ELSEVIER

Parallel Computing 20 (1994) 1627–1638

PARALLEL COMPUTING

Meiko CS-2 interconnect Elan-Elite design

Jon Beecroft *, Mark Homewood, Moray McLaren

Meiko Limited, 650 Aztec West, Bristol, UK

Received 10 September 1993

Abstract

The architecture of the interprocessor communications subsystem in the Meiko CS-2 MPP is discussed. This utilises a 'fat tree' network constructed from high performance crosspoint switches. Processing Elements interface to this network via a communications co-processor which contains intelligence to handle virtual addressing and ensures very low message start up times. Measured performance figures are given for the system running a variety of different programming models.

Keywords: Meiko CS-2 MPP; Interprocessor communication; System architecture; 'Fat tree' network; System performance

1. Introduction

The first generation Meiko computing surface (CS) proved to be an early competitor in the emerging massively parallel processing market. Since this introduction the market has gained acceptance and followers, both vendors and customers, [4,6] and [5]. During this time perceptions and requirements have changed somewhat and, more importantly, technical capabilites have improved enormously. The Meiko CS-2 has been developed to track this changing market place.

The predominant concern in the development of the second generation Meiko Computing Surface (CS-2) interprocessor communications subsystem was to broaden the applicability of massively parallel processing (MPP) systems by improving critical system characteristics. Three areas were targetted:

* Corresponding author.

SSDI 0167-8191(94)00052-C

Firstly, application suitability (Improved bandwidth and latency), reducing the latency for short messages improves the scalability of applications, increases the number of suitable applications and may make it possible to exploit further levels of parallelism within a given application. Bringing the network bandwidth closer to that of the central processor unit (CPU) memory system reduces the cost of communications. Obtaining real improvements in latency and bandwidth is made more challenging by the continued improvements in CPU performance, since these factors must be considered as relative to CPU speed.

Secondly, support for multiple programming models, since there is no one preferred paradigm for parallel application programming. Different models may be suited to different applications due to internal consideration, such as the structure of the application and to external considerations such as portability of own or third party applications. The basic communications mechanisms should therefore not make assumptions about application usage.

Thirdly, system availability and fault tolerance, as MPP systems have moved to the point where they are deployed as production systems, fault tolerance has acquired increased significance. While true fault tolerance requires consideration of all aspects from base hardware and operating system through to application, the feasibility of building high availability systems is dependent on a certain basic level of provision in the communications hardware.

2. System architecture

The Meiko CS-2 range of machines are third generation distributed memory MIMD machines. They are built from multiple Processing Elements (PE) comprising industry standard SPARC microprocessors (with optional vector processors for vectorizable applications), running standard Solaris 2.x kernels. A full Unix implementation was chosen over a micro-kernel since all elements in the system can be configured to have disk and I/O devices, and are required to support the full set of system calls. Having a full Unix on every PE has the important consequence that CS-2 systems do not require a host machine and single points of failure are obviated.

The basic structure of the system (Fig. 1) is that every PE is configured with a communications co-processor, known as the Elan chip. This provides the interface between the PE and the communications network. Two 50 Mbyte/s per direction links are provided into two independent switch networks. The switch networks are multi-stage networks based on an 8×8 crossbar component – the Elite network switch.

The two device approach was taken in order to decouple the PE architecture from the issues of network topology. In principle the Elite switch can be used to construct a variety of networks including grids and hyper-cubes. CS-2 uses a 'fat tree' topology for reasons which will be discussed later in this paper. The two components, Elan and Elite correspond closely to the T9000 transputer virtual channel processor (VCP) and C104 router, [1] and [2].

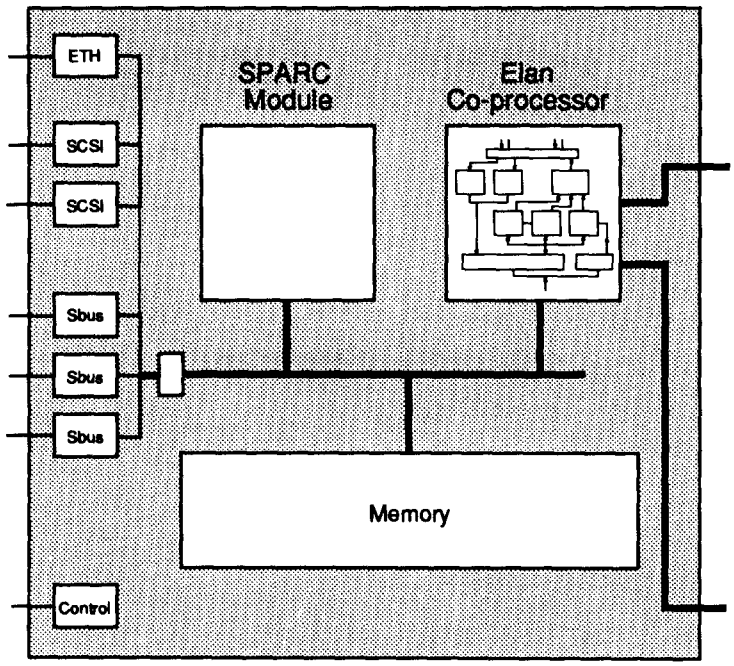

Fig. 1. CS-2 processing element.

The communications subsystem of the CS-2 is a sharable resource. Multiple parallel applications may share the network without compromising security. This is utilised by the operating system which can communicate securely between PEs using an SVR4 Data Link Provider Interface (DLPI) to the communications hardware. This means that a separate communications network is not required to provided standard Unix networking functionality. TCP/IP network operations are provided across the Elan/Elite network. This provides, for example, a boot path for PEs with no local disk. Up to 64K independent parallel applications can be supported on the network, 32 of which are reserved for system services.

3. Communication sub-system

The most significant feature of CS-2 communications is the use of network read/write and synchronisation primitives, rather than virtual channel operations. There are several reasons for this:

Ease of handling incoming data a write transaction consisting simply of a virtual address and a block of data contains all the information required for handling as it comes off the network. By comparison a virtual channel would require a look up of the channel descriptor and an update to that descriptor in addition to writing the data. The simplicity of the CS-2 network protocol allows for a simple and fast hardware input engine.

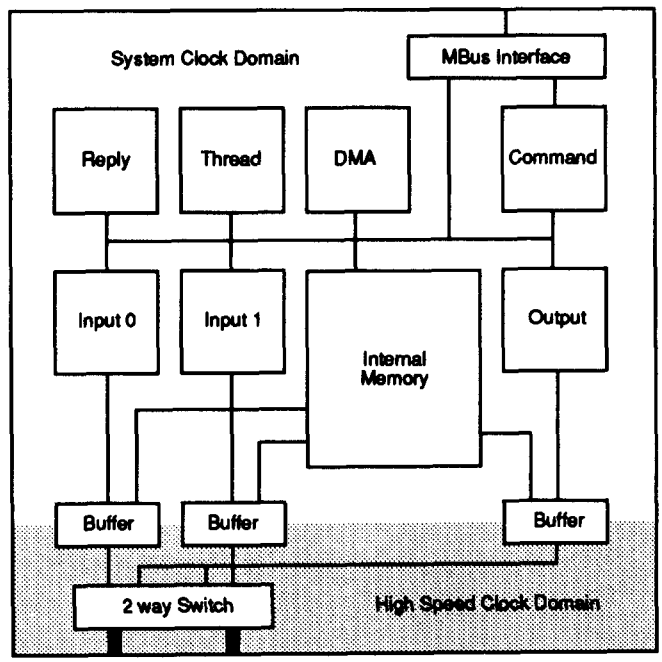

Fig. 2. Elan processor schematic.

Exception and error handling, the basic protocols must be tolerant to loss of packets from data transmission errors or when packets are discarded due to page faulting on the inputer. In a channel protocol, the state of a communication is distributed between the sender and the receiver and requires a higher level protocol to handle errors. Where network write is the basic operation, the state of a communication is held by the processor performing the network DMA.

Support for shared memory models, given that both shared memory and channel style communications are required, it is easier to emulate channel communications over shared memory primitives than to emulate shared memory over channel style operations.; the latter case requiring a thread schedule to service every network memory operation.

3.1. Communications co-processor

The Elan is a memory mapped co-processor such that any commands and data are transferred between the main CPU and the Elan in memory. Hardware cache coherency is implemented between the two processors to avoid the requirement for cache flushing.

The Elan consists of five separate functional units (Fig. 2):

Inputers (2). Receives packets in from the network and carries out the required operation on main memory or schedules an operation on another Elan functional unit.

DMA engine. Executes a queue of DMA descriptors held in main memory. A DMA transfers data from local memory to the memory of a remote PE. The DMA processor handles fragmentation of long transfers and time-slices between very long transfers.

Thread processor. The thread processor executes a queue of threads held in main memory. This uses a simple RISC instruction set optimised for fast context switching. The thread processor is used to handle message protocols to minimise the number of interrupts taken on the main processor.

Reply processor. Maintains a queue of read responses and transfers them back to the requester over the network.

Command processor. Watches the main processor memory bus for physical memory transaction in a fixed area. These are interpreted as commands to schedule operations on the other functional units.

In the current implementation of Elan all five of the processors are implemented on a single microcoded engine capable of single cycle context switching.

3.2. Virtual operation

To minimise communication start up latency a user process can issue instructions directly to the Elan without going through a system call. One consequence of this is that Elan must handle and check virtual addresses.

The Elan has its own memory management system and an on-chip translation lookaside buffer (TLB). The separate set of page tables maintained for the Elan is kept in step by system software. The advantage of having a separate set of tables, rather than sharing the main CPU tables, is that different permissions can be set. This allows, for example, pages to be mapped as readable but not writable over the network.

In a parallel application, a similar virtual abstraction is applied to the process space. User programs communicate with virtual process numbers which are checked and translated by the Elan before packets are output onto the network. The absence of physical address or processor locations in user space provides a mechanism for migrating processes around the system and for making them placement independent.

3.3. Network transactions

The basic communication transcation is the write block. This consists of a 16 bit transaction type field, 16 bit context, a 32 bit virtual address, 32bytes of data and a 16 bit CRC (Cyclic Redundancy Code). Data is always sent cache block aligned to optimise memory bandwidth and simplify virtual memory exception cases. In addition to the above there are four other classes of transactions.

Event operations. An event is a double word primitive which is used for synchronisation. By performing a *wait* on an event, a thread or DMA may be suspended on that event. A set operation occuring across the network will cause anything suspended on that event to be scheduled. A common use of an event is to reschedule a thread on completion of a network DMA.

Read operations. Short reads of 1 to 3 words are supported directly. Remote reading of larger amounts of data is handled by scheduling a DMA remotely.

Atomic operations: Atomic swap, atomic increment and *test and set* are supported for shared access to data structures across the network.

Test operations. A network test operation allows semaphores to be poled acros the network. Combining this with the network hardware broadcast function allows creation of block synchronisation primitives.

3.4. Low level protocols

At the lowest level data is transferred over a byte wide bi-directional link. The choice of a byte wide link protocol is dictated by a number of factors. The link needs to have sufficient bandwidth so that a single link can meet the requirements of a PE. The main limitation on link data width is that a very wide link format would mean that the I/O pin limited Elite switch device could only satisfy a very low order crosspoint. The actual implementation uses 20 wires for each bi-directional link, 10 in each direction. When clocked at 70 MHz this permits 100 Mbytes/s of bi-directional bandwidth after allowance for protocol overheads. Both directions of the links can be used simultaneously, on independent operations, at 50 Mbytes/s. The format and level of performance is also appropriate for conversion into optical fibre for long distance communications. The link can be converted into a single 700 MHz data stream for this purpose.

The use of bi-directional links permits flow control and acknowledge tokens to be multiplexed on to the return link. The low level flow control allows buffering at the line level of data so that communications clock frequencies higher than the round trip delay can be used. The interface is asynchronous and is tolerant to a 200 ppm frequency difference between ends. This means that each end can have its own clock, substantially simplifying construction of large systems.

3.5. The switch component

The Elite switch (Fig. 3) is capable of switching eight independent links of the above format. The switch is a full crosspoint so any permutation of inputs and outputs is allowed. The total sustainable switch bandwidth is therefore 400 Mbytes/s. For each data route through the switch a return route for packet acknowledges exists. This effectively creates an independent return network for acknowledges that shadow the data network. This means that acknowledges are never affected by congestion on the data network.

The switch component contains a broadcast function. Any input can be forwarded to any contiguous range of outputs. The switch contains logic to perform the logical recombination of acknowledge or not acknowledge tokens from all the broadcast destinations. To allow broadcasts to ranges of outputs over multiple switches the switch topology must be heirarchical.

Routing within the switch is byte steered. On entry into a switch the first byte of any packet is interpreted as the destination output or range of outputs. This byte is

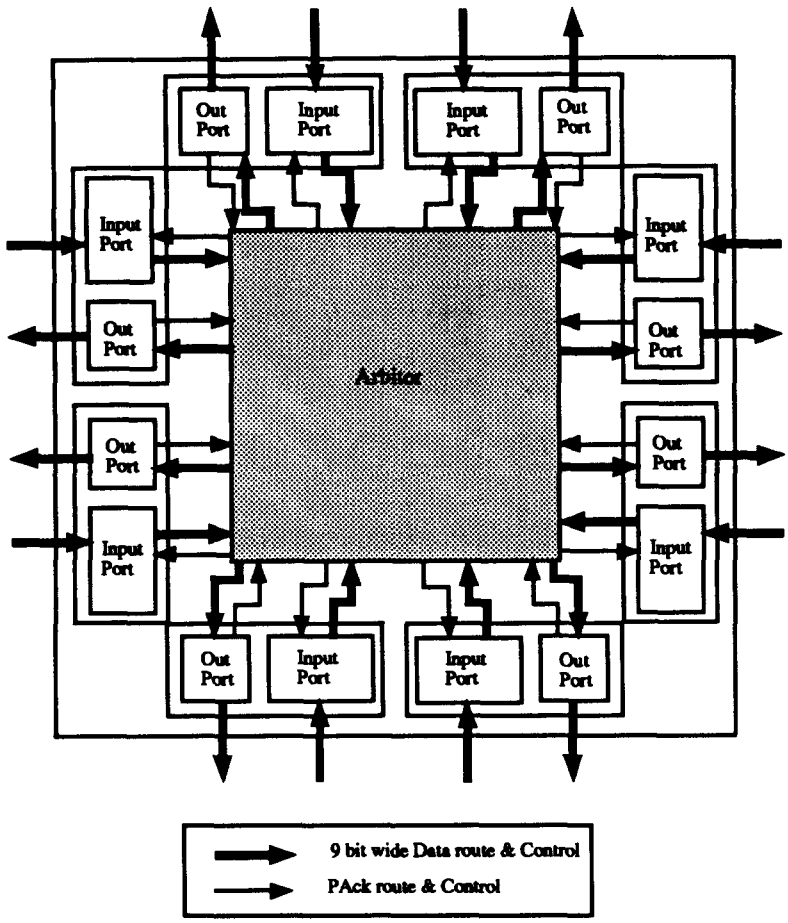

Fig. 3. Elite crosspoint schematic.

stripped off within the switch so that the next byte is used for routing in the following switch. The latency through each switch device is 7 clock cycles for outgoing data, and 5 cycles for returning acknowledge tokens. The switch contains no routing tables of any sort. The translation between destination processor and route information is performed entirely on the communications processor where it can be more easily modified or updated. The switch requires no configuration state of any sort to become operational.

4. The CS-2 network

The CS-2 Network (Fig. 4) is a 'fat tree' structure [8]. In the CS-2 implementation there is always sufficient bandwidth up at each stage to carry all the traffic from lower levels; bandwidth is never reduced.

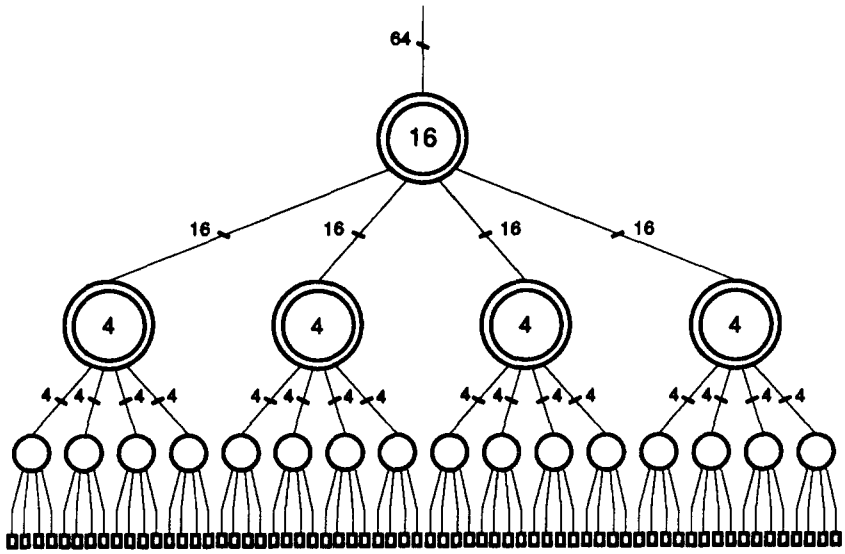

Fig. 4. One layer of a 64 processor CS-2 network.

There are three main reasons for the choice of topology.

Firstly, Fault tolerance, fat tree structures contain many alternate routes between PEs. Moreover, all the alternate routes have identical characteristics in terms of numbers of switches traversed. In the CS-2 this is further augmented by having two identical networks.

Secondly, process placement issues are removed. Arguments about locality of reference in communications patterns can be made to justify reducing the network bandwidth at higher stages of the fat tree. However, this has the effect of making process placement more critical. CS-2 provides a sufficiently rich interconnect to minimise the effects of process placement.

Finally, scaling characteristics, the fat tree is essentially a logarithmic network. Bandwidth maintained at the higher levels of the tree are characterised by a bi-sectional bandwidth that scales with the number of processors, for an N log N growth in cost. For a grid topology, scalability of bisectional bandwidth is sacrificed to ensure a linear growth in cost [3,7].

Mathematically, the network can also be considered as a Benes network folded around its centre line. The significance of this is that this network is equivalent to a full crosspoint for any given permutation.

4.1. Routing algorithms

Although the network has the capacity to emulate a full crosspoint there still remains the problem of allocating the routes in a non-contending fashion. Route selection for Elite networks is made on the way into the network by the communications co-processor. This means that the routing algorithm can be changed easily.

The translation process on the Elan processor uses a table look up indexed by destination. As each source has its own routing table, any function of source and destination address can be used. Each table entry consists of four possible routes, one of which is selected at random. This function can be used to distribute traffic evenly over a network. Random routing is easily disabled by setting all routes to the same value. Having the route function on the PE simplifies the problem of modifying the routing tables to avoid failed routes.

One simple routing function routes all data for the same destination through the same node at the top of the hierarchy. The rationale for this is that the routing network is essentially performing two functions; data distribution; and arbitration where many senders wish to talk to the same destination simultaneously. Directing all traffic for the same end PE through the same point at the top of the heirarchy ensures that if blocking occurs, then it occurs as soon as possible, using as little resource as possible. Using this simple algorithm has the effect of reducing the network to an Omega network – essentially the paths back down the 'fat tree' are guaranteed non-blocking and perform a simple data ordering operation. This routing strategy has the same blocking behaviour as an Omega network, and is non-blocking for arbitrary shifts and fast fourier transform (FFT) style permutations.

5. System performance

The CS-2 system is designed to support many different programming paradigms efficiently, [9]. Indeed multiple paradigms can be used from within the same application. The most appropriate is determined by machine dependent factors such as performance and external factors such as portability.

The measured performance for the CS-2 system under a variety of conditions is set out below:

Latency on a quiet network increases by 170 ns per switch traversed – the longest route in a 1024 processor data network is through 9 switches, adding 1.5 microseconds to the endpoint latency.

All figures are for communication between Unix user level processes. Bandwidth is largely independent of the relative positions of the communicating

Table 1
Measured performance between two Processing Elements (PE) over a single switch

Programming model/Op.	Latency	Sustained bandwidth
Direct DMA	9 μs	44 Mb/s
Channel comms.	24 μs	44 Mb/s
PARMACS	78 μs	44 Mb/s
PVM send/receive	78 μS	44 Mb/s
NX/2	78 μs	44 Mb/s
Barrier sync.	28 μs	N/A Mb/s

processors. In fact communication between 'distant' processors achieves higher bandwidths than indicated because of extra buffering in the data network.

6. Implementation

The Elan communications processor and Elite network switch were designed at the Meiko European subsidiary in Bristol, UK.

The two ASICs are implemented on 1.0 micron drawn, three layer metal, CMOS sea-of-gates gate arrays. Both components use a 110,000 gate base array which incorporates more than 440,000 transistors.

The communications processor achieves a utilisation of approximately 75% (representing around 83,000 gates), the network switch utilises approximately 55% (61,000 gates), the switch is a pad limited design.

Both components are packaged in 208 pin PGAs.

6.1. Design approach

The network design was subjected to extensive parallel simulation using models written in 'C'. The network components were designed using an interactive top down approach.

A gross functional model was written in the Verilog HDL (hardware description language) to simulate the behaviour of the Elan. This was rewritten and refined to produce a cycle by cycle functional model encompassing the full state required, together with a module hierarchy. These modules were converted into a stylised RTL (Register Transfer Language) subset of the Verilog HDL, which aided their synthesis into gate level logic. This synthesis/conversion was done by hand for the majority of modules. The development of the Elite followed a similar design flow to the Elan.

The gate and functional models were automatically compared using a number of techniques, and were kept in step throughout the design. The two models have different simulation properties. The functional model consumes considerably less memory, however the gate level model simulation executes faster if the entire data set is present in memory during the simulation (for Verilog-XL). Simulations of CS-2 data networks (multiple communications processors and network switches) were conducted using functional models.

The gate level logic was targeted at a vendor independent technology library. Vendor selection was based on the two qualities of cost and speed requirement. Re-targeting of the implementation to a different vendor/process would be relatively easy to perform.

6.2. Optimisation

The 70 MHz operating frequency of the Elan-Elite network required extensive optimisation of logic delay using a timing analyser (Cadence Veritime) to give

guaranteed worst case temperature, process, and voltage operation. The layout associated with these high frequency circuits was carefully floor-planned to give such high clock frequencies from a relatively conservative technology.

Both designs will operate at 50 MHz and 100 MHz respectively on 0.8 micron processes. The use of static timing analysis provides guaranteed achievement of these frequency rates, without ad hoc iterations to improve high frequency yield.

6.3. Clocking

The Elan operates in two separate clock domains, one synchronised to the host processor's memory interface, the other to the 70 MHz communications clock. Appropriate synchronisation occurs when data is transferred between the two domains.

The Elite also synchronises data from each link to its local 70 MHz clock, the data is sampled and regenerated at each switch. The Elan and Elite do not therefore require a globally synchronised clock distribution scheme.

6.4. Re-hosting

The interface between the Elan and the processing element is through the processor's cache coherent memory protocol. The initial version of the Elan implements the SPARC MBus protocol, however re-targeting of the communications processor to a different host memory bus would require only minor modifications to the communications processor, since the memory interface is a well contained module within the design consisting of around 5,000 gates.

7. Conclusions

The CS-2 interprocessor communication systems delivers a high proportion of the peak performance to user programs. The basic design is flexible in terms of usage and supports multiple programming paradigms effectively. The implementation is both scalable and fault tolerant and provides the basis for machines with up 4096 processors.

References

[1] J. Lewis and P. Thompson, The IMSC104 Packet Routing Chip, INMOS Limited, 1993.
[2] The T9000 Transputer Hardware Reference Manual, INMOS Limited, 1993.
[3] A. Chien, A cost and speed model for k-ary n-cube worm-hole routers, Proc. Hot Interconnects Conf., Stanford University (Aug. 1993).
[4] G. Bell, Ultracomputers, a teraflop before its time, *Commun. ACM* 35(8) (Aug. 1992) 27–47.
[5] CM5 Technical Summary, Thinking Machines, 1991.

[6] H. Burkhardt III, Announcing the KSR1 supercomputer, Henry Burkhardt News. comp. arch, 1992.
[7] W.J. Dally, Performance analysis of k-ary n-cube Interconnection Networks, *IEEE Trans. Comput.* (June 1990) 775–785.
[8] C.E. Leiserson, Fat-trees: Universal networks for hardware-efficient supercomputing, *ICPP IEEE* (1985) 393–402.
[9] V.S. Sunderam, PVM: A framework for parallel distributed computing, *Concurrency Practice and Experience* (Dec. 1988) 315–339.