RL-TR-95-223
Final Technical Report
November 1995

SYSTEM, PERFORMANCE, AND APPLICABILITY ASSESSMENT OF A HIGH-PERFORMANCE COMPUTER SYSTEM

State University of New York at Binghamton

Dr. Kanad Ghose

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

Rome Laboratory
Air Force Materiel Command
Rome, New York

This report has been reviewed by the Rome Laboratory Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be releasable to the general public, including foreign nations.

RL-TR-95- 223 has been reviewed and is approved for publication.

APPROVED: John Grieco

JOHN GRIECO
Project Engineer

FOR THE COMMANDER:

DELBERT B. ATKINSON, Colonel, USAF
Director of Intelligence & Reconnaissance

If your address has changed or if you wish to be removed from the Rome Laboratory mailing list, or if the addressee is no longer employed by your organization, please notify Rome Laboratory/ (IRAA), Rome NY 13441. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	November 1995	Final Jun 92 - Jun 93

4. TITLE AND SUBTITLE
SYSTEM, PERFORMANCE, AND APPLICABILITY ASSESSMENT OF A HIGH-PERFORMANCE COMPUTER SYSTEM

5. FUNDING NUMBERS
C - F30602-92-C-0148
PE - 62702F
PR - 4594
TA - 15
WU - P6

6. AUTHOR(S)
Dr. Kanad Ghose

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
State University of New York at Binghamton
Department of Computer Science
P.O. Box 6000
Binghamton NY 13902-6000

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Rome Laboratory/IRAA
32 Hangar Rd
Rome NY 13441-4114

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
RL-TR-95-223

11. SUPPLEMENTARY NOTES
Rome Laboratory Project Engineer: John Grieco/IRAA/(315) 330-4024

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

The objective of this effort was to perform an evaluation of the first-generation RL/IRAA VHSIC speech multiprocessor. In particular, the shortcomings of the system were highlighted so that modifications could be made to the second-generation VHSIC speech multiprocessor. A detailed comparison of CPUs was made for use in the second-generation processor, and a specific choice was justified.

14. SUBJECT TERMS
High-performance computing

15. NUMBER OF PAGES
36

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

TABLE OF CONTENTS

1. INTRODUCTION

This report describes the work done under contract No. F30602-92-C-0148, entitled *System, Performance and Applicability Assessment of a High-Performance Computer System.* The objectives of this project were slightly revised from the objectives described at the time of submitting this proposal as a result of discussions between the principal investigator and the technical contacts in the speech lab in the RL/IR directorate at Rome Labs. The revised objectives are summarized here:

- Perform a critical evaluation of the first generation of the VHSIC speech multiprocessor. In particular, highlight the shortcomings of the system and indicate potential features to be included in the second generation of speech processors that are missing in the existing VHSIC speech multiprocessor based on the AT&T DSP32 signal processors.
- Undertake a detailed comparison of CPUs that can be incorporated in the second generation of VHSIC multiprocessors.
- Justify the choice of a particular CPU for use in the second generation of the speech multiprocessors.
- Outline a design for the main processing section of the second generation of the speech multiprocessor based on the candidate CPU, whose choice was justified.

This report has five major sections. The following four sections mirror the four major goals listed above. The last section represents our overall conclusions.

2. A CRITICAL EVALUATION OF THE EXISTING VHSIC SPEECH MULTIPROCESSOR

The current generation of the VHSIC speech multiprocessor was custom designed for the RL/IR directorate by Martin Marietta corporation. This section of the report is based on the information available from the manuals provided by Martin Marietta, as well as information available from the technical personnel at Rome labs and Martin Marietta.

2.1 An Overview of the First-Generation VHSIC Speech Multiprocessor

The first generation of the VHSIC speech multiprocessors are based on technology that is at least 5 to 6 years old. The main computing engine used in this system are the AT&T DSP32 signal processing multiprocessors. Figure 1 shows the main components of the first generation VHSIC multiprocessors. The main components in this system are:

- The DACU - Data Acquisition and Conversion Unit. This provides the analog-digital interface for the VHSIC system.

- One or more APUs - Array Processing Units. The APUs provide the main processing for the system.

- One or more MUs - Memory Units. The memory units provide data storage for recognition templates and code book data.

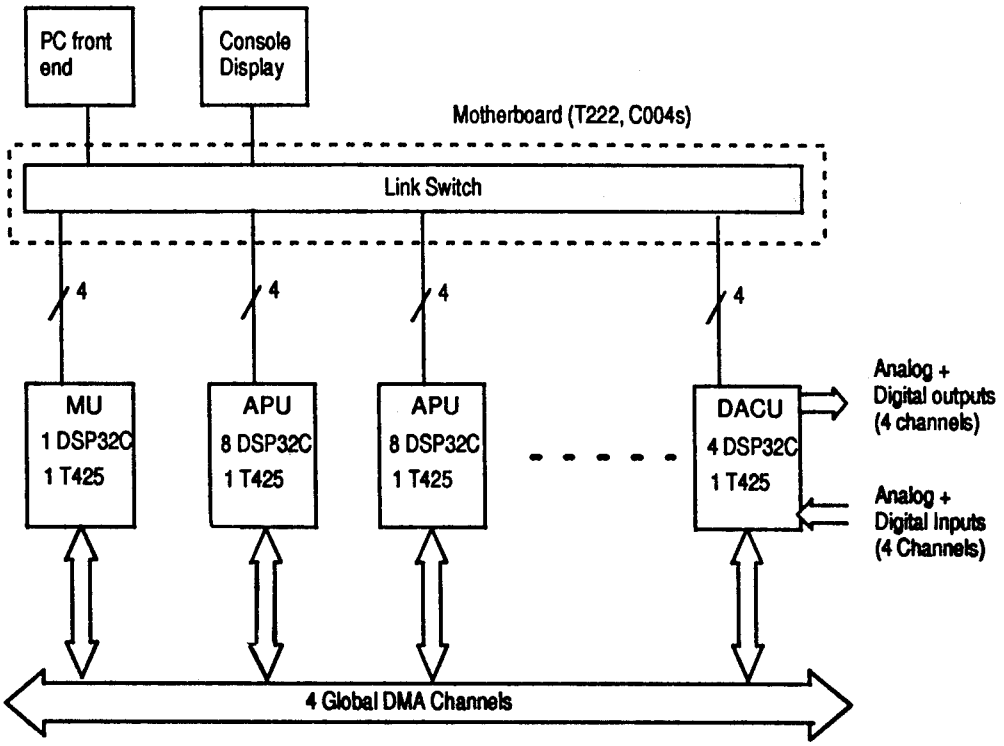

Figure 1. The First-Generation VHSIC Multiprocessor

- One or more motherboards. The motherboards allow the various processing elements (PEs) (on the APUs) and MUs to be interconnected using serial links each rated at 20 Mbits/sec. The motherboard also provides the interface functions for the system and allows the entire system to be dynamically reconfigured. In particular, it allows the interconnectivity among the PEs and MUs to be re-configured into a variety of topologies including meshes, trees, hypercubes, shuffles, cube-connected cycles etc.

- Four sets of global DMA busses (described as **external** busses in the Martin Marietta manuals) that connect the various units (and motherboards). Each DMA bus is rated at 80 Mbytes per second (peak).

- Four serial links, each rated at 20 Mbits/sec. from the DACU, each APU and MU to the motherboard.

In the next section, we describe the various units in the VHSIC speech multiprocesor is some detail.

2.2 Major Units in the VHSIC Speech Multiprocessor

The role of the various subsystems/units in the first generation system are as follows:

DACU - Data Acquisition and Conversion Unit: This is the analog interface and data acquisition unit for the system. Each DACU can also handle four channels of digital inputs. Analog-digital conversion, filtering and signal conditioning is performed on the analog inputs available on up to four input channels. The conditioned data is packetized and sent over a custom DMA bus to the main processing boards, the APUs. The DACUs also provides four channels of analog and digital output. The main digital computing engines in a DACU are:

- One DSP32C for each of the four analog channels. These operate at a 40 MHz. clock and provide a peak floating point power of 20 MFLOPs (IEEE 32-bit floating point ops) per DSP32C. For each one of the four channels handled by a DACU, the associated DSP32C performs FIR filtering (for input and output, with 201 coefficients, with a maximum allowable phase shift of 10 degrees), decimation (rates: 10K, 8K), interpolation (by 4 or 5) and automatic selection of gain ranges.

- One T425 Transputer. The T425 plays the role of the overall supervisor for the DACU and is used to dynamically reconfigure the DACU as needed. The T425 also supervises data I/O to/from the DACU to other components via four serial Transputer links.

Each channel on the DACU has a frequency response of 20 Hz. to 4.2 KHz. (within 3 dB) and each channel has a sharp cutoff beyond the specified frequency range (96 dB attenuation at 5 KHz.). A 40 KHz. sampling frequency is employed for each channel to provide a 16-bit digitization of the analog inputs. The DACU also incorporates a variety of analog/hybrid filters and gain-controlled amps (gain range: 0.5 to 18K), various buffers and four 16-bit DACs (that provide the analog outputs on the four channels). The use of gain controlled front-end amps provide a dynamic range in excess of 84 dB. Several internal busses and DMA controllers are also incorporated within the DACU.

Each DACU has access to the four global DMA channels.

APU - Array Processing Unit: These units are responsible for performing the signal processing on the digitized data available from the DACU. Each APU incorporates 8 DSP32Cs (as four hybrid multi-chip modules, each with two DSP32Cs) to perform

either:
(a) 8 sets of speech coding or speech enhancement processes. (As we will see later, these processes are not as independent as they appear to be at first glance because of a master-slave hierarchy between the two DSP32Cs on the hybrid package.)
or
(b) One speech recognition function using template matching/dynamic time warping.
The main processing elements within a APU are:

- Eight DSP32Cs, in four hybrid packages with two DSP32Cs to a single hybrid package. The DSP32Cs on the hybrid packages are clocked at 50 MHz., leading to a peak IEEE 32-bit floating point performance of 25 MFLOPs for each DSP32C. Consequently, the peak floating point performance (32-bit IEEE) of a APU is 200 MFLOPs. The two DSP32Cs within a hybrid package are *not configured symmetrically* - one DSP32C is set up as a master and the other as a slave. Only the master DSP32C has direct access to an external DMA bus. The slave DMA has direct access to a DMA bus within the APU. The master and slave DSP32Cs communicate via data registers and buffers within the hybrid package.

- One T425 Transputer. This serves the same role as the T425 Transputer on the DACU, viz., providing board-to-board connectivity and local supervisory functions.

Besides these computing elements, each APU board also incorporates a variety of buffers, DMA controllers (implemented within programmable logic arrays), and local memory devices. The APUs are programmable in assembly and C. The 1024-point complex FFT benchmark runs on the 8 DSP32Cs on an APU in 0.41 msecs. (In contrast, if the same benchmark is run on a single DSP32C within the APU, the time taken is 2.87 msecs.)

MU - Memory Unit: These units are used for holding recognition templates used for speech recognition and code book data (used for VQ-based speech coding). The bulk storage within the MU is in the form of 32 Mbytes of dynamic RAMs. The use of a fully-associative cache (implemented with 1K X 32 content addressable memories and 128 Kbytes of static RAMs) speeds up the access time to the dynamic RAMs. The MU can access the four global DMA channels but allows only one DMA channel to be operate at a time. Like the APU and DACU, the MU incorporates a T425 Transputer for supervisory and reconfiguration functions, as well as for inter-unit connections (using four serial links). The peak data rate available from a MU is limited by the global DMA channel bandwidth - the on-board cache thus provides adequate performance.

Motherboard: The motherboard provides the external interface to the host PC and display unit. In addition, it incorporates a T222 Transputer and a number of C004

switches that allow the processors on the APUs and MUs and the DACU to be connected in a variety of topologies using the four serial links available from each unit.

In the next section, we describe the shortcomings of the first-generation VHSIC multiprocessor and indicate features that are desirable for the second generation of this system.

2.3 A Critique of the First-Generation VHSIC Multiprocessor

We will divide up the critique on the VHSIC multiprocessor into three subsections, the first being an architectural critique of the system, the second a programmers critique and the third an end-user critic for the system.

Architectural Critique

In this part, we critique the system-level design of the VHSIC prototype. In particular, we do not focus on the shortcomings that are a consequence of using the DSP32C as a processing element.

Two architectural shortcomings are fairly serious in the VHSIC multiprocessor. Both of these are concerned with the APUs:

1. The master-slave configuration of the two DSP32Cs on the same hybrid package results in an asymmetry that impacts performance as well as programming. If the two DSP32Cs in a hybrid package are assigned to independent speech coding and speech enhancement tasks, the tasks although logically independent, are in reality not decoupled. If, for instance, speech coding is running on the slave DSP32C, the code book data will have to come from the MU via the global DMA bus. This requires the master DMA – running an independent task – to handle the I/O for the slave. Consequently, the task running on the master can take a performance hit. This situation is particularly aggravated when both the master and slave are running tasks that require the use of the global DMA channels (both running speech coding, for example).

2. The four channels processed by the VHSIC speech multiprocessor share resources – in particular APU resources – that does not allow the processing for a channel to be reconfigured while the other channels are operational. As an example of this scenario, consider speech coding tasks for two channels, say channels 2 and 3, running on an APU. If channel 2 is reconfigured to implement speech enhancement on the same APU, the operations for channel 3 are affected. This is because reconfigurations affect entire APUs at a time.

There are some additional architectural problems in the first-generation VHSIC multiprocessor. These are listed below:

3. The serial links that are used for connecting the processing elements and MUs in a specific topology (hypercube, mesh, tree etc.) are Transputer links and as such have a fairly high communication overhead - 3 control bits are needed for every byte that is transferred - this amounts to a packet overhead of 38%. The 20Mbit/sec. Transputer links can be potential bottlenecks - especially if higher throughput CPUs are used. Further, all 8 DSP32Cs in a single APU share the four serial links - the topologies for processor to processor interconnection are thus extremely limited and poor. In addition, only the master DSP32Cs in an APU can communicate directly with the T425 Transputer - communication between the T425 and the slave DSP32Cs are thus relatively slower, making the inter-processor connections asymmetric. The ability to harness the power of CPUs operating in parallel is thus limited. The interconnection strategy used within an APU is also handicapped - these are evident from the FFT benchmark timings. Speedup achieved using all 8 DSP32Cs within an APU on the FFT benchmarks is about 6.5 times compared to the timing on a single DSP32C. Since the FFT code is cleanly parallelizable, one would expect the speedup to be higher - say 7.5 or thereabouts (but still lower than the ideal speedup of 8). Since the communication time between the master and slave DSP32Cs within a hybrid package is fast, one would suspect that the degradation in speedup is due to deficiencies in the inter-package connections within a APU. It is also reasonable to surmise other applications that are not as parallelizable as the FFTs can take a higher hit in the speedup of going from one to 8 CPUs.

4. All 8 DSP32Cs in an APU share four Transputer links to the C004 switch array on the motherboard. It is thus *physically* not possible to connect the CPUs and MUs in the topologies that are required (meshes, hypercubes, shuffles etc.). As an example, if 16 CPUs on two APUs are to be connected in a four-dimensional hypercube, four physical links are needed from *each* CPU in an APU - this is clearly not possible as all 8 CPUs in an APU share only four Transputer links. This true embeddings of the various interconnection topologies (as claimed in the manuals) are not possible. Instead, all the logically required connections among CPUs and MUs will have to be simulated using the four relatively slow Transputer links to/from an APU. This can take a serious performance toll in applications that require CPUS to communicate with each other frequently.

5. The transfer time on the global DMA busses can be approximated as:

$$T = F + D*BW$$

where D is the datasize (maximum of 32 Kbits) and BW is the channel bandwidth (80 Mbytes/sec.). F represents a flat startup/arbitration overhead. In the first generation VHSIC multiprocessor, F is about 50 μsecs. and the total time taken to transfer a 1K word data packet on an external bus (T in the above equation, for D=1K word) is about 59.2 μsecs. This shows that the flat overhead F of the global bus dominates the

overall transfer time. *The relatively large flat overhead is a consequence of implementing the bus protocols with the transputers.* In the current system, the large DMA transfer time is not a bottleneck since the DACU essentially drives the other units and does not saturate the DMA channels. (The DMA transfer rate can become a bottleneck in the first-generation VHSIC multiprocessor if several APUs are doing voice recognition.)If, in the future generation of the VHSIC multiprocessor, the DACUs accommodate more channels and the APUs use DSPs rated at higher speeds, the global busses can become a bottleneck. One can not take care of this problem by simply adding more global busses, since additional global busses can put a severe demand on the already high number of connections to a board. If the global bus protocols are implemented in random logic, the flat overhead can be easily cut down by at least a factor of 10.

To summarize, problems 1, 2 and to some extent 3 and 4, as described above are fairly serious problems that have to be addressed in future designs. Problem 4 can be taken care of relatively easily.

Programmer's Critique

In this section, we describe some limitations of the VHSIC system as perceived by the programmers. *We would like to note that there are, quite possibly, other programming limitations in the system, that we could not identify in this section, since we had no access to an actual VHSIC multiprocessor.* What we have listed here as potential problems that face the VHSIC system programmer is based on a study of the VHSIC multiprocessor at the system level:

1. The VHSIC speech multiprocessor is a parallel system in all senses of that word. Yet, virtually no programming environment is included to support parallel programming in the first generation prototype. The onus is on the programmer to partition the application into parallel components and "schedule" these components over the various units/processors - an all too formidable task prone to errors.

2. The programmers responsibility is further complicated by the fact that in many applications asymmetries (in the form of the master-slave DSP32Cs in an APU) and the fact that inter-CPU links have to be simulated (item #4 in the architectural critic), complicate the already difficult task of programming a multiprocessor system without any parallel programming environment.

3. A parallel debugger is also not available in the system, complicating the program development process further.

4. Any parallel system, where applications have to be fine-tuned to optimize performance requires sufficient high-level instrumentation to be built into the

programming environment to time various operations of interest. Such facilities are clearly lacking in the first-generation of VHSIC multiprocessor system.

6. Sufficient support, including customized programming environments (that incorporate some of the features listed in items 1 through 5) to support the programming of the Transputers are lacking.

As noted earlier, since we have not actually programmed the VHSIC system, we are unable to identify some other potential problems that affect the programmer.

End-User Critique:

Once again, as we have not actually used the VHSIC systems, we cannot mention the inadequacies in the system as perceived by the end-users. However, judging from the manuals, the visual interface to the system appears to be very primitive. Better visual interfaces can certainly be designed using X-windows and Motif. We also suspect that the end-user support for reconfiguring the channels may not be very versatile, given the architectural limitations mentioned earlier.

3. A COMPARISON OF CANDIDATE PROCESSORS FOR THE SECOND GENERATION OF SPEECH MULTIPROCESSORS

In this section, we compare several candidate CPUs for incorporation in the second generation of VHSIC multiprocessor. Before we present the actual comparisons, a short review of the multiprocessing requirements for advanced speech processing is included in Section 3.1.

3.1 Advanced Speech Processing and Multiprocessing

The surprisingly large throughput demanded by *real-time* speech processing applications require two features in the processing system:

1. A high processing throughput - the processor(s) must perform the processing at a rate that can provide responses within a reasonable time.

2. The system must have adequate I/O bandwidth. This requirement is dictated by the fact that unless adequate I/O rates are available from the transducers or to the I/O devices, the I/O system can become the bottleneck.

Many speech processing applications that require a real-time response can easily demand a processing throughput in excess of several hundred MFLOPs. Unfortunately, the fastest uniprocessors available today cannot deliver such floating point processing power (in uniprocessor configurations). Table 1 shows the MFLOPs ratings (peak) for some contemporary uniprocessors including signal processing microprocessors. It is to

be emphasized that the numbers in Table 1 correspond to the peak throughputs. The actual sustained throughputs are expected to be significantly lower.

PROCESSOR	PEAK FLOATING-POINT THROUGHPUT
DEC 21064 (Alpha Implementation)	200 MFLOPs
Inmos T9000 Transputer	25 MFLOPs
Analog Devices ADSP 21020	100 MFLOPs
Texas Instruments TMS 320C40	50 MFLOPs (110 MFLOPs Counting Floating-Point Moves)

Table 1. Peak Processor Throughputs

It is clear from the figures given in Table 1 that a single processor cannot sustain the throughput requirements of many real-time speech processing applications: a multiprocessor configuration is thus mandated. The need for a multiprocessing configuration for these system opens up several critical questions - choosing the manner in which the processors interact, the problem of physically connecting the processors, the degree of autonomy in the interactions and the architecture of the I/O subsystem.

3.1.1 Interaction Models

There are two commonly used models for interactions among processors in a parallel/multiprocessing configuration. These are the so-called *shared memory* (SM) model and the *distributed memory* (DM) (aka message-passing paradigm).

In the SM paradigm, processors interact with each other by reading and writing shared variables. The shared variables are implemented within a physically shared address space. Interaction among the processors is kept fast by using a high speed interconnection network (dynamic switches) to the shared memory system or by using a multi-ported shared memory. Since the interaction time among processors in a shared memory multiprocessor can be kept quite low (typically, anywhere from a few tens to at most a few hundred processor cycles), shared memory multiprocessors can accommodate applications that require the processors to interact very often with each other. Consequently, shared memory multiprocessors are often dubbed as tightly-coupled multiprocessors. Obviously, shared memory multiprocessors can also handle applications that do not require frequent interaction among the processors with equal ease.

In the DM model of parallel processing, processor do not share any address space. Instead, processors interact via message passing. The network used for message passing

is usually a point-to-point static network among the processing nodes. (A processing node consists of a processor, its private memory and its network interface.) Since message passing takes several hundreds to a few thousands of processor cycles in typical systems, distributed memory multiprocessors are usually suited for applications that do not require frequent interaction among the processors - i.e., applications that are loosely-coupled.

Speech processing applications can be designed to suit a shared memory configuration, as well as a distributed memory configuration. If a large number of processors are required, the shared memory model, despite its generality and ease of programming, may not be a good choice, since the interconnection network can become very expensive and since the system can be difficult to scale because of the formation of hot spots. If the parallel system requires a few processors, a shared memory model may be preferred, because of its ability to accommodate loosely-coupled as well as tightly-coupled applications. The fact that shared memory systems are relatively easier to program than distributed memory systems can also be decisive in this regard. For systems using a small number of processors, a hybrid system using message passing as well as shared memory can be a good choice since the performance and generality of the shared memory model can coexist with the scalability and easy reconfigurability of the distributed memory systems.

3.1.2 Interconnection Network

A second critical factor in the design of a multiprocessor/parallel processor hinges on the choice of the physical interconnection network among processors. The interconnection network plays a critical role in deciding such things as the latency of the interactions among the processors, the scalability of the system and the ease with which the interconnection topology can be altered.

For shared memory systems, to allow interactions among processors to be fast, the interconnection network should have a low latency. Additionally, it should have the ability to be reconfigured dynamically, since it is usually expensive to dedicate an interconnection path among all pairs of connected entities. This dictates the need to use dynamic interconnection networks. Dynamic interconnection networks can be as simple as a single bus to something as expensive as a crossbar switch; other cost-effective solutions (such as multistage switches) fall between these two extremes. The choice of a dynamic interconnection network is dictated by such things as cost, latency, the number of connection permutations realized and so on.

Another solution for connecting processors in a shared memory configuration is to use multi-ported memory modules. Few real systems use this approach, since the multiported memory devices have a cost that varies almost exponentially with the number of ports.

Distributed memory systems typically use a point-to-point static interconnection network. The relatively large latencies of such interconnections disallow tightly-coupled applications to be run on these systems, as noted above. Typically, the topology of the interconnection allows direct links among only a subset of the nodes. A connection required between an arbitrary pair of nodes will be forced to routed through intermediate nodes if a direct link does not exist between the source and the destination. Several topologies exist for such static interconnection networks, such as linearly-connected topology, rings, meshes of two or more dimensions, trees, hypercubes etc. The choice of a static interconnection network is dictated by such things as

- The number of fixed links per node (aka node degree): this decides the wiring complexity, as well as the interconnection topologies that can be supported. Because of the latter, it also decides the performance of the system.

- The diameter of the network (the maximum number of links to be crossed in going from one node to another). The diameter of the static interconnection network determines the time it takes to broadcast a message from a node to all other nodes.

- The number of interconnection patterns embeddable into the network. A good static interconnection network should be able to subsume the interconnection patterns required by common algorithms. As examples, the parallel implementation of FFT requires a "butterfly" interconnection among processors; speech recognition on a multiprocessor can require a tree interconnection among the processors; many convolutions can benefit from a two-dimensional mesh interconnection etc. If the connection pattern required by the parallel application is not embeddable into the static interconnection pattern, the application will run slowly since the logical link required between two processing nodes will have to be simulated on the existing links.

- The scalability and cost of the network. The scalability of the network is a measure of how easy or difficult it is to add processors (i.e., nodes) to the system.

It is to be pointed out that related to the choice of an interconnection network is the routing mechanism. The network may support store-and-forward (packet) routing, circuit switched routing, wormhole routing or their variation. For a given topology for the interconnection topology, the choice of the routing pattern depends of the typical message size, average propagation distances and the communication latency that can be tolerated.

3.1.3 Communication Support And Autonomy

In a multiprocessor system, raw processor throughput by itself does not determine overall performance. Ideally, the processing node or the processor itself should have the following features:

a. Enough ports to accommodate the connections to the interconnection network.

b. Autonomous controllers that oversee the communication among processors without tying up the processor for the duration of the communication.

Note that these features are important regardless of whether one uses a shared memory model or a distributed memory model. Consider for example, a processor that does not have autonomous communication facilities. When an interprocessor communication is initiated, the processor remains tied up, supervising the communication. In a shared memory system, this supervision can take the form of performing handshakes to the interconnection and waiting till the communication can be initiated. In a distributed memory system, the supervision may involve the allocation of a space in the message buffer, running a routing algorithm to decide which outgoing link is to be used and examining the status of an outgoing link and waiting till the message can be launched. Thus, in either case, the processor is tied up in doing mundane chores related to communication instead of performing actual processing. The situation worsens as the degree of communications in the applications grow.

The number of ports available on a processor also plays a critical role in deciding how good the processor is in a multiprocessing configuration. This is particularly true if a distributed memory architecture is employed. If enough ports are not available on the processing node, then the interconnection network topology will become highly restricted, leading to some potential performance loss because of the inability to embed commonly used interconnection patterns.

If a processor does not incorporate support for autonomous communication or enough communication ports, its usefulness in a multiprocessor system becomes questionable. If the processor lacks these abilities, these can be certainly implemented externally (i.e., off the processor chip) using external logic. However, the off-chip signal propagation delays will invariably ensure that the overall performance will be considerably less than the performance of a processor that incorporates these facilities within the (processor) chip.

3.2 A Framework for Comparing the Candidate CPUs

The nature of the speech processing done on the first generation of the VHSIC speech multiprocessor in the RL/IR directorate at the Rome Labs and the need for a second generation system with enhanced capabilities suggests that only the state-of-the-art components be used to implement the second generation machine. The choice for the processing elements can thus be reduced to a fairly short list:

1. The prototype of the DEC Alpha processor running at 200 MHz. internally. This is the fastest general-purpose microprocessor available at present.

2. The T9000 Transputer, the top-of-the line in the Inmos transputer family.

3. The AD 21000 series of signal processors - this is the signal processor with the highest througput from the AD family. However, its usefulness in a multiprocessing configuration is questionable since it does not possess good connectivity and I/O bandwidth.

4. The TMS 320C40 signal processor, the top-of-the-line from Texas Instruments well-known and long standing 320 line. Although the raw throughput of this processor is lower than that of the 21K, it has several on-chip features that gives it an enormous I/O bandwidth of 320 Mb/sec through a large number of on-chip ports and an on-chip multi-channel DMA co-processor. These features are likely to make it perform considerably better than the AD 21K in a multiprocessing configuration.

In this section, we compare the candidate processors for the second generation of the VHSIC speech processors being developed by the RL/IR directorate. This comparison is based mainly on the following criterion:

- Potential to support multiprocessing
- Facilities for processor interconnection
- Special features in the instruction set to support signal processing
- Availability of parallel compiler/environments and partitioning tools

Other factors besides these have also been considered in this evaluation; these are incorporated within the sections that describe each processor.

In the next section, we present a fairly detailed comparison of the processors that can be used in the second generation of speech multiprocessor for the RL/IR directorate.

3.3 The DEC 21064 (Alpha implementation)

The DEC 21064 represents the prototype implementation of the DEC Alpha architecture. The 21064 is a superscalar, superpipelined architecture clocked internally with a 200 MHz. clock. Figure 2 depicts the major components within the DEC 21064.

The main features of the 21064 are as follows:

- Peak performance of 200 MFLOPs. This is achieved by taking in a 100 MHz. external clock and doubling it internally.

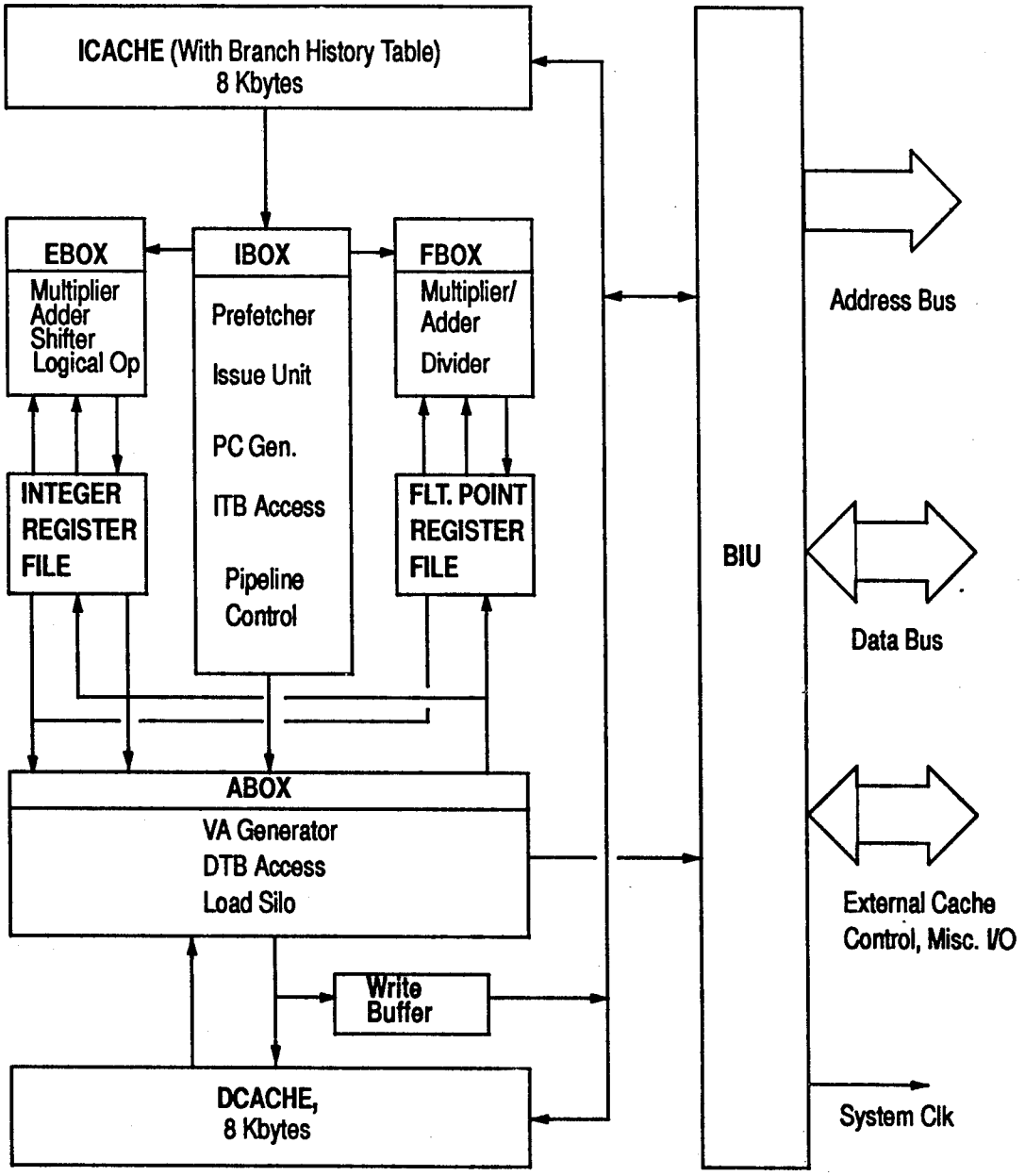

Figure 2. The DEC 21064 (Alpha protitype Implementation)

- Implements IEEE (and DEC) floating point arithmetic.
- Independent floating point and integer registers.
- Substantial on-chip storage in the form of instruction and data caches that are 8Kbytes each. In addition, write buffers are incorporated.
- Support for virtual memory in the form of virtual address translation buffers (TLBs - translation lookaside buffers).
- Single external bus, with support for coherent memory interface in a shared memory configuration.

The 21064 is the fastest general-purpose microprocessor available at present. It, however, lacks many features that can prevent the exploitation of its impressive throughput in multiprocessor-based signal processing applications. The missing features include the availability of addressing modes (modulo addressing, bit reversal etc.) useful for signal processing, the lack of on-chip DMA co-processors and the lack of independent communication ports for processor-to-processor interconnections. The prototype chip also has an unusually high power dissipation (23 to 30 watts!).

3.4 The Inmos T9000 Transputer

The Inmos T9000 Transputer was announced in early 1991 - it is still not available openly at the point of writing this report. The T9000 represents a significant improvement over the T805. Figure 3 depicts a block diagram of the T9000. The T9000 incorporates a 16 Kbyte on-chip data and instruction cache, a superscalar processing unit and the ability to support virtual channels on four physical channels. The main features of the T9000 are as follows:

- Peak throughput of 200 MIPs (with a 50 MHz clock)
- Peak floating point performance of 25 MIPs
- 16 Kbyte cache: organized in 4 banks. Each fully-associative bank consists of 256 entries, each 16 bytes long. The on-chip cache can be also used as on-chip RAM. When used as a RAM, the key field of each entry can be programmed to emulate any address.
- Four inter-transputer links supporting virtual channels.
- Combined link bandwidth of about 70 Mbytes/sec. on all four links
- Virtual links support variable length messages

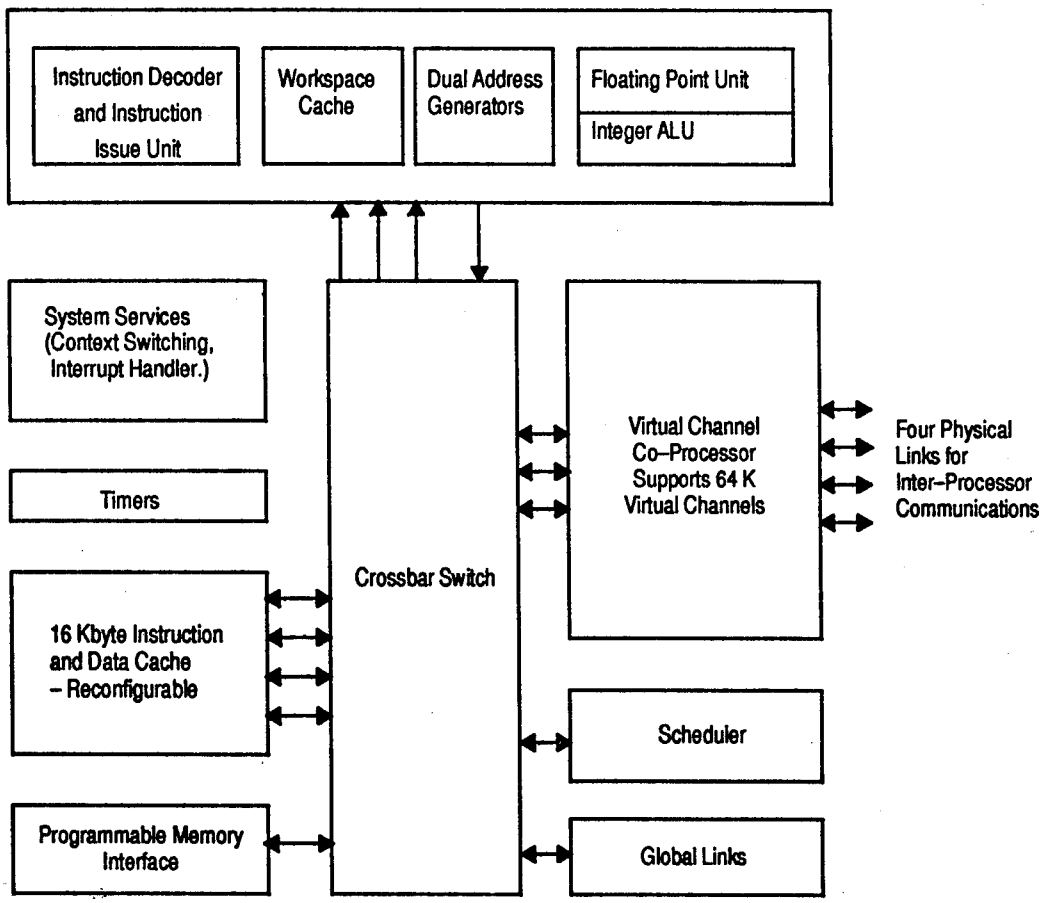

Figure 3. The Inmos T9000 Transputer

- Dedicated hardware for multiplexing virtual channels
- Two global channels to allow system wide data transmission and broadcasts
- I/O Concurrency: four concurrently used virtual channels (on four physical links), two global channels and one external memory access: these can operate concurrently with the computation units.

The availability of dedicated on-chip context switching hardware and hardware supported virtual channels make the T9000 excellent as a building block for distributed memory style systems. However, so far as signal processing applications are concerned, the T9000 shows some fairly serious deficiencies:

1. The peak floating point performance of 25 MIPs per T9000 is well below what signal processing microprocessors provide.
2. No instruction set level (or hardware) support is provided for features that can speed up signal processing applications significantly (such as modulo and bit-reversal addressing, zero overhead looping, fast floating point conversions etc.)

In defence of the T9000, it must however be pointed out that a variety of parallelizing compilers and environments exist for the Transputer family. One would expect that a parallel compiler for the T9000 will be available soon that will exploit the new features in the T9000.

3.5 The Analog Devices ADSP 21020

The ADSP-21000 series of signal processing microprocessor is the latest offering from Analog Devices Inc. This series is a follow up to the earlier 2100 series. The ADSP 21020 is the top-of-the line in the 21000 series and is the only member of the series that includes floating point hardware. The ADSP 21020 is a superscalar machine that uses a 33 MHz. clock. The basic features of the ADSP 21020 are as follows:

- Superscalar CPU with IEEE 32-bit and 40-bit support.
- Dual external busses (two sets of address and data busses)
- Three independent computation units (ALU, Multiplier and Shifter)
- 100 MFLOPs peak performance
- Register file with 16 registers
- On-chip instruction cache (192 bytes, 32 X 6 bytes)
- I/O Concurrency: Instruction fetching plus two concurrent memory accesses through the two external busses.

- Operational concurrency (3 - ALU, shifter, multiplier; other non-compute type of operations can also be in progress concurrently with these)
- Zero-overhead looping, multiply-and-add/subtract instruction
- Automatic sequencing for external addresses through dual address generators for external address busses. A variety of address generation modes are possible (indirect, immediate, bit-reversal, modulo wraparound).

Figure 4 depicts the major components of the ADSP 21020 processor. The superscalar ADSP 21020 uses a three stage fetch-decode-execute pipeline clocked at 30 nsecs (33 MHz. clock). In the superscalar mode, three instructions are issued simultaneously into the pipeline, yielding a peak throughput of 100 MIPs. Since the ALU, multiplier and shifter can be in operation simultaneously, the peak floating point performance is also 100 MFLOPs.

The time taken to do a 1024-point complex FFT on a ADSP 21020 is claimed to be about 0.58 msecs. (It is not clear whether this number includes I/O time.) While this appears impressive when compared with the execution time of the same program on a TMS 320C40 (1.54 msecs.), one must not judge the performance of these processors solely on the basis of these numbers. When it comes to a multiprocessing configuration, the ADSP 21020 shows some fairly serious limitations:

1. The ADSP 21020 is geared only for a shared memory style multiprocessor, since it lacks direct processor-to-processor communication ports. As explained in item 2 (below), the latency of the interactions are increased due to the use of an external shared bus. This essentially means that the multiprocessing mode of operations will not support tightly-coupled applications efficiently. Thus one of the advantages of shared memory multiprocessing is lost.
2. The only facility for processor-to-processor communication will be the two external busses. Consequently, external arbitration logic is needed for access to the bus. The arbitration delays can significantly add to the latency of the transfer.
3. No autonomous DMA co-processor is included in the ADSP 21020 chip - this implies that the processor has to be involved in the I/O operation. The benefits of having a high speed computation unit is thus seriously jeopardized. (A new version of the ADSP is, allegedly, going to incorporate this facility.) *Even if an on-chip DMA co-processor is incorporated, the lack of on-chip RAMs will result in the generation of frequent (internal) interrupts for the processor, since large blocks (such as samples) cannot be moved into the CPU in one step.*
4. Other than a simple test-and-set instruction, no mechanisms are provided for processor synchronization. The external bus arbitration mechanism has to be augmented to provide any additional facilities.

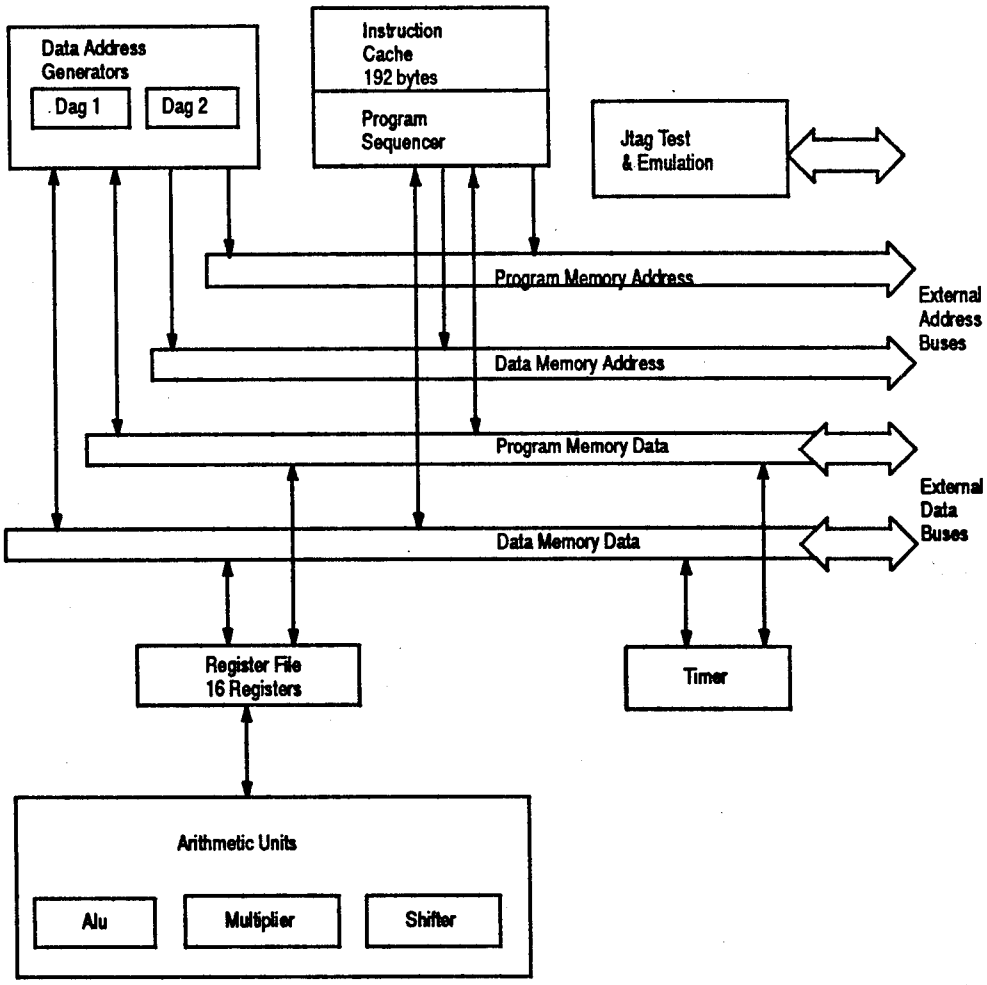

Figure 4. The Analog Devices 21020

5. The use of a bus interconnection severely limits the interconnection topologies that are required; it also limits operational concurrency quite significantly. As an example, consider an application where the processors are to be logically connected to form a tree with a processor making up a node in the tree. The dual shared bus interconnection supported by the ADSP 21020 can logically implement this tree interconnection by using shared memory locations. However, if the application requires the processors to operate in a pipelined fashion, level-by-level, then the shared bus fails to implement the required concurrency. (Note that speech recognition algorithms can make use of a pipelined tree interconnection.)

6. The overall interprocessor communication bandwidth, assuming that the two external busses are run at the processor clock rate and zero arbitration delay), is 11 bytes per 30 nsecs. (since the external bus data widths are 40 and 48 bits). This translates to a total bandwidth of 300 plus Mbytes per second. Although this seems very high and seemingly more than adequate, some things should be kept in mind - the non-zero arbitration delays will result in a useful bandwidth well below this figure. As an example, if the external arbitration delay is 100 nsecs. on the average, then the peak I/O bandwidth is 11 bytes/130 nsecs., i.e., 84 Mbytes/sec. (A 100 nsec. arbitration delay is going to be typical, since the ADSP 21020 uses external arbitration logic.)

Besides these limitations in a multiprocessor configuration, the ADSP 21020 lacks some features (found in the TMS 320C40, for instance), that can enhance its usefulness:

- No on-chip RAMs are available; neither does it have a generous number of registers. Consequently, coefficient tables, weight tables, recognition templates etc. will have to be stored externally, requiring a slow off-chip access.
- The size of the on-chip instruction cache is relatively small - thus instruction cache hit ratios are going to be small, implying a corresponding performance penalty.
- The ADSP 21020 has one timer that has to be shared by the two external bus interface. This can be restrictive for some applications.

Besides the architectural limitations mentioned above, the ADSP series is handicapped by the lack of a parallel compiler or partitioning tools. It also lacks pervasive development systems. The lack of a parallel compiler forces the programmer to write serial code and manually partition them across processors and then manually inserting synchronization primitives. This is an error-prone and tedious process and affects performance and programmer productivity adversely.

3.6 The Texas Instrument TMS 320C40 processor

The Texas Instrument TMS 320C40 is a member of the long standing 320 family of signal processors. The feature that makes the 320C40 unique among processors in its class is

on-chip support for multiprocessing in the form of autonomous DMA controllers and six independent communication ports. The high connectivity that can be obtained using these features make the TMS 320C40 an ideal engine for a signal processing multiprocessor. No other signal processor comes close to the 320C40 in terms of interconnectivity. The TMS 320C40 comes in two versions - one with a 20 MHz clock and another with a 25 MHz. clock. Figure 5 depicts the major components within a TMS 320C40 processor. The salient features of the 320C40 are summarized below:

- The 320C40 has a pipelined superscalar architecture with hardware support for 32-bit and 40-bit IEEE floating point arithmetic.
- There are 32 internal data registers (including 8 registers that are used for extended precision arithmetic).
- The 320C40 has a substantial amount of on-chip storage in the form of two independent 4Kbyte on-chip RAM banks. The on-chip RAM can be used to hold a variety of data, including large coefficient tables or encoding tables. The ability to store such tables on-chip will allow applications such as filters, speech encoding and speech recognition to be speeded up significantly.
- The 320C40 has two independent sets of external address and data busses.
- Six independent communication ports, each with 8-bit wide data lines and 4 control lines. Each port has its own set of input and output FIFO queues and its own protocol controller. The built-in protocol controller in each port allows a 320C40 to be directly connected to another 320C40 without the need for any external logic. Each of the six communication ports have a bandwidth of 20 Mbytes per second. These ports are usable as processor-to-processor interconnection or as connection ports to other devices.
- The TMS 320C40 features a full-fledged on-chip DMA co-processor that can support six simultaneous I/O channels. The DMA co-processor can be used to support channels for internal as well as external-internal memory-to-memory transfers. DMA channels can be set up through the external busses, through the six communication ports, as well as through the busses and the ports. This allows the main processing engine to concentrate on processing while highly parallel I/O operations are in progress. The peak floating point throughput is 50 MFLOPs (110 MFLOPs if floating point load/stores are included) and the peak operation throughput is 275 MOPs.
- The peak I/O bandwidth for a TMS 320C40, using the six communication ports and the two external busses is 320 Mbytes per second - an impressive figure by any standard. *Realized I/O bandwidths are going to be very close to this because of the presence of the DMA co-processor and large on-chip RAMs.*

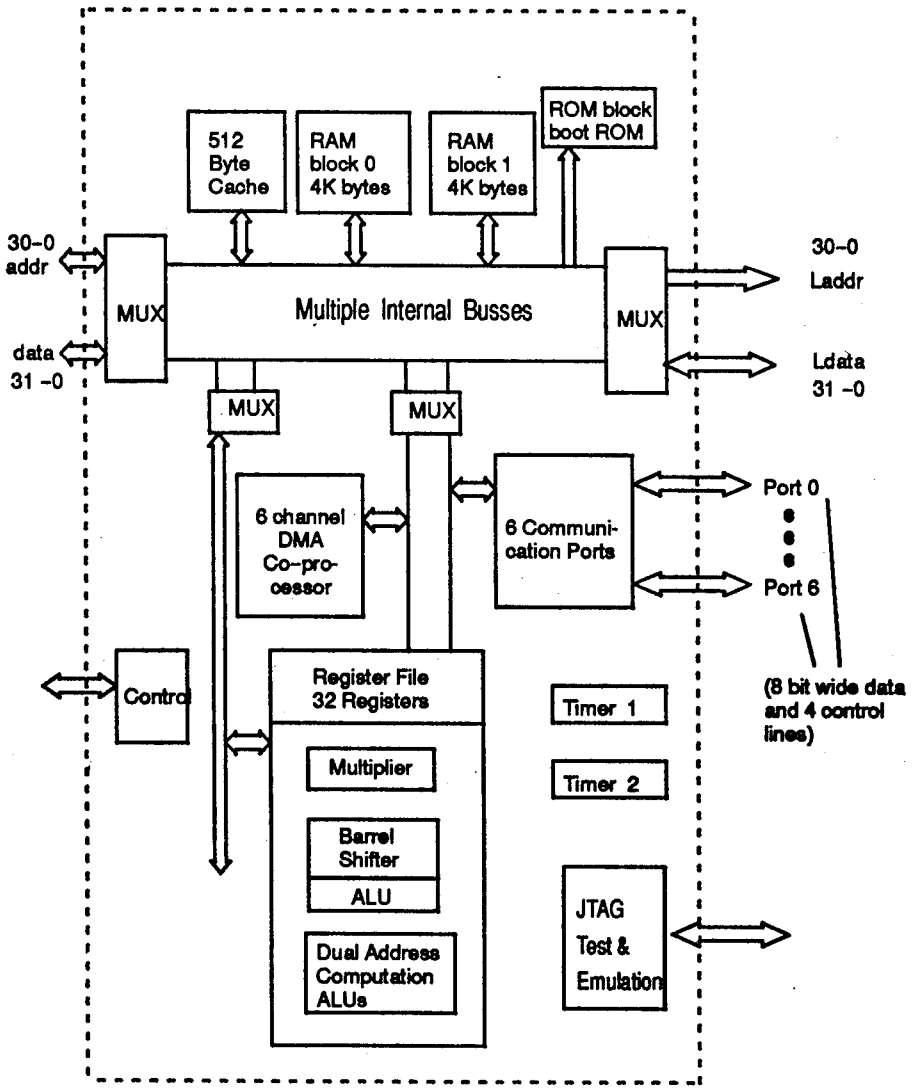

Figure 5. The Texas Instruments TMS 320C40

- The TMS 320C40 incorporates several independent processing sections – these are the integer ALU, a multiplier, a shifter, the DMA co-processor and two independent address computation units. The address computation units can generate a variety of address patterns that are useful in signal processing (such as modulo addressing, bit-reversal etc.).

- A 512-byte on-chip instruction cache is included to speed up the throughput. Fairly large loops can be cached in their entirety in this cache.

In addition to the above features, the 320C40 sports **two** independent timers and single cycle branch, call and return instructions. An on-chip boot ROM allows boot programs to be executed from any of the six communication ports or from either of the two external busses.

The TMS 320C40 does not have the limitations of the ADSP 21020 mentioned in Section 3.3. Published data shows that a 1024-point complex FFT takes 1.54 msecs. Although, this is slow compared to the execution time on an ADSP 21020, a few caveats should be added:

1. The FFT performance does not allude to the real performance, especially in a real-time multiprocessing configuration, where the I/O operations can be a potential bottleneck.

2. The 320C40 processor is more than 2 years old at this point – it is expected – given that the 320C40 is so pervasive – that versions that support higher speed clocks and architectural enhancements will be available soon.

The six communication ports with built-in queues and protocol controllers allow the 320C40 to be usable in a distributed memory multiprocessor configuration and also in a shared memory configuration through the two external busses.

3.7 A Brief Comparison

A brief comparison is warranted here to identify potential candidates for the processors to be used in the second generation of the VHSIC speech processors. Clearly, the T9000 can be ruled out as a candidate on at least the basis of its relatively poor floating point performance of 25 MFLOPs peak. The DEC Alpha prototype (21064) has a very impressive floating point throughput of 200 MFLOPs. It is however, handicapped in several respects. First, its I/O performance – considering what is required in the real-time speech processing domain – is rather limited; it has only one external data bus that can support only a shared memory style multiprocessor configuration. Second, several features useful in speech processing (such as modulo addressing, bit-reversal addressing) are not supported. Third – and perhaps the most serious drawback – is the fact that the Alpha prototype has a 23 Watt (nominal)/ 30 Watts (peak) power requirement,

that can seriously jeopardize its use in a portable unit for aircrafts. Thus, we are left with two contenders - these are compared in some depth in Section 4.

4. THE TMS 320C40 VERSUS THE ADSP 21020 IN ADVANCED SPEECH PROCESSING APPLICATIONS

The applications processed on the current VHSIC speech processor for the RL/IR directorate at the Rome Labs have several characteristics that are worth repeating here for the sake of the comparison between the TMS 320C40 and the ADSP 21020:

a. These applications require a very high floating point throughput.

b. These applications demand a large bandwidth between the data acquisition section(s) and the processing section(s).

c. The inevitably required multiprocessing configurations demand a variety of connections between the CPUs (such as trees, meshes, linear and tree-based pipelines etc.) specific to the applications.

d. Many applications require the use of tables (for coefficients, codes, recognition templates etc).

The ADSP 21020 meets only the first of the four requirements given above. It certainly has a superior floating point performance than the TMS 320C40. However, the ADSP 21020 falls far short of meeting the other three requirements of the VHSIC speech processor. Overall, the TMS 320C40 appears as a clear winner - its floating point performance is commensurate with its I/O bandwidth and connectivity. The presence of a full-fledged DMA co-processor within the TMS 320C40 allows the processor to concentrate on processing without the need to stay in the I/O loop. The TMS 320C40 is thus more likely to exploit is floating point performance in the applications it runs. The presence of the six independent communication ports allow the TMS 320C40 to be configured in a wide variety of interconnection topology. The ports on the 320C40 can also bring in the boot code, obviating the need for dedicated booting logic/hardware (such as the Transputer used in the first generation VHSIC system) or interconnections. If processor-to-processor interconnections use the six communication ports, the dual external busses are kept free to load data into the sizeable on-chip RAMs. The dual 4Kbyte on-chip RAMs in the TMS 320C40 allow data, especially tables of code or coefficients and recognition templates to be stored on-chip and accessible at a very high speed. However, applications need to be run on a scaled down prototype or a development system to allow these performance potentials of the TMS 320C40 to be substantiated.

The TMS 320C40 has another advantage over the ADSP 21020 system that can make a substantial difference for the end users. Parallel compilers and parallel program development environments are available for the TMS 320C40 from a number of vendors.

At the present moment, such tools are not available for the ADSP 21020. Programmers writing applications for a multiprocessor based on the ADSP 21020 processor will therefore have to parallelize the program, partition the data over the processors and fine tune the parallel application by hand. The complex process of parallel debugging will force users to eventually end up writing a parallel debugger, since parallel programming environments are not available for the ADSP 21020.

To summarize, it appears that the TMS 320C40 is a better choice for the computing engine for the second generation of the speech processors for the RL/IR directorate for the following reasons:

- The use of the TMS 320C40 processors result in reduced parts count and overall power reduction. This is because the TMS 320C40 has an autonomous DMA controller, six CPU-CPU connection ports (that do not require *any* external logic for interconnection) with built-in FIFO buffers in each direction , 8 Kbytes of on-chip RAM and a large On-chip instruction cache. Further, spare communication ports can be used to bring in the boot code, obviating the need for a Transputer (as used in the current VHSIC processor).

- Processor boards can have an uniform structure because of the ability to use the hypercube interconnection among the CPUs. (See proposed design in Section 5).

- *The ADSP 21020 will be seriously handicapped when its has to transfer the 512 bits (or larger) samples from the DACU. This is true even if the new version of the 21020 includes a DMA channel.* Because the on-chip storage on the ADSP 21020 is restricted to just 16 registers, no more than 16 words (=96 bytes) can be transferred at any time by the on-chip DMA logic. Thus several I/O transactions will be needed to bring in just one sample from the DACU. *This problem is completely absent in the TMS 320C40.* In fact, because the two 4 Kbyte on-chip RAM banks on the TMS 320C40 are independent, a DACU sample can be loaded into one RAM bank, while processing and output-ing of the results for the previous sample is going on using the data stored in the other bank.

- Parallel compilers and parallel programming environments are readily available, making it convenient for the end users to write code.

- The I/O and processor-to-processor interconnection bandwidth is commensurate with processor throughput in the TMS 320C40. This and the existence of autonomous DMA coprocessors allow the TMS 320C40 to be an ideal engine for the speech multiprocessor.

- The spare ports of the TMS 320C40 CPUs on a processing board can be used to replace the Transputer used in the VHSIC multiprocessor prototype.

- The TMS 320C40 is supported by numerous third party vendors. Single board components as well as software is available from a variety of sources.

In the next section, we sketch out the design of the processing board for a prototype multiprocessor system using the TMS 320C40.

5. A POSSIBLE DESIGN FOR A TMS 320C40 BASED MULTIPROCESSOR PROTOTYPE

In this section, we briefly outline the design of the main processing boards for the second generation of the VHSIC speech processor based on the TMS 320C40s. The proposed design has the following features:

1. Every processing board has an uniform structure, allowing interoperability with ease.

2. Each processing board incorporates four TMS 320C40s that are connected in a two dimensional hypercube topology locally on each bus. Additional connections are provided from each processor on a board to the backplane to allow the eight processors on two neighboring boards to be connected in a three dimensional hypercube topology. Interconnection topologies demanded by the speech applications (such as trees, meshes, linear and pipelines etc.) are embedded within the hypercube topology and can be used readily without the need to reconfigure the interconnection. The applications will be designed to use the embedded interconnection pattern that is the most advantageous. *Most certainly, this design can be extended to support 8 CPUs to each board.*

3. The design, because of the homogeneous design of the boards, the partitionability of hypercube interconnection and because of the autonomy of the six ports on each CPU, *allows a channel to be reconfigured while the other channels remain operational.*

4. No master-slave relationship exists among the processors - all processors see an identical I/O structure and have equal access to all resources on the board.

5. Since processor-to-processor interactions use only three of the six communication ports (a three dimension hypercube ties up three ports on each TMS 320C40), the dual external busses are used as follows:

   - One external bus is used to access the local memory on each board.

   - Another external bus forms a global bus common to all boards. This global bus is part of the system backplane.

Arbitration logic for both busses can be incorporated on each board. Each bus will have a throughput of 100 MBytes per second. A second global bus, interfacing to each local bus on a board can be added.

6. The three unused communication ports on each TMS 320C40 can be used for a variety of things:

   - as a diagnostic port
   - as a serial global port
   - as a port carrying re-configuration information, with the TMS 320C40 dasiy-chained. (The daisy chaining requires two ports on each TMS 320C40.)
   - as ports that can configure the processors on two adjacent boards into a four dimensional cube etc.

Figure 6 depicts the main components in the proposed design on each board.

The proposed board structure will have the following characteristics:

- A power requirement of 2 watts (@ 5V) (nominal) for each TMS 320C40. The external on-board RAMs, bus arbitration and reconfiguration logic will require another 4 watts. This translates to an overall power requirement of 10-15 watts per board.

- Each board will require a minimum of 122 signal lines on the backplane (74 for the global bus and 12 for each of the four communication ports that connect eight CPUs on two neighboring boards). If the optional second global bus is incorporated, the signal line requirement jumps to 196. If additional lines for the communication ports are bought out, this number will be higher.

- The peak processing power available on each board is 4*275 = 1100 MOPs per seconds.

- The peak floating point performance available on each board is 4*50 = 200 MFLOPs. If floating point loads and stores are counted as contributors to floating point throughput, the peak floating point power from each board is 440 MFLOPs.

- Each CPU on a board is connected to three other CPUs - two on the same board and a third one on a neighboring board. The processor-to-processor communication bandwidth in this configuration is 60 Mbytes per second from *each processor*. Additionally, the local external memory bus can be used to implement shared memory style communication at 100 Mbytes per second between any two CPUs on the same board.

- Unused communication ports can be used to replicate links among processors on the same board to improve the inter-processor communication rate. In particular, this

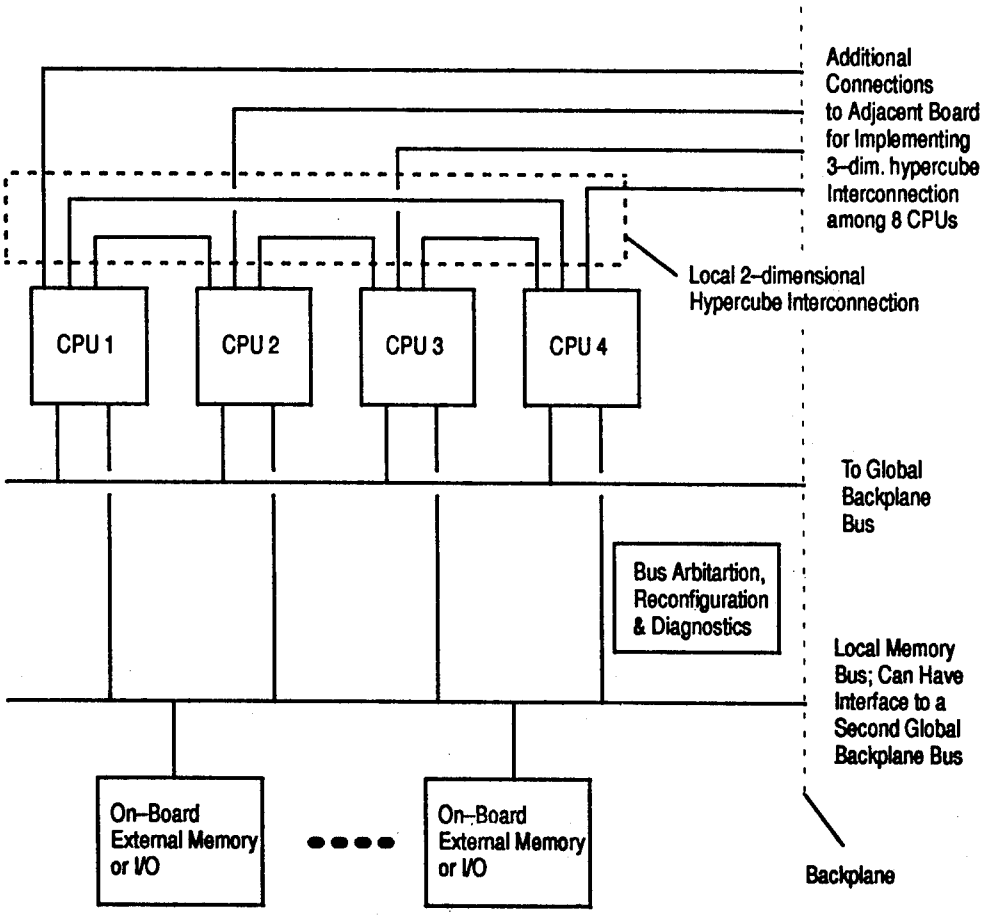

Figure 6. Components in the Proposed Processing Boards

will allow each CPU to play the role of the Transputer on the current VHSIC APU boards – a Transputer will thus not be required on each board, reducing the component count, power requirements and enhancing reliability. *None of the flexibility of the Transputer will be lost in this process.*

Note that the proposed design based on the TMS 320C40 uses distributed memory style interaction among the processors through the hypercube interconnection. In addition, shared memory style interaction is supported through the busses. Both flavors of multiprocessing are thus incorporated in the proposed design.

6. SUMMARY AND RECOMMENDATIONS

The first generation of VHSIC multiprocessors were critically evaluated in this report and a number of problems in these systems, some fairly serious, have been identified. A comparative study was also performed among several candidate CPUs to identify a potential CPU for use in the second generation machines. An initial design for the main processing boards based on the CPU selected was also presented to reveal the potential advantages of using that CPU and the advantages in the proposed design.

Based on the problems that we have identified in the first generation of the VHSIC speech multiprocessors, we make the following recommendations pertaining to the design of the second generation of these systems:

- The CPUs should be in a symmetric configuration – in particular, the master–slave configuration of CPU pairs should be abolished. The use of a symmetric configuration should ease the performance and programming problems cited in Section 2.3.
- The logic for each channel in the system should be logically and physically independent to allow the reconfiguration of one channel while the other channels are operational.
- To derive meaningful performance from the higher throughput CPUs, the inter–processor communication structure has to be significantly upgraded. In particular, the use of the first–generation Transputer links to interconnect the CPUs in the desired topologies, such as hypercube, mesh, quadtrees etc. should be abandoned.
- The use of the relatively slower Transputers for implementing the arbitration protocols of the global (external) DMA channels should be abandoned. It may be better to implement the channel arbitration protocols in dedicated random logic (or a field programmable logic array).
- A full–fledeged parallel compiler and program development environment should be provided for the new generation of the speech multiprocessors.
- A more user–friendly user interface (such as one based on X–windows/Motif) should be incorporated.

Our recommendations are to use the TMS320C40 CPUs in the second generation of machines, since they allow all of the above recommendations, as well as the programming and performance requirements to be easily met. The use of the TMS320C40 also obviates the need for including Transputers and external logic for inter–processor and global connections. (Please see Section 4 for detailed justifications.)

*U.S. GOVERNMENT PRINTING OFFICE: 1996–710–126–20141

MISSION

OF

ROME LABORATORY

Mission. The mission of Rome Laboratory is to advance the science and technologies of command, control, communications and intelligence and to transition them into systems to meet customer needs. To achieve this, Rome Lab:

a. Conducts vigorous research, development and test programs in all applicable technologies;

b. Transitions technology to current and future systems to improve operational capability, readiness, and supportability;

c. Provides a full range of technical support to Air Force Materiel Command product centers and other Air Force organizations;

d. Promotes transfer of technology to the private sector;

e. Maintains leading edge technological expertise in the areas of surveillance, communications, command and control, intelligence, reliability science, electro-magnetic technology, photonics, signal processing, and computational science.

The thrust areas of technical competence include: Surveillance, Communications, Command and Control, Intelligence, Signal Processing, Computer Science and Technology, Electromagnetic Technology, Photonics and Reliability Sciences.