NATIONAL RADIO ASTRONOMY OBSERVATORY

SOCORRO, NEW MEXICO

November 10, 1975

VLA COMPUTER MEMO #127

SPECTRAL LINE MAP MAKING BY DIGITAL COMPUTERS

A DESIGN STUDY

B. Clark

I - INTRODUCTION

In the process of spectral line system design, I have made an independent study of the spectral line sort-merge problem, based on disks competing with the IBM 3350. Specifically, W. R. Burns suggested the consideration of the Calcomp 1035/235-4 disk, which stores 400 M bytes on a nonremovable spindle.

I have not attempted to optimize any procedure in any way nor have I searched for any sophisticated algorithms. I merely took reasonable approaches to every problem until I found one which worked without having a great deal of unused capacity. For this reason, this is not the usual conservative computer design study. The reader will note safety factors of the order of 30% instead of the 100% usual in real-time computer design. The system, bought and programmed as described here, would very probably not work. However, it is about right, and a more careful look should pick up about as much from optimization as is lost to items I have omitted.

I have evaluated three approaches:

1) the classical disk-based sort,
2) storing the spectrum at an address on disk so that it lies in the appropriate (u,v,w) cell, adding the spectrum to any previous one at the same (u,v,w) address (an approach suggested by L. D'Addario),
3) forming the (u,v,w) solid, in large part, in a large memory area in the DEC-10.

I have presumed we are trying to handle 256 frequency channels, each channel output (in u,v,w space) consisting of two 16 bit numbers. For simplicity, the information passed for every baseline every sample time is assumed to be exactly 1024 bytes. The not-very-useful end channels are dropped and replaced by the values of u,v,w, baseline, time, status information, etc. at an early stage in processing.

For each approach, I have looked into sample times of 10 seconds, 20 seconds, and 40 seconds. The maximum usable numerical fields of view (NFOV's) are about 2048, 1024, and 512 respectively. In the last case, where the chore is not too onerous, one might wish to use a 1024 field of view to better suppress aliased sidelobes of distant sources. One should also be aware that the outer parts of the field of view are corrupted by various instrumental effects due to gridding and aliasing, and, depending on the desired accuracy, the usable NFOV may be significantly less than the computational NFOV, which is all I consider here. If these fields of view are used at two points per beam at 21 cm wavelength, we can calculate the angular field of view in degrees.

NFOV \ Config.	A	B	C	D
2048	.57	1.20 (ant beam)	-	-
1024	.28	.92	-	-
512	.14	.46	1.20 (ant beam)	-
256	.07	.23	.72	1.20 (ant beam)

From this follows the required number of steps in the third dimension ≈ (numerical field of view)x(angular field of view)/4

NFOV \ Config.	A	B	C	D
2048	6	11	-	-
1024	2	5	-	-
512	1	2	3	-
256	1	1	1	2

Further, one may use the Hermitian property to cut the required number of w samples by a factor of about 2; more specifically, to (N+1)/2. As is shown below, handling an appreciable number of w-samples gets very expensive, and it is well to take a closer look at the requirement of 6 samples for the B array and 2048 NFOV. If the mapping is made to the first null of the antenna pattern, we can have, with 2048 NFOV, about 3 points per array beam. If we go back to 2, we are really using only about 1400 NFOV, and the number in the above table drops from 11 to 7, or to 4 samples, using the Hermitian property. This number is the one used in the discussions below. Specifically, I design, for the three sampling intervals, a system for the u,v,w dimensionalities given in the following table.

Sample Interval	Sample Data Rate Bytes/ sec.	uvw Dimensionality	u,v,w Data Rate (12 hr obs) Bytes/sec.	x,y Data Rate (12 hr obs) Bytes/sec.
10^s	36 K	2048 x 2048 x 4	398 K	50 K
20	18	1024 x 1024 x 3	75	13
40	9	512 x 512 x 2	13	3.2

It should be emphasized that, with proper connection of the special purpose hardware to a mass store, the choice of "pipeline" sample time does not preclude utilization of shorter sample times, at the price of not being able to reduce spectral line observations in real time.

At this point, also, the properties of the Calcomp 1035/235-4 disk can be summarized:

1600 cylinders/spindle
19 tracks/cylinder
400 M bytes/spindle
1200 K bytes/sec. peak transfer rate
789 K bytes/sec. long record mean transfer rate
4 ms single track seek
16 ms max latency
30 ms mean access time
55 ms max. access time

Cost $17.5 K per spindle.

Approx. $20 K controller and interface.

The Calcomp interface to the Varian minicomputers is already designed, so these computers have been costed. The compute times do not appear to be a limiting factor, so the high numbered members of the family are not required. On the other hand, memory capacities beyond 32 K words are required, so the V71 may not be used. The V72 is, therefore, used in these estimates. The 1.6 microsec per word maximum disk transfer rate appears to me too high to try to run into 1.2 microsecond core memory, so I have used 660 ns core in the cost estimates.

I have estimated no programming costs.

II - SYSTEM BLOCK DIAGRAMS AND COSTS

A. Ten Second Sampling

In all of the possible ten second sampling systems, much of the cost of the system lies in components occurring after the data is first given to the DEC-10. These components depend only on the map size, and so are in variant to the sort algorithm. The cost variation of the different algorithms is thus small compared to the total cost, but the minimum seems to be attained by the classical disk based sort, and a system based on it is described below.

Doing the sort really classically would mean making up little records with keys of frequency, u,v,w and contents of baseline, time as well as real and imaginary parts. This is a little too classical, as the overhead in storage required is about 300%. The approach I take here is to do a classical sort on records of 256 frequency channels, and, at the end, commutate the frequency channels into individual partitions. It is likely that the optimum procedure is somewhere between the two, to move some of the work of the final commutation pass into one of the other passes. I have not investigated this at all.

The technical details of the processes taking place in the various system processors are given in the technical appendix. The system block diagram is given in Figure 1 and the rough costing in Table I. A more detailed cost breakdown is in the technical appendix. It will suffice to give a brief summary here.

The system is a pipeline processor, which pumps the data through the various stations at a sufficiently high rate that no backlog is accumulated. The pipeline processor is also presumably associated with a mass store which ideally can be accessed from several points in the pipeline, so that, if something breaks down, the data flow may be diverted into a reservoir in the mass store without interfering with observation. When operating in the pipeline mode, the DEC-10 CPU and array processor are heavily subscribed, and other programs will slow down by a large factor. If twelve hours of data on a single source are run into the pipeline, the first map appears on the output display about 4½ hours after the end of the observation. After this, the maps appear about every two minutes.

1) The Sort Station. This device accepts, from the existing Modcomps, 351 records of length 1024 bytes every 10 seconds, and, after receiving about 8000 such records, puts them in u,v,w order and outputs this sorted string (about 8 million bytes) to the next station.

2) The Merge Station. This device accepts the 8 million byte sorted strings and combines them into a final output sorted string through four phases of four input merges.

3) The Commutation Station. This device converts the sorted string of 1.5 million records of 256 channels per record into 256 sorted strings with one channel per string. This would be an appropriate place to archive data.

4) DEC-10/Spectradata System. These existing components grid and perhaps apodize the u,v,w data, do the Fourier transform in the third dimension, and generally control the data flow from the sort system through to the display system.

5) Very Fast FFT. The Spectradata is not sufficiently fast to handle the Fourier transforms needed in this system. An external FFT, of which there are several on the market of sufficient speed (2048 complex vector transformed in 5 ms), is required.

6) Transpose Station. A two dimensional Fourier transform is most conveniently handled by conventional devices by a transform of the row vectors followed by a transpose of the half transformed matrix and a second transform of the row vectors. I suspect that the least costly device for performing this transpose is a large CCD shift-register memory (1.4×10^8 bits). However, a system constructed from existing half megabit cards is more expensive than the alternate system priced here of four minicomputers, each with a disk.

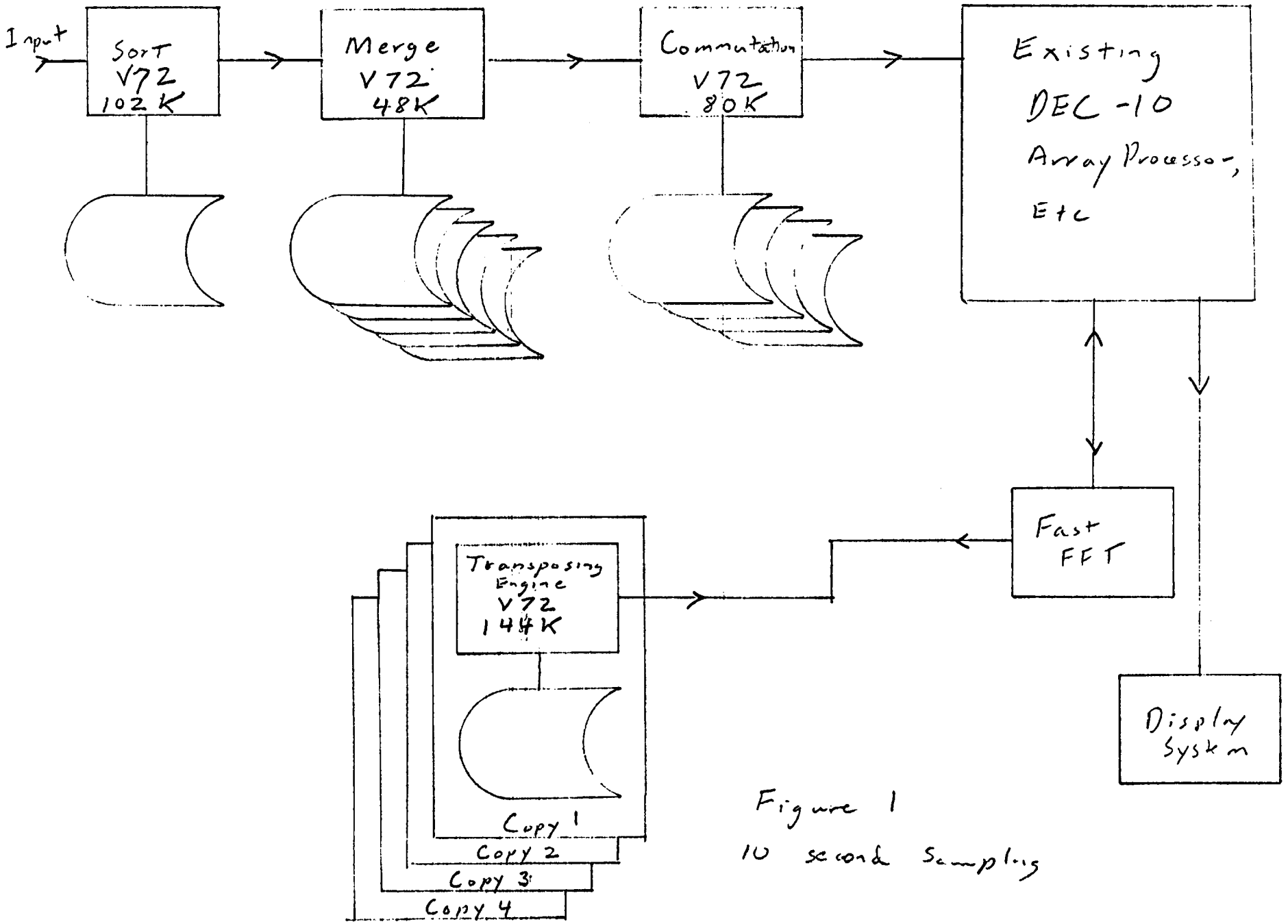

Figure 1

10 second Sampling

Table I
Cost of 10 Second Sampling Classical Sort System

Component	
Sort Station	$ 107 K
Merge Station	147
Commutation Station	144
Transpose Station	488
Fast FFT Station	150
Spares @ 15%	155
System Integration, Documentation, Purchased Software @ 10%	119
Total	$1,310 K

B. Twenty Second Sampling

As stated in the introduction, the map size appropriate for this sampling is 1024 x 1024 x 3. The most significant savings occur in the resulting smaller data handling problem in the DEC-10 and beyond, due to this smaller map size, rather than to the decreased data rate in the sort machine per se.

1) Description of a 20 Second Data System.

The most effective algorithm for the 20 second sort involves some degree of direct sorting into DEC-10 memory. That is, data is input to a minicomputer at the rate of 351 baselines every 20 seconds, and is then run into a radix sort - the sort used by a card sorter. The computer has time to do three passes of a four 'bin' radix sort. At the end of the three passes, there are 64 'bins', each containing, say, the data for all v's and w's and a range of 16 (out of 1024) in u. The records are then commutated (as in the 10 second system) to convert their organization from having a value from a given baseline/time contiguous, to having frequencies grouped together. Then, in the DEC-10, the 48 rows of the u,v,w (16 u's and 3 w's) solid are formed in place. The data can then be apodized by the Spectradata processor.

With the 20 second sampling, the map making process becomes just easy enough to do entirely in the DEC-10/Spectradata system, but both CPU's and the disk channel would be running at saturation, and time sharing would cease. This is intolerable. I think it is necessary to install an aide to both CPU's. This means an external FFT, and a single transposing machine of the sort discussed with 10 second sampling.

The costs of the system are given in Table II, and the block diagram in Figure 2.

Table II
Cost of 20 Second Data System

Sort/Commutate Station	$ 127 K
64 K Words Core for DEC-10	66
Transpose Station	122
Fast FFT Station	150
Spares @ 15%	70
System Integration, Documentation, Purchased Software at 10%	53
Total	$ 588 K

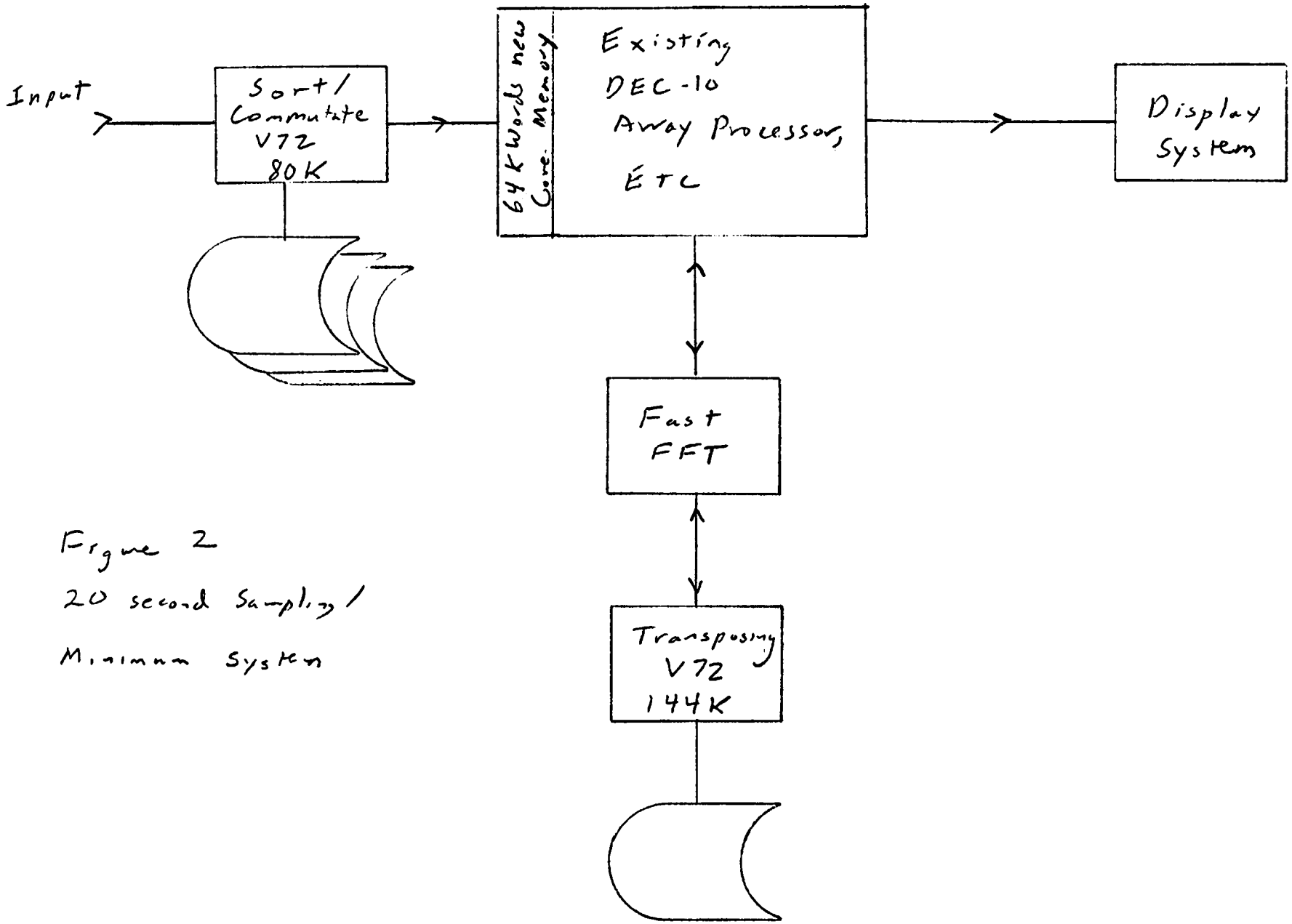

Figure 2

20 second Sampling

Minimum System

2) Use of the System for Processing 10 Second Data.

The system described above can clearly be used to emulate the full ten second system at rates less than real time. The only minor hang up is that the three spindles of disk on the sort/merge/commutate station can hold only about 9 hours of 10 second data; this can presumably be resolved by appropriate use of the mass store. The rates data can be processed are nearly proportional to the number of CPU's involved. Therefore, this system will require about three times real time on the sort/merge/commutate computer and four times real time on the transpose computer to process full 10 second data.

3) An Enhanced 20 Second Data System.

If the four times real time is felt to be too much, an intermediate step is possible, which can process 10 second data in twice real time.

This machine differs from the one above by employing the solid state memory transposing engine rather than the minicomputer. Also, the sort/merge/commutate engine consists of two minicomputers, which may do the entire sort for 20 second data, eliminating the additional core for the DEC-10.

When used in the 10 second emulate mode, during real time the two minicomputers do the sort and merge phases, and output completely sorted but uncommutated data to the mass store. Then, during the second pass (also requiring about real time) the sort/merge engine is split apart, and the computer with three disk spindles is used for commutation, and the other is used in conjunction with the solid state memory for transposing. The solid state memory can be interfaced to provide sufficient throughput to transpose four 1024 x 1024 quadrants of the 2048 x 2048 plane in 30 seconds, and in a similar time the minicomputer can combine them to form the final transpose plane.

The cost of this system is shown in Table III, and the block diagram in Figure 3.

Table III
Cost of Enhanced 20 Second Data System

Sort/Merge Station	$ 107 K
Merge/Commutate Station	127
Solid State Transpose Station	299
Fast FFT	150
Spares @ 15%	102
System Integration, Documentation, Purchased Software @ 10%	79
Total	$ 864 K

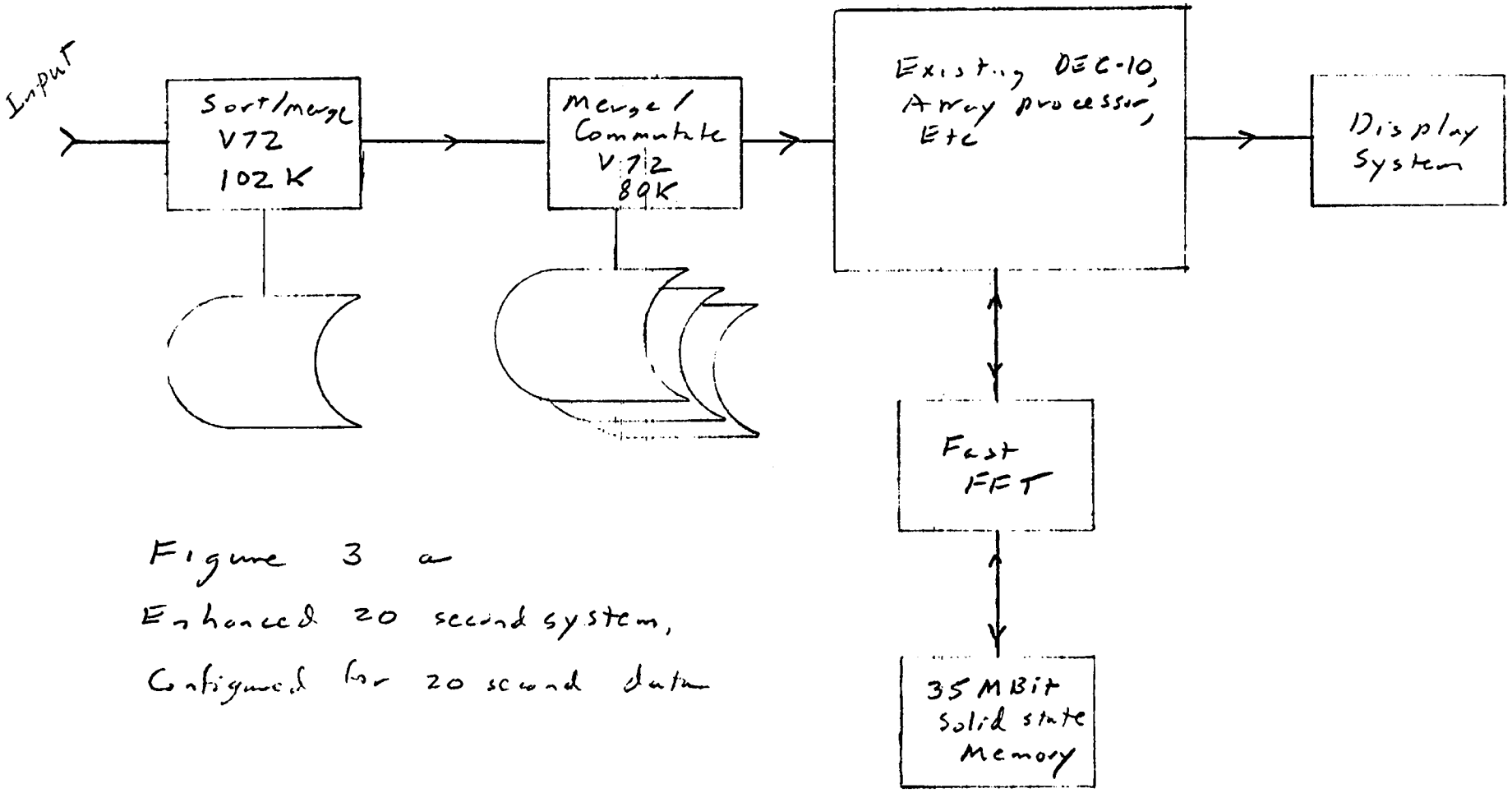

Figure 3 a

Enhanced 20 second system,

Configured for 20 second data

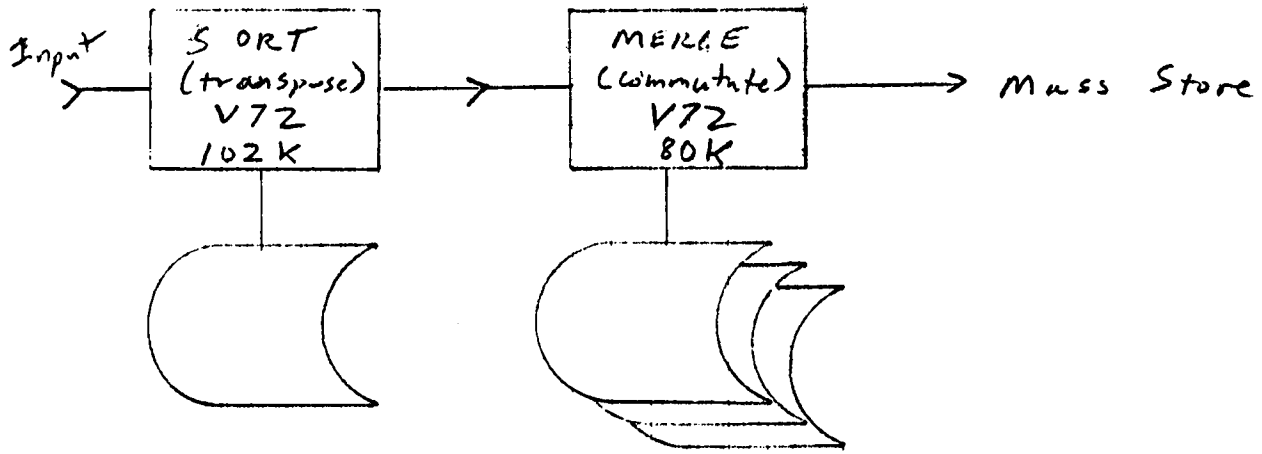

Figure 3b - Enhanced 20 second system during observing of 10 second data.

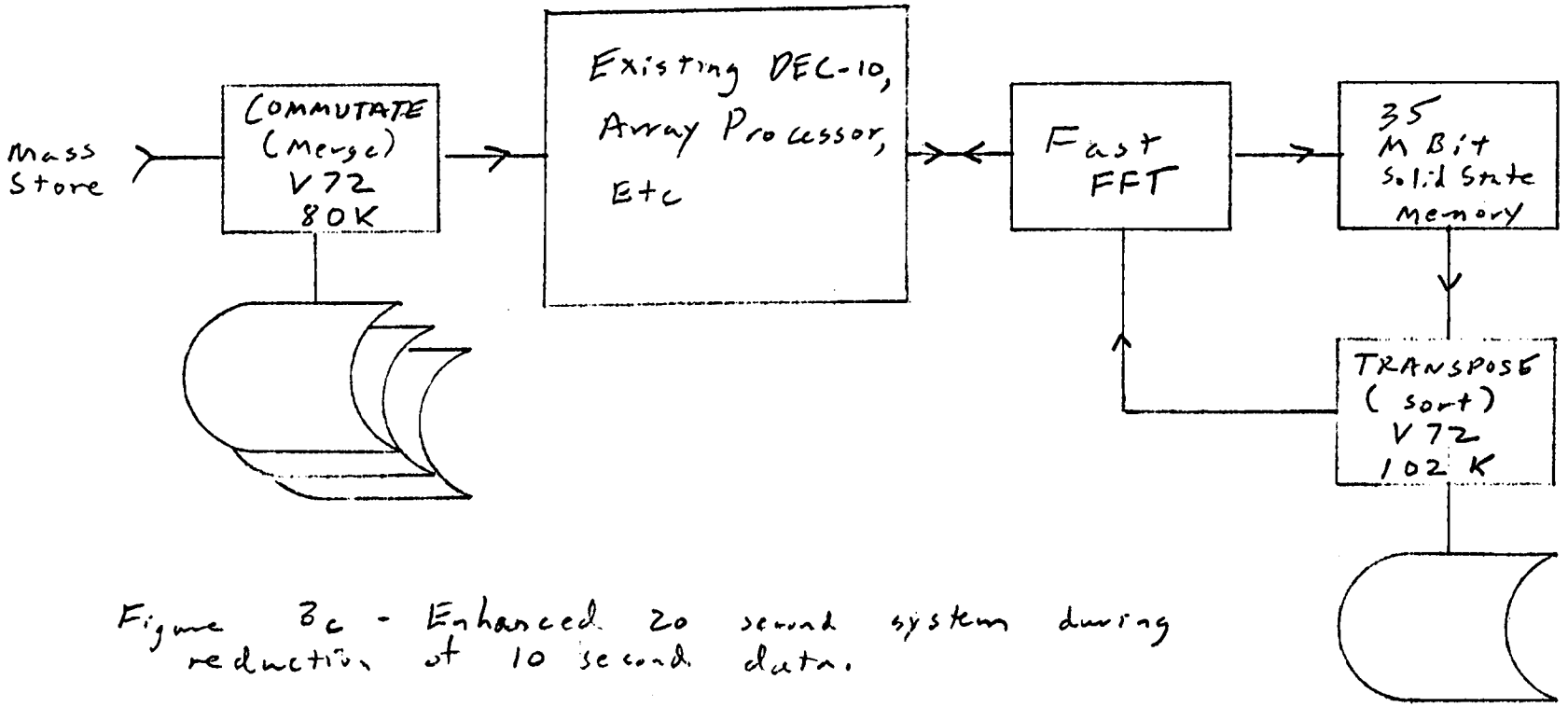

Figure 3c - Enhanced 20 second system during reduction of 10 second data.

C. Forty Second Sampling

With this sampling time we may go to a NFOV of 512 x 512 x 2. This is yet another reduction of a factor of 5 in the amount of map data it is necessary to process. The DEC-10/Spectradata now may process the entire mapping problem alone. A small assist from a sort/merge device is still required at the front end, and this comprises the entire addition to the existing system. The cost is $160 K, including spares.

However, when one of the faster sampling rates is wanted, the impact falls heavily, and most undesirably, on the DEC-10. The time taken by the Spectradata processor to do the FFT's alone is:

512 x 512 x 2	1.7 hrs.
1024 x 1024 x 3	10 hrs.
2048 x 2048 x 4	54 hrs.

There is likely to be a comparable impact on the DEC-10 CPU, even if all sorting and transposing are done by the single minicomputer.

With this system, processing 20 second data will take about three times real time, and 10 second data about seven times real time.

III COMMENTS AND OPINIONS

Although the 40 second sampling system is adequate (barely) to map the full beam of the C configuration at two points per beam, it impresses me that it has much too little margin for all the little variations of processing we might want to try. On the other hand, considering the small range of problems for which it is necessary, the full ten second system impresses me as overkill. I would, therefore, recommend either the twenty second system or enhanced twenty second system.

At the moment, the high price of the solid state memory makes the enhanced system look rather undesirable. However, the price I have estimated for the memory is based on integrated memory boards whose price is seven times the cost of the memory chips they contain. This cannot be a permanent situation. If the cost of the solid-state memory transposing unit falls from $300 K to $200 K, then the additional cost of the enhanced system over the 20 second system, $165 K, begins to look like a worthwhile investment.

The 40 second system is acceptable, I think, only if it is side-by-side with a good, convenient, and flexible optical processor which has output of sufficient quality that the desire to make digital maps at all is very significantly reduced. If a ten- or twenty-second system is constructed, the role of any optical processor would be essentially to provide a remarkably sophisticated display. The additional advantages of providing displays without the pipeline lag and of being able to inspect maps during observation partake of the same nature.

The costs in this memo are, of course, not perfectly accurate. It is my hope that the necessary items I have omitted would be counter-balanced by the items which could be saved by a more careful investigation and optimization of the systems discussed.

IV THINGS THAT OUGHT TO BE LOOKED INTO

Before a final system is designed, there are several areas which ought to be considered further.

A. I have assumed that the 3-dimensional transform is the computationally easiest method of taking account of sky curvature. The mosaic method should be looked at, wherein small NFOV maps are generated at various phase tracking centers. This method involves much more computation, but the transpose problem (the most severe problem for much of the above system design) is much less severe.

B. Several constants are rather uncertain from theoretical discussion. If general agreement is not reached, simulation studies must be undertaken to discover them. Most important of these are the number of points per beam required, the percentage of calculated map which is 'usable', the upper limit for NFOV*integration time, and the number of w-slices for given numerical and angular fields of view.

C. For the various sample times, I have taken the worst case computationally, Configuration C for 40 second data and B otherwise. Clearly, most of the spectral line observations will not be made at these worst-case configurations. The value of having a 'ten second machine' or a 'twenty second machine' must be looked at very carefully in terms of how often we would use the facility, and the slowdown (beyond real time) when we do not provide the full machine.

D. I did not realize until preparing this report for final typing that I have egregiously misconfigured things for the use of solid-state memory in the transposing system. In this case the Hermitian conjugate property should be invoked in the u or v direction rather than in w. For instance, in the ten second use, instead of transposing four 2048 x 2048 matrices, one should transpose eight 2048 x 1024 matrices. The throughput remains the same, but the amount of memory required at any one time is halved. The impact on capacity calculations elsewhere in the system is, I think, minimal.

V TECHNICAL APPENDICES

Those of you without a considerable interest in computer programming may stop reading at this point. You are to be congratulated for getting this far.

A. Cost of Varian 72. The prices used here are taken from the Datapro Report of April, 1975. In my tables, I have lumped together the following basic items as a single line:

72-1201	V72 CPU + 8K of 660 ns parity core	$12.5K
72-3100	3 block transfer channels	4.5
72-3300	Memory Map	2.5
70-8301	2 digital interfaces	1.0
70-6402	Console CRT	3.2
	V 72 System	$23.7K

It is assumed that the 8K of core in these configurations is just sufficient to hold a minimal operating system and program and that buffer space is a separate line in the cost tables.

I assume that communication with all other parts of the system are via the digital interfaces. Communication with the mass store is also probably by such an interface.

Presumably there would be some mechanism for, say, the Boss Modcomp to force a load, and a bootstrap loader for the disk (which, incidentally, I have not priced).

B. Full Ten Second Sampling

1) Classical Disk-based Sort: The system I have considered consists of the three minicomputers whose functions are indicated in Section II A. A more detailed discussion of just what is happening in each machine is given here. A similar discussion for the transpose machines is given in Section V B.4.

a. The Sort Station.

The sort station accepts 351 1024 byte records every 10 seconds and performs a first stage sort on the way to disk. As the data goes by, the sort key (6 bytes) and disk address (2 bytes) are preserved in core. Eight thousand pointers are accumulated (representing 8 million words of data) and sorted by a partitioning sort. The data are accessed from disk and output in sorted, 8 million byte strings (corresponding to 220 seconds of observing time) to the merge station.

i) Input Stage.

This stage requires a 72 K byte buffer divided into eight areas holding nine records (the 9th record area is an overflow area used to hold a new input during writing of the previous eight). The incoming record is directed to one of the eight areas, depending on its key (with equal probability over the ensemble of VLA observations). When the area has eight records, they are written to disk. The disk time required to write the eight records is 71 ms (two accesses are figured, since the heads must be returned to the operation in progress). A 71 ms access time every eight records corresponds to an accumulated time of 3.6 hours for the data taken in twelve hours.

ii) Partitioning Sort Stage.

Eight thousand pointers of eight bytes each stored in a 64 K byte buffer. A partitioning sort of eight thousand items typically requires 90 thousand comparisons and 50 thousand exchanges. A comparison on the V 72 computer typically requires about 200 memory cycles (including the overhead of getting to unit item partitions) and an exchange 25. The partitioned sort will run in about 14 seconds. This corresponds to a total time of 0.7 hours in a twelve hour observation.

iii) String Assembly and Output Stage.

This stage is allocated a 64 K byte buffer, requiring 64 more or less randomly located records to fill it. Because of the input strategy, however, these records will lie in only one-fourth of the whole 8 million word area - on only 12 cylinders of the disk. In a typical case, most of the information will fall on five cylinders of the twelve. Making a 30% allowance for CPU overhead and track switching, about ten records per track can be written, so, in a given record slot and cylinder, there is an everage of 1.6 records to be recovered. The mean worst case among the ten record slots has four records in the slot, so the disk must rotate through four turns to pick up all the data. The dwell time on the other seven cylinders is assumed to average 1½ turns. The total disk time is thus

Initial access		.030^s
Cylinder access	.008	
Dwell 4.6 turns	.077	
Times 5		.425
Initial access to second area		.020
Cylinder access	.008	
Dwell 1.5 turns	.025	
Times 7		.231
		.706^s

The whole 8 million byte record goes in 91 seconds, representing 4.8 hours for a twelve hour observation. The time of the sort station is thus spent as follows for a twelve hour observation:

Input state	3.6 hours
Partitioning sort	0.7 hours
Output stage	4.8 hours
	9.1 hours

Note that, although the data storage requirement is small, a 3330 type disk is still required because the high track density and fast cylinder access are required.

Note also that the total buffer memory requirements for this station are about 200 K bytes.

b. Merge Station.

This station accepts the presorted strings of length 8 million bytes and goes into four phases of four input merges. Each phase consists of reading 16 K byte records from each of four input areas and writing a 64 K byte record on the output area. In order to have a completed output record and four partially used input areas, a total buffer area of 128 K bytes is needed. For each phase, the time economy is as follows:

Access time	.030 s	
16 K byte input time	.021	
	.051	
Times four		.204 s
Access for output		.030
64 K byte output time		.083
		.317 s
for 64 K bytes		
or, for 12 hours of data,		2.02 hours
Total for four merge phases		8.1 hours
Input from sort station, output to commutation station		2.0 hours
		10.1 hours for a 12 hour observation

c. Commutation Station.

When the sort is complete, the disk is read into half of the 128 K byte buffer (one-fourth cylinder), commutated into the other half, so that it is organized into 256 records of 256 bytes each, and rewritten on the same cylinder.

The time budget is as follows for each cylinder:

Initial seek		.006 s
Rotational latency	.016	
Input 64 K bytes	.083	
Commutate	.130	
Rotational latency	.010	
Output 64 K bytes	.083	
	.322 x 4 =	1.288
		1.29 s/cylinder
		0.57 hrs/spindle
		2.28 hrs total

Commutation time is figured on the basis that each two bytes take a load, store, and miscellaneous overhead, each of two 660 ns memory cycles.

The records can be stored so that the four records pertaining to one frequency channel are reasonably adjacent in rotation. There are 1024 bytes per channel on each cylinder, occupying 0.079 of a turn; suitable spacing between records should occupy about .05 turn. An adjacent cylinder seek takes about 0.3 turn. Therefore, if adjacent cylinders are oriented at 180° (i.e., if channel 1 is at 0° on even cylinders, it is at 180° in odd cylinders), a pack can be searched at the rate of 120 cylinders per second (assuming the accepting device is always ready). The total readout time is thus 13-1/3 seconds per pack per channel, about one minute per channel total, four hours to output the entire half day's data base. The input time to the commutation station is about another hour.

The total time required by the commutation station is thus 7½ hours for a twelve hour observation.

d. Cost estimate is given in Table IV below, from which the numbers in Table I are derived.

Table IV
Cost Estimate, Classical Sorting Scheme, 10 Second Data

Sort Station	
V 72 System	$ 24 K
96 K Words Memory	45
Disk Controller	20
1 Spindle	18
	$107 K
Merge Station	
V 72 System	$ 24 K
32 K Words Memory	15
Disk Controller	20
5 Spindles	88
	$147 K
Commutation Station	
V 72 System	$ 24 K
64 K Words Memory	30
Disk Controller	20
4 Spindles	70
	$144 K

2) Store in place in u,v Plane: This sorting system replaces the sort station and merge station by more severe data storage and commutation problems. Holding 256 channels of 2048 x 2048 x 4 u,v,w solids requires 1.7×10^{10} bytes equals 43 spindles. Since the cost of this alone is greater than the classical sort system by a factor of nearly two, it was not considered further. It is possible that it might be more competitive if it were implemented with partial sorting and disk staging onto a mass store, but this line was not pursued.

3) Direct Gridding: In this procedure the commutation stage is very much the same as in the classical sort, but the sort stage is done entirely by indexing in the DEC-10.

In order to minimize cost, only one-eighth of a map need be present in the DEC-10 at one time - this is equivalent to a 32 way radix sort occurring in the commutation device, which does not degrade its performance intolerably, although the safety factor allowed is uncomfortably small (twelve hours data handled in ten hours).

The maximum memory must be installed on the commutation engine in order to minimize the DEC-10 memory required. With buffers approaching 512 K bytes, the commutation described in the classical sort processor can be implemented and the four w-slices may be routed to different disks as well. Then, when data is read sequentially, data is sorted, crudely, into eight v ranges. Only the first is sent to the DEC-10. The other seven are written out to disk again as long records of 64 K bytes. These long records can then be reread and forwarded very easily.

The time budget is, for a 12 hour observation,

Input Data	1.7 hrs
Commutation	2.3
Search and Read	4.0
Write and reread unwanted part of (u,v) plane	1.9
	9.9 hrs

Six disks are specified - four are the commutation area, one is reserved for the rewrite process, and the sixth accumulates the first part of the next observation while the commutation of the previous one is occurring.

The DEC-10 must be able to hold one-eighth of a u,v map - half a million words.

The cost of this approach is given in Table V below. This is to be compared with the cost for the entire sorting system of $398 K (sum of the entries from Table IV).

Table V
Direct Gridding to Core Cost, 10 Second Data

Commutation Station		
V 72 System	$ 24 K	
24 K Memory in CPU Chassis	21	
224 K Words External Memory	105	
Disk Controller	20	
6 Spindles	105	
		$275 K
Half million words, DEC-10 Memory		264
		$519 K

4) Map Making: The map making process consists of the following steps: a) grid, average and taper; b) Fourier transform rows; c) transpose x and v directions of the half transformed map; d) Fourier transform the rows of the transposed matrix (i.e., column transform the matrix); e) access the similar rows of the w slices and do the DFT (not FFT) in the third dimension, and store the resulting map for display and analysis.

a. Grid, Average, and Taper.

This step is done by the DEC-10-Spectradata system. The exact division of labor is not clear.

The complex visibility is read from the sorting system, along with the control channel giving the u,v location. The u coordinate is used to index the location within the row of the matrix where the data is to be added. If uniform weighting (independent of integration time within the cell) is desired, a count of data could also be generated. I presume the Spectradata could then be persuaded to divide the two rows. In any event, the Spectradata can easily superpose a Gaussian taper by multiplying by a constant (representing $\exp - \beta v^2$) and by an invariant vector ($\exp - \beta u^2$).

The order of magnitude times involved are:

Input data, (u,v)	3	μs/word
Index data by u and add	10	
Count data entries	5	
Looping overhead	8	
	26	μs/word

Total 2.7 hrs DEC-10 Time

The effort required to divide by the data count and apodize depends on details of the Spectradata machine which I do not have ready to hand. It is possible that it can do the job. I doubt it. However, it might well lie within the capability of the external fast FFT, or, failing that, not too much effort to implement as a separate external special purpose device.

b. Fourier Transform Rows.

A total of 4096 Fourier transforms of length 2048 must be done for each w-slice, or 4 million altogether. The FFT device must have an operation time (total time = setup time + (length)*$\log_2$ (length)*operation time) of 0.3 microseconds to do this in eight hours, not counting setup time and I/O time.

c. Transpose Rows and Columns.

To accomplish the transpose in real time is a remarkably difficult task. To illustrate, let us look at the throughput. To make 256 maps, each with four w-slices and each slice 2048 x 2048, in, say, eight hours, the throughput rate is 149 thousand values per second, or (with 16 bit word length), 4.8 million bits per second. This is about 0.75 times the sustained writing rate of a 3350 disk, so the data cannot be written to a single disk and recovered in real time.

There are two approaches to implementation of a transposing engine. We could have a large (1.4×10^8 bits) CCD shift register memory, at a cost of nearly a million dollars. Alternatively, we may simply set several mini-based disk systems down side by side, and round-robin schedule to get the desired throughput.

For instance, with four minicomputers, each with a 3330 type disk, we can solve the problem as follows: Using a 256 K byte buffer in each machine, and with each machine receiving a quarter of the data, the buffer will hold 128 rows. After transpose, the buffer is written sequentially to disk. When the data is to be output, the buffer can be refilled with 32 columns of length 2048 by 16 accesses. The time adds up as follows:

Initial input and transpose	$3^s.0$
Disk write	$6^s.6$
Disk read	$14^s.3$
Output	2.0
	26^s
For 4 x 256 maps	$7^h.4$

The cost of a single minicomputer system is given in Table VI below - the cost of the entire transpose station is four times as great, or $488 K.

Table VI
Cost of a Transpose Engine

V 72 System	$ 24 K
128 K Words Memory	60
Disk Controller	20
1 Spindle	18
	$122 K

d. Fourier Transform Columns.

This step was discussed under c.

e. Fourier Transform in the Third Dimension.

With reasonable buffer sizes, the disk access times for the four w-slices can be made decently small and I shall ignore it. The function to be performed is

$$B(x,y) = \mathrm{Re}\{\sum_{n=0}^{3} e^{inz}\, B(x,y;n\Delta w)\}$$

$$\text{where } z = \sqrt{1 - x^2 - y^2}\ \omega\Delta w \simeq \frac{1}{2}(x^2+y^2)\ \omega\Delta w.$$

The transform is done by the array processor a column at a time, multiplying first by the constant $\exp \frac{1}{2}\, i\, x^2\, \omega\Delta w$ and then by the tabulated vector $\exp \frac{1}{2}\, i\, y^2\, \omega\Delta w$. Finally, the four contributions are summed and the real part taken. Presuming that the array processor takes $\dot{3}$ microseconds for a complex multiply, this consumes 24 seconds per w-slice, 96 seconds per map or 6.9 hours total. If, in fact, the Spectra-data processor requires a significantly longer time, which seems likely, this function must also be farmed out to an external device.

C. Twenty Second Sampling

The data rates in the sort section are lower by a factor of two, resulting in, approximately, a factor of two less compute being needed. The data rates in the map making section are a factor of five lower, which makes the more significant saving in cost.

1) Classical Sort Approach: It is not quite possible to combine all three stations of the 10 second system into a single device. A two station pipeline is still required.

To sufficient accuracy, times can be scaled from the discussion of the 10 second system, with the following results:

Initial Sort Stage	4.4 hours
Merge Phases @ 1.1 hours	4.4
First Commutation Phase	1.7
Second Commutation Phase	2.0

A reasonable split of the effort would be to assign the initial sort and first merge to the first station, and three stages of merge and commutation to the second. The second system needs enough storage for 16 hours observation - 12 hours being commutated and 4 hours to store incoming data during commutation. This is three spindles. It is possible but uneconomic in the long room to replace the 3330 type disk on the input station by a 2314 type. The cost is shown in Table VII below:

Table VII
Cost of Classical Sort System for 20 Second Data

Sort/Merge Station	
V 72 System	$ 24 K
96 K Words Memory	45
Disk Controller	20
1 Spindle	18
	$107 K
Merge Commutate Station	
V 72 System	$ 24 K
64 K Words Memory	30
Disk Controller	20
3 Spindles	53
	$127 K

2) Direct Gridding to Disk: The data storage required to support u,v,w solids 1024 x 1024 x 3 amounts to nine spindles. About 80% of this space is empty. With 128 K bytes of buffer commutation (first phase) can be done (at the rate of 0.6 hrs/spindle) in 5½ hours. During this transpose, the data can be compressed, so that the output phase can be done in the same 2 two hours as in the sort case.

Inputting the data will cause about one hour interference with this process, because it will usually be done on separate access arms, and it will only interface with the commutation process during the times when the record to be written falls at the rotational position on disk where the commutation is currently working.

The data rate of the machine in this mode - one 1 K byte record every 57 ms - is sufficiently slow to permit a mean access, read, add, rewrite operation (47 ms), in a straightforward way.

The cost of the system is shown in Table VIII below.

Table VIII
Cost of Direct Gridding to Disk, 20 Second Data

V 72 System	$ 24 K
128 K Words Memory	60
Disk Controller	20
9 Spindles	175
	$279 K

3) Direct Gridding to Core: The commutation phases require a total of 3.7 hours for a twelve hour observation. There remains time in the same engine to do three 4-way radix sorts. This results in 64 strings, each containing 16 rows by three w-slices of the 1024 x 1024 x 3 u,v,w solid. This last sort, into 48 individual rows, requires very little additional effort on the part of the DEC-10, but about 48 K additional memory.

Table IX
Cost for Final Gridding to DEC-10 Core, 20 Second Data

Sort/Commutate Station	
V 72 System	$ 24 K
64 K Memory	30
Disk Controller	20
3 Spindles	53
	$127 K
64 K Memory for DEC-10	$ 66 K

4) Map Making: The map making process becomes just easy enough to do entirely in the DEC-10/Spectradata system, with both CPU's and the disk channel running near saturation.

I think it is sufficiently undesirable to have either processor saturated that I would include both an external FFT and a transposing station. The transposing station has less work to do than in the 10 second case by about a factor of (2048 x 2048 x 4)/(1024 x 1024 x 3) = 5.3; a single stage of the 10 second minicomputer will handle it nicely, for a cost of $122 K (Table VI).

5) Handling 10 Second Data: It is obviously desirable to be able to handle 10 second sampling times at less than real time rate. A small price premium would attach to being able to do this in the most convenient possible way.

a. Direct Grid to Core System.

The limiting time factor in this case is the transposing station. Working with the full 2048 x 2048 x 4 maps and twelve hour observations, it can transpose at only about 1/4 of real time. At this rate, there is no reason why the sort system should not be used to successively emulate the three stations of the ten second pipeline, sending data to the mass store each time instead of to the next station. This minimizes the heavy load on the DEC-10 and results in a total rate of four times real time for the processing.

b. An Enhanced System.

Choosing the more expensive option in a couple of places in the design considerably enhances its capability of handling 10 second sampling.

i) Sort rather than Direct Grid.

If the sort approach (Section IV C.1) is taken, and a fourth spindle is added to the second station, the sort pipeline can emulate in real time the first two stations of the 10 second pipeline. Then, a second pass through the second station may emulate the third station of the 10 second pipeline. Thus, in twice real time (with a spare CPU the second time through) the 10 second data may be sorted and commutated.

ii) Solid-State Memory Transposer.

A 1024 x 1024 map w-slice, with 16 bit values in real and imaginary parts consists of 32 M bits of information. If we provide a solid state memory of this size, we can write a row at a time and read back a column at a time.

I estimate a price for the solid-state transposer in Table X on the basis of a half M bit RAM board (mounting and driving Intel 2416 CCD IC's). This board sells for $3500.

Table X
Solid State 1024 x 1024 Transposer

64 ½ M Bit Memory Boards	$224 K
Controller, Interface	50
Power Supplies, Packaging	10
Engineering	15
	$299 K

If one uses this concept for transposing a 2048 x 2048 matrix, one must record three quarters of the data and transpose a quarter of the data at a time. That is, of each 2048 value row, 512 entries would be inserted into the 1024 column positions in the solid state memory row (the first value, say, going alternately into columns 0 and 1). Then, when the memory is full, it would be read two columns at a time to extract the 2048 values of a single column of the large matrix. The time budget for a single w-slice is about as follows:

Write 3 quarter rows with 64 K bytes/record	22^s
Read back with 192 K bytes/record	16^s
CCD memory dump time	$\sim 6^s$
	44^s

Processing for four w-slices, 256 channels thus requires 12.5 hours, only slightly more than real time.

D. Forty Second Sampling

The data rate here is sufficiently low that the process could be handled entirely within the DEC-10. However, this would have an extremely severe and probably unacceptable impact on the DEC-10 resource availability.

1) Sort by Minicomputer.

Given a minicomputer front end, either of the two complete sort algorithms (classical sort, direct grid to disk) operate with pleasantly large amounts of surplus time. For a 12 hour observation processed by the classical sort:

Initial Sort	2.2 hours
Merge Phases	2.2
First Commutation Phase	0.9
Second Commutation Phase	1.0
	6.3 hours

or processed by direct gridding to disk:

Access, read, add, rewrite, return access arm for each record	5.0 hours
Commutate	1.9 hours
	6.9 hours.

2) Map Making

Making a 512 x 512 x 2 map is not an extremely painful job for the DEC-10, without any minicomputer help whatsoever. The time is about as follows for a 12 hour observation:

Gridding	0.8 hours
Apodize (Spectradata)	1.5
Fourier Transform (Spectradata)	1.6
Transpose (DEC-10	0.4
(Disk Channel	1.0

3) Use of the Equipment for 20 Second Data.

With the use of 48 K core from the DEC-10, the system can keep up with the sorting and gridding chores, or without the extra core, can run slightly slower than real time. The Spectradata will be the limiting item, requiring about one times real time for apodizing and one for Fourier transforms. The impact on the DEC-10 CPU (not counting memory interference from the Spectradata) is not too bad, about 0.4 real time and the interference with the disk channel is also moderate, about 0.6 real time.

4) Use of the Equipment for 10 Second Data.

The Fourier transforms for a complete set of maps amounts to 54 hours, five times real time. There is another time increment, about twice real time, for apodizing and the third dimension transform.

Since the Spectradata operates at one-seventh of real time anyway, it would seem logical to utilize the single minicomputer CPU configuration to simulate, in turn, the seven stations of the 10 second pipeline discussed above. Thus, without excessive load on the DEC-10 (except memory interference with the Spectradata), the 10 second data may be reduced in about seven times real time. The first map does not emerge from the pipeline until after the sort is complete, about three times real time.