JIVE UniBoard Correlator Memo 14: The Delay Model

Des Small

June 30, 2014

1 Introduction

This document outlines the calculation and use of the current proposed delay and phase-delay models supplied to the UniBoard by the JIVE UniBoard Correlator Control System (JUCCS).

In correlating a VLBI experiment, a delay is applied to the signal from each station in order to virtually relocate the signal to the centre of the earth. The delay incorporates a geographical component arising from the position of the station on the earth's surface at the time of the observation and also atmospheric delays due to the ionosphere.

In the case of the Jive UniBoard Correlator (JUC) these delays are calculated using the CALC program[1]; CALC is also used with the SFXC correlator and was used in the Mark IV correlator system.

As well as applying delays in the time domain we also have to correct the phase of samples since the delay is applied at baseband frequency rather than sky frequency. Since the UniBoard does not have floating point arithmetic, and in fact does not even know which frequencies it is correlating, the control system generates coefficients for phase correction polynomials for each band as well as the coefficients for a time delay polynomial.

On the UniBoard the delay is calculated in units of clock ticks (where one clock tick is the time between samples); the integer and fractional parts of the delays are treated separately, as described below and more fully in the correlator design document[2].

Integer delay When the delay value changes by ± 1 tick between successive FFT segment, a sample in the latter FFT segment is skipped or repeated.

Phase correction Each sample is also rotated to compensate for phase delay (because the sampling is done at baseband frequency rather than sky frequency). When the integer delay value shifts by ± 1, the phase is rotated by $\pm 90°$.

Fractional delay A form of fractional bit shift is implemented by rotating each FFT segment at the output of the polyphase filter bank.

2 Summary of recommendations

The current UniBoard personality will operate with 8, 16 or 32 MHz subbands[2]. This implies a Nyqvist sample frequency, f_N, of 16, 32 or 64 MHz. The sample time t_s is $\frac{1}{f_N}$.

[1] http://gemini.gsfc.nasa.gov/solve/

The sample time, or *tick*, is the unit used for delay calculations.

The current recommendations for delay calculations are:

- Linear polynomials over an integration (an interval of up to one second);
- Units of ticks, i.e., $\frac{1}{32} \times 10^{-6}$ s;
- 48-bit fixed-point coefficients for delay;
- 8 bits after binary point for delay;
- 40 bits after binary point for delay-rate;
- May be necessary to use quadratic polynomials for RadioAstron correlations, or for picosecond delay resolution

The recommendations for phase calculations are:

- Quadratic polynomials over an integration (an interval of up to one second);
- 64-bit coefficients in units of cycles;
- All 64 bits after a notional binary point;
- Units of cycles; i.e., 1 cycle corresponds to 2π radians.

2.1 A note on the current implementation

The current implementation in the UniBoard firmware does not match what is described in this section. It was decided to leave changes to the model until other parts of the UniBoard implementation of correlation had been thoroughly debugged.

The current – transitional – implementation is described in Section 4 below.

2.2 From CALC to Erlang

Delay model calculations are done using the CALC program. Delays are calculated at a one-second interval. Parameters (scan times, station and source positions) are taken from the JIVE experiment database, which is filled from VEX files.

The version of CALC used in the JUCCS uses a slightly different output format from that used by SFXC, but that is the only change, and it only affects the top-level C wrapper, not the FORTRAN core of the program.

The calculation of the CALC delays is intended to be done offline as part of the preparation for correlation; the results are stored in a directory structure for the JUCCS that is modelled on that used in the existing Mk IV correlator control system.

When a correlation is started, an Erlang model-reader component reads the CALC output file and computes the coefficients of interpolating polynomials for delay and phase for a one second interval. The delay polynomial is linear, so it needs two delay points; the phase polynomials are quadratic, so they use three consecutive delay points, and a separate polynomial is calculated for each subband.

In the case that the integration time is less than a second, the polynomials sent to the UniBoard are derived from the one-second polynomials by selecting the appropriate "parent" polynomial and adjusting the coefficients such that they represent the same polynomial over the integration time, effectively shifting the origin. In the rationales given in Section 3 we consider only the most demanding case for accuracy, namely the case where the integration time is a full second.

2.3 Subband frequencies

Frequencies are ultimately taken from the VEX file from the experiment (via the experiment database). The `$FREQ` block stores the sky frequency which maps to 0 Hz in the BBC output for upper and lower subbands and the bandwidth of the channel. We use the centre of the subband as the reference frequency to calculate phase correction coefficients.

For each subband, the delays computed by CALC are multiplied by the subband's reference frequency (using double-precision floating-point arithmetic) to calculate phase delays (for each second), which are used to calculate the coefficients of an interpolating polynomial.

2.4 Discretization and packetization

All polynomials are initially calculated using floating point arithmetic and then rescaled to the appropriate units before discretization: ticks, in the case of delay. The phase polynomial coefficients are simply reduced to their fractional part, since they are calculated in cycles.

The only subtlety in discretization is rescaling with binary points. The delay rate coefficient is to be stored in a 48-bit register with 20 bits after the (notional) binary point. This simply means that we multiply the floating point value by 2^{20} and convert the result to an integer. The delay coefficient has 40 bits after the binary point (of which the first twenty bit are known to be zero) so it is multiplied by 2^{40} before conversion to an integer. Note that the position of the fixed point is not stored anywhere in the data: it is simply a convention that the UniBoard and the control software share.

Since the coefficients for phase polynomials only store fractional values in 64-bit registers, we multiple the values by 2^{64} and convert them to integers. The conversion to integers is done by Erlang's built-in rounding function which converts to the nearest integer, as opposed to truncating.

The delay coefficients for each station are collected into a packet and written to the correlator; likewise, for each station and subband a packet of phase delay coefficients is written to the correlator. The format of the packets is given in the correlator definition document; it is transparent from the Erlang code and is therefore not reproduced here.

2.5 On the UniBoard

The quantum of data processed by the UniBoard is the FFT segment, typically made up of 2048 samples, although other lengths can be configured in software, for a given station and subband. For each FFT segment the UniBoard calculates an integer delay for the whole segment, using the delay model coefficients provided. The calculation uses the time of the central point of an FFT segment to calculate the delay. The delay is calculated using fixed point arithmetic, and the integer and fractional parts are used separately by the UniBoard.

The integer delay is used to retrieve the data from an appropriate location in the DDR memory, effectively shifting it in time in the appropriate way. The highest eight bits of the fractional part of the delay are used as an index into a look-up table of phase slopes used to rotate the products in the FFT segment.

The choice of eight bits after the binary point for the constant term in the delay polynomial was originally made when we planned to use a 4-bit lookup table for fractional

delay compensation; it makes possible an accuracy of 2^{-8} ticks $= 1.2 \times 10^{-10}$ s in delay calculations, which is safely more than is actually needed in that case.

The nine most significant bits of the phase delay for each subband are input to the appropriate subband mixer at the input to the polyphase filter bank. For 9-bit accuracy to be maintained using the same phase delay polynomial coefficients over a one-second interval we show below that the coefficients must be stored to at least 59 bit accuracy, which it is convenient to round up to 64 bits.

3 Justifications

3.1 Fixed point

Fixed point arithmetic is much cheaper to implent in VHDL than floating point arithmetic. If we *had* the luxury of floating point arithmetic on the UniBoard there would be no need for any decisions about scaling, and we would not need the control system to multiply delays by frequencies to calculate phases: that could all be done on the UniBoard itself.

3.2 A priori estimates of terrestrial delay coefficients

We are going to use quadratic polynomials to approximate station delays relative to the centre of the earth. In order to get bounds for the worst case for coefficients of the polynomials we consider a Taylor's series expansion of the diurnal delay due to rotation of the earth. Again in the interest of studying the worst case we assume a station on the equator. The delay is approximated by the polynomial

$$\text{delay}(t + \Delta t) = c_0 + c_1 \Delta t + c_2 \Delta t^2, \tag{1}$$

with coefficients c_i given by

$$c_0 = \text{delay} \quad = \frac{R}{c} \cos \Omega t \tag{2a}$$

$$c_1 = \frac{d}{dt} \text{delay} \; = -\frac{R}{c} \Omega \sin \Omega t \tag{2b}$$

$$c_2 = \frac{1}{2} \frac{d^2}{dt^2} \text{delay} = -\frac{1}{2} \frac{R}{c} \Omega^2 \cos \Omega t \tag{2c}$$

Plugging in the numbers ($\Omega = 2\pi/(24 \times 60 \times 60) = 7.3 \times 10^{-5}$ radians/s, $R = 6.4 \times 10^6$ m) we get maximum values for the coefficients (in second-based units) of

$$\max(c_0) \approx 0.02 \text{ s} \tag{3a}$$

$$\max(c_1) \approx 1.6 \times 10^{-6} \text{ s/s} \tag{3b}$$

$$\max(c_2) \approx 5.6 \times 10^{-11} \text{ s/s}^2. \tag{3c}$$

Substituting new variables Delay and T, with scaling Delay $= L$delay and $T = Lt$, we get

$$\text{Delay} = L \, \text{delay}(t_0) + \frac{d}{dt} \text{delay}(t_0) \delta T + \frac{1}{2} \frac{1}{L} \frac{d^2}{dt^2} \text{delay}(t_0) \delta T^2 \tag{4}$$

In particular, scaling to the JUC's internal unit of ticks for time gives $L = 3.2 \times 10^7$ ticks/s, giving new coefficients C_i, with values

$$C_0 = \quad L \text{ delay} \approx 6.4 \times 10^5 \text{ ticks} \tag{5a}$$

$$C_1 = \quad \frac{d}{dt} \text{ delay} \approx 1.6 \times 10^{-6} \text{ ticks/tick} \tag{5b}$$

$$C_2 = \frac{1}{2}\frac{1}{L}\frac{d^2}{dt^2} \text{ delay} \approx 1.8 \times 10^{-18} \text{ ticks/tick}^2 \tag{5c}$$

3.3 Coefficient sizes and resolution

3.3.1 Goals

In the absence of floating point arithmetic there is a trade-off between the size of the coefficients for delay and phase polynomials and the accuracy that can be maintained. This section gives the details of this relationship.

It has been suggested that it is desirable to have a resolution of 1 ps or better for delay calculations. The current specification discussed in Section 4 below does not satisfy this requirement, in part because the requirements for the correlator were revised after an internal review in 2013 and the firmware has not yet caught up with the new requirements. But it is also the case that the JUC treatment of fractional delays is implemented using an 8-bit lookup table, and it is thus limited to a resolution of 1/256 samples. At a 32 MHz sample rate this corresponds to a delay accuracy of 120 ps.

3.3.2 Constant delay coefficient

For a sample rate of 32 MHz, a clock tick is 31 ns. To achieve picosecond resolution, we need $-\log_2\left(1 \times 10^{-12} \cdot 32 \times 10^6\right) \approx 15$ bits after the binary point. With the current 8 bits after the binary point we achieve 100 ps resolution, which is at the limit of acceptability for an 8-bit lookup-table for fractional delay correction. (It was originally specified to match a 4-bit fractional delay correction.)

For terrestrial experiments, the maximum delay of 0.02 s corresponds to 19 bits before the binary point at 32 MHz; this means that to also have picosecond resolution we would need at least 34 bit coefficients. To also enable space experiments with a maximum delay of 2 s, we would need an integer part of the coefficient of 19 bits, for a total of 41 bits.

These are readily accommodated in a 48-bit coefficient with 20 bits after the binary point, which we propose to use to replace the transitional 32-bit coefficient.

3.3.3 Linear delay coefficient

To achieve picosecond accuracy over a maximum integration time of T_{max}, which for the JUC is limited to 1 s, we require the error in representing the linear term in the delay coefficient, e_1 to satisfy

$$T_{max} \cdot e_1 = 1\text{ps},$$

so that $e_1 \approx 1^{-12} \approx 2^{-40}$, and we need to keep 40 bits after the binary point. (Note that this is true at any sample rate.)

However, for terrestrial experiments the maximum delay rate is 1.6×10^{-6} s/s which means (independent of sample rate) that the first 19 bits after the binary point will be empty. For space experiments with a maximum delay rate of 50×10^{-6} s/s, this means

that the first 14 bits will be empty. So in practice we only need to transmit 26 bit coefficients, where the coefficient starts 19 bits after the binary point.

3.3.4 Quadratic delay coefficient

For terrestrial experiments, the maximum delay acceleration is 5.6×10^{-11} s/s^2. At a sample-rate of 32 MHz, this is 1.8×10^{-18} tick/tick^2. This corresponds to 58 empty bits after the binary point.

For an resolution of 1/256 samples, to match the fractional delay correction, at an integration time of 1 s $= 32 \times 10^6$ tick we would need the error, e_2, in the delay acceleration term to satisfy

$$e_2 \cdot T_{\text{max}}^2 \approx t_{\text{res}} \tag{6}$$

So that here

$$e_2 \cdot \left(32 \times 10^6\right)^2 = \frac{1}{256},$$

so that $e_2 \approx 2^{-57.9}$. This is less than the number of bits left empty, so the quadratic term could be omitted altogether. This is done in the transitional case.

For picosecond accuracy, however Equation 6 gives $e_2 \approx 2^{-64.8}$ so that we would need 65 bits after the binary point, of which the first 57 will be zero, leaving an effective 8 bit coefficient offset by 57 bits.

3.4 Remarks on delay coefficients

As remarked above, the fractional delay correction is currently limited to a time resolution of 1/256 of a sample, and this is far coarser than a picosecond. However, if it is nonetheless desirable to calculate delay polynomials to picosecond accuracy then for terrestrial observations it would be possible to pack the 20-bit effective delay rate and the 8-bit effective delay acceleration into a single 28-bit space, which is less than the transitional design uses for the linear term alone.

3.5 Phase coefficients

We also wish to assess the implications of quantization error for phase coefficients, using a similar methodology to that of Section 3.3. Clearly, the quadratic term in the phase polynomial will dominate quantization effects – it must accommodate a value of T^2 from zero to $(32 \times 10^6 \text{ ticks})^2 = 1.024 \times 10^{12}$ ticks^2.

It is proposed to add the nine most significant bits of the phase value to the inputs of the polyphase filterbank, so our polynomial must have this level of precision over an interval of one second.

Denoting the quantization error in the second-order coefficient for the phase polynomial by $\text{Err}(c_2)$, it follows that

$$\text{Err}(c_2) T_{\text{max}}^2 \leq 0.5 \times 2^{-9} = 9.8 \times 10^{-4}, \tag{7}$$

so that with $T_{\text{max}} = 32 \times 10^6$ ticks we require

$$\text{Err}(c_2) \leq 9.54 \times 10^{-19} = 0.55 \times 2^{-59}, \tag{8}$$

which is to say that the second-order coefficient needs 60 bits.

It is also appropriate to reconsider the *a priori* estimates of Section 3.2 in the context of phase delays. For this purpose, we use a maximum frequency of 20 GHz. The

constant term is not very useful, since given that we use units of cycles we effectively discard the integer part as a result of periodicity.

$$\max(c_1) \approx 3.6 \times 10^4 \,\text{cycles/s} \approx 0.011 \,\text{cycles/tick} \tag{9a}$$

$$\max(c_2) \approx 1.12 \,\text{cycles/s}^2 \quad \approx 1.1 \times 10^{-15}, \text{cycles/tick}^2. \tag{9b}$$

This shows that there isn't any scope for shifting the phase rate register left. The phase acceleration register, on the other hand, will have zeros in its 24 most-significant bits for terrestrial experiments; it might be possible to exploit this fact to reduce the register size back down to 48-bits by shifting the register left as is done with the first-order coefficient of time delay, but it remains unclear what the phase acceleration might be for space experiments.

4 Transitional implementation

The current UniBoard firmware expects delay and phase-delay coefficients for each frequency, with 32-bit coefficients for delay and 48-bit coefficients for phase.

This reflects the original design, and we have been waiting until the existing correlation firmware is mature and debugged before changing it. The existing control system code calculates delays over a interval corresponding to one second and then copies and adjusts the coefficients to make new coefficients (representing the same polynomial) for each integration. When, as is usually the case, there is not an integer number of integrations in a second, the second in which an integration starts is used to calculate its delay and phase polynomial coefficients.

References

[1] http://gemini.gsfc.nasa.gov/solve/

[2] Jonathan Hargreaves and Harro Verkouter, *EVN Correlator Design* Version 3.3, 11 June, 2014