format) can be made either with `rdbetsm` (see Sec. 6.1).

3.1.1 Multicast data format

The RDBE server program multicasts switched power in a binary format that differs a bit between the PFB and DDC personalities. In all cases the transmitted data use big-endian (Motorola) byte ordering.

Every PFB switched power packet contains data for *all* polyphase channels, including the awkward end channel which contains 16 MHz from each of the top and bottom of the sampled band for a total of 32 sets of numbers. The first 16 bytes contain an Binary Coded Decimal (BCD) encoding of the time:

Code 1 *PFB switched power header.*

```
struct SwitchedPowerSetPFB
{
        char timeBCD[16];       /* YYYYDDDHHMMSSxxx (xxx are dummies) */
        struct SwitchedPowerPFB switchedPower[0]; /* points to the data */
} __attribute__((packed));
```

The BCD timestamp has a value 1 second larger than the actual end time of the switched power integration. The zero-length array `switchedPower` is a C language construct that makes access to the switched power array easier without adding length to the header. This header is followed immediately by data for 32 channels, each being a 24 byte long structure:

In the DDC personality switched power is only generated for the channels being created which can number between 1 and 4. Like the PFB, each multicast switched power packet has a header and a small structure for each channel which are defined below.
In the DDC the two count values are added to the header and allow for edge cases where unequal amount of data are counted for the 'on' and 'off' states to be properly normalized.
Here the array index provides the channel number so that does not need to be explicitly transmitted.
Sampler bias is accounted for in the `vOn` and `vOff` parameters.

Code 2 *PFB channel data.*

```
struct SwitchedPowerPFB
{
        uint16_t ifNum;         /* 0 or 1 */
        uint16_t chanNum;       /* 0 to 15 */
        uint16_t dummy[2]; /* not used */
        uint64_t pOn;
        uint64_t pOff;
} __attribute__((packed));
```

Code 3 *DDC switched power header.*

```
struct SwitchedPowerSetDDC
{
        char timeBCD[16];       /* YYYYDDDHHMMSSxxx (xxx are dummies) */
        uint32_t countOn;       /* used to normalize ON data */
        uint32_t countOff;      /* used to normalize OFF data */
        struct SwitchedPowerDDC switchedPower[0]; /* points to the data */
} __attribute__((packed));
```

Code 4 *DDC channel data.*

```
struct SwitchedPowerDDC
{
        uint64_t pOn;
        uint64_t pOff;
        int64_t vOn;
        int64_t vOff;
} __attribute__((packed));
```

3.1.2 Archival switched power data format

Data produced by `rspr` is stored in compressed (with `xz`) text files. These text files currently support 2 kinds of information: comments and data. Comments begin with a pound sign (#) and are ignored by downstream software. Some of the information in the comments may be useful for diagnostics. Data lines contain all the switched power information for one RDBE at one time stamp and thus can be quite long. Each line has 3 columns of header information followed by 5 columns of data per channel. The header consists of the antenna name (2 characters), and start and stop time of the Tsys integration period represented in MJD. Each channel has the following information:

Column	Value	units	comments
1	sky frequency	MHz	+ for net USB, − for net LSB
2	bandwidth	MHz	
3	polarization		R or L
4	$P_{\rm on}$		$= \sum v_{\rm on}^2$
5	$P_{\rm off}$		$= \sum v_{\rm off}^2$

3.1.3 DiFX extracted system temperature

It is possible to command DiFX to extract switched power from 2-bit quantized baseband data. This was used before the RDBE switched power extraction was working. The primary disadvantage in the DiFX-based switched power extraction is that it can have a significant impact on correlation performance. For this reason the DiFX-based system temperature measurements are not used, though they remain an option if needed.

3.2 Gains

With the exception of the VLA, gain data for HSA antennas continues to be collected from the same source as used by the legacy (pre-RDBE) system (see Sec. 2.3). In the VLA case, a antenna-specific gain file will be created for each phased-VLA experiment. `difx2fits` will look for files of the form *project*`.antenna.gain` when building IDI-FITS files. Such a file, if found, will override values found in files in `$GAIN_CURVE_PATH`.

3.3 Pulse calibration

There are two sources of pulse calibration information for RDBE data. The first is derived in real-time by software running on the X-cube switch. The second is the DiFX correlator. The legacy system pulse calibration data should not be used for RDBE data as the signal paths differ.

3.3.1 X-cube derived pulse calibration

The X-cube software switch which is installed at each VLBA site and the GBT sees all of the recorded baseband data and contains significant computing resources. Software running on this machine, `pcalxcube`, determines the RDBE configuration[2] by listening to emitted archive messages

[2]Currently it is assumed that the SSLO of the IFs are multiples of the pulse cal interval. At this time the SSLO is always a multiple of 100 MHz, so there is no concern. In the future it is possible that flexible LOs could violate this assumption. It will be possible to handle most of the likely cases but some additional work within `pcalxcube` will be needed.

and automatically configures itself for appropriate pulse cal extraction[3]. A time-domain folding algorithm is used to attain a computational load roughly equivalent to one floating point addition per baseband sample. Parallelism both through use of multiple CPU cores and through use of vector instructions enables even the highest sample rates, 1024 Msps aggregate, to be comfortably handled.

Every second accumulated time-domain data is Fourier transformed. The spectrum that is produced from this DSP technique typically contains hundreds or thousands of frequency bins of which a small fraction contain frequencies containing pulse cal tones. Due to the folding, only frequencies within a fraction of 1 Hz (the inverse of the accumulation period) of the bin center are coherently combined; the choice of fold parameters is such that the tones always land in these sweet spots.

This efficient algorithm can extract *all* of the tones within the bands with essentially no additional overhead beyond the cost of the first tone extracted, however there is a cost to transporting and storing the pulse cal information. To reduce the data volume, up to four tones are extracted from each band. For bandwidths less than four times the tone spacing all tones in the band are extracted. For DDC channels larger than four times the tone spacing, two tones are placed near the band edges as close as possible to $\Delta\nu/8$, one tone is placed as close to the band center as possible, and the final tone is placed one tone away from the one near the center. For PFB channels in the case of 1 MHz tone spacing, the four tones to be extracted are 5, 17, 18, and 27 MHz from the SSLO. The center tone (at 16 MHz) is avoided as the PFB has an artifact tone at mid-band. Tones residing on the baseband channel edges are never extracted as they are useless for multiple reasons.

The extracted pulse cal tones are packed into a binary structure containing data for up to 64 tones and multicast using port 41222 and group 225.2.2.2. In cases where two DBEs are in use, pulse cal data for each DBE will be packaged in a separate packet.‘ These messages are picked up by `rspr` which writes them to text files. `rspr` then handles these files exactly the same way is it handles switched power data. Ultimately the extracted pulse cal data ends up in `/home/vlba/metadata/xcube_pcal`.

The binary data multicast from `pcalxcube` to `rspr` consist of a header:

Code 5 *Binary pulse cal data header.*

```
struct PulseCalRecord                 /* 16 bytes for the header */
{
        double mjd;                   /* MJD of the center of the integration */
        float dur;                    /* duration [sec] of the measurement */
        uint16_t nTone;               /* number of pulse cal tones represented here */
        uint16_t version;             /* version number */
        uint8_t toneSpacing;          /* [MHz] 0 means no tones */
        uint8_t flagState;            /* 0 is unflagged.  Otherwise flagged */
        uint16_t dummy;

        struct PulseCalTone tone[0];  /* use the zero-element trick for unknown size */
};
```

followed by n_{Tone} tone records:

[3]As of this draft, 20150730, the state of the pulse cal generator is not accessible to `pcalxcube` and 1 MHz is universally assumed.

Code 6 *Binary pulse cal tone record.*

```
struct PulseCalTone          /* 20 bytes per tone ; must be multiple of 4 */
{
        float re, im;        /* complex parts of the tone measurement */
        uint32_t skyFreq;    /* frequency [MHz] of the tone */
        uint8_t channel;     /* channel number of the RDBE (0-based) */
        uint8_t rdbeId;      /* RDBE number (currently 0 or 1) */
        char pol;            /* 'R' or 'L' */
        char sideband;       /* 'U' or 'L' */
        uint16_t threadId;   /* only 10 LSBs are used.  6 MSBs reserved for future */
        uint16_t dummy;
};
```

Future needs can be handled by updating the format and changing the header version number. As of this writing the version number is 3.

One could snoop on the extracted pulse cal data in real time by logging into the X-cube unit and running `testpulsecalreceive` program. Every second a new batch of values will be displayed on the screen.

3.3.2 Archival pulse cal data format

The pulse cal records that end up in `/home/vlba/metadata/xcube_pcal` are formatted in text as follows. Comment lines begin with # and contain no vital information, although some of the information in comments is formatted such that it can help software interpret the pulse cal data. Data lines always begin with 6 fields describing the content of that data line:

Column	Value	units	comments
1	day	days	Time centroid of measurement (MJD, inc. fraction)
2	dur	days	Duration of measurement
3	version		Version number of this record type (currently version 2)
4	nTone		Number of pulse cal tones detected per band per polarization
5	toneSpacing	MHz	Pulse cal generator setting. Either 0 (off), 1 or 5
6	flagState		If not zero, a data validity flag has been raised

Following these four fields is a record for each tone, *nTone* total. Version 2 records (the first to contain usable data) consist of the following 8 fields:

Column	Value	units	comments
1	re		Real component of the pulse cal tone
2	im		Imaginary component of the pulse cal tone
3	rdbeId		RDBE number providing this data (0 for dbe-1, 1 for dbe-2)
4	thread		VDIF thread containing the channel (0 for Mark5B format)
5	channel		Channel number (0 based), inferred from RDBE setup
6	pol		Polarization, either `R` or `L`
7	sideband		BBC (FIXME) sideband `U` or `L`
8	freq	MHz	Sky frequency of tone

Note that this format differs from that which is generated and used by DiFX. This raw format is intended for use in diagnostics and in generating ANTAB files which may be sent to correlators that do not generate pulse cal data themselves.

3.3.3 DiFX derived pulse calibration

DiFX has the ability to extract all pulse cal tones within all observed baseband channels. DiFX pulse cal extraction is documented in detail at http://cira.ivec.org/dokuwiki/doku.php/difx/pcal . This section summarizes some relevant aspects.

DiFX extracts pulse calibration tones before the fringe rotation and Fourier transform steps. The calculations are primarily performed in the time domain with a single transform to the frequency domain at the end of the calculations. Depending on the numerology of the tone separation, the sample rate and the baseband channel tuning (modulo sample tone separation) one of three algorithms is used. The algorithms used in DiFX are very nearly the same as those used in `pcalxcube`. The most general algorithm adds the capability to reduce the fold buffer sizes by applying a complex-valued rotation on each pass through the folding (somewhat equivalent to making the fold buffer a non-integer length).

Extracted tones are stored within the `.difx/` directory created for each DiFX job correlated. A separate pulse cal file will be written for each antenna. The pulse cal data files are systematically named in the form: `PCAL_`*day_sec_ant*, where *day* is the 5 digit integer MJD of the start of visibilities, *sec* is a zero-padded 6 digit number of seconds since the MJD midnight, and *ant* is the 1 or 2 letter antenna name in capital letters. There is potential for these text files to have very long lines (more than 10,000 bytes) when many pulse cal tones are extracted.

The format of DiFX-extracted pulse cal data is described in the DiFX Users Manual (http://cira.ivec.org/dokuwiki/lib/exe/fetch.php/difx/difxuserguide.pdf).

3.4 Cable cal data

Information on the round trip cable calibration is collected by the L103 module and is injected in to the legacy system monitor stream via the VME station computer. Cable cal data is captured by `mon2xml` and sent to the new `vlbampts` database (see Sec. 5). A python program `db2cc` can extract the relevant records if given the `.vex` file for an experiment. Normally this program will be called by `rdbetsm` and humans need not be involved. Cable cal files will be left in the `jobs/` directory. One file will be created per antenna with the name *project*.*station*`.cablecal`, all lowercase.

As some point the VME replacement program will transfer cable cal monitoring to a system that bypasses the VME and `mon2xml`, but everything downstream of that will remain unchanged.

While the cable cal data still comes from the legacy system it remains relevant for RDBE operation.

3.4.1 Historical note

In the past, the legacy program `tsm` extracted cable cal data and merged into the pulse cal table in the legacy VLBA calibration (*project*`cal.vlba`) file. For a legacy project `vlog` would extract the cable cal and place that data within the VLBA-wide `pcal` file (or multiple *project*.*antenna*`.pcal` files, one per antenna) along with the pulse calibration data. For RDBE projects where the legacy system pulse cal data is not of value the cable cal data was extracted to a file called *project*.*antenna*`.cablecal`.

3.5 Weather

Weather data follows the same general path as cable cal data. The VME receives data from the weather station and emits it as legacy monitor data which `mon2xml` collects sends to the `vlbampts` database (see Sec. 5). A python program `db2wx` can extract the relevant records if given the `.vex` file for an experiment. Normally this program will be called by `rdbetsm` and humans need not be invoked manually. Weather data files will be left in the `jobs/` directory. One file will be created per antenna with the name *project*.*station*.`weather`, all lowercase.

As some point the VME replacement program will transfer weather data monitoring to a system that bypasses the VME and `mon2xml`, but everything downstream of that will remain unchanged.

Weather data from both the VLA and GBT are gathered by `db2wx` in similar, but not identical, manners. GBT weather data ends up in the `vlbampts` database and VLA weather data gets to the `evlampts` database. Both have different monitor point names than used by the VLBA.

3.6 Flags

System data validity flags, or simply "flags", are used by astronomers to mask potentially imperfect data. Common flags include "source change in progress", "synthesizer unlocked", or "ellipsoid position error". There are two streams of flags in the VLBA system. The first, for subsystems controlled by the VME, is through `mon2xml`, and the second, for subsystems under executor control, are through MIBs or software MIB emulators. In both cases emitted flags are captured (see Sec. 5) and placed in the `vlbampts` database.

A python program called `db2flag` extracts relevant information if given the `.vex` file for an experiment. Normally this program will be called by `rdbetsm` and humans need not be invoked manually. Flag files will be left in the `jobs/` directory. One file will be created per antenna with the name *project*.*station*.`flag`, all lowercase.

4 Other sources of non-RDBE calibration data

The new calibration path allows independent sources of calibration data for each antenna in a project. There are two supported data formats.

The first is ANTAB format for system temperature (and its associated formats for flags, weather and pulse/cable cal). Like for the VLBA, a combined "cal" file containing multiple calibration tables is supported. `vlog` can be run directly on as many such files as needed to produce a full set of calibration information. Since the new behavior is to write separate files for each antenna there should be no concern about overwriting the data for the VLBA.

The second acceptable format (one less likely to be used) is DiFX-formatted per-antenna per-table calibration files. The format of these files is described in the sections above.

Antenna naming is a possible source of error in transferring calibration. The 1- or 2-letter antenna code used both inside and in the name of the DiFX-formatted calibration files must match that used for that antenna within the `.vex` file. These names are treated as case-insensitive.

4.1 VLA system temperature

System temperatures for the phased VLA are created within AIPS. A special AIPS task VLAMP us used to create an ANTAB file compatible with `vlog` . Once this file is created[4] the analysts

[4] As of this writing Amy Mioduszewski does this.

should run `vlog` on this file to produce *project*`.y.tsys` .

5 Monitor data transfer

A program, `RemoteMonitorDataServer`, running on the control computer, is responsible for transferring monitor data, pulse cal data, and switched power data back to appropriate directories under `/home/vlba/metadata`, which live on the computer "filehost" in Socorro. Ten days of past records are stored on the control computer to allow for recovery if a data transfer failed. The mechanism for recovering this data is documented (where? FIXME!!!).

6 Post-observation software

The sections that follow describe the software tool chain that is used to get calibration into the IDI-FITS output files. See Fig. 2 for a big picture view of how the various parts fit together.

Upon the completing of `rdbetsm`, the programs `tsm` and `vlog` are no longer needed. The original `tsm` program should not be run anymore. In certain cases running `vlog` on non-NRAO stations may be useful so as to produce calibration data that can be fed straight into `difx2fits`.

The following sections show sequence of commands that are representative of a normal VLBA observation. Details may vary from project to project. Each program provides some help information if needed by supplying `-h` or `--help` command line arguments.

6.1 rdbetsm

The program `rdbetsm` is responsible for gathering data from many parts of the VLBA system and formatting them for later use. Two output data types are produced by default: a familiar `cal.vlba` (e.g., ANTAB) file, and file suitable for ingestion into `difx2fits` to complete calibration transfer to the output FITS files.

Five different types of calibration and editing data are handled by `rdbetsm` as described in the following sections.

6.1.1 Tsys

Computation of system temperature occurs after observing is complete by `rdbetsm`[5]. This program takes as input a `.vex` file for an experiment and produces a set of files with systematic names *project*.*station*`.tsys` in a format that `difx2fits` can handle. This format is documented in the DiFX Reference Manual. `rdbetsm` first averages the switched power. The averaging periods are determined by taking the data valid period for each scan as depicted in the `.vex` file and dividing it into equal sized chunks as close to 30 seconds in duration as possible (the 30 second default can be overridden if desired though command line arguments to `difx2fits` or to `makefits`). Validity flags, as determined by `rspr`, are honored by `rdbetsm`. These flags do not adjust the averaging interval, so it is possible for some data to have as little as 1 second of data averaging. Noise diode noise temperatures, $T_{\rm cal}$, are required to convert switched power into $T_{\rm sys}$

$$T_{\rm sys} = \frac{1}{2}\frac{P_{\rm on} + P_{\rm off}}{P_{\rm on} - P_{\rm off}} T_{\rm cal}. \quad (1)$$

[5] `rdbetsm` grew out of a program `rdbetsys` which served only to produce system temperature data. This program is still available but `rdbetsm` should be considered its successor.

`rdbetsm` finds these values in files found in a directory pointed to by environment variable `TCAL_PATH` . Currently this points to `/home/jansky3/vlbaops/TCAL` .

In addition to adding this data to the output ANTAB file, A separate output file is made per station with name *project*.*station*.`tsys` which is placed in the project `jobs/` directory. The format of this file is as follows. Comment lines begin with # and contain no vital information. Data lines begin with four columns:

Column	Value	units	comments
1	antId		Station name abbreviation, e.g., `LA`
2	day	days	Time centroid of measurement (MJD or day of year, inc. fraction)
3	dur	days	Duration of measurement
4	nChan		Number of recorded baseband channels

After these three columns there is a pair of columns for each recorded baseband channel, listed in "natural channel order", meaning that the names of the channels, as seen in the `.vex` file, are put in alphabetical order. The first column for each channel is the system temperature (in K), and the second is the name of the receiver (e.g., `20cm`).

6.1.2 Pulse Cal

No calculations beyond averaging are needed to produce usable pulse cal data. Averaging is performed in the same manner as for Tsys data. Validity flags, as determined by `rspr`, are honored by `rdbetsm`. These flags do not adjust the averaging interval, so it is possible for some data to have as little as 1 second of data averaging, which is typically enough to measure phase to about 1 degree.

Because it is expected that DiFX will produce pulse cal data that will end up in the FITS file `rdbetsm` does not package pulse cal data for calibration transfer purposes.

6.1.3 Cable Cal

A helper program, `db2cc`, is used to extract cable cal monitor points from the `vlbampts` database. The database stores the raw monitor values which are converted to delay (in picoseconds) according to a recipe that can be found on page 9 of VLBA Technical Report #5. Note that additional scaling by $512.0/36.0 \times 3276.8$ is required due to scaling performed in `mon2xml`. The cable cal value reported in the output ANTAB formatted file is determined by linear interpolation to the averaging center for pulse cal data.

A separate output file is made per station with name *project*.*station*.`cablecal` and is placed in the project `jobs/` directory. The format of this file is as follows. Comment lines begin with # and contain no vital information. Data lines always have 4 fields:

Column	Value	units	comments
1	antId		Station name abbreviation, e.g., `LA`
2	day	days	Time centroid of measurement (MJD or day of year, inc. fraction)
3	dur	days	Duration of measurement
4	cableCal	ps	Cable calibration measurement

There may be two additional columns that contain debug information.

6.1.4 Weather

A helper program, `db2wx`, is used to extract weather data from the `vlbampts` database for VLBA and GBT, and from the `evlampts` database for the VLA. Recovered values are used without further recalculation, averaging, filtering, or interpolation.

A separate output file is made per station with name *project*.*station*.`weather` and is placed in the project `jobs/` directory. The format for the output per-station weather files is as follows. Comment lines begin with a # and contain no vital information. Data lines always contain 9 columns.

Column	Value	units	comments
1	antId		Station name abbreviation, e.g., `LA`
2	time	days	Time of measurement (MJD, inc. fraction)
3	T	C	Ambient temperature
4	P	bar	Pressure
5	dewPoint	C	Dew point
6	windSpeed	m/s	Wind speed
7	windDir	deg	Wind direction (measured E of N)
8	precip	cm	Accumulated rain since UT midnight
9	windGust	m/s	Maximum wind gust over collection period

6.1.5 Flags

Data validity flags are collected from the `vlbampts` database by helper program `db2flag`. There are two sources of such data:

1. MIB- or MIB-emulator-based hardware emit alerts that are directly collected by `RemoteMonitorDataServ`
   As of this writing the following MIB-based flags are generated:
   - (M402) ellipsoid position error
   - (L404) synthesizer out of lock
2. SIB-based hardware is still controlled by the VME. A new program, `mon2xml` converts relevant monitor data into EVLA messages, which are then collected by `RemoteMonitorDataServer`. As of this writing the following SIB-based flags are generated:
   - (L104) synthesizer out of lock
   - Source change in progress
   - Antenna pointing error
   - Focus/rotation mount out of position
   - Antenna out of point mode

As the VME replacement program continues the monitor points will eventually migrate entirely to the MIB path.

A separate output file is made per station with name *project*.*station*.`flag` and is placed in the project `jobs/` directory. The format of DiFX flag data is as follows. Comment lines begin with a # and contain no vital information. Flag lines always consist of exactly 5 columns.

Column	Value	units	comments
1	antId		Station name abbreviation, e.g., `LA`
2	start	day	Beginning of flagged period (MJD, inc. fraction)
3	end	day	End of flagged period (MJD, inc. fraction)
4	recChan		Record channel affected; -1 for all (zero based)
5	reason		Reason for flag, enclosed in single quotes, truncated to 24 chars

6.1.6 Running rdbetsm

Shortly after[6] an observation completes the VLBA operators can run `rdbetsm` to extract and `tsmplot` to plot system temperatures collected over the previous experiment.

⟶ `cd /home/vlbiobs/astronomy/`*MonYr*`/`*project*

⟶ `rdbetsm` *project*`.vex` *project*`cal.vlba` *# to be done by operators*

Database access is not immediate so running `rdbetsm` can take many minutes. At this point the `jobs/` subdirectory should contain `.tsys`, `.cablecal`, `.weather`, and `.flag` files for the experiment. An ANTAB file (ending in `cal.vlba` containing all of this information plus pulse cal, if extracted with `pcalxcube`, will be left in the project directory.

6.1.7 Plotting rdbetsm output

The `cal.vlba` file can be plotted, resulting in the familiar `tsm` plots:

⟶ `tsmplot` *# to be done by operators*

∗ `IN` *project*`cal.vlba`

∗ `PL` *project*`/ps`

∗ `/`

∗ `EX`

∗ `/`

Perhaps at this point the `rdbetsm output` (the `cal.vlba` file) would be gzipped to save space:

⟶ `gzip` *project*`cal.vlba`

6.2 vex2obs

This program retrieves clock, EOP, and Mark5 module usage information from the `vlbampts` database and appends the information at the end of the `.vex` file.

⟶ `vex2obs` *project*`.vex` *# to be done by analysts*

This produces a `.vex.obs` file which should end up in the `jobs/` directory.

6.3 vlatsys

For experiments involving the phased VLA, one or more "VLAMP" files will be made. Process these to produce system temperature and gain files for the phased VLA. This will produce a file

⟶ `vlatsys` *vexFile VLAMPfile(s)* *# to be done by analysts*

Make sure that the `.tsys` file that is produced ends up in the `jobs/` subdirectory before correlator jobs are queued.

[6] Sometimes up to 20 minutes is required for completion of transfer of the switched power data from the sites to Socorro.

6.4 difxqueue

Correlation and IDI-FITS file building at the VLBA DiFX correlator is done within a staging area. These directories are within `/home/swc/difx/queue`, defined by environment variable `DIFX_QUEUE_BASE` and are distinct from the long-lived `/home/vlbiobs/astronomy` file system. All files needed for both the correlation and IDI-FITS building need to be present in the staging area, including the calibration data discussed in this document. The data in question is copied to the correlator staging area with the `difxqueue` command. There are no user-visible changes to `queueVex` to support the new per-antenna per-table calibration files.

⟶ `cd jobs` *# change to the jobs/ subdirectory where this should be run*
⟶ `difxqueue add` *pass* *# to be done by analysts*

6.5 showcal

A python utility has been created that can be used prior to FITS file generation to display which source of calibration data will be used for each table for each antenna. This should be run after correlation for the most accurate results.

⟶ `cd jobs` *# change to the jobs/ subdirectory where this should be run*
⟶ `showcal` *vexFile passName* *# to be done by analysts*

Things to look for are:

- Antennas lacking certain calibration information, especially those where it is expected to be found. Foreign stations won't have any calibration data unless it was explicitly delivered to the analysts and properly processed (possibly through `vlog` acting on an ANTAB format file).
- Mark5C projects where pulse cal data comes from a source other than DiFX. Some projects do not enable pulse calibration; in these cases it is expected that this column is empty.

6.6 makefits and difx2fits

`difx2fits` is the core DiFX program that converts the native output from `mpifxcorr` and calibration data to IDI-FITS format. A wrapper program called `makefits` is used by the VLBA analysts to run `difx2fits`. This program adds a layer of accountability, changes filenames to match what is expected by downstream software (e.g., `difxarch`), prepare output for the sniffing process and check for some possible errors.

⟶ `makefits` *pass* *# to be done by analysts*

7 Unfinished work

As of this writing several details have yet to be properly addressed.

- `pcalxcube` does not know the state of the pulse cal generator and simply assumes 1 MHz tone spacing. This will be temporarily solved by adding a virtual monitor point, `L110.Pulse_Cmd` to the RDBEs. Longer term either the L110 MIB emulator will be able to handle this, or the X-cube MIB will be resurrected and a control/monitor point will be added here to the same effect.

- Handling multiple VLAMP files as would be produced for cases where the frequency setup changes during a phased VLA observation is not fully supported. Merging of the system temperature data into proper time average is complete but proper linkage to gain table entries is not fully working. At this time it is not possible to perform calibration transfer in cases where the frequency setup of a VLBI+VLA observation changes.
- `db2flag` does not generate flags for VLA or GBT.

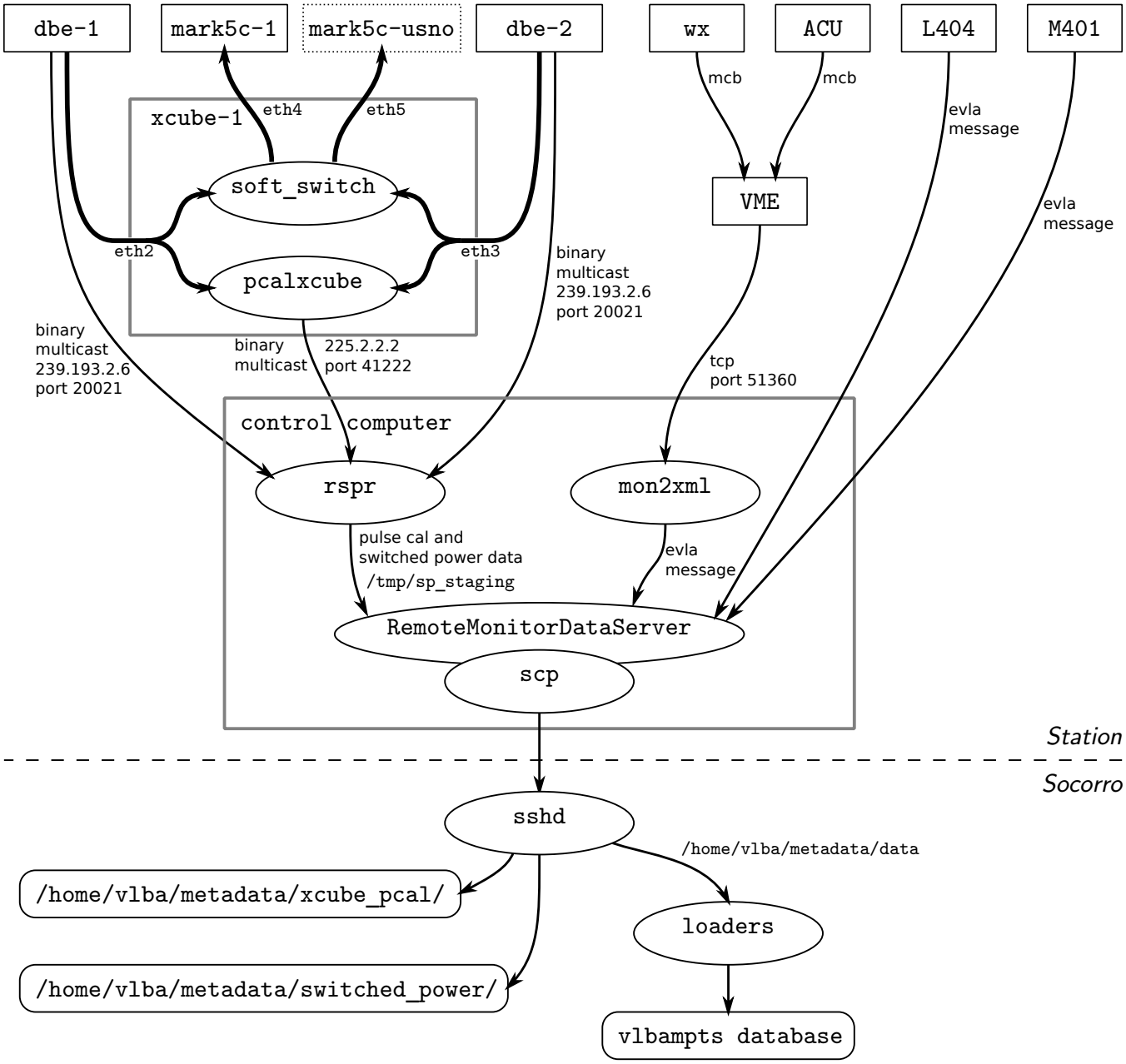

Figure 1: *The real- and near-real-time portions of the VLBA calibration and data validity system.* In this diagram arrows indicate direction of data flow. Thick arrows (upper left area) represent baseband data while other arrows are low bandwidth monitor data. Ovals represent pieces of software. Overlap of ovals represents one program directly calling another. Rectangles represent pieces of hardware. Note that not all devices are listed. Rounded rectangles represent long-term data storage. Other data paths are temporary. Control data paths or real-time monitoring paths are not shown.

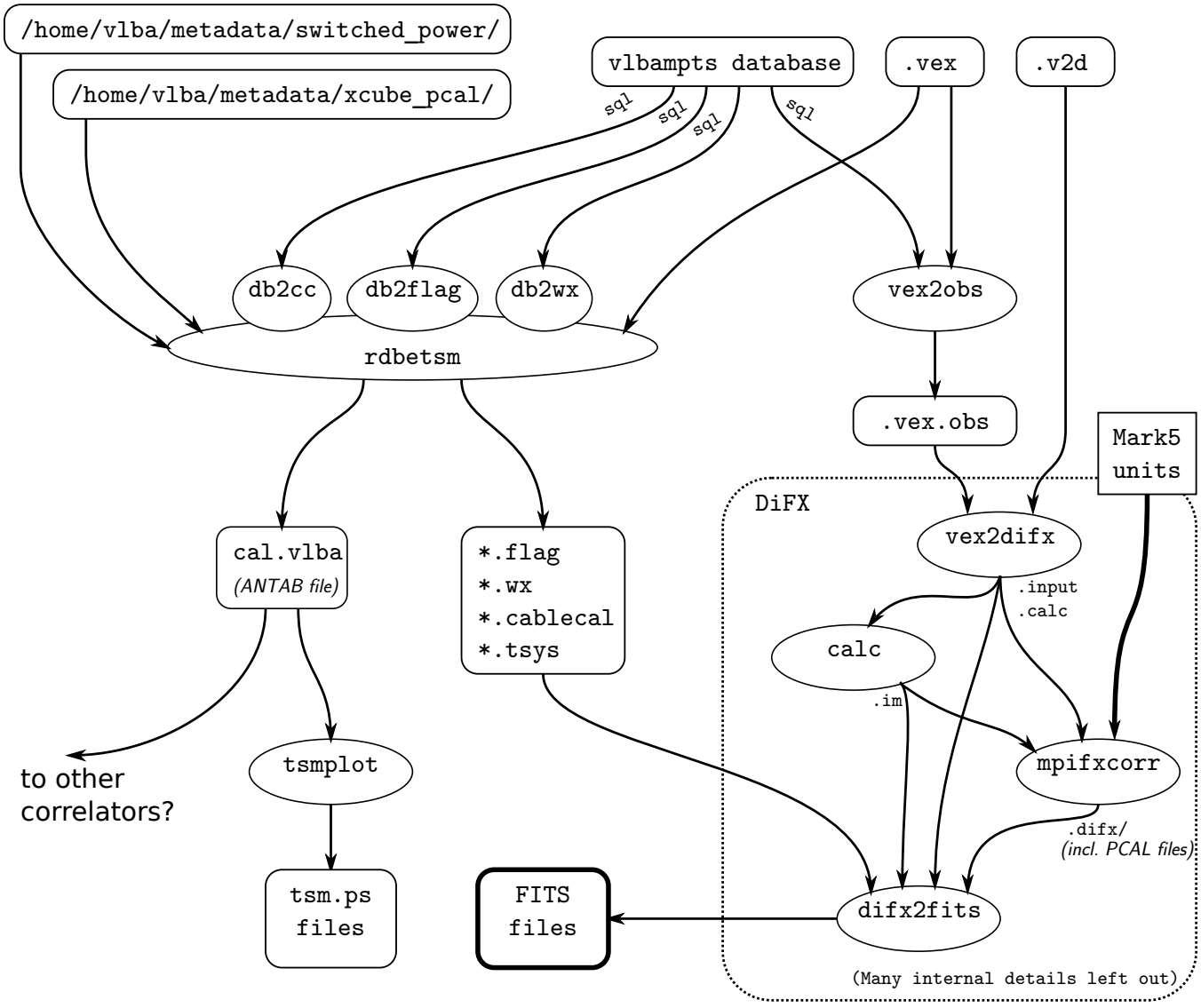

Figure 2: *The post-observing portion of the VLBA calibration and data validity system, including DiFX.* In this diagram arrows indicate direction of data flow. See Fig. 1 for details on nomenclature.