Submillimeter Array Technical Memorandum

Number: 65
Date: April 16, 1993
From: Colin Masson

Correlator Data Processing

Summary

This memo gives a brief calculation of the buffer sizes, calculation loads and data transfer rates for the SMA correlator, along with a sketch of a possible layout for the DSP code, and some comments on the integration of the overall computer system. 1 Mbyte of RAM per card is sufficient for our needs, and a single TMS320C40 chip has plenty of power to do the calculations. Since there is little load on the crate controller computers, it may be possible to have one cpu handle the whole correlator over a single extended bus.

Introduction

The SMA correlator is proposed to consist of 8 crates, each containing 8 cards and one CPU. Each card will contain 32 correlator chips, and a TMS320C40 DSP. Each chip has 5 12 lags. The SMA signals will be demultiplexed by a factor of 4, so that each chip will provide 128 effective lags after the data have been condensed.

The receiver LOs will be adjusted to track the astronomical fringe phase, leaving a DC phase at the correlator. However, superposed on the LOs will be a phase switch pattern over 4 quadrature phases 0°, 90°, 180°, and 270°, according to Walsh functions. For each period of 10 msec, the Walsh function will be in a fixed phase. At the end of 10 msec, there will be a short blanking period of a few μsec while the correlator counters are unloaded and the LO phases are adjusted. To handle 8 antennas x 2 receivers x 2 patterns (0/180 and 0/90), there will be 32 periods in a full Walsh cycle of 320 msec.

The A/D samplers will operate at fixed times, with no hardware vernier delay adjustment. All delays will be handled by the digital system, with fractional delays corrected in software. There is also a residual phase shift across the IF band which will be corrected in software.

At the end of each 320 msec Walsh cycle, the FFTs must be calculated and the phase and delay adjustments applied. The results of the 320 msec cycles are then accumulated for typically 30 sec before being dumped to disk. There may be requirements for faster readout, but it is acceptable to compress the data in some way (averaging over channels, or selecting channels).

It is planned that all these computations will be performed on the DSP chips, with data being passed off board only after accumulation for 30 seconds. All data processing should be double-

buffered and pipelined so that there are no gaps between the 30 second integrations.

Buffer Sizes

The data should be double buffered at all stages. All numbers following are 32 bit fixed or floating words. I have kept separate copies of lags and spectra in case we need them. The dumps from the correlator chips will include a few extra words for parameters, but I have ignored them in the calculations. The following buffer sizes are required:

Chip Dump

1 per lag. 32 bit fixed point.

32 chips x 512 lags	16k per buffer	32 k (double-buffered)

Walsh cycle accumulators

4 phases required. 32 bit fixed point.

32 chips x 5 12 lags x 4 phases	64k	128k (double-buffered)

Condensed lags

Condense 4 copies of each lag, and the 2 opposing phases, convert to float. This is the place where the extra numbers read from the chips will be discarded, and the lags sorted into contiguous groups.

32 chips x 128 effective lags x 2 phases	8k	16k (double-buffered)

Spectrum Buffers

Quantization correct, sort into USB/LSB, do FFT

2k complex channels x 2 sidebands	8k	16k (double-buffered)

Spectrum accumulators

Accumulate for 30 sec.

2k complex channels x 2 sidebands	8k	16k (double-buffered)

Total	208 k (double-buffered)

A few other small spaces are needed for twiddle factors, etc., and a few k will be needed for the program. A total space of 256 kword = 1 Mbyte per board will give us plenty of room for all--we need to do for the SMA.

Data Rates

DMA dump

The chips will be dumped into accumulation buffers every 10 msec, at a time determined by a hardware sync pulse (BOCF).

16k transfers to DSP per 10 msec	1.6 Mword/sec/board

Add to accumulators

Two reads and 1 write on the DSP memory bus. Program in cache.

48k bus accesses per 10 msec	4.8 Mword/sec/board

FFT, etc

Basic rates are much slower, since these operations are done every 320 msec, and the buffers are smaller. FFT is a significant computational load, however.

VME Bus

Dump all boards every 30 sec.

8k words x 8 boards	2.1 kword/sec/crate

Crate 0 throughput

Pass all data to crate 0, format and send to disk. Total rate is a good fraction of an Ethernet bandwidth, but maybe not prohibitive.

8 crates x 2.1 kword/sec	*17* kword/sec 550 kbit/sec

Computational Loads

These are crude estimates. Real effect will depend on use of program cache and on-chip fast RAM. The on-chip RAM should probably be reserved for the FFT.

Add into Walsh buffers

Every 10 msec the DSP will add the correlator outputs to one of 4 buffers. The choice of buffer depends on the net phase of the Walsh functions for that baseline, which will be communicated to the DSP in advance by the crate controller.

512 x 32 fixed point additions per board per 10 msec — 1.6 Mega adds/sec/board

Condense buffers and float (SMA responsibility from here on)

At the end of a complete Walsh cycle (320 msec), the buffers will be aggregated, and the 4 different sets of data corresponding to each lag will be summed. The correlator offset will automatically cancel at this stage. There are 4 (buffers) x 512 x 32 numbers to add, resulting in 2 (quadratures) x 128 (effective lags) x 32 output buffers.

4 x 512 x 32 fixed point additions per board per 320 msec — 0.2 Mega adds/sec/board

2 x 128 x 32 floats — 8 kilo floats/sec/board

Quantization correction

For each chunk, the transfer function must be calculated, and a fifth-order polynomial applied to each point, after normalizing it. Ignoring the overhead of calculating the coefficients, we have about 5 multiplies and 5 adds per datum, giving

2 x 128 x 32 x 10 floating operations per card per 320 msec — 0.26 Mega Flops/board

Windowing

We may want to weight down the longer lags to improve our spectral sidelobes. Assuming that the windowing function has been precomputed, this is only 1 multiply per datum, giving

2 x 128 x 32 x 1 floating operations per card per 320 msec — 0.03 Mega Flops/board

Fourier Transform

The worst case is if all the lags on each baseline are in a single long transform. There are two baselines per card, giving a maximum length of 128 x 16 = 2048 for the transform. There are 4 of these transforms to do, one for each baseline, and one for each of the two sidebands separated out by the phase switching. (1 of the 2 buffers is USB+LSB, and the other is USB-LSB). The FFT takes some number of operations $xN \log_2 N$. I estimate that 4 ops are required, based on the numerical recipes book, but this could easily be wrong by a factor of two, and we need to look at the exact DSP chip we use. The TMS320C40 will do a 1024 point complex transform, which should be equivalent to our 2048 point real, in 2.26 msec, if the on chip RAM is used. Our total

time for the FFT would therefore be about 10 msec every 320 msec, since our longest transforms fit exactly into the 2k words of on-chip RAM.

2 (baselines) x 2 (sidebands) x 2048 x 11 x 4 per 320 msec 1.13 Mega Flops

Sideband separation

We just add and subtract the 2 buffers, giving 128 x 32 real or 64 x 32 complex numbers per sideband per card.

2 x 128 x 32 x 2 per 320 msec 0.05 Mega Flops

Fractional bit shift, and residual phase rotation

Each complex visibility must be multiplied by a complex number. If these have been precomputed elsewhere, the cost is 6 operations per complex visibility

2 x 64 x 32 x 6 per 320 msec 0.08 Mega Flops

30 seconds

The values are accumulated for a period of typically 30 seconds before passing to mass storage. We may want to dump faster for special experiments, but 30 sec is adequate for Nyquist sampling of fringes at the first zero of the antenna beam at the longest baseline, and is the shortest we need to support fully. We may want to permit faster sampling only for a limited part of the data, or after averaging of channels. I have ignored arithmetic done at this time period, for passband calibration, etc.

Example DSP software scheme

Interrupt driven processes

Int 0 ***10 msec BOCF***

Clock tick++
Start DMA
Return

Int 1 ***IO msec DMA complete***

Sum into Walsh buffer
If Not 320 msec -> Return
Else -> Last Buffer
Start reentrant tail of SMA code
Wait for quantization parms
Process data
Return

Int 2 ***Random :VME Bus***

Put/Get data to/from memory
Return

Reset ***From slot 0, or hardware***

Chip reset
Jump to init routine

Bootup ***From slot 0, or hardware***

"Keyboard" level process

Wait for VME input

Act on it:
Set static parameters
Send monitor info
Set control flags

Set Walsh tables
Prepare for integration
Stop integration
etc...

Return to wait loop

Notes:

1) Putting SMA code as a reeentrant tail of the 10ms interrupt is good for synchronization and time sharing of the DSP, but means that the VME needs an interrupt mechanism to get the quantization parameters to the right place in the code at the right time.

2) Setting up the board will only be done while data taking is not going on.

3) On chip RAM is a scarce resource, and we need to decide how to partition it between the data accumulation step and the FFT.

4) The DSP code is simple enough that we do not need an OS. The total load on the DSP is probably about 20-30%, and limited by memory accesses.