VLA COMPUTER MEMORANDUM NO. 141

VLA Post-Processing: Phase I Continued

Eric W. Greisen

September 21, 1977

I. Introduction

In a memorandum dated March 25, 1977, Burns and Greisen discussed the progress of, and defined the longer range plans for, Phase I in the development of a VLA post-processing capability. In the present memorandum I will bring the progress report up to date, discuss the equipment requirements of a Phase I VLA map analysis center, and outline a preliminary software plan for this center. The latter portions of this memorandum represent solely my current view of the project and are not intended to imply that there is general agreement on the points presented.

II. New Accomplishments

In the March memorandum we described a program package designed to reduce data produced by the synchronous subsystem of the VLA. During the past 6 months we have cleaned up this package and added a few routines. In particular, the calibration programs VLACORR, VLACAL, and VLAPCAL have been refined and tested and now appear to work in an acceptable fashion. We have added the program VLAMEM which produces maps using a maximum entropy procedure and the subroutine ANT-TABLE which prints visibilities in antenna-oriented tables. A restartable map cleaning program has nearly been completed. Programs to convert the Green Bank interferometer format to the format used by this package

and to convert this package's format to the VLA Export Format have been developed. The documentation, in the form of a user's manual, is complete for all programs in the package except VLACORR, VLACAL, VLAPCORR, and VLAMEM.

During the past six months, we have developed for the IBM 360/65 a package of PL/1 programs to process data in the VLA Export Format. This format, produced by the Dec-10, is generally used for data which have been fully calibrated and, at least, partially edited. The package is designed to appear to the user to be similar to the Green Bank interferometer and VLA synchronous system data reduction packages. User control is via Job Control Language (for data sets), PARMS cards (for option selection), and INCLUDE/EXCLUDE cards (for data subset selection). The last of these provides a particularly powerful and versatile tool for editing the data at all stages of the processing. I will now briefly describe the various parts of the package. The program DECINDEX is used to provide a summary of the data set while DECCOPY is used to copy data sets or to carry out "manual" editing via the INCLUDE/EXCLUDE cards. DECPOS adjusts phases for changes in phase reference position. The observations of each source may be averaged to selected integration times using DECAVE and sorted on a wide variety of parameters using DECSORT. The program DECSCRIB provides a diverse choice of printer displays. Time-ordered data may be listed in compact baseline-oriented or antenna-oriented forms or plotted. Data in user-selected order may also be plotted or tabulated in several ways. The user may chose to have the correlator data converted to Stokes, or relative Stokes, parameter forms before printing in any of the displays. There are two u-v plane model fitting programs: DECPTSRC

for unconfused point sources and DMODFIT for models consisting of sums of elliptical gaussians and for the polarizations of such models. The program DECMODEL allows these models to be added to, or to replace, the observed visibility functions in the data set. Maps may be produced by standard FFT processes using DECMAP or by a maximum entropy process using DECMEM. The map processing programs VMCLEAN, VMCOPY, VMINDEX, VLAPRINT, VLADRAW, and VLAPIX, which were described in the March memorandum, may be applied to the output of DECMAP and DECMEM. A format conversion program, DECSCANS, may be used to convert the source-sorted export format back to a fully time-ordered form while retaining the general export format.

The current status of the export-format package is really quite good. The programs have been tested and all appear to work as advertised. The user documentation for these programs is complete and may be found in the "User's Guide for VLA Data Reduction in Charlottesville", NRAO User's Manual Series Number 29 by Eric W. Greisen and Frederic R. Schwab. This manual is maintained on Pandora text editor files in Charlottesville and copies are available upon request. The final testing of this package will have to wait until we receive a correct and complete export tape from the VLA site. After the basic tape-writing problems are solved, there will probably remain two further site-oriented problems which are annoying, but not catastrophic. To wit, the export format calls for antenna station and position information which are apparently unavailable in the current Dec-10 data base and for time to an accuracy of $\pm$ 0.66 seconds which appears to be ~8 times better than is currently provided.

III. Map Processing

The intention of Phase I is to develop a prototype map processing system. In such a prototype environment, one may be obliged to "make do" with limited hardware and software in various aspects of the problem. The prototyping goal will be met so long as the consequences of these limitations may be understood fully. However, if the limitations are severe, then their consequences cannot be known and the prototyping effort will fail. I will belabor this point some by citing some obvious examples.

(1) At present, there is no employee with a primary responsibility to maintain NRAO's computer equipment. We obtain the services of Bill Wireman for very limited periods of time and only on the sufferance of the VLBI development project. If equipment is allowed to break and remain broken for indefinite periods of time, as is currently the case, then the prototyping effort will be delayed well beyond the time when we will have to determine the characteristics of the "final" post-processing system.

(2) The final system is intended to be interactive - at least for most of its capabilities. In order to get scientists to use the prototype system, it must respond to their commands with reasonable rapidity. If inadequate core means that the making of a hard copy photograph via the Dicomed (which requires 3-15 minutes) precludes all other operations, then the system will not be fast enough to get people to evaluate its capabilities. Similarly, a lack of

an adequate number of disks will lead to severe head conflict between simultaneous tasks and yield an erroneous impression of the intrinsic capabilities of the system.

(3) It is clear to all concerned that an array processor would have a considerable impact on the computational speed of map analysis. Such processors should make possible operations which are intolerably slow when done by traditional computer methods. To prototype a post-processing computer without an array processor would be a mistake.

The prototype system is expected to do both computational and display processes in independent and mixed forms. It is very difficult to specify how the program should be designed because the Modcomp offers a very wide choice of task structures and because we really do not know what, if any, additional equipment will become available. Perhaps the best way to get a handle on the problem is to specify what display devices ought to be present in the system and how they might be used. (The order given below is not intended to imply a priority ordering.)

(1) Refresh display (e.g., television display with associated microprocessors): The device should have at least a 512x512 display area, pseudo-coloring microprocessor, track ball, and, preferably, other microprocessors to perform standard map manipulations such as zoom, rotation, translation, and

the like. The device should also have an independent 1-bit overlay for annotation, contours, et al. This device is very fast and will normally be used to follow the progress of an operation as well as to provide a clear display of the results of operations. Because of the speed, the data to be displayed should be in core, if possible. This forces the task which controls the display to be a large task (e.g., > 16K words). Because of the speed, it should be possible to suspend other tasks while the driving task is working. However, because the task will be used very often it should reside in core.

(2) Memory display (e.g., TEK 4010, 4014 type CRTs): A 4010 would be acceptable initially, but the larger screen and addressing space and the more flexible display format of the 4014 make it the more desirable display device. The ability to obtain hard copies from the TEK screen is essential. The TEK would be used for line drawing displays (e.g., contours, profile plots, histograms, etc.) and limited numerical output. Because the TEK is slower than the refresh device it will not be used as a continuous monitor and, hence, the data to be displayed may be found on disk. Furthermore, the user is likely to stick with a particular kind of TEK display for a fairly extended period.

Thus the task which drives the TEK may be a small task (< 16K) with overlays. The TEK will be used enough, however, that its current overlay should reside in core if possible.

(3) Photo recorder (e.g., Dicomed): The task which drives the Dicomed should provide all of the capabilities currently present in the NRAO Image Recording System plus the ability to convert standard maps to the fully annotated and interpolated forms used for photographs. The Dicomed is a very slow device requiring 3 (black and white) to 15 (color) minutes to record a picture. For this reason, the driving task, after acquiring sufficient information from the user, must allow other tasks to execute while it is executing.

(4) Text CRT (e.g., TEK 4023): The text CRT should have full page buffering to allow menu-prompting forms of user communication. The device would also be used for numerical and textual output. The device needs to be available to all tasks, but care must be taken to prevent both I/O conflicts and confusing sequences of messages.

(5) Printer/plotter (e.g., Versatec): The Versatec suffers from the disadvantage that the paper will darken rapidly when it sits immobile near the print bar. Thus, the paper must be forward spaced after each collection of output. Therefore, output should be held on disk and spooled out one page at a time. The Versatec is a point rather than a

line plotter. Thus plots must be built up as bit arrays on disk before they can be output. This requirement is consistant with the spooling of the printed data mentioned above. The Versatec-driving task can probably be a small task and, depending on the usage of the Versatec, it may not have to reside in core continuously.

(6) IBM/Modcomp interface: The nature of this interface is currently unknown and is controlled by what the 360 system will allow. If the two computers are to work closely together, passing information regularly, then the controlling task must be core resident and ready to respond to transmissions from the 360. On the other hand, if the Modcomp is used principally to enter jobs in the 360 which execute in the regular 360 queues and which place their output on the 360 output devices, then the interface may be handled by an overlayed subroutine in the main computing task.

The task structure, if we have enough core, has now become clearer. The Modcomp IV/25 allows 3 large and 4 small tasks to be present and executing simultaneously. These could be devoted as follows:

Map 0 (large):	operating system	(OS)
1 (large):	user communication, computation	(UC)
2 (large):	refresh display	(TV)
4 (small):	TEK 4014	(TEK)
5 (small):	Versatec	(V)
6 (small):	Dicomed	(PIX)
7 (small):	? (we won't have the core anyway)	

The intrinsic core requirement for this structure is roughly twice the core which we presently have. Thus, considerable care and heavy use of overlays and assembly language will be required even in the initial programming attempts. The core situation will be assisted, at least temporarily, by the absence of a refresh display and the Versatec. To simplify program modification and transmission, I will write the UC task principally in Fortran. For many reasons, including core, speed, system flexibility, and the uniqueness of the device data structures, I will probably do the device driving tasks in assembly language.

The UC task will be the main program which the user will see. It will be a "BATCH" task replacing the standard batch program provided by Modcomp. The root overlay, which will be attentive whenever the system is restarted and whenever the requested operation has finished, will be a user to computer communication program. This program will allow the user to select his data set, define the desired operation, and choose parameters. It should be some form of menu prompting using a structure of standard menus plus special summary and explanatory menus available upon request (e.g., "HELP!"). The program defaults should reflect the user's previous choices and the user should be able to save and retrieve parameter sets built up during the current and previous runs on the system. (These properties have proved useful in the VLA Asynchronous subsystem.) When execution is to begin, this program will be overlayed by "computational" programs (e.g., CLEAN, disk/tape transfer routines, and many more). The main problem which I forsee in this program is the need to allow the user to interrupt the operation at an arbitrary point, alter the parameters, and resume execution. This need may force us to make

the user communication overlay a separate task from the computation task. Since we already forsee severe core problems, this is not a desirable solution and the matter requires further study. So far as I can tell, there is not room in this system for a "deep background", batch-oriented computational task. Such jobs will need to be sent to the 360 via the Modcomp/IBM interface.

The program outlined here is certainly naive since I lack experience with real-time systems. However, I believe that it is a structure which can grow as I gain experience and which is sufficiently modular in concept to allow the various pieces to change without destroying the whole. In particular, I will first code the programs to run without time sharing. When enough routines are known to work, I can then experiment with a gradually increasing range of simultaneous operations. The user communication overlay will initially be rather primitive. When page handling becomes available for the 4023, I can try more elaborate and, hopefully, better schemes. When an array processor comes on line, it may be added to the computational task overlays in a gradual and systematic fashion.

A detailed time sequence for program development is hard to provide since I have so much learning and experimentation to do. Because of this need to learn and because the interactive portion of this project has received little support, other than moral, it is not reasonable, at present, to give time scales and dates for the completion of various stages of the phototyping effort. In fact, because a lot of the hardware is currently malfunctioning, I cannot even predict with any accuracy when I will be able to begin the first step listed below. The initial sequence of steps is probably:

1. Gain experience with test programs

2. Preliminary system generation
3. Design basic global common definition and transfer formats
4. Tape to disk overlay
5. TEK task limited to contours
6. Sub map extraction overlay
7. Clean overlay
8. Upgrade user communication overlay
9. Basic Dicomed task
10. Attempt initial simultaneous operation (e.g., Dicomed/contour)

IV. Summary

If we are serious in our desire to maximize the scientific output of the VLA, we must expend considerable effort to develop data examination and processing facilities. A major portion of these facilities may consist of a minicomputer/graphics system attached to a large computer. To test this idea and to acquire the experience needed to design a "final" VLA post-processing center, we need to develop a prototype system such as that proposed in the March memorandum. Considering the VLA construction schedule, we are at least two years behind in starting this prototype system.

To carry out a reasonable prototyping operation and to attempt to recover some of this lost time, substantial support from management, the scientific staff, and the technical divisions is required. The proposed project will require an investment in hardware including, at least, a graphics display processor, an array processor, additional core and disc, and lesser items such as a better TEK terminal and heat print device. The project cannot

progress, yet say succeed, without enough hardware assistance to maintain the devices in fully operable condition and to construct any needed special devices such as interfaces and external interrupt connections. Scientific support is needed in the form of user testing of the programs followed by detailed constructive advice. This means that scientists will have to spend substantial time using the system in order to assist the project rather than only when the system will contribute to their current scientific experiments. Mathematical and algorithmic advice from the scientific and programming staffs will be very useful. Direct software assistance will also be needed particularly for the IBM/Modcomp interface and for I/O and interrupt handlers.

This is a significant project. If the support outlined above is provided in an abundant fashion without continuous bickering, then I should be able to handle the remainder of the task either by myself or, if necessary, with one additional programmer. If the support is not freely supplied, then we should admit that we cannot handle VLA post-processing and we should encourage outside groups to take on the problem.

APPENDIX

Since acquisition of additional hardware items is essential to the success of the prototyping effort, it will be useful to list the approximate costs of the items. I do not wish to imply any commitment to the particular manufacturers or models listed below. However, I feel that the devices listed are representative of the types of devices which we should obtain. The prices given are probably current, but they are also probably incomplete (i.e., delivery, installation, and minor "options" needed to make the selected options work may have been overlooked). Furthermore, because the technology of graphics and array processors is changing rapidly, it is hard to predict the nature and costs of the devices which will be available by the time we are actually shopping for them.

1. Additional core (Modcomp IV/25):

first 32K words	$17,500
memory extender	8,500
additional ($\leq$ 4) 32K words (1-4X)	17,500
	$96,000 maximum

(We usually receive a 10-13% discount from these list prices.)

2. Disc (through Modcomp):

The need for disc may well be infinite. In a practical sense, we need at least 1 more 12 M-word disc to help avoid head conflict. These list at $14,000. Large 3330-type discs are available for Modcomps at costs of

single with controller	$32,000
dual-access with controller	46,000
single, no controller	25,000
dual, no controller	33,000

Such discs hold about 42M words.

3. Refresh graphics (based on Information Systems Incorporated VDI Series):

640x512 8-bit processor	$20,000	
color generator	5,250	
memory (320K bytes)	25,560	
memory coupler	1,500	
character generator	3,000	
graphic overlay generator	2,250	
cursor/target generator	1,250	
option expansion	500	
color monitor	3,000	
monochrome monitor (high resolution)	2,000	
trackball	2,600	
enclosures	2,600	
interface	1,250	?
delivery	1,380	
	$72,140	

4. Array processor (based on Floating Point Systems Incorporated):

processor (AP-120B)	$29,500	
fast data memory (32K words?)	42,800	
table memory (4 x 512 words)	2,580	
program source memory (7 x 256 words)	11,165	
parity option (for 32K words)	4,525	
interface	4,700	
	$95,270	
less 12% GSA discount	11,432	
	$83,838	
plus spare parts and installation	5,000	?
net	$88,838	

5. Memory screen (based on Tektronix):

TEK 4014-1	$10,995
minibus extender	110
enhanced graphics	750
joystick	65
4631 hard copy unit	3,995
	$15,915

6. Bulk memory (Modcomp Memory +):

Instead of the core and disc listed above we might consider bulk memory. Modcomp's bulk memory acts like a disc but with a latency time $\leq$ 8.5 micro seconds. Such speeds allow one to overlay code very extensively and allow the maps, while we are acting on them, to be kept in bulk memory rather than the more expensive computer core.

controller		$ 5,000
first 128K words		20,000
additional 128K words	(0-7x)	10,000
		$95,000 for 1 M words