Harvard-Smithsonian
CENTER for ASTROPHYSICS
60 Garden Street, Cambridge, Massachusetts 02138

To: Submillimeter Telescope Memo Series October 3, 1989

From: Mark Reid

Re: Computing Requirements for the Submillimeter Array

MEMO #7

ABSTRACT

The computing requirements for the Submillimeter Array will probably be driven by off-line computer systems involved in taking observed visibilities and generating digital images. Off-line computing will be dominated by the number of spectral (and polarization) channels and by the number of pixels needed to map a source. The computing load is **not** significantly affected by the number of antennas in the array. Thus, the computational load of the Submillimeter Array will probably be comparable to that of the proposed NRAO millimeter array. I recommend a computational power equivalent to about 30 Convex C1 XP's or about 200 Mflops (million floating point operations per second). Today, systems such as a few fully configured (4 processors) Convex C2 computers with large amounts of memory ($\sim$ 500 Mbytes) and disk space ($\sim$ 50 Gbytes) could provide this power.

The on-line computer requirements could be handled by about 5 mini-computers in the class of μVax III's or Sun 4's, but with an interrupt structure conducive to on-line control of instruments. Such systems could not only control the array electronics, but also crudely calibrate data, transform to the image plane, and display images of data just collected in "real time".

I. INTRODUCTION

In this memo, I will attempt to estimate the computing load needed for the Submillimeter Array. I will break the discussion into two sections dealing with the "on-line" computers needed to control the telescopes, receivers, correlators, etc., and the "off-line" computers needed to analyze the data. All numerical calculations will be based on the array parameters given in Table 1.

Table 1. Assumptions for the Submillimeter Array

Description	Parameter	Value
Number of antennas	N_{ant}	6
Antenna diameter	D_{iam}	6 m
Maximum baseline	B_{ase}	600 m
# spectral channels per pol	C_{hans}	1000
# polarizations correlated	P_{ols}	4

II. OFF-LINE SYSTEM

For a conventional cross-correlation radio interferometer (e.g., the VLA) the major steps in the analysis of the raw visibility data are (1) calibration, (2) image construction, (3) deconvolution, and (4) self-calibration. In addition, we anticipate that the Submillimeter Array will for some fraction of the observations analyze ("mosaic") multiple pointings on a source whose angular extent exceeds that of the primary beam of a single antenna.

1. Calibration

For the standard parameters given in Table 1., a total observing time, T and an integration time per record, Δt, the number of computer operations needed to calibrate interferometer data, $\mathcal{N}_{cal}$, can be approximated by the relation

$$\mathcal{N}_{cal}=C_{al}\ N_{visrec}\ C_{hans}\ P_{ols} \qquad (1),$$

where C_{al} is the number of floating point operations needed to calibrate one complex visibility, and N_{visrec} is the number of visibility records per spectral/polarization channel:

$$N_{visrec} = \frac{N_{ant}(N_{ant}-1)}{2}\frac{T}{\Delta t} \qquad (2).$$

2. Image Construction

The major tasks in making an image from calibrated visibilities are to sort the visibilities, convolve them onto a regular grid, and Fourier transform the spatial frequencies (i.e., u, v) to sky coordinates (i.e., α, δ). The sorting/griding step requires $\mathcal{N}_{grid}$ operations and is given by the relation

$$\mathcal{N}_{grid}=\mathcal{G}_{rid}\ N_{visrec}\ C_{hans}\ P_{ols} \qquad (3),$$

where $\mathcal{G}_{rid}$ is the number of floating point operations needed to sort and grid one complex visibility.

The grided $(u - v)$ data requires a complex Fourier transform to obtain the image. Using an FFT algorithm to make "$C_{hans}P_{ols}$"-images requires $\mathcal{N}_{FFT}$ floating point operations given approximately by the relation

$$\mathcal{N}_{FFT} = C_{hans}\ P_{ols}\ 2\ L\ (8\ L\ log_2 L) \qquad (4),$$

where L is the number of pixels per side of an image. The term "$8Llog_2L$" is the number of floating point operations needed to do an L-point *complex* FFT; and "$2L$" such FFT's are needed to generate the image. (The factor of "8" in the term "$8Llog_2L$" was determined empirically by methods described in Appendix I.)

3. Deconvolution

Images made from radio interferometers by Fourier transforming the visibility data can be "improved" by deconvolving the interferometer response ("dirty beam") from the raw ("dirty") map. The most widely used algorithm is the implementation of CLEAN by B. Clark, which is used in the AIPS package. The major computational load of this algorithm is Fourier transforming between the (u-v) and image plane (and not point-source subtractions from the dirty map); hence this deconvolution algorithm effectively multiplies the imaging step by the number of transform steps, $\mathcal{D}_{econv}$.

There are other deconvolution techniques, such as the maximum entropy method (MEM), that have different computation loads. For the MEM case the computation load is also heavily dependent on iterative mapping and while it may have a different characteristic $\mathcal{D}_{econv}$ value this should not greatly affect our estimate of the computational load.

Also, mosaicing from observations with different pointing positions in a large source may affect the computational load associated with deconvolution. Mosaicing produces the best results when deconvolution of all pointing positions is done simultaneously. To first order, mosaicing results in increased image sizes, and this is the dominant factor in increasing the computational load.

4. Self-calibration

Self-calibration will be important for the Submillimeter Array. Any source that has sufficiently strong emission will likely undergo some form of self-calibration. Self-calibration leads to iterative imaging and recalibration of the visibility data. Hence the computing load from calibration, imaging, and deconvolution is increased by a multiplicative factor, $\mathcal{S}_{elfcal}$, equal the the number of self-calibration iterations. However, assuming that self-calibration is only needed for one channel in any data base (e.g. a continuum channel or a strong line channel), the computational load is increased approximately by replacing "$C_{hans}P_{ols}$" with "$\mathcal{S}_{elfcal} + C_{hans}P_{ols}$". However, in almost all cases important for the estimation of the computational load of the array, $\mathcal{S}_{elfcal} << C_{hans}P_{ols}$.

Table 2. Example Observational Parameters

Description	Parameter	Value
Calib operations per vis rec	$\mathcal{C}_{al}$	10
Sort/grid oper's per vis rec	$\mathcal{G}_{rid}$	20
# deconvolutions iterations	$\mathcal{D}_{econv}$	25[a]
# self-cal iterations	$\mathcal{S}_{elfcal}$	4
Image size (primary beam only)	L	512[b]
# pointing positions	$P_{ointings}$	25[c]
Observation time	T	20,000 sec
Data accumulation time	Δt	10[d] sec

Notes:
[a] The number of major clean cycles for CLEAN or the number of iterations in an MEM deconvolution.
[b] At 4 pixels per synthesized beam, $L \approx 4\frac{B_{ase}}{D_{iam}}$, rounded up to a power of 2.
[c] With 25 pointings at half-beam spacings, a mosaiced map would be three primary beams across.
[d] See discussion associate with equation (9) in section III.

The total computational load, $\mathcal{N}_{total}$, can now be characterized from equations (1) through (4) and the deconvolution factor, $\mathcal{D}_{econv}$:

$$\mathcal{N}_{total} = \mathcal{N}_{cal} + \mathcal{N}_{grid} + \mathcal{D}_{econv}\mathcal{N}_{FFT} \qquad (5).$$

Equation (5) can be expanded with appropriate use the self-calibration factor, $\mathcal{S}_{elfcal}$, as follows:

$$\mathcal{N}_{total} = (\mathcal{S}_{elfcal} + C_{hans}P_{ols})\{(\mathcal{C}_{al} + \mathcal{G}_{rid})N_{visrec} + 16\ \mathcal{D}_{econv}\ L^2\ log_2 L\} \qquad (6).$$

The two terms within the braces ({ }) in equation (6) can be of vastly differing magnitudes for different types of observations. The first term is proportional to the number of visibilities for each spectral/polarization channel (N_{visrec}), and the second term is approximately proportional to the number pixels (L^2) needed to map the desired portion of sky. For example, for the observational parameters given in Table 2, the "visibility" term is $\approx 10^6$ operations, whereas the "imaging" term is $\approx 10^9$ operations. (It is important to note, however, that vector and/or parallel processing computers will not achieve the same throughput for all applications. For example, a Convex C1 XP can accomplish about an order of magnitude more floating point operations in a given time doing the mostly "vectorized" FFT's needed for the imaging term than doing the "scalar" operations involved, for example, in sorting the visibilities.)

For rare observations involving long integrations and very small fields to be mapped (e.g., point source detections), the first term in equation (6) can dominate the computational budget. However, for the small number of telescopes anticipated for the array, the second term will dominate. **Thus, the (off-line) computational load of the Submillimeter Array is only weakly dependent on the number of antennas and will probably be comparable to that of the NRAO millimeter array.**

Since the large field mapping case can yield far more extreme computational loads for the array, we can simplify equation (6) by dropping the dependence on the number of visibilities in comparison to the number of pixels and dropping the number of selfcal iterations in comparison to the number of spectral and polarization channels:

$$\mathcal{N}_{total} \approx 16\, C_{hans}\, P_{ols}\, \mathcal{D}_{econv}\, L^2\, log_2 L \qquad (7).$$

For the parameters given in tables 1 and 2, $\mathcal{N}_{total} \approx 4 \times 10^{12}$ floating point operations to map the primary beam of a 6-meter diameter antenna with 600-m baselines. To carry out the off-line processing in a time equal to a reasonable observing time of 20,000 sec ($\approx$ 6 hr) requires an effective computational power of about 190 Mflops. If a mosaiced field of 3 by 3 primary beams is synthesized, about 840 Mflops are needed. (Note that this mosaiced example can be squeezed in a 1024 by 1024 pixel image and, hence, does not require a full order of magnitude more compute power than the single beam image.)

By comparison, for these applications a Convex C1 XP yields approximately 6 Mflops when running the AIPS mapping task MX assuming that the FFT's dominate the computing load. The effective Mflops of the Convex C1 XP for mapping applications was calibrated by assuming the theoretical numbers of floating point operations given by equation (7) and by timing MX on a data base of 18,000 visibility records for each of 63 spectral channels. See Appendix II for details of this timing study. By way of comparison, a test on a *fully* vectorized problem indicates that the C1 XP also achieves 6 Mflops, hence MX makes nearly optimal use of the C1 XP

Clearly the 4000 channel, 600 meter baseline, mosaiced example calculated above is nearly a "worst case" for the Submillimeter Array. One could reasonably conclude that this would be a rare case (or even not allowed). More typically the total number of channels ($C_{hans}P_{ols}$) needed to be mapped might be about 500 and baselines (B_{ase}) might be less than 200 m. This more "typical" case would yield image cubes of 500 by 128 by 128 and would result in a significant decrease in computational load; it would require 1.5 Mflops(0.25 Convex C1 XP's) for single beam imaging or 24 Mflops (4 C1 XP's) for the mosaiced example. Only the single beam (non-mosaiced) case could be handled with the computers presently in the division.

Table 3 summarizes the examples just discussed, including the disk (or other mass storage) space required to hold the images. The computation (CPU) speeds are those required to keep up with 6 hour observations by processing them in an equivalent amount of CPU time. The disk storage requirements are for the final image-cube only; clearly about 2 orders of magnitude more disk space will be required for the full system!

Table 3. Examples of Computation Load

Total Channels $C_{hans}P_{ols}$	Image Size L	CPU Speed (Mflops)	Disk Storage[e] (Gbytes)
4000	1024[a]	840	16
"	512[b]	190	4
500	512[c]	24	0.5
"	128[d]	1.5	0.03

Notes:
[a] Appropriate for $B_{ase} = 600$ meter baselines, $D_{iam} = 6$ meter antennas, and $P_{ointings} = 25$ to mosaic 3 by 3 primary beams.
[b] Appropriate for $B_{ase} = 600$ meter baselines, $D_{iam} = 6$ meter antennas, and no mosaicing.
[c] Appropriate for $B_{ase} = 200$ meter baselines, $D_{iam} = 6$ meter antennas, and $P_{ointings} = 25$ to mosaic 3 by 3 primary beams.
[d] Appropriate for $B_{ase} = 200$ meter baselines, $D_{iam} = 6$ meter antennas, and no mosaicing. [e] Disk space to hold the final image cube only.

In addition to the considerations mentioned above, this analysis has not allowed for a "reprocessing factor" which experience has shown is probably at least a factor of 3, nor has it allowed for increasingly complex imaging algorithms which will almost certainly be used in radio interferometric imaging in the next 5 to 10 years. **Thus one should plan on a factor of about 10 larger compute power than that estimated in the theoretical manner given above when specifying a system.** Thus, the first case cited in Table 3 may be beyond the system that could be purchased for the array, unless major breakthroughs in parallel processing and mass storage are made in the next few years. The other cases may not be unreasonable, provided that any 4000 channel, 600 meter baseline cases are rare.

Therefore, I recommend a computational specification equivalent to about 30 Convex C1 XP's ($\sim$ 200 Mflops). This system could allow for limited processing of the 4000 channel, 600 meter baseline, **non-mosaiced** example given in line one of Table 3. It would probably comfortably handle the 500 channel, 200 meter baseline, examples in Table 3, even allowing for a factor of 3 each for reprocessing of data and for future increased algorithm complexity.

It is important to remember that observing wavelength does not significantly enter into the above discussion, and hence it is likely that the computational load will be driven by lower frequency observations that will have higher sensitivity (and hence dynamic range possibilities) and will be possible for large amounts of the observing time. This will tend to increase the number of computationally intensive observations. Also, the CPU speeds calculated above are those required to keep up with 6 hour observations by processing them in an equivalent amount of CPU time. Sorter observations would lead to more maps and require more computer power.

III. ON-LINE SYSTEM

The on-line computer system will be required to control the individual antennas, front-end equipment, and correlators. In addition, it would be highly desirable to allow for some form of on-line image display. Such image display would allow more efficient use of the system than conventional interferometers (e.g., the VLA) for two reasons. First, an on-line image display provides a useful method of assessing data quality, including the opportunity to catch setup errors (e.g., wrong source coordinates). Second, it allows one to tailor the integration time by taking into account the source characteristics and the quality of the observing conditions.

The computational requirements for the on-line system will be directly coupled to the maximum correlator output rate, R_{cor}.

$$R_{cor} = C_{hans} P_{ols} \frac{N_{ant}(N_{ant}-1)}{2} \frac{1}{\Delta t} \qquad (8).$$

The integration time per visibility record, Δt, must be short enough so that phase changes for emission far from the phase (pointing) center are small. Adopting an angular offset equal to the primary beam FWHM, leads to the approximate condition that

$$\Delta t < \frac{D_{iam}}{B_{ase}} \frac{1}{\Omega_{\oplus}} \quad (9),$$

where $\Omega_{\oplus}(= \frac{2\pi}{86400} \text{ sec}^{-1})$ is the angular rotation rate of the Earth.

For the parameters given in table 1, we find that equation (9) yields $\Delta t < 140$ sec. The degree to which this inequality must be satisfied depends on the dynamic range required in the image. Having a strong source far from the pointing center will limit the dynamic range in an image because of inaccuracies in phase from the effects just discussed (as well as amplitude errors from imperfect knowledge of the antenna primary beam and pointing). We will adopt a factor of 14 shorter Δt to satisfy the inequality in equation (9), recognizing that further study will be needed to understand what factors will ultimately limit the image dynamic range. Thus, integration times as small as about 10 seconds may be required for the on-line system. While the standard correlator dump time would probably be 10 seconds or longer, the system should allow for much more rapid correlator dumps for rapidly varying phenomena, such as solar flares also needs consideration, or to allow frequent self-calibration on strong sources.

The total correlator dump rate will therefore be about 6000 visibilities per second; at 8 bytes per complex visibility this requires about 50 kbytes per second. For a 20,000 second (6 hr) observation the data would occupy 1 Gbyte. This sort of data flow and storage is within the capabilities of today's minicomputers. Thus the basic antenna and correlator operation could be accomplished by a small number of μVax III's or Sun 4's.

The display of on-line data, including real-time calibration and imaging, probably could be accomplished by fast minicomputers. Ignoring mosaicing for the on-line display, the problem is to (crudely) calibrate a small subset (e.g., a few spectral channels) of the 6000 visibilities per second, and Fourier transform to the image plane for accumulation at about 1 minute intervals. In addition, some simple deconvolution steps (e.g., CLEAN) would be necessary. Achieving these steps for the 512 by 512 pixel images specified in table 2 at minute intervals is within the reach of minicomputers.

The total on-line computer requirements can, therefore, be met with about 5 fast mini-computers with reasonably large mass-storage systems. The choice between a number of mini-computers and one more powerful system is not obvious. On the one hand, one should avoid as much as possible the somewhat fragile networks linking multiple computer systems, especially in a real-time environment. On the other hand, if tasks can be conveniently split among different computers, there are benefits from multi-computer systems, including less costly options for having spare systems to minimize down time due to the failure of one system.

APPENDIX I

An L-point Fourier transform can be performed using of order "$L\ log_2 L$" operations via an FFT algorithm. I performed timing tests on a μVax II and a Convex C1 XP and find that the FFT codes, such as the one used in AIPS, do *complex* FFT's in approximately "$8L\ log_2 L$" operations. The results of the timing tests are given in the table below:

MicroVax II Timings

Program	Operation	CPU time
Cmplx Vec Mult	1024-pt	0.029 sec
FOURG	512-pt complex FFT	0.17 sec
QXFOUR	512-pt complex FFT	0.19 sec

Convex C1 XP Timings

Program	Operation	CPU time
Cmplx Vec Mult	1024-pt	0.00098 sec
FOURG	512-pt complex FFT	0.025 sec (code not vectorized)
QXFOUR	512-pt complex FFT	0.0058 sec

The complex vector multiply test performed 4 multiplications and 2 additions (6 floating point operations) per vector element. Therefore the μVax II performed these operations at a rate of 0.21 Mflops. The Convex performed these operations at a rate of 6.3 Mflops, or about 30 times faster than the μVax II. This is as expected for a vector application.

Assuming that the μVax II does FFT's (using FOURG for example) at the same 0.21 Mflopsas complex vector multiplies, the 0.17 seconds required to do the 512-point complex FFT implies that it is doing about "$8 \times 512 \times log_2 512$" operations. Assuming this number of operations for the FFT, the AIPS program QXFOUR on the Convex runs at 6 Mflops. (Note that FOURG has been optimized for scalar machines and the code does not vectorize well.) Since QXFOUR is vectorized code this result is consistent with the Convex'es performance on complex vector multiplies.

APPENDIX II

The following CPU times were obtained on a data set consisting of 17850 visibility records for each of 63 spectral channels. A Convex C1 XP computer with 32 Mbytes of memory was used. The MX task created 512 by 512 pixel maps with 1000 clean components (using FACTOR=0).

AIPS TIMINGS

Task		CPU time
UVLOD (entire data base)		70 sec
UVSRT (entire data base)	(TB—XY)	172 sec
MX (each channel)	Get data	10 sec
	make beam	14 sec
	map and clean	170 sec

The 17850 visibility records would correspond to a 3.6 hour observation with a 6-antenna array that integrates data for 10 seconds. For a 4000 channel, 512 by 512 pixel field, the MX task would require about 200 hours of CPU time. Assuming that a 6 hour observation (30000 visibility records) would have only slightly larger CPU time requirements, making these maps would require the power of $\frac{200}{6} \approx 30$ Convex C1 XP's.