BACKGROUND

This report is a description of the hardware design, construction and operation of a parallel computation system made up of circuit boards containing several Digital Signal Processors (DSPs). The computation system operates as an attached processor to a host computer which in turn is accessed via the UNIX operating environment. The system is called the Ring Array Processor (RAP).

The RAP was envisioned as a system for supporting our research in continuous speech recognition algorithms [1]. It is designed to efficiently use the multiply-accumulate (MAC) operations of several DSPs in parallel along with a high-performance interprocessor communication channel to achieve top performance for the algorithms of interest. The particular algorithms used in the speech work are based on artificial neural networks and map efficiently to the RAP architecture, as described in [2]. Additional reports describing the software and user interface of the system are available in [3][4]. This work has also appeared in [5].

HARDWARE DESIGN GOALS

Embarking on a design project the size of the RAP first required consideration of several issues. One particular concern affecting the hardware design was how the RAP might be realized quickly with a small engineering team and without sacrificing the overall goal of 100x the performance of a Sun Sparcstation 1, for a four board system. As the system architecture evolved, this concern was translated into several concrete decisions:

1. Only standard, commercially available components are used. (The logic families represented on the RAP include bipolar TTL, CMOS, BiCMOS, ECL and GaAs.)

2. The computational power of the system comes from using several high-performance processors, not from complex interconnection schemes or sophisticated peripheral circuits. To that end, the processing nodes are kept simple to allow the maximum number (4) per circuit board.

3. The memory system uses only a single bank of dynamic RAM (DRAM) and static RAM (SRAM) at each processing node. This decision allows greatly simplified memory control logic and provides the minimum electrical loading of the processor data lines.

4. The backplane bus chosen is a standard VME bus using only the 32 address and data lines on the primary interface. The clock signals and communication channel needed by the RAP use separate connectors and cables at the front panel in order to maintain strict compatibility with the VME bus.

5. As much of the logic as possible is implemented in Programmable Gate Arrays (PGA[1]). This decision reduces parts count, simplifies circuit board design, and allows flexibility for later design enhancements. PGAs are used for two functions at each node: the interprocessor communication channel, and the memory control for DRAM. One additional PGA is used in the VME bus interface.

6. The RAP is all-digital, and does not directly support analog I/O. By using only digital circuits, we simplify board layout and sidestep choices of the particular analog devices required. Analog I/O can be added later without redesigning the RAP by using a pair of

[1] PGA is a registered trademark of Xilinx Incorporated.

connectors that are provided for an add-on board which connects to the DSP serial ports. These high-speed ports can support data conversion rates (A/D or D/A converters) as high as 1 MB/second.

RAP ARCHITECTURE

The RAP is an array of processing nodes connected in a circle network via a synchronous, unidirectional communication channel called a Layer Distribution Ring (LDR). The name LDR is taken from its use in multilayer perceptron algorithms where all of the processors simultaneously perform the calculations of a single *layer* of an artificial neural network. When the layer calculations are complete, the results are distributed to all of the other nodes over the LDR. The number of processing nodes in a RAP system is a multiple of 4, depending on the number of boards used. The architecture of the RAP system (for N nodes) is shown in Figure 1.

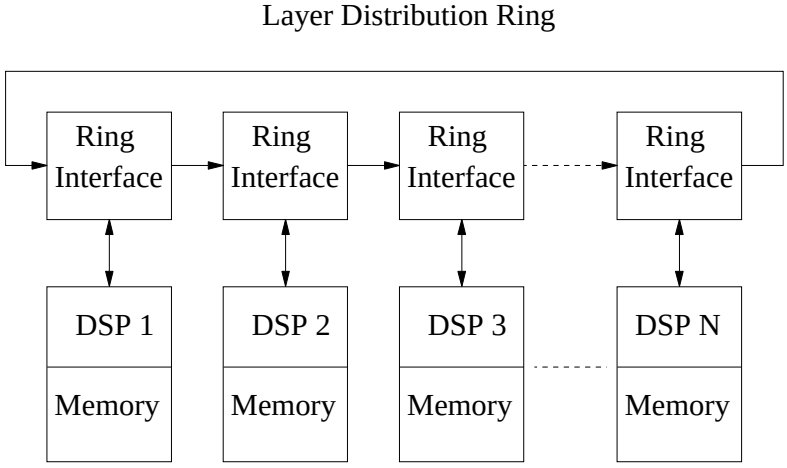

Figure 1. Ring Array Processor Architecture

Each RAP *node* consists of a DSP chip, the local memory associated with that chip, connections (the LDR internode bus) to other nodes, and miscellaneous control logic. Figure 2 shows additional detail of a single processing node. Four interconnected nodes are contained on a single circuit *board*, with the internode busses also brought to connectors at the board front edge. At the next higher level, several boards are plugged into a common backplane *bus* (VME) with all LDR busses connected using flat ribbon cable.

A reasonable maximum expected *system* configuration is 16 boards, giving a total of 64 nodes. Such a 16 board system would have a peak rating of 2048 MFLOPs and a peak interprocessor communication bandwidth of 4096 MBytes/second.

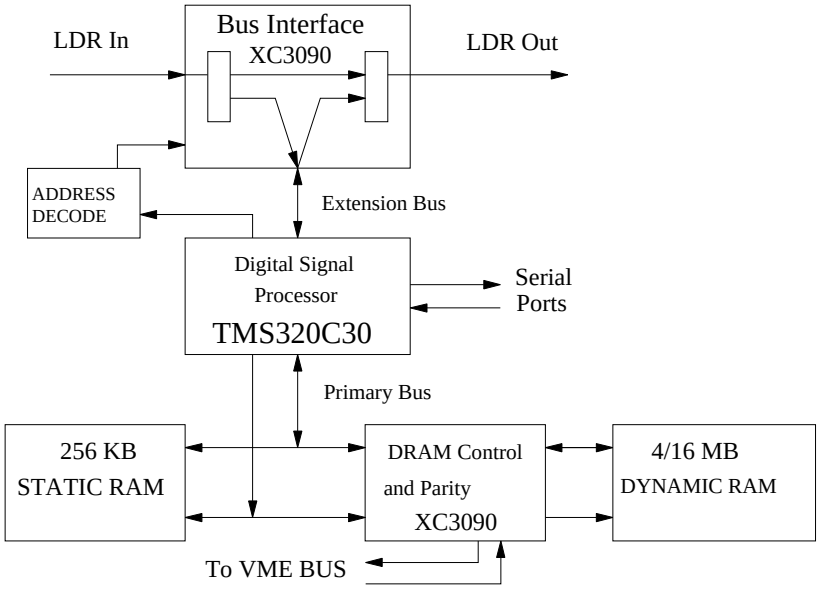

Figure 2. RAP Node Block Diagram

Figure 3 indicates how a four board RAP with 16 processing nodes connects to a local disk, additional memory and a host processor in a standard VME backplane. The user interface to the RAP is through a workstation connected over the Ethernet.

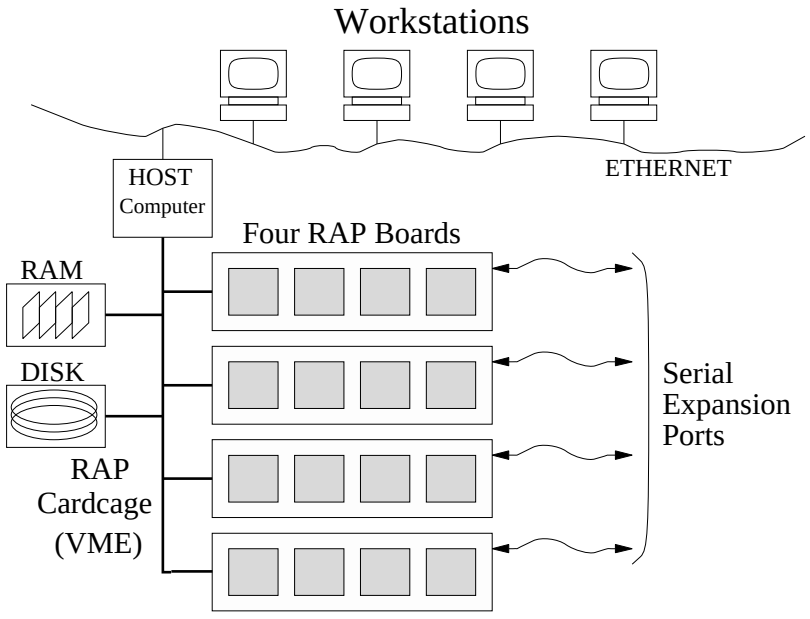

Figure 3. RAP System Block Diagram

HARDWARE DETAILS

The major hardware design subsystems for the RAP are presented in the following sections. These subsystems include: DSP, memory, LDR and expansion options using the serial ports. The technique used for clock distribution is also described.

DSP Selection

Digital signal processors are specifically designed for high throughput on problems that primarily use multiplication and addition operations. For our requirements, these operations must use a 32-bit floating point format. Additionally, the items in the following list were considered:

1. Performance in MFLOPs (Million of FLOating Point operations per second).
2. Language support (both assembler and C) with development and debug utilities.
3. Level of design support from manufacturer, including software simulator.
4. Cost and availability.
5. Accessibility of DSP via special ports or instructions for debugging.
6. Simplicity of internal execution pipeline and suitability for assembly language programming.

Because of the floating point requirement, very few DSPs merited serious consideration. Although additional DSPs have been introduced since the RAP design was complete, the Texas Instruments TMS 320C30 best met our criteria at the time, and was selected. Table 1 lists the most important features of the TMS 320C30 for our application.

Table 1. TMS 320C30 Features

Performance:	16 MIPs/32 MFLOPs
Address Range:	64 MB
On-chip RAM/I-Cache:	8 KB/256 Bytes
Secondary External Bus:	32-Bit Data; 13-Bit Address
Fastest RAM Required:	35 ns
Serial Ports/Speed:	Two: 8 Mb/sec each
Additional On-chip Peripherals:	Complete DMA 2x32-Bit Timer/Counters
Integer Multiply/Barrel Shifter:	24 x 24-Bit/32-Bit
Registers:	8 x 40-Bit FP/Int 8 x 32-Bit General 12 x 32-Bit Control
Unusual Addressing Modes:	Bit Reversed PC-Relative Circular Buffer
DO/LOOP Instruction Overhead:	4 Cycles

Memory Subsystem

The memory system design takes advantage of different technologies, with consideration for the speed, costs and board area involved. The resulting memory architecture includes three hierarchical levels within the RAP, plus additional levels located on other VME-compatible memory and offline storage devices. The three levels of memory system organization within the RAP are:

- 8 KB of on-chip memory for each DSP
- 64 KB of fast static RAM for each node
- 4 MB or 16 MB of dynamic RAM for each node

The memory hierarchy is identical on all RAP nodes, and the memory is local to a node. All of this memory, described in detail below, occupies at most slightly over one-quarter of a DSP's 64 MB address space. The SRAM and DRAM are accessible to the host processor over the VME bus.

Tables 2 and 3 show the DSP memory map for a single node. All addresses are word (32-bit) addresses in the 24-bit DSP address space. Byte and halfword addressability are not supported by the TMS 320C30. The DRAM design allows usage of either 1 megabit or 4 megabit chips, and the two center columns in Table 2 show the maps for both choices. The timing column indicates if wait states[2] are added by the DSP to account for slow memory accesses.

Table 2. Node Memory Map

Address	w/1Mb DRAMs	w/4Mb DRAMs	Timing
00 0000	Page 0 SRAM	Page 0 SRAM	0 Waits
01 0000	- unused -	- unused -	
10 0000	|||	|||	
20 0000	|||	|||	
30 0000	- unused -	- unused -	
40 0000	4 MB DRAM		Three
50 0000		16 MB	Wait
60 0000	- unused -	DRAM	States
70 0000			
80 0000	I/O, LDR Bus & Internal	I/O, LDR Bus & Internal	Various
81 0000	- unused -	- unused -	
|| ||||	|||	|||	
F0 0000	- unused -	- unused -	

[2] Each wait state adds an additional 62.5 ns delay.

Table 3 shows more detail for the area starting at address 80 0000 (Page 80). Since this section of the memory map is not dependent on the type of DRAM used, the 1 Mb and 4 Mb cases are not shown separately.

Table 3. I/O, LDR Bus & Internal Memory Map

Address	Memory	Timing
80 0000 80 1000	Expansion Memory: LDR Bus	LDR Bus Linked[3]
80 2000 80 3000	- Reserved -	
80 4000 80 5000	Expansion I/O: Control Registers	VME Linked
80 6000 80 7000	- Reserved -	
80 8000 80 9000	On-Chip Peripherals & 2 KW Memory	0 Wait
80 A000	- unused -	
|| ||||	| | |	
80 F000	- unused -	

On-Chip RAM and Registers. The TMS 320C30 contains two banks of 1024 words each of read-write memory. (8192 bytes total) In addition, there are 28 registers available internally for address calculations, I/O interface, arithmetic operations and other temporary storage. Each internal memory bank operates at the same speed as the DSP. Because there are separate address and data busses for the two banks, the DSP can actually access internal RAM three times during one cycle.[4] Generally, critical execution loops and frequently used data will be put in the on-chip RAM to allow for maximum overall performance.

To help sustain this performance, the TMS 320C30 contains a 64 word instruction cache. The cache is two bank set associative, with one entry of 128 bytes (32 words) in each set. A cache miss on an instruction fetch causes a normal memory access to occur, with the resulting instruction also being written to the cache. The logic for determining which set to use relies on a Least Recently Used (LRU) register stack within the chip.

Fast SRAM. For the next higher memory level, fast static RAMs are used, allowing the DSP to operate at full speed (without wait states). The architecture of the TMS 320C30 allows only a single access to the main memory bus per cycle, so instructions that need more than one SRAM access will automatically take additional cycles. The SRAM array for each node requires nine chips, each 64 Kb by 4 bits, for a total per node of 256 KB of data. One of the nine chips is used to support byte parity.

[3] For certain parallel operations, the LDR will operate at 0 wait states.

[4] One instruction fetch and two data accesses.

To allow the DSP to run at full speed without unnecessary delays, the SRAM array consists of only a single bank of parts. In other words, each data line from the DSP connects to the minimum number of devices: one SRAM, one DRAM, and a data bus buffer for VME connections. This simplification eliminates the delays associated with decoding addresses for selection of multiple RAM banks, and minimizes the timing delays associated with multiple connections to the data lines. The number of SRAM load connections for each address line is nine.

Node DRAM. Each node may include either 4 or 16 MB of DRAM, depending on whether 1 Mb or 4 Mb chips are used. Operation of each node's DRAM controller is independent, and does not require synchronization between nodes. Control for the node DRAM is incorporated in a single PGA which will later include error detection and correction logic. Due to internal constraints on the controller design, nine additional DRAMs are required to store the check bits for the 32-bit word, instead of the theoretical six additional chips. The error correction method used will correct single-bit errors, and flag all double bit errors.

The DRAM cycle time in the RAP is 250 ns, necessitating the addition of three wait states for each access. An additional one cycle wait is added whenever access is switched from DRAM to SRAM or vice versa.

VME Bus Memory & Offline Storage. When problems are formulated that require memory beyond that described above, commercial memory boards for the VME backplane can be added. Such boards have the advantages of large capacity and competitive pricing. The usage of this memory would be under the control of the host processor and not be accessible directly from the RAP. Offline storage, in the form of a disk, is also desirable to connect through the VME bus. Using a local disk would considerably speed up loading input training patterns at the beginning of a run when compared to moving the same data over the Ethernet.

Memory Implementation. Control for the memory system at each node is implemented using a PAL[5] and a Xilinx XC3090 PGA. The PGA is referred to as "MEM" and performs the following functions:

- Multiplexing of the address lines and generation of the address strobes as required for the DRAMs.
- Controlling VME access and data buffering to the node memory.
- Generating and checking parity for the SRAMs.
- Performing error checking and correction (ECC) for the DRAMs. (currently unimplemented)
- Generating refresh cycles required by the DRAM.

The primary function of the memory control PAL is to generate the READY signal required by the DSP. According to the specifications for the DSP, the READY must be valid 9 ns after the address is available in order to avoid adding a wait state. A 7.5 ns 16L8 PAL is required to meet this specification.

[5] PAL is a registered trademark of Advanced Micro Devices.

Layer Distribution Ring

In evaluating alternative approaches to obtaining high performance in a parallel processor, internode communication was found to be a common bottleneck. Realizing that a custom high speed bus was required, initial work focused on a comparison between broadcast and ring topologies. Further examination revealed serious technical problems with implementing a hardware-level broadcast capability that would support a bandwidth commensurate with the DSP performance. For designs that includes separate physical connections (wires) for the internode communication, the cabling problems are formidable for small systems, and impossible for large ones. The traditional broadcast topology, a single shared communication bus, loses significant performance due to the bus arbitration process, and also suffers from engineering constraints in large implementations.

Closer analysis of the algorithms needed for a layered feedforward network with back propagation using multiple processors suggested that the pipelined ring topology would be a satisfactory compromise between implementation difficulty and overall performance [2]. After the activation levels for a layer are computed in the forward direction, each node's output must be distributed to all of the other nodes for use during the next layer calculations. Later, during the back propagation phase of the algorithm, an individual node does not have all of the parameters (partial sums) required to complete an error calculation, and must send intermediate results to all other processors for additional computation.

These two requirements are quite different, with the forward direction suggesting a broadcast mechanism, and the back propagation phase more favorable to a pipeline ring architecture. Ultimately, engineering considerations prevailed and an augmented pipeline ring was adopted with features to maximize "broadcast" performance. The final design includes a pair of 32-bit unidirectional bus connections (input and output) at each node to the two nearest neighbor nodes. These bus connections include pipeline registers, and are connected together to form the LDR.

For the algorithms the RAP was designed to solve, the LDR architecture offers the following advantages over a bus-based broadcast mechanism:

- Access and handshake control is simplified with the LDR. Since each node is only connected to its two neighbors, there is no need for bus requests and no requirement for global handshaking.
- For very regular problems, all DSPs are doing the same thing at the same time, (SIMD-style) simplifying debugging. With a broadcast bus, one DSP is talking, and the other 15 are listening.
- In a hardware realization, the LDR is potentially faster than a broadcast bus because the interconnections are both local and unidirectional.

The LDR is implemented with two chips at each node: a Xilinx XC3090 PGA and a fast PAL . The PGA includes three 32-bit data bus connections (LDR in, LDR out and DSP), a pair of 32-bit pipeline registers, address and function decode logic, and connections to the PAL. The handshaking between nodes, along with READY signal generation for the DSP, are handled by the fast PAL.

Some of the design features of the LDR are:

1. Bus transfer time for one word is the same as one DSP instruction time.

2. Handshaking between nodes is hardware interlocked to prevent illegal reads or data overruns. This feature provides hardware synchronization that encourages SIMD-style coding for maximum overall efficiency.

3. The LDR is accessed from the TMS 320C30 as a small number of memory locations (I/O mapped), with the address indicating the function desired.

4. Only 32-bit data word transfers are supported on the internode bus. Eight additional bits of *tag* are included for future experiments.

5. Bus transfers are initiated only by a DSP; no other bus control hardware is included. However, the DMA controller inside the DSP can be programmed to use the LDR independently from the main code stream.

6. The handshaking method between the bus interface and the DSP is changeable to accommodate other transfer models.

LDR Performance. In our design, the TMS 320C30 executes one instruction in 62.5 ns (16 MHz clock). However, the number of cycles required to access an external memory location can vary (even with zero wait state RAM) from one to three cycles, depending on the order of read and write operations. The worst performance occurs when reads and writes alternate on successive cycles. The best performance is attained when consecutive READs are executed.

To maximize performance when accessing the LDR, we devised a technique that allows distributing the outputs to all the other nodes efficiently using a sequence of special "READ-SHIFT" instructions. This instruction uses a READ LDR operation with a special address as a gating function which transfers the same data from the input to the output of an LDR node. The sequence is:

1. Each DSP WRITEs its first output to the LDR.

2. Each DSP node performs a READ-SHIFT operation, which reads the LDR and places the data in local memory. (For maximum performance, the destination should *not* be DRAM.) The data is automatically shifted to the output register of the LDR node upon completion of the READ-SHIFT instruction.

3. Step 2 is repeated for a total of $N-2$ times, where N is the number of nodes in the system. By repeating an operation that is just a READ (to the DSP), no additional delays are added, and each transfer takes place in one cycle.

4. Finally, each DSP node executes a READ instruction for the last operation of the sequence. This does not do a shift, and instead "removes" the data from the LDR and clears the handshake signals.

The actual data transfer time of the LDR is slightly less than one clock cycle, but it is overlapped with the DSP operation to appear to take zero time. In practice, this means data written by one node can be read by the node on the output side during the very next cycle. When the pipeline is full and all the DSPs are doing read-shifts, the peak bandwidth of the LDR is 16 MW/sec per node. The timing constraints of the TMS 320C30, however, add a three cycle overhead penalty to this method as the DSP changes from WRITEs to READs.

For a 16 node system, adding in the three cycle overhead means that 15 words will be distributed (broadcast) to 16 nodes in 19 cycles. (The 16^{th} word does not need to be distributed.) This provides an effective cost of less than 1.3 cycles per unit broadcast. Stated another way, the aggregate communication bandwidth for a 16 node system is 202 MW/sec, or 808 MB/sec. Overall system performance is a complicated function involving DSP speed, memory allocation, partitioning of the parallel operations between nodes and memory system speed. For more detail on performance for our neural network algorithms, see [2].

LDR Implementation. Handshaking between pairs of DSPs is hardware interlocked to prevent data overruns. If for any reason one DSP is out of synchronization with the others, this interlock mechanism will stall the fast DSPs until the slow ones catch up. To minimize this interlock bottleneck, there are two pipeline registers between each pair of nodes, and both registers are examined by the handshaking hardware.

When a DSP initiates an LDR READ, at least one of the pipeline registers on its input side must contain data if wait states are to be avoided. If both registers are full, the DSP can read both words on successive cycles, with the handshaking and data transfer taking place automatically between registers. The situation is analogous for WRITE instructions, with the handshake hardware monitoring the condition of the two output pipeline registers. The READ-SHIFT operation utilizes both the READ and WRITE handshaking functions, actually monitoring four pipeline registers.

Within one RAP board, the LDR is permanently connected as three unidirectional links between the four nodes. The fourth and fifth logical connections from the "end" nodes go off-board, linking multiples of four nodes with flat ribbon cables. The number of nodes in a system using the LDR is determined by this exterior cabling, and will always be a multiple of four.

All of the data path and most of the control logic for implementing one node of the LDR connection is contained in the Xilinx PGA. This device has 144 I/O pins, 928 internal flip flops and 320 logic blocks available for user customizing. Although our application requires less than 10% of the internal logic, the high I/O pin count justifies using the PGA to reduce board wiring and area.

In addition to the LDR connection function, the XC3090 contains a serial interface to the VME bus interface logic for communication with the host processor. This connection was designed to support up to eight 32-bit registers per node used for passing messages between the host and the DSP. In the prototype system, these registers have not been implemented.

One feature of the PGA is that configuration programming is done at power up time, and upon demand thereafter. Programming is accomplished by shifting a configuration bit stream into the PGA, which is then stored in internal static RAM. The configuration process is handled by the controlling host processor over the VME bus. This configuration capability has been valuable during the initial bringup of the prototype board by allowing the LDR functions to be partitioned and debugged one at a time. We also expect to take advantage of the configurability in later experiments with different communication protocols on the LDR.

The handshake control of the LDR logic is implemented with a GaAs 22V10 PAL at each node. The primary function of the PAL is to generate the READY signal required by the DSP based upon decoding the address lines. In addition, the PAL contains two simple state machines that generate the high-speed handshake signals for transferring the data between the nodes.

Serial Ports

Each TMS 320C30 includes a pair of bidirectional serial ports which are currently unused in our application. These ports are fully synchronous up to 8 Mb/second, with selectable word length up to 32 bits. In addition, the DMA controller in the TMS 320C30 can supervise transfers between the serial ports and memory under interrupt control.

To provide for user expansion, these serial ports (two per node) are brought to connectors at the edge of each board for later use. The connectors allow for either attaching a cable or plugging in a daughter board parallel to the RAP board. A clear area of approximately 3"x7" is reserved on the RAP for the daughter board to occupy. Texas Instruments and others offer a variety of analog-to-digital and digital-to-analog converters designed for a serial interface with a minimum of additional logic. With a 16-bit converter, each serial line could support an A/D or D/A sample rate of nearly 500 ksamples/sec.

Clock Distribution

The hardware-interlocked handshaking used in the LDR requires that all nodes be clocked synchronously. Excessive clock skew between nodes has the potential of causing the handshake signals and data to get out of sync, with disastrous results. Just making the circuits faster does not solve the problem, because, as with any shift register design, the setup and hold times of individual stages need to overlap, *after* clock skew is factored in. Faster circuits reduce both the setup and hold times, aggravating the problem of skew. Asynchronous, or "self-timed" circuits might be a technique for avoiding clock skew, but would incur an unacceptable time penalty in the RAP. Ultimately, it is primarily clock skew that limits the size of a RAP that could be constructed.

The TMS 320C30 requires an input clock of 32 MHz and produces an output clock of 16 MHz. All LDR and memory operations are synchronized using the 16 MHz clock from the DSP. Unfortunately, the timing relationship between the input and output clocks of the DSP is not tightly controlled, and would not allow using a single master 32 MHz system clock for the RAP.

The solution to this problem is to include a phase locked loop (PLL) in each node, locking the clock coming out of each DSP to a master 16 MHz clock. This master 16 MHz clock is generated on one RAP board and distributed to all boards (including the generator board) using a special cable carrying a differential ECL signal at the card edge. A quad ECL receiver on each board converts the distributed master clock to TTL logic levels, and sends four copies to the four nodes on a board. The clock signal lines to the four nodes on the board are of equal length to ensure similar delays for the distributed master clock.

This replicated master clock is then sent to a digital phase comparator implemented inside of the LDR PGA. The other input to the phase comparator is the output clock from the node DSP. The phase comparator output is then low-pass filtered and sent to the control input of a 32 MHz crystal-stabilized voltage controlled oscillator. The output of the oscillator

next drives the clock input to the node DSP, closing the feedback loop and phase-locking the two 16 MHz clocks. Measurements on the prototype system show a clock skew window (on a single board) of less than 2 ns between all nodes.

VME HOST INTERFACE

The RAP is controlled by a host processor operating on the VME bus. To date, we have used the RAP with a 68020-based CPU board running a real-time operating system, and with a Sparc-based Sun workstation. The host processor accesses the RAP by reading and writing areas in the address space that map to each node. The address mapping is controlled by a unique PAL on each board that decodes 4 bits of address from the host, giving a system limit of 16 boards (64 nodes).

For each RAP board, four different address areas are used. Two of these areas are used for individual node access of memory and control registers, and the remaining two are used for board control and status functions. Accessing node memory is via the "extended" addressing mode of VME, which uses a full 32-bit address. All of the register and control functions use the "short" addressing mode, requiring only a 16-bit address. The four different address areas are:

- 8 MW of extended access read/write memory above VME address 8000 0000 for SRAM and DRAM on each node.
- Eight words of short access address for control and status registers on each node. Currently unimplemented, these registers will allow a clean interface between host and DSP software, and are referred to as the *node registers.*
- Four words of short access address for control and status registers on each board. These are called *board registers.*
- Three words of special-purpose address in the short access range used for "broadcast" (write) and "broadcall" (read) functions. These addresses are the same for all boards and will be used for system configuration and synchronization functions that benefit from the parallel access mechanism. These registers are presently unimplemented.

The RAP boards also generate VME interrupts and provide 8-bit interrupt vectors to the host. Only 32-bit data transfers are supported between the RAP boards and the CPU, and the RAP has no provision for being a bus master.

VME Address Map

Details of the four address areas used by the RAP are shown in Table 4. Three of the areas use the board identification number, labelled BID, while the fourth address access is independent of the BID. The BID is any 4-bit value from 0x0 to 0xF, and all values must be unique within a single VME card cage. In addition to the BID, the register accesses that use the short address mode utilize one of two 4-bit board address qualifier (RRAQ or RBAQ) to prevent conflict with other VME boards. The prototype RAP system uses RRAQ=0xF and RBAQ=0xE. All of these enabling qualifiers are programmed into an address decoder PAL for easy modification in later systems.

Table 4. VME Board Address Map			
Type	Memory Used	VME Access	Address Formation
RAM	8 MWords/node	Extended (32-bit)	A[31] = 1 A[30:27] = BID A[26:25] = Board Node # A[24] = SRAM*/DRAM A[23:2] = RAM Location
Node Register	8 Words/node	Short (16-bit)	A[15:12] = RRAQ A[11:8] = BID A[7] = 0 A[6:5] = Board Node # A[4:2] = Register #
Board Register	8 Words/board; only 4 used	Short (16-bit)	A[15:12] = RRAQ A[11:8] = BID A[7] = 1 A[6:5] = Board Node # A[4:2] = Register #
Broadcast & Broadcall	3 Words/board	Short (16-bit)	A[15:12] = RBAQ A[11:8] = 0000 A[7:5] = don't care A[4:2] = Register #

Interrupt Operation

The interrupt mechanism on the VME bus allows for interrupting devices to be processed according to two "prioritizing" choices. Seven separate interrupt signals are available on the backplane bus for any board to use, with a predetermined priority among the seven. In addition, the installed order of the boards in the backplane gives priority to the one closest to the bus master in the event of two boards interrupting on the same level simultaneously.

In this design, only one of the seven interrupt signals is used, to be shared by all RAP boards. As fabricated, the boards use interrupt level 3 on the VME bus. By making a minor hardware modification, this can be changed to level 6. Interrupts are used for two different situations on the RAP: board interrogation (also called autoconfiguration), and "normal" program operation, when a DSP generates the interrupt. The following sections detail the two cases.

Board Interrogation. Although each board is assigned a unique 4-bit identification number (BID), the order of the boards in the backplane and the connections between boards are not constrained.[6] In order to allow flexibility, a technique for board interrogation is supported that uses the VME interrupt daisy chain to ascertain the board order. Once the order is determined, a diagnostic program will use the LDR to discover the arrangement of the cabling between the interboard LDR connectors.

[6] With the exception of the interrupt daisy chain, which must be continuous.

The board interrogation method uses the broadcast address in the VME short access range to generate a simultaneous interrupt on all RAP boards. During the interrupt acknowledge sequence, the order of response is based on the proximity of the responding board to the bus master CPU board. When a board responds to its acknowledge operation, it returns an 8-bit vector to the CPU which includes the 4-bit board identification number. As long as the interrupt daisy chain is not broken, the CPU will continue to service all the interrupts and end up with an ordered list of BIDs corresponding to the placement of the boards in the backplane.

DSP ⇒ Host Interrupts. Each DSP has the capability to send an interrupt to the host using the "normal" mechanism. This is accomplished by the DSP writing a bit to an I/O register, which in turn generates the interrupt on the VME backplane. When the host executes an interrupt acknowledge cycle, the RAP supplies an 8-bit vector to the interrupt handler routine on the host. At the current level of system development, the interrupt is serviced by passing messages between the host and DSP using the node SRAM. Ultimately, the system software may use the (as yet unimplemented) node registers for performing this function.

VME-RAP Registers

Table 5 shows the complete address of the VME-accessible registers on the RAP. The prototype RAP system uses RRAQ=0xF and RBAQ=0xE as the address qualifiers.

Table 5. RAP Board Register Addressing

Byte	Address Bits					Access	Register Access	
Address	15:12	11:8	7	6:5	4:2		R/W	Name
0-1C	RRAQ	BID	0	0		Node 0	These node registers will be used for host-DSP control in later versions of the system software.	
20-3C	"	"	0	1		Node 1		
40-5C	"	"	0	2		Node 2		
60-7C	"	"	0	3		Node 3		
80	RRAQ	BID	1	x	0	Board	W	Board Configuration
80	"	"	1	x	0	Board	R	Board Status
84	"	"	1	x	1	Board	R/W	Interrupt Vector
88	"	"	1	x	2	Board	W	Xilinx LDR Load
8C	"	"	1	x	3	Board	W	Xilinx MEM Load
90-FF	"	"	1	x	4-7	Board		[reserved]
	RBAQ	0000	x	x	0	System	W	Broadcast
	"	0000	x	x	1	System	R	Broadcall
	"	0000	x	x	2	System	W	Board Interrogation

Node Registers. Only one of the eight available node registers (on each node) is currently implemented. Bit 0 of the node configuration/status register is used to RESET the DSPs. This register is located at byte addresses 0x18, 0x38, 0x58 and 0x78 for the four nodes, and the signal is active-LOW.

Although the RAP has been debugged using simple host-DSP communication mechanisms, development will continue on software that can take advantage of the node registers. These registers offer a potential performance advantage over memory-based message passing because the DSP need not stop during the host access of SRAM.

Board Configuration & Status Register. The board configuration and status register (byte address 80) is used to load the Xilinx MEM and LDR chips. The bit assignments are shown in the following table.

Table 6. Board Configuration & Status Register

Bit #	Name		Function
0	XMDPM	W	MEM START PROGRAM - Active Low
0	"	R	MEM PROGRAM DONE STATUS - High When Done
1	XMRSTN	R/W	MEM RESET - Active Low
2	XPIP	R	Xilinx Programming In Progress - Active Low
3	-	R/W	*Not Assigned*
4	XLDPN	W	LDR START PROGRAM - Active Low
4	"	R	LDR PROGRAM DONE STATUS - High When Done
5	XLRSTN	R/W	LDR RESET - Active Low
6	-		*Not Assigned*
7	GVIE	R/W	Global VME Interrupt Enable - Active High
8-31	-	-	*Not Assigned*

Xilinx Loading. Loading the Xilinx MEM and LDR chips is handled by the host processor at the time power is turned on. One 90 KB file containing the configuration information for all four MEM chips is sent, a word at a time, to byte address 8C. Additional operations are required before and after the code is downloaded to ensure the process is successful. The same procedure, with a different 90 KB file, is repeated for the LDR chips using byte address 88. The details of this procedure are described in the Appendix.

Interrupt Vector Register. The interrupt vector sent to the host processor is controlled by the value loaded in this register. At power-up, the register is loaded with a default vector value of 0x0F, which is interpreted by the host as an "uninitialized interrupt". Once the host processor writes to the register, the written value replaces the 0x0F. The register is readable by the host on demand as well.

Register Implementation. The registers described in the preceding sections were designed to provide a reasonable software control mechanism for individual nodes without adding an undue amount of hardware. The board registers are implemented completely inside the Xilinx VME interface chip, while the node registers are located in the LDR Xilinx devices.

The DSP accesses the node registers in the I/O address space beginning at 80 4000. Since these are the only devices residing in the I/O address space, the I/O STROBE signal from the DSP is a convenient gating signal. The DSP also treats the timing of I/O operations differently (than memory), adding an extra clock cycle to I/O READ instructions.

The eight node registers (per node) are located in the Xilinx LDR chip for that node. Because of pin count limitations of the LDR chip, access of the node registers from the VME is through a serial communication link to the Xilinx VME chip. The data clocking rate for this link is 16 MHz, enabling a word transfer to or from a node register in 2μs. Because this communication mechanism is expected to be used primarily for signalling the host and doing print functions, this rate is not a performance limitation.

VME Interface Hardware

Aside from the address decoding PAL mentioned above, the interface to the VME bus is implemented using a pair of bus-specific devices, some address and data buffers, and one Xilinx XC3064 PGA. The bus-specific devices are the VME2000 and VME3000 manufactured by PLX Technology. These chips handle the normal bus handshaking and interrupt control, and conform to the VME specifications for timing and bus driving level.

The XC3064 provides additional address decoding required to access the individual nodes on a board. In addition, the PGA provides the 32-bit shift register interface used to download the MEM and LDR chips at power up.

CONSTRUCTION and TEST

In order to make the most efficient use of the design effort, random logic is minimized, bus loading is carefully monitored, and architectural decisions have been made to ease the board layout process. The RAP boards adhere to the large (9U) VME specification and are designed to plug into Sun backplanes. This size allows including all of the features described above without requiring extreme constraints on layout or fabrication. The boards are 6 layers total, with 2 of the layers as power and ground. Each board requires 252 chips, and contains over 10,000 holes.

The TMS 320C30 is designed with an internal scan path serial shift register that is accessible via 6 control pins. The RAP includes a connector to these pins at each node allowing it to be attached to a hardware "in-circuit" emulator TI provides for the TMS 320C30. By using this technique, all of the signal pins on the DSP as well as the internal RAM bits are available for reading and writing through the emulator. This in-circuit emulator was quite useful for debugging both hardware and software.

SUMMARY

We sought to build a fast processing system for our research and the RAP is the result. Now that the RAP is functioning, this report documents (from the hardware perspective)

what it is and how it got that way. Although development work is continuing in both hardware and software, the RAP has already been a success in performing the task for which it was designed.

ACKNOWLEDGEMENTS

The RAP system design was the result of a team effort over the period of eighteen months: architecture - Nelson Morgan, Joachim Beer and Eric Allman, all of ICSI; software design and implementation - Jeff Bilmes and Philip Kohn of ICSI; hardware design and review - Joel Libove of Ultraview and Peter Alfke of Xilinx; board layout - Lewis Lopez of Pactite. Special thanks to the following companies for assistance in realizing the prototype RAP system: Texas Instruments, Cypress Semiconductor, Toshiba America and Xilinx Incorporated. The support of ICSI for this project is gratefully acknowledged.

APPENDIX

The steps required to load the Xilinx LDR and MEM devices are as follows:

1. For the Xilinx parts to be downloaded, write the following sequence of bits to the configuration register (byte address 80):

    RESET = Low, PROGRAM = High
    RESET = High, PROGRAM = High
    RESET = High, PROGRAM = Low
    RESET = High, PROGRAM = High

All other bits in the register should be High.

The above sequence pulses the RESET line, and then pulses the PROGRAM line. If the board status word is now read, (byte location 80) the PROGRAM bit should still be Low, held there by the reset Xilinx parts until the downloading is complete.

2. Next, write the file with the Xilinx configuration data to the appropriate register address (byte address 88 or 8C). After sending each word, read the XPIP bit, waiting until it returns LOW before sending the next word.

3. To verify that the process worked properly, read the board status word (byte address 80) to see that the appropriate PROGRAM bit is now High. If it is still Low, the download was not successful.

REFERENCES

[1] N. Morgan and H. Bourlard, "Continuous Speech Recognition Using Multilayer Perceptrons with Hidden Markov Models", *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp 413-416, Albuquerque NM, April 1990.

[2] N. Morgan, "The Ring Array Processor (RAP): Algorithms and Architecture," International Computer Science Institute, TR-90-047, September 1990.

[3] P. Kohn and J. Bilmes, "The Ring Array Processor (RAP): Software Users Manual," International Computer Science Institute, TR-90-049, September 1990.

[4] J. Bilmes and P. Kohn, "The Ring Array Processor (RAP): Software Architecture," International Computer Science Institute, TR-90-050, September 1990.

[5] N. Morgan, J. Beck, P. Kohn, J. Bilmes, E. Allman and J. Beer, "The RAP: a Ring Array Processor for Layered Network Calculations," *Proc. of International Conference on Application Specific Array Processors*, pp. 296-308, IEEE Computer Society Press, Princeton, NJ, 1990.

Index

Table of Contents

Table of Figures

Table of Tables