Developing New Architectures for the Block II VLBI Correlator System

J. C. Peterson
DSN Data Systems Section

This article focuses on the overall LSI (large-scale integrated circuits) architecture design and current status of the VLBI (very long baseline interferometry) Block II correlator. The VLBI correlator algorithms demand a computing system that provides a throughput of hundreds of millions of instructions per second to perform cross-correlation detection for six baselines. LSI technology lights the way for the computation of complex parallel process and is raising the upper bound of computerization.

I. Introduction

The Block II VLBI (very long baseline interferometry) correlator can best be understood by reference to Fig. 1, which shows a simplified block diagram for the acquisition and processing of VLBI data. The data acquisition facility consists of an antenna pointed at the signal and low-noise microwave amplifiers and receivers for receiving the signal. Analog-to-digital (A-D) converters convert the data to digital form for recording. This recording requires large volumes of data so the data are recorded on a digital instrumentation recorder, the digital recorder assembly (DRA). These tapes are then shipped to the data processing facility, which employs special-purpose hardware for computer-controlled correlation. The correlation algorithms demand a special-purpose computing system that provides a throughput of hundreds of millions of instructions per second.

The input data rate for a 6-station VLBI correlator, as shown in Fig. 1, is 1344 million bits per second. The fast Fourier transform (FFT) algorithm, which is only one of many algorithms in the system, requires 12 million instructions per second to adequately handle data reduction. Systems with such unprecedented throughputs have been named "super-systems."

Advances in LSI (large-scale integrated circuits) technology have significantly reduced the cost and enhanced the reliability of computer components. It is now feasible to design reliable modular computing systems containing more hardware resources than ever before, resulting in a sharp increase in the number of instruction and data streams that may be computed by a parallel system. LSI technology has provided powerful means of improving throughput, such as architectural adaptation and increased parallel computation.

Since existing systems have failed to provide such enormous throughputs, let us look at the potential of adaptable data flow architectures, which are now feasible with the advent of LSI and promise to be the architectures of future supersystems. This article describes a complex parallel system equipped with the architecture for creating this exceptional throughput. The design work is for a 28-channel, 3-station VLBI correlator system with transfer of hardware to operations scheduled for April 1982. The expansion to a 28-channel, 6-station system is possible shortly after this date. The system design must be characterized by allowing an increase in system complexity.

II. Problems of System Complexity

The VLBI correlator requires a system capable of computing 168 (28 channels × 6 stations) parallel streams, each of which may have rates in the range of 1 to 8 million bits per second. A system operating at this speed must also maintain a high reliability since the cost of failure will create a backlog of data and thereby other associated problems.

Research on supersystems has shown that these extreme throughputs have been difficult to obtain by merging existing complex computers into a system (Refs. 1 and 2). Let us consider the question associated with this system merger.

Since including a very large number of computers in a single supersystem leads to enormous complexity, the supersystem is thereby degraded by worsening reliability. It then becomes unsuitable for computing many complex real-time algorithms that require high reliability.

In addition, complex real-time algorithms are characterized by an increase in the number of data streams involved. The supersystem then has to respond by an increase in the number of computers it incorporates. This then leads to an increase in delay or adds complexity by introducing interconnecting logic. This complexity is shown with a VLBI correlator system; a N-station, 28-channel correlator will require $28N(N\text{-}1)/2$ copies of a module for simultaneous, all-baseline cross-correlation. As the data streams increase ($28N$), the complexity of the system involved increases by $(N^2\text{-}N)/2$.

To reduce the number of interconnections required or the delay between connecting modules, a new type of architecture has to be developed to offset the problem. The new architecture has to be capable of performing a dynamic adaptation to the computational requirements of the algorithms being processed. This will allow the system to increase throughput with the same resources, but without changing the system's complexity. In this way, supersystems will become suitable for new algorithms requiring higher throughput and reliability. In adaptable architectures, this purpose is accomplished by microprogrammable, reconfigurable, and dynamic adaptations to algorithms.

Another new type of architecture, data flow architecture, increases the efficiency of computation by the architecture's ability to conform to the computational flow graph of the executing algorithm. Both of these architectural principles require distributing the processing among multiple processors, memory, and input/output (I/O) units connected via one of the interconnection networks.

III. New Architectures

Making new architectures feasible requires the effective distribution of the algorithm among multiple resources. The problem is, given the multiple resources as individual modules, one has to allocate processor and memory resources to them in a way that minimizes time and system complexity.

Real-time algorithms provide a processor architect with clues as to what types of adaptations must be performed. These algorithms are characterized by a variable number of both data and instruction streams. In the VLBI correlator, the system is designed to process variable-number lengths of concurrent data streams. Since each module unit requires its own process, it is characterized by a variable lag length for the different data streams. This lag length, characterized by its own instruction and data streams, is able to adapt so that it may redistribute the available lags into another cross-correlation module unit. For example, a VLBI correlator incorporating 420 8-lag cross-correlators must be capable of a software-controlled reconfiguration into 240 16-lag, 120 32-lag, 60 56-lag, 30 112-lag or 15 224-lag cross-correlators. To meet these reconfigurations, the FFT algorithm also has to match the redistributed cross-correlator and process 240 16-point FFT, 120 32-point FFT, etc. As a result, the VLBI correlator increases its throughput while maintaining the same resource complexity, thus preserving its level of reliability.

Distributed processing enhances system performance by employing many processors to handle the processing load. A representation of the VLBI correlator distributed processing system is shown in Fig. 2. The key elements in this system are a set of modules assigned to the same processor.

Only three unique processors are necessary in this distributed system: station, correlation and FFT processors. These processors are distributed in the system as needed, depending on the number of station streams to be processed. Each station data stream input requires a station processor and sharing of correlation and FFT processors, which can handle up to 3 baselines of data. The network of processors for a 6-station, 28-channel system then assigns 6 station processors and 5 correlation and FFT processors to a VLBI correlator system. By employing these distributed processors in the

VLBI system, the processing load is contained in the correlator hardware and not in the controlling computer. It is expected that the FFT processor, for example, will yield performance superior to that of a general-purpose computer on FFT operations. The FFT algorithm requires 20 million instructions per second for a 6-station processor. The algorithm load is distributed into 5 FFT processors, each requiring 4 million instructions per second. The performance of a Digital Equipment Corporation VAX 11/780 computer, for example, is capable of about one million FFT-type instructions per second. A speedup of 20 times is achieved by going to specialized distributed processors for just this algorithm, which is one of many in the system.

These processors in the network are homogeneous; they are fully connected by control entry points and the effect of precedence relationships among modules can be ignored. This assigning or grouping of modules with high intermodule communication can be used for module allocation, subject to real-time and memory constraints. This clustering of modules to a processor is called fusion. In an algorithm, after finding a group of modules, we check to see whether this fusion satisfies the real-time and memory constraints, the fusion of which will eliminate the greatest amount of interprocessor communication. That module group is then represented by a module cluster for the next iteration. This fusing process continues until all eligible modules are allocated. The goal is to balance the load among the set of processors, yielding high throughput and faster response time.

The 28 modules fused to the station processor (Fig. 2) are tailored to the real-time constraints. Each module is actually a group of subcircuit-modules assigned to one channel of data. The module load is divided up into these different, well-defined subcircuits. They make up the bit and frame synchronizer, buffer delay and calibration tone detector. The main real-time burden in the station hardware falls on these subcircuits. This is true also in the correlator hardware, but here the modules make up the cross-correlator and time domain integration.

The task allocation problem is important in the initial design phase of a distributed processing system, in the normal operational mode, and in the reconfiguration phase of a dynamic reallocation environment.

During the design phase, it is necessary to evaluate competing design configurations including network topology, channel bandwidth, and number of processors. Task allocation allows determination of the value of these parameters to achieve a desired level of performance.

In the normal operational mode of the VLBI correlator, it is important to assign modules to processors to meet critical timing constraints. It should not be assumed that all processors are ready and available at task allocation time. Task allocation is also important during reconfiguration due to changing operating environment. These changes may be a result of changes in task input rate, processing and channel capacity.

Because of the computational complexity of the correlator system, a need has arisen for a simplified approach to the task allocation problem. This is generally a heuristic approach where one balances the loading constraints with real-time constraints. Figure 3 shows the task and communication allocation assigned to the three different processors. The processors in this environment communicate among themselves via an interconnection mechanism. This communication is essential for overall system throughput. All the modules and processors are related by two interrupts: the frame rate interrupt (FRI) and correlator output interrupt (COI). Every 20,000 data stream bits a FRI is generated, and every 100,000 data stream bits a COI is generated. Assigning these two interrupts to all modules and processors speeds up the overall processing time by establishing a mutual relationship among all elements.

IV. Lengthening the Life Cycle with New Architectures

Many complex real-time algorithms may require an increased number of data and instruction streams. In the VLBI correlator, for instance, some algorithms face an increased number of lags and an increased level of each FFT that must be handled in real-time. To be adequate, a supersystem must be able to increase its throughput by integrating new resource units into the existing system. Such an increase in complexity, however, may lead to a deterioration in reliability. Thus, its life cycle for the application may be shortened since the supersystem may become incapable of providing highly reliable computations within a specified time limit.

The architecture adapts to an increase in lags and FFT via dynamic redistribution of the available resources, and it will augment the concurrent computer resources that will be needed with the system hardware. This additional processing power is gained without adding equipment, and the associated increase in complexity, to the supersystem. Consequently, the VLBI correlator system is capable of maintaining the same level of reliability that existed before. Its life cycle thus lengthens since its computational adequacy is sustained in spite of the increased processing requirements encountered in the algorithms. It should be noted here that this dynamic redistribution is not just designed into the cross-correlator modules and FFT processor but into some of the other modules and processors as well. Only in the cross-correlator modules and FFT processor is the life cycle lengthened so remarkably because most of the hardware encountered is in this area of the system. This is shown in the following examples.

The VLBI cross-correlator module is designed for an application in which the data requires a lag length of either 28 channels – 8 lag points, 14 channels – 16 lag points, 7 channels – 32 lag points, 4 channels – 56 lag points, 2 channels – 112 lag points or 1 channel – 224 lag points. The system is capable of adapting its cross-correlator resources on instruction to any of the above modes. This adapting is handled by a recursive feedback design between cross-correlator modules. Its total complexity of lag resources is equivalent to 28 X 8 = 224 lag.

Now let us find the complexity of a VLBI correlator system that computes the same algorithms but performs no adaptations for cross-correlator lags. Since it must handle the same required lag length, it must have 1 channel – 224 lag points, 1 channel – 112 lag points, 2 channels – 56 lag points, 3 channels – 32 lag points, 7 channels – 16 lag points and 14 channels – 8 lag points. Thus, the overall channels are still 28 but the resources complexity of lag has increased to 96 – 8 lag cross-correlator. Its total complexity is equivalent to 96 X 8 = 768 lag.

Consequently, to maintain the same level of concurrency, the system with conventional architecture must increase its complexity of lag resources to 3.43 times that of a system which can perform adaptations on parallelism. An 8-lag cross-correlator module requires about 25 integrated circuits (ICs). To realize the level of hardware required for a 6-station by 28-channel adaptable cross-correlator, one must have 25 ICs X 28 channels X 15 baselines = 10,500 ICs. A cross-correlator with no adaptations is 3.43 times this amount or 36,015 ICs. Since new adaptable cross-correlator modules must communicate with each other, new connecting elements must also be added, their number depending on which recursive feedback design is utilized. This addition is very small compared to the overall cross-correlator module hardware.

V. Detailed Design Summary and Status

This section summarizes the detailed design and implementation work now proceeding for the high-speed digital cross-correlator. For the following discussion, reference to Fig. 4 should be made. This diagram shows the overall data flow for one station/correlator baseline. The input data, at the left, enters from magnetic tape with 28 data streams per station and at rates up to 8 million bits per second per stream.

A. Emitter-Coupled Logic (ECL), Frame Sync and Delay Buffer Circuits

1. **ECL.** The data from tape head line drivers first enters the ECL circuit board. This board has 30 28-to-1 ECL multiplexers, one for each of the 28 channels and two for self-test. Any tape track may therefore be routed into any data channel for processing. The data then enters a digital bit-synchronizer circuit which generates a clock reference from data. Data and clock for all channels are converted to TTL (transistor-transistor logic) level before leaving the board.

Total ICs for this board	450
Total power required	160 W

Design and check-out of ECL prototype is complete.

2. **Frame Sync.** The data stream first enters FIFO (first in/first out) memory, which removes the effects of recorder jitter and synchronizes the data to an internal clock. A sync word is decoded from the data stream to locate frame boundaries and generates a BOF (buffer output frame) timing interrupt. This decode circuit uses only two Programmable Array Logic (PAL) ICs. Loss of frame synchronization is detected when the decoder finds a sync word at the wrong bit count. Resynchronization is then started, a frame error generated and counted.

3. **Delay Buffer.** Buffering of the data streams, for correct alignment before cross-correlation, is implemented with 16 X 1 dynamics RAMs (random access memory). 8-RAM ICs are used for data delay or a total of 0.131 million bits of delay. Three additional RAM ICs are added for parity delay and delaying the BOF/FRI (frame read interval) timing interrupts. FRI is a master interrupt generated by the hardware every frame period. The delayed time difference between BOF and FRI is a delay correction made to the delay buffer by the station's phase processor. A delay update is made every 5k bits. This buffer circuit is upward compatible to 64k X 1 or 256k X 1 dynamic RAMs (2.097 million bits of delay), when it becomes cost effective to do so.

Additional FIFO memory is added for storing time labels, auxiliary data words, 12-bit cycle redundancy check character and frame error counts.

Total ICs for 15 modules (one board)	600
Total power per board	250 W

Design and checkout of FRAME SYNC/ DELAY BUFFER is complete.

B. Calibration Tone Detectors

The data at the output of the delay buffer is converted back to a serial data stream for mixing and cross-correlation. The cal-tone is also extracted at the buffer's output. The serial data is mixed with 8 bits of phase from the station's phase processor. This gives 127 levels to the sine and cosine waveforms. The fringe rate generator word length is 24 bits and is updated every 5k data bits. The output of the mixing is then accumulated in 16-bit counters before transfer to the station phase processor for additional processing.

Total ICs for 28 modules (one board)	620
Total power per board	275 W

Design is complete.

Additional board slots are provided so that up to 4 cal-tones per channel may be tracked simultaneously.

C. Station Process Controller

The station process controller is designed to handle all of the high intermodule communication between the tailor data modules and data processor. These data modules, i.e., frame sync, buffer and cal-tone detectors, are assigned to a controller so that all the real-time constraints are off-loaded from the data processor. This clustering of modules to a processor is called fusion. All frame sync, buffer and cal-tone detectors are fused to the station process controller for communication I/O at the data stream bit rate. The design of the controller throughput has to be 8 million instructions per second for a data stream rate of 8 million bits per second. To provide that kind of throughput, special microprogrammable hardware was designed. The hardware's microword length is 96 bits by 512 words. Built into the firmware are also fault detection and diagnosis, capable of fault isolating down to subcircuits within a data module.

Total ICs required	60
Power required	25 W

Design and checkout of prototype is complete.

The hardware for this process controller is identical with the correlator process controller, with about 80% of the firmware transferable.

D. Station Phase Processor

The station phase processor will compute the calibration fringe rate for all cal-tone modules and compute the buffer delay for all channels. Calculation is based on a 64-bit-word-length algorithm which is completed every 100k data bits. Typical cal-tone accumulation dumps also occur every 100k data bits. From this data the phase processor computes the phase (arctan i/r) for four tones per channel. The computed phase is than transmitted to the cal-interface for transfer to the cross-correlator. Here the phase is applied as a correction to the cross-correlated output every 100k data bits. New model updates to the station phase processor from the computer are typically made every 25 seconds. The data word length of the processor is 32 bits and is capable of doing a 32-bit by 32-bit multiply in 333 nanoseconds. Throughput is six million instructions per second.

Total estimated ICs	175
Total estimated power	75 W

Design is 80% complete.

The hardware design for the phase processor and the design for the corralator phase processor are identical, with about 60% of the firmware transferable.

E. Station Rack

The number and type of boards in the station racks are one ECL board, two frame sync-buffer boards, one cal-tone detector board and one control/processor board. A total of 11 boards may be housed in the station rack along with the Honeywell recorder electronics.

F. Digital Interfaces

1. **Data Interface.** Data stream interfaces between the station racks and correlator rack are made of twisted-pair flat cable, 14 data and 14 invalid plus clock twisted-pairs per cable. All interfaces are designed for digital transmission over balanced differential lines. Data speeds are less than 10 million bits per second per data stream pair.

2. **Calibration interface.** The calibration interface transfers cal-phase between station racks and correlator rack, also using twisted-pair flat cable. This interface transfers 16-bit data words over a common interface. Each station rack sends 120 words every 100k data bit times. The data rate is about 1 million bytes per second.

G. Cross-Correlator

After both data streams are buffered by a programmable delay in the station racks, the data is sent to the correlator for cross-correlation. One stream is multiplied by 3-level approximations of the sine and cosine functions, then cross-correlated with the other data stream. The cross-correlator has 8 lags per channel and 16-bit accumulation. After cross-correlation the correlator process controller dumps the correlated data to the correlation signal processor. The fringe rate generator word length is 24 bits and is updated every 5k data bits.

The cross-correlator is designed to process a variable-number correlated length of concurrent data streams. So different numbers of data streams processed by the cross-correlators will be characterized by variable lag lengths. The cross-correlator may be reconfigured into 1 channel – 224 lags, 2 channels – 112 lags, 4 channels – 56 lags, 7 channels – 32 lags, 14 channels – 16 lags or 28 channels – 8 lags. The algorithms in the digital signal processor will also have to match this redistribution in processing the data.

The cross-correlator has been designed without the use of custom LSI chips. The design uses a number of the programmable array logic family ICs for reducing package count.

Total ICs for 14 modules (one board)	400
Total power per board	175 W

Design and checkout of prototype is complete.

H. Correlator Process Controller

The correlator process controller, like the station controller, is designed to handle all of the high intermodule communication between the tailor data modules and data processors. The design of the controller throughput has to be 8 million instructions per second for control of data stream rate up to 8 million bits per second. One controller can handle up to 3 baselines – 28 channels and can also fault detect with fault isolating hardware down to subcircuits within a cross-correlator module.

Total ICs required	60
Power required	25 W

Design and checkout of prototype is complete.

The hardware for this process controller and the hardware for the station process controller are identical, with about 80% of the firmware transferable.

I. Correlator Phase Processor

The correlator phase processor will compute the fringe rate for all 3-baseline, 28-channel cross-correlator modules. Calculation is based on a 64-bit-word-length algorithm, which is completed every 100k data bits, as in the station phase processor design. The processor also calculates the fractional bit correction for each cross-correlator. The output of the fractional bit correction algorithm is a phase value which is added to the cal-phase values every 100k data bit times. These added phase values are then passed along to the correlation signal processor for data correction. The values are sent over the cal-correction interface between phase processor and correlation signal processor. New model updates to the correlator phase processor from the computer are typically made every 25 seconds. The data word length of the processor is 32 bits. The processor is capable of doing a 32-bit by 32-bit multiply in 333 nanoseconds. Throughput is six million instructions per second.

Total estimated ICs	175
Total estimated power	75 W

Design is 80% complete.

The hardware design for the phase processor and that of the correlator phase processor are identical, with about 60% of the firmware transferable.

J. Correlation Signal Processor

The correlation signal processor uses an FFT directly to compute estimates of the cross-correlation function for lags/channel, where lags/channels are the size and sequence of the transform used. Using an N-point FFT, the power density spectrum may be estimated at N uniformly spaced frequencies around the unit circle. In order to improve the estimate at the finite set of frequencies, a correction window must be used to reduce the effects of fraction bit and oscillator phase uncertainty. This correction window is supplied by the correlator phase processor as phase values, and is updated every 100k data bit times. The windowed FFT output is then boxcar-filtered (integration) for typically one second before outputing to the computer or tensor processor.

It is advantageous to reduce the output to the computer as much as possible so that storage and arithmetic operations can be minimized in the computer. Sample rate reduction can be performed with no loss in sensitivity by passing the data through a digital low-pass filter. Then only every N samples will be retained for output. The filter will be a IIR (infinite impulse response) recursive design type with a selectable reduction ratio. The reduction is independent for each group of three baselines processed by the correlation signal processor. The filter will operate on the 32-bit boxcar-filtered data.

The data word length of the processor is 16 bits, and it will do a 16-bit by 16-bit multiply in 166 nanoseconds. Throughput is six million instructions per second.

Total estimated ICs	175
Total estimated power	75 W

Design is complete.

K. Tensor Processor

The tensor processor has previously been called 'phasor' and has been renamed to prevent confusion with the Mark 2 / Block O phasor software program.

The output of the correlation signal processor may be routed to the tensor processor for additional processing in the hardware. The additional processing requires filling a bandpass buffer holding 2048 complex points. Points in the buffer which are not loaded with data are set to zero. The bandpass buffer will be Fourier transformed from frequency domain to time domain. If a priori information allows one to estimate delay, one can select a delay window that defines the delay range and keep that data for further processing. Mutiple windows may be defined. This windowed data then fills a programmable delay buffer. When full, a Fourier transform is performed on the columns of the delay buffer. After the transform, the array contains visibilities on a grid of synthesized delay and fringe rate. Data in the array may be output to the computer on a selective basis. This is according to the delay window already selected and according to a priori fringe rate information. The tensor processor reduces overall I/O data rates between the computer and hardware correlator, for normal processing, by 1000 to 1 plus facilitating storage economics in the computer. The tensor processor implementation is planned to handle three baselines.

In the design of the correlation signal and tensor processor it was attempted to include sufficient flexibility to support all the known observational requirements. In the extreme case, it will be possible to bypass all the functions described here, and dump the 'raw' cross-correlated data directly to the computer.

Total estimated ICs	400
Total estimated power	160 W

The design is preliminary at this time.

L. Correlator Rack

A 3-baseline, 28-channel cross-correlator requires 6 boards; two additional boards are required for control and processing. A total of 28 boards can be housed in one correlator rack. The rack is a VAX 11/780 double-wide CPU (central processing unit) cabinet.

M. I/C Summary

For one station, ICs total 2505, plus an additional 200 for recorder control and interface, or

ICs	2705
Power	1135 W

For one three-baseline correlator system, ICs total 3210, plus an additional 100 for interfaces or

ICs	3310
Power	1485 W

The JPL/CIT hardware design for a 4-station 6-baseline correlator system uses 17,440 ICs and consumes 7510 watts of power.

The Haystack Mark III processor design is based on a straightforward modular concept of one module per baseline per track. Thus, 28 (350 ICs) modules are required per baseline. The Haystack hardware design for a 4-station 6-baseline correlator system uses 58,800 ICs and consumes 6470 watts of power. The Haystack hardware design is an existing VLBI cross-correlator design using mostly SSI technology ICs and designed without new data-flow architectures.

VI. Conclusions

We have shown that to increase throughput and lengthen the life cycle of supersystems, new architectures must perform distribution and adaptation to algorithms not implemented in traditional systems. The proliferation of both types of architectures was encouraged by advances in large-scale integrated circuits (LSI) technology that significantly reduce the cost and enhance the reliability of processor components. These advances in LSI technology have made distributed processing a practical system design approach. The modularity, flexibility, and reliability of distributed processing make it attractive to many types of users.

The VLBI correlator system has been designed without the use of custom LSI chips. In the near future the advantages provided by this custom LSI technology will allow even faster and less expensive computations for processor systems. This technology will become part of the accepted processing repertoire of future system designers.

We must assume that, in the end, system resources are limited. That is, the number of available processors, processor speed, memory capacity, and number of modules are fixed and limited by available system resources. In real-time applications, allowed elapsed time is also a limited resource. In any case, system design methodology should strive for simplicity balanced by a need for being fast enough to meet system performance requirements.

References

1. Wittle, L.D., "Efficient Message Routing in Mega-Microcomputer Networks," *Proc. Third Annual Symp. Computer Architecture,* 1976, pp. 136-140.
2. Vick, C. R., "Research and Development in Computer Technology, How Do We Follow the Last Act?" *Proc. 1978 International Conference on Parallel Processing,* August 22-25, 1978, pp. 1-5.
3. Kurtashev, S. P., and Kartashev, S. I., "Adaptable Pipeline System with Dynamic Architecture," *Proc. 1979 International Conference on Parallel Processing,* pp. 222-230.
4. Peterson, J. C., and Dillon, J. W., "Implementation of the DSN VLBI Correlator," *The DSN Progress Report 42-50,* Jet Propulsion Laboratory, Pasadena, Calif., January 1979, pp. 226-236.

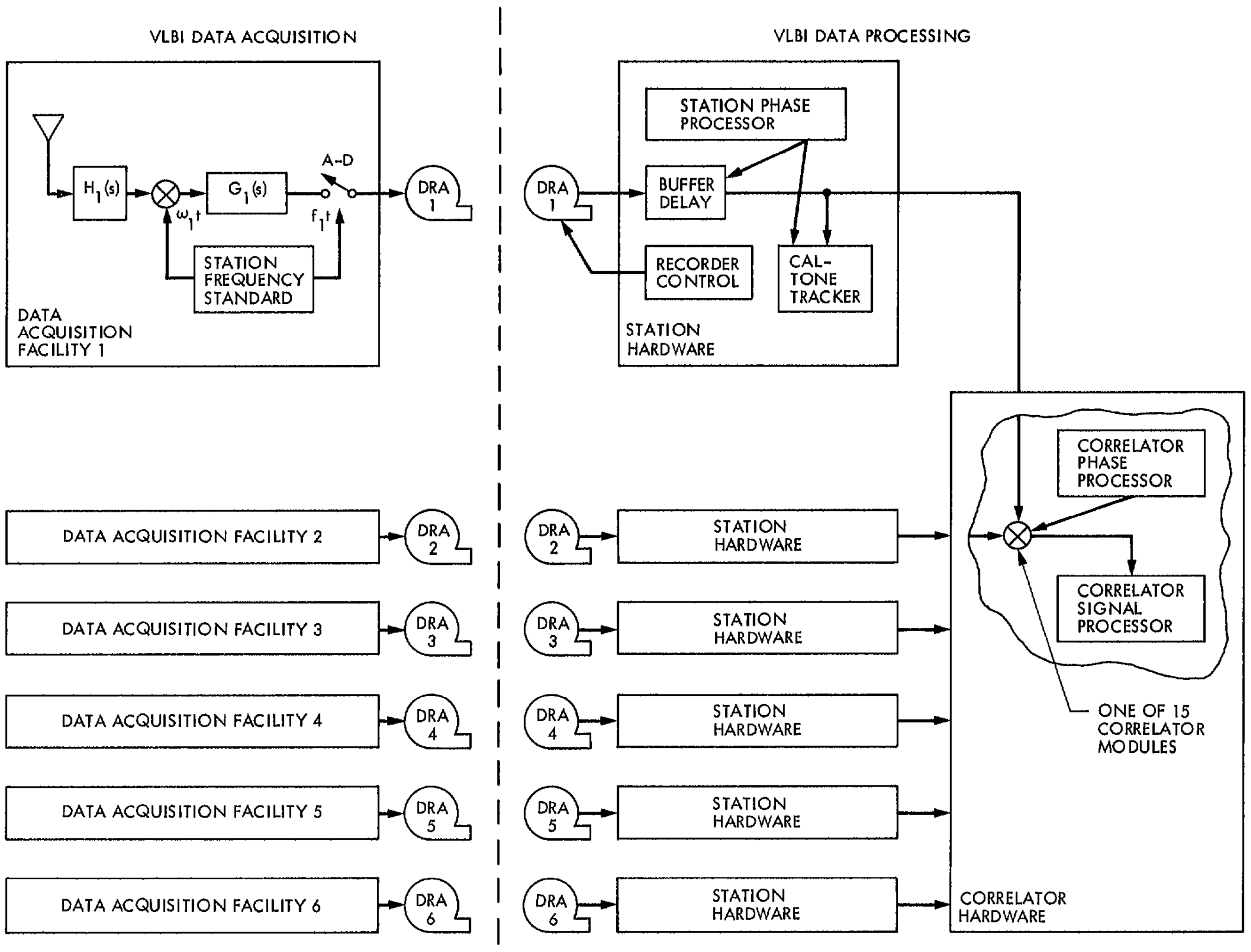
VLBI DATA ACQUISITION
VLBI DATA PROCESSING
$H_1(s)$
$G_1(s)$
A-D
$\omega_1 t$
$f_1 t$
STATION FREQUENCY STANDARD
DATA ACQUISITION FACILITY 1
DRA 1
STATION PHASE PROCESSOR
BUFFER DELAY
RECORDER CONTROL
CAL-TONE TRACKER
STATION HARDWARE
CORRELATOR PHASE PROCESSOR
CORRELATOR SIGNAL PROCESSOR
ONE OF 15 CORRELATOR MODULES
CORRELATOR HARDWARE
DATA ACQUISITION FACILITY 2
DRA 2
DATA ACQUISITION FACILITY 3
DRA 3
DATA ACQUISITION FACILITY 4
DRA 4
DATA ACQUISITION FACILITY 5
DRA 5
DATA ACQUISITION FACILITY 6
DRA 6
STATION HARDWARE

Fig. 1. VLBI data acquisition and processing

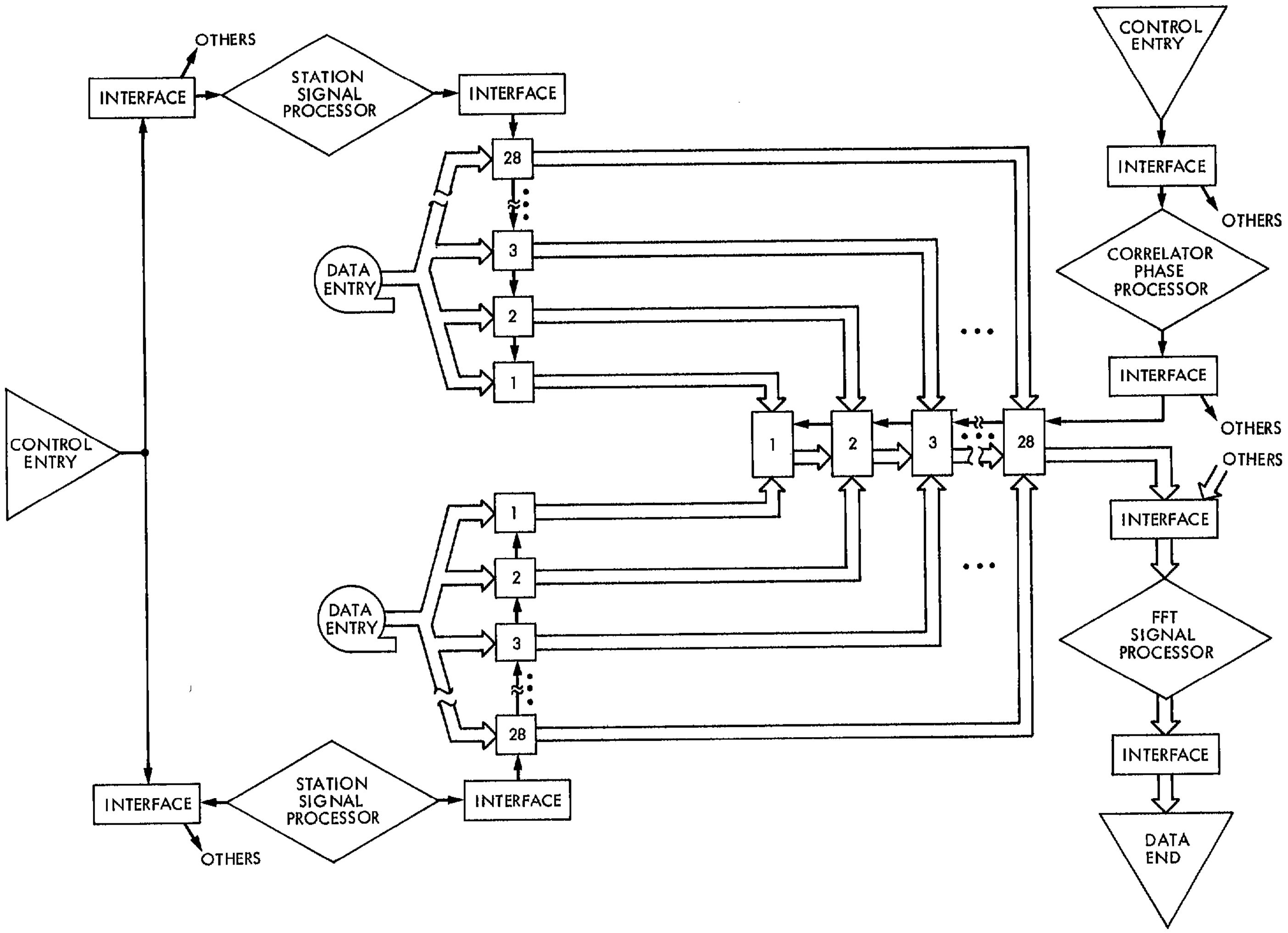

Fig. 2. VLBI distributed correlator system, 2-stations/28 channels

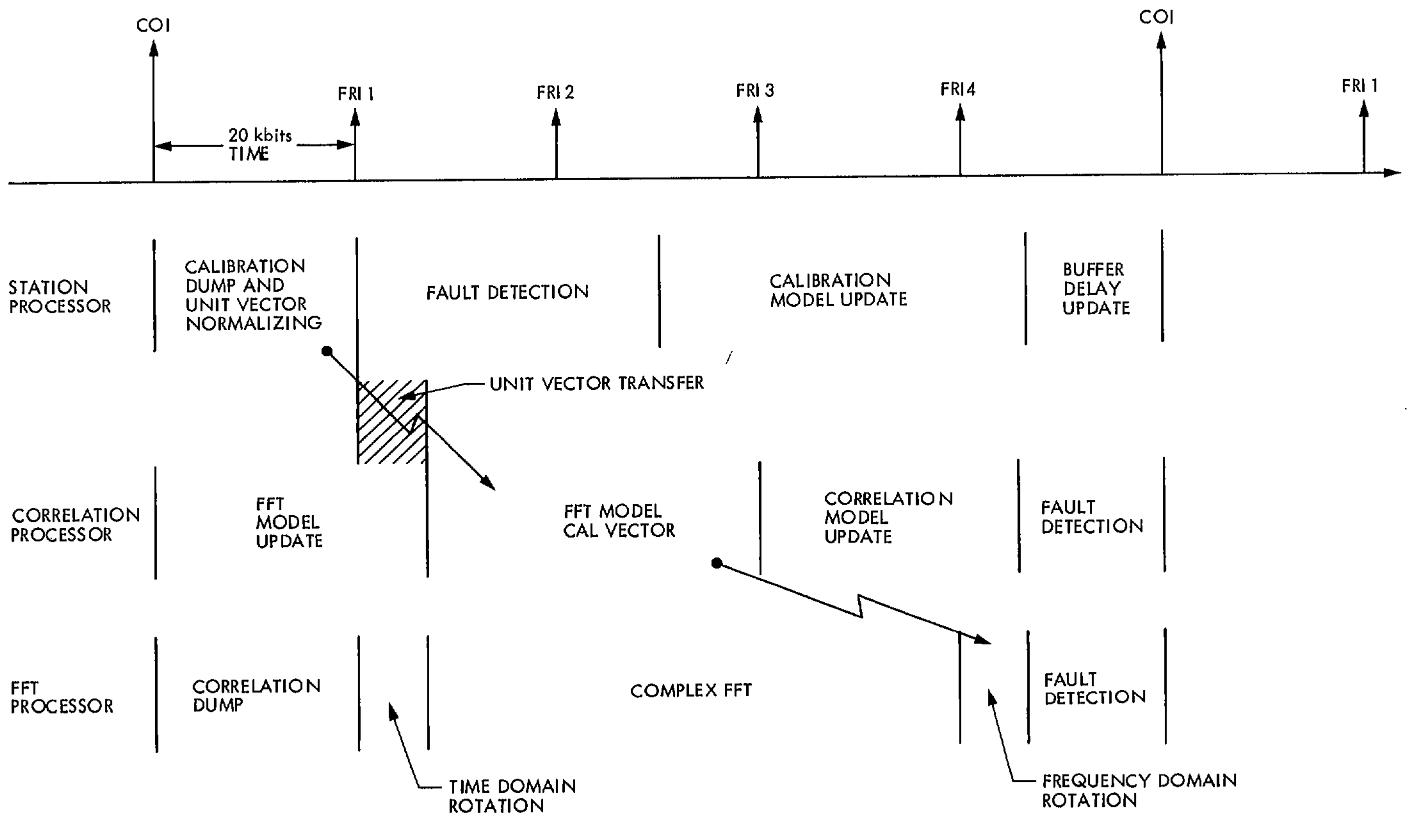

Fig. 3. VLBI correlator system, task and communication allocation

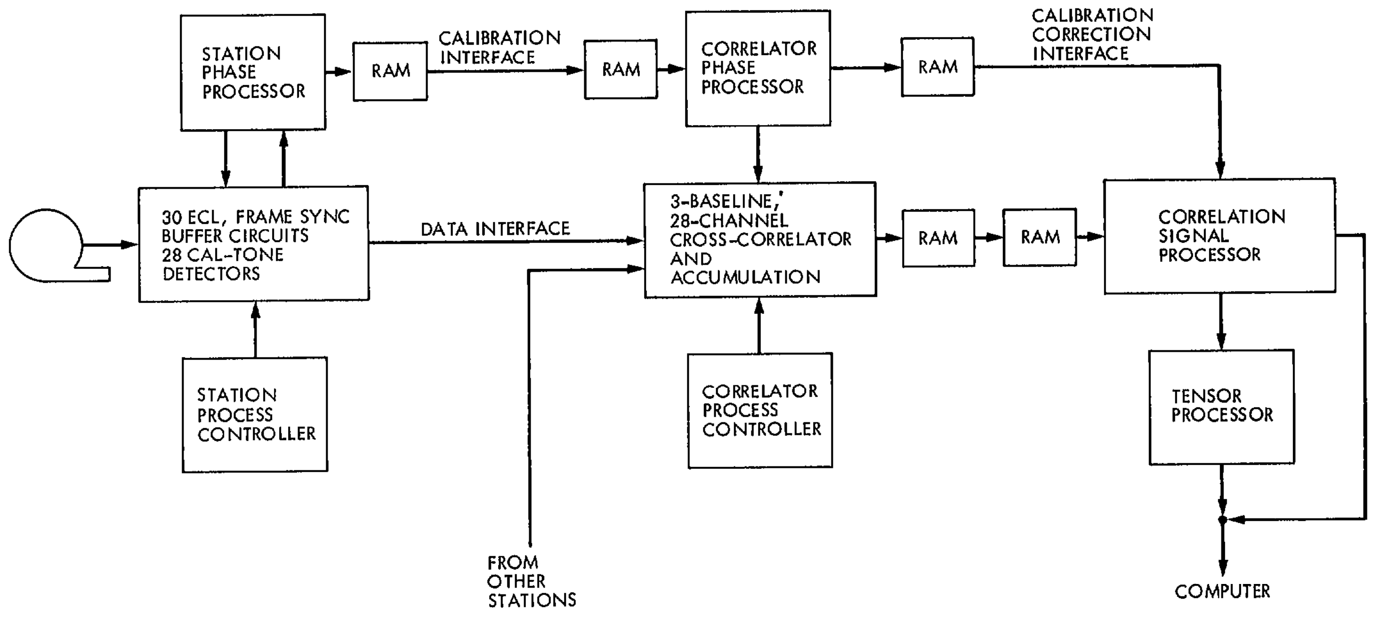
STATION PHASE PROCESSOR
RAM
CALIBRATION INTERFACE
RAM
CORRELATOR PHASE PROCESSOR
RAM
CALIBRATION CORRECTION INTERFACE
30 ECL, FRAME SYNC BUFFER CIRCUITS 28 CAL-TONE DETECTORS
DATA INTERFACE
3-BASELINE, 28-CHANNEL CROSS-CORRELATOR AND ACCUMULATION
RAM
RAM
CORRELATION SIGNAL PROCESSOR
STATION PROCESS CONTROLLER
CORRELATOR PROCESS CONTROLLER
TENSOR PROCESSOR
FROM OTHER STATIONS
COMPUTER

Fig. 4. Block VLBI correlator data flow block diagram