Hathi-2 multiprocessor system

M Aspnäs, R J R Back and T-E Malén present a complete project review of the design, implementation and use of the transputer-based Hathi-2 multiprocessor

Hathi-2 is a reconfigurable general-purpose loosely-coupled MIMD multiprocessor system consisting of 100 32-bit IMS T800 transputers and 25 16-bit IMS T212 transputers. Hathi-2 has a distributed switching network and a distributed control system which contains an interrupt system and hardware support for distributed monitoring. This report describes the architecture of the Hathi-2 system and the design goals of the system. The system software that has been developed for the system is also presented.

multiprocessors computer architecture transputers

Hathi-2 is a reconfigurable general-purpose multiprocessor system built of 100 Inmos T800 transputers. The system can be characterized as a loosely-coupled MIMD multiprocessor, with a reconfigurable distributed interconnection network and a modular design. Due to its modularity, Hathi-2 can be expanded by adding more modules, i.e. processor boards, to the system.

The Hathi-2 multiprocessor system was developed as part of the Hathi project at Åbo Akademi and the Technical Research Centre of Finland (VTT/TKO) in Oulu. The Hathi project started in August 1986 and ended in December 1988. One goal of the project was to build a massively parallel multiprocessor system based on transputers. The design of the Hathi-2 system began early in 1987 and the system was delivered to Åbo Akademi in April 1988. The Technical Research Centre of Finland in Oulu was responsible for designing and building the hardware system[1,2], while Åbo Akademi has designed the system software and application programs for Hathi-2[3]. The theoretical parallel performance of the Hathi-2 multiprocessor system is 150 MFLOPS or 1000 MIPS.

A number of applications have been implemented on Hathi-2 as a part of the project. Among the applications are full-text retrieval in a distributed database[4], fluid dynamics simulations[5], geometric image transformation of satellite data[6], cluster identification in a three-dimensional data space, and parallel execution of production systems.

The reader is assumed to be familiar with the transputer architecture[7,8] and Occam[9,10], so this will not be presented here. The paper contains an overview of other transputer-based multiprocessor systems. The architecture of the Hathi-2 system and the motivations for choosing this architecture are described. Later sections describe the hardware environment in which Hathi-2 is operated, i.e. the host computers from which it is accessed and the peripheral units attached to it. The software environment of Hathi-2, i.e. the programming languages and other utilities that can be used to write parallel programs for the system, and the software developed for the control system are presented. Also covered are the Transputer Network Topology administrator, a utility with which the Hathi-2 system can be (statically) reconfigured, and the Transputer Network Monitoring System, with which the user can monitor the activities in a parallel program running on Hathi-2. Finally, the paper evaluates the architecture of the Hathi-2 system and identifies its strengths and weaknesses.

RELATED WORK

Several transputer-based multiprocessor systems have been announced during the last few years. This section gives a short description of the architecture of two other, somewhat similar, transputer-based multiprocessor systems.

Meiko Computing Surface

The Meiko Computing Surface[11,12] is a modular, reconfigurable, general-purpose transputer-based multiprocessor system, manufactured by Meiko Ltd of the UK. A Computing Surface consists of one or more modules each containing 10–40 boards. At least one of the boards in a module is a local host board, carrying one transputer, 8 Mbyte of memory and various interfaces for connection

Åbo Akademi, Department of Computer Science, Lemminkäisenkatu 14, SF-20520 Turku, Finland
Paper received: 7 July 1989. Revised: 13 November 1989

with external devices. The local host is responsible for carrying out maintenance tasks such as controlling the electronic routing network, hardware reset, monitoring for hardware errors and hosting the interactive program development environment. The transputers in a module are connected to the local host through a supervisor bus and the local hosts are connected to each other in a chain using two of the links, thus logically forming a global supervisor bus. The supervisor bus is used as a link-independent debug channel through which messages can be routed to the standard output device from every processor. The other boards in a module can be compute boards which contain four T800 transputers with 1–8 Mbyte of memory per transputer, compute boards which carry a single transputer with up to 48 Mbyte of memory, graphics boards with one transputer and dual-ported video RAM and SCSI controller boards for interface to mass storage devices.

The Meiko Computing Surface uses a custom VLSI switching chip for reconfiguring the topology of the transputers in the system. The interconnection network, formed by the switching elements and the transputer links, allows the transputers in the system to be configured as if they were all in the same module. The reconfiguration capability is mainly restricted by the fact that each transputer can be connected to at most four other transputers.

The Computing Surface has a multiuser Unix-like operating system. Each user is allocated a smaller, logically independent part of the system, a domain, which is totally controlled by the user. The user logs into the system and executes the program development environment on a *user-seat*, which is a transputer dedicated to this task. The user-seats share access to the resources in the system, i.e. to the compute servers (the domains) and the disc servers.

Esprit P1085 (Supernode)

A reconfigurable transputer-based multiprocessor system, called the Supermode, was developed in Esprit project P1085[13]. The Supermode architecture has been commercially exploited by Parsys of the UK and Telmat in France, who manufacture and sell general-purpose multiprocessor systems based on this architecture.

The Supernode has a hierarchical architecture. The basic building block is a computing module consisting of one T800 transputer, 256 kbyte to 4 Mbyte of memory, a control bus interface and a performance monitoring module. Eight computing modules are mounted on a *worker*board. Six worker boards can be plugged in to a switching backplane, which connects the worker transputer links to a custom-designed 72-way crossbar switch. Each backplane requires a controller card with one transputer, which is responsible for controlling the switch. Other types of boards can also be plugged into a backplane, i.e. mass storage boards with a disc interface, link buffer cards for extending links away from the backplane, or boards with one transputer and a very large amount of memory. A single backplane can support up to 36 transputers, including the control transputer, disc servers and external links.

Two backplanes can be connected to each other through a switch in a tandem rack, containing up to 72 transputers. For larger systems, several tandem racks can be connected by replacing half of the worker cards in each rack with link buffer cards, whose links are connected to an interrack switch. The interrack switch is controlled by a transputer, thus forming a three-layer switching network. The switching network in the Supernode architecture is universal, i.e. it is able to establish any graph of transputer interconnections. The switching network is reconfigurable but blocking.

All working transputers in the Supernode architecture are connected to a separate low-bandwidth byte-wide control bus, which is used for system housekeeping (such as reset, analyse, error and link speed selection), and for sending monitoring data about link and CPU utilization to a controlling transputer. The control bus can also be used for synchronizing a group of modules with the controlling transputer. The maximum data rate on the control bus is around 100 kbyte s^{-1}.

HATHI-2 ARCHITECTURE

Hathi-2 is a transputer-based multiprocessor system. A transputer-based system generally consists of three main hardware components (see Figure 1):

- A host computer system, which can be a PC (e.g. IBM-PC AT or Apple Macintosh) or a graphical workstation (e.g. Sun-3). The host computer is a normal computer system with discs, screen and keyboard and an operating system (e.g. MS-DOS or Unix). The processor in the host computer (e.g. Intel 80286 or Motorola 68020) is referred to as the host processor. The host computer provides I/O to the multiprocessor system and interaction with the user through a server process, which executes on the host processor.
- A host transputer, which is installed in the host computer system. The host transputer is connected to the bus of the host computer, usually via one of its links, and is thus able to communicate with the host processor. The host transputer is normally mounted on a transputer board (e.g. Inmos B004, Transtech MCP1000 or Levco Translink), which is plugged into one of the available extension slots in the host computer system. The programming environment TDS and the compilers for transputers are executed on the host transputer and use the host computer as an I/O unit.
- A transputer network, consisting of a number of transputers connected to each other via links. The transputers can be connected to any desired topology (e.g. ring, mesh, tree, cube). One restriction on the topology of the network is that the number of links on each transputer is limited, typically four links per

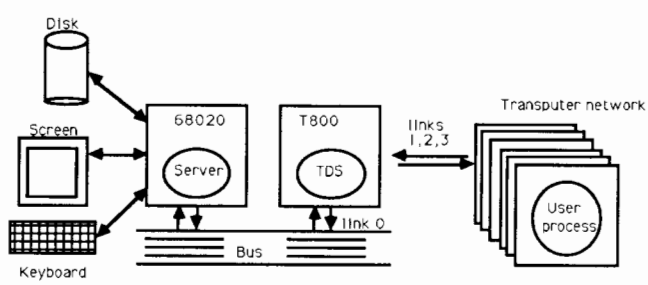


Figure 1. Components in a transputer-based multiprocessor system

transputer. The parallel program is executed on the transputer network and I/O is handled by the server process running on the host processor.

Hathi-2 extends this basic scheme with some additional features, such as a reconfigurable switching network, the ability to partition the system between several users, and hardware support for distributed monitoring of the system.

Design goals

When the design of the Hathi-2 system began at the beginning of 1987, the T800 transputer had just been introduced and only preliminary data was available for the C004 crossbar switch. Large general-purpose transputer-based multiprocessor systems were not commercially available to the same extent as today.

Before the Hathi-2 system was built, a smaller, 16-transputer multiprocessor system, Hathi-1, was constructed from Inmos B003 boards. This system was used for over a year, and much of the system software was designed on this system, using a Hathi-2 architecture simulator developed during the project.

Hathi-2 was intended to be used as a general-purpose multiprocessor system, mainly for research and program development. It was expected that the users would have very different, even contradicting, demands on the system. To meet these demands, the architecture of the system must be flexible, so that users can adapt the system to their specific demands. It was especially important to give the users the ability to reconfigure the topology of the system. The system must also be able to support more than one user at a time.

The proposed design should be modular and as simple and straightforward as possible, in order to minimize the costs of building the system and to improve its reliability. The architecture of the system should also be fully compatible with the existing program development software (e.g. TDS). This requires that the reset, analyse and error control signals are treated in the same way as in Inmos transputer board products.

Experiences from the Hathi-1 multiprocessor system showed that debugging and load balancing of parallel programs was difficult; some tools for monitoring the activities in the multiprocessor system were needed. However, monitoring a distributed system produces a large amount of data and can require extensive amounts of computation. So as not to interfere with the main computation in the system, the architecture contains hardware dedicated to monitoring system activities during program execution. Monitoring data needs to be processed and presented by a separate subsystem, otherwise monitoring will affect the behaviour of the distributed program.

Based on these requirements the following design decisions were made for the architecture of the Hathi-2 system:

- To keep the system design simple and modular, the multiprocessor system is built of one type of board. The number of processors in the system can thus be expanded simply by adding new boards. To extend the system with peripheral units, every board has a number of I/O links dedicated to this purpose.
- The communication links between the processors in the system are connected to each other via a reconfigurable switching network. Since there is only one type of board and all boards are identical, there is no central switching system. The switching network is distributed over the boards and built from Inmos C004 crossbar switches.
- The system is made compatible with existing program development tools by connecting the control signals, Reset, Analyse and Error as a pipeline through all transputers in the system. The system can be shared between a number of users by partitioning the system into a number of independent subsystems. This is done by disconnecting the control signal between the partitions, using switches on the backplane, and connecting the control signals from the user's host to the partition. This gives the user full control over his partition of the system, but prevents him from affecting other partitions.
- To control the distributed switching network, a separate distributed control system was built. The control system consists of one additional transputer on each board, connected to each other in a ring. One of the links on the control system transputers is used for sending commands to the C004 switch. The control system can also handle other supervisory tasks, such as monitoring the behaviour of the system. The control system is separated from the rest of the system, i.e. it is controlled by a separate host computer. Each processor in the system is equipped with some hardware support for monitoring. Gathering and forwarding of monitoring data may not affect the computations in the system, and thus must be performed by some hardware dedicated to this task. Monitoring information is forwarded from the calculating transputers to the control system by two additional hardware components: the FIFO buffers and the CPU load meter.
- An additional interrupt subsystem, based on the transputer's EVENT pin, is included in the control system. This can, for instance, be used to provide a global clock synchronization signal for the system.

Hathi-2 board

Hathi-2 consists of 25 identical extended 3E Eurocard boards. Each board contains four 32-bit IMS T800 transputers with up to 4.25 Mbyte of external memory, one IMS C004 crossbar link switch and one 16-bit IMS T212 transputer.

The T800 transputers are connected pairwise to each other via link 0. The three remaining links from each T800 and link 2 of the T212 are connected to the C004 crossbar switch. The C004 is controlled by link 3 of the T212 transputer (see Figure 2). The T212 controls the setting of the C004 switch by sending commands on the control link. The two remaining links on the T212 (links 0 and 1) are used to connect the T212 transputers into a ring, thus forming the distributed control system.

Of the remaining 19 links on the switch, three links are used as I/O links, i.e. to connect users' host computers and peripheral units to the system. The remaining 16 links are used to form a statical torus connection between the boards in the Hathi-2 system.

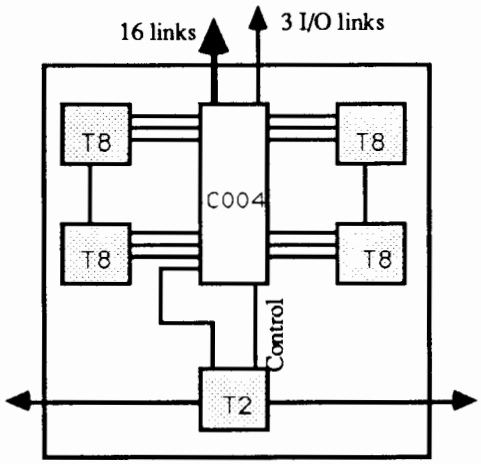


Figure 2. Hathi-2 board architecture

Distributed switching network

Hathi-2 has a two-level connection scheme, with a static and a reconfigurable level. The static connection scheme is formed by the physical interconnections between the boards. The boards are statically connected to each other in a torus connection, with four links between every pair of boards (see Figure 3). From the switch on every board, four links are connected to the upper, lower, right and left neighbour switches respectively. This connection between boards is static and is implemented by physical wiring on the backplane of Hathi-2. It can only be modified by manually changing the link connections between the boards on the backplane.

The second level of connection is through the C004 switches on each board. This connection scheme allows the processor interconnection topology to be changed by software. Two transputers on the same board can be connected to each other through the on-board switch. If the transputers reside on neighbouring boards, the link connecting the two transputers goes through the two switches on the corresponding boards, using one of the four links in the static torus connection between the boards. If the two transputers do not reside on neighbouring boards, the link connecting them can be routed through an arbitrary number of intervening switches (see Figure 4). The performance penalty associated with this is a 20% communication capacity reduction for each C004 switch the link is routed through.

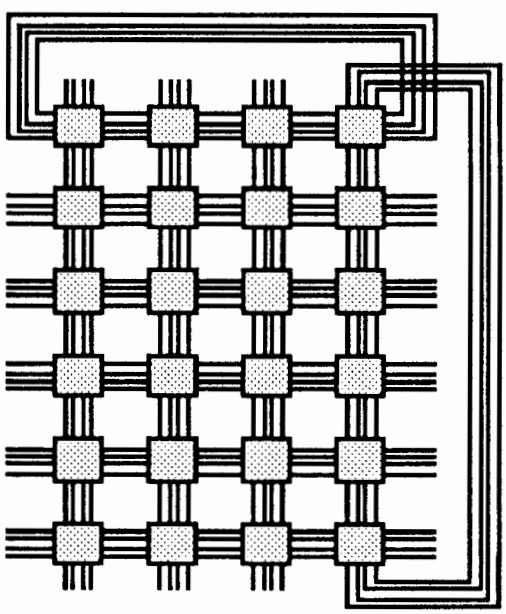


Figure 3. Hathi-2 board connections

Partitioning the system

Hathi-2 can be used by several users simultaneously. Each user gets their own subsystem, — a *partition*. A partition is allocated to one user and used as a separate multiprocessor system. The user can reconfigure the partition and execute code on it independently of all other users. The smallest unit that can form a partition in Hathi-2 is one board, i.e. four T800 transputers. Thus, Hathi-2 can be partitioned into at most 25 independent subsystems. The largest partition that can be formed consists of 96 T800 transputers. Partitioning is done by a set of manual switches on the backplane of Hathi-2 and can not be done during run time, as the system has to be rebooted after partitioning.

All T800 transputers are connected into a pipeline by the control signals Reset, Analyse and Error (see Figure 5). A partition is formed by disconnecting its control signals from the rest of the system, which is done by cutting off the RAE-signals by a manual switch on the backplane of Hathi-2. This isolates the partition from the rest of the system and prevents the RAE-signals from continuing to the transputers outside the partition. The user is connected to his partition with the RAE-signals and at least one link from the user's host transputer. Partitions must be formed so that the boards in one partition follow each other along the control signal line (see Figure 5). The board to which the user is connected (the first board in the partition) is called the 'master' board and the T212 transputer on this board is called the master T212.

Figures 3 and 5 show that the boards in Hathi-2 are organized as a 4-by-6 torus. One board (the 25th) is not part of the torus connection, and its links are not connected to any other board. This board is used as a separate partition consisting of four T800 transputers. Therefore, Hathi-2 is always partitoned in at least two subsystems.

Control system

Hathi-2 is logically divided into two separate subsystems: the multiprocessor system and the control system. The multiprocessor system consists of 100 T800 transputers. The control system consists of 25 T212 transputers, the C004 crossbar switches, the monitoring hardware and the interrupt system. The T212 transputers in the control system are connected to form a ring and are controlled by a separate host computer — *the control system host* (see Figure 6). The topology of the control system is static and cannot be changed. Partitioning Hathi-2 does not affect the control system. The user's host transputer is connected via one of the I/O links to the master T212 of the partition. This allows the user to communicate with the control system software using a set of predefined interface procedures.

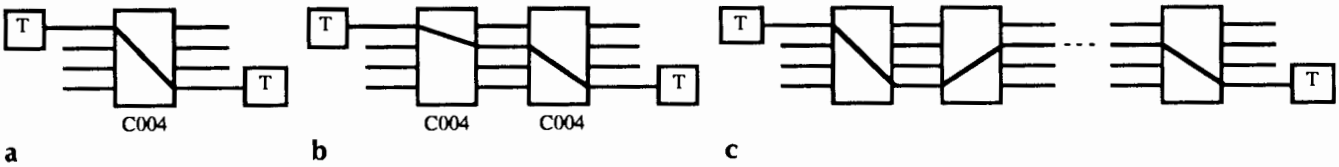


Figure 4. Connecting transputer links via C004 switches

The control system has two primary tasks:

- To control the settings of the C004 switches, i.e. to manage the reconfiguration of the system. The reconfiguration software is described below.
- To support monitoring of the activities in the system, i.e. to forward monitoring data from the T800 transputers to the control system host. The monitoring software is described below.

Hardware support for monitoring

The Hathi-2 architecture contains hardware support for monitoring the activities in the multiprocessor system. A process running on a T800 in the system can send information about its activities to the controlling T212 on the same board with a minimal overhead on the processor. Monitoring data is sent from a T800 to the T212 via a memory-mapped FIFO buffer accessible by the two transputers. The T800 can only write to the FIFO buffer and the T212 can only read from it (see Figure 7). The FIFO buffer is 512*9 bit.

A process running on a T800 can put data into the FIFO buffer by writing a byte (8 bit) to the memory location where the FIFO buffer is mapped. The T212 can get data from the FIFO by reading the buffer's memory location. The T212 has an empty-flag for each FIFO, indicating that the FIFO queue is empty. The T800 transputer has a corresponding 'full' flag, indicating that the FIFO is full. Additionally, when the buffer is full, an interrupt is generated on the T800 (described in the next subsection). If data is inserted into the FIFO when it is full, old data will be pushed out, thus causing loss of data.

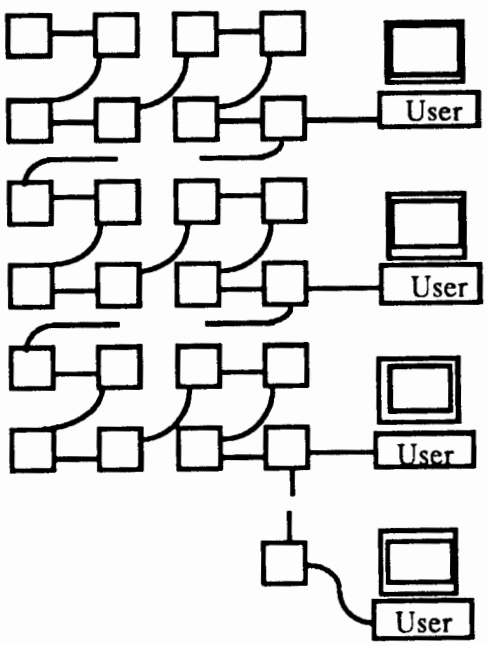


Figure 5. Reset, Analyse and Error signals

When monitoring activities in a partition, a synchrounous clock pulse — a *sync. interrupt* — can be generated by the master T212 in the partition. The sync. interrupt reaches all processors in the partition simultaneously, both the T800 and the T212 transputers. It is used for generating a global clock pulse, dividing the time into short intervals. On each sync. interrupt the *interval bit* on the transputer is complemented. The interval bit is used as the 9th bit in the FIFO, thus identifying to which time interval the monitoring data in the FIFO belongs. The T212 can also read the status of the interval bit from a register in its memory.

Hathi-2 also contains hardware support for measuring the load of the CPUs. This is done by observing the activity on the transputer bus. A high activity on the bus indicates that the CPU is busy; a low activity indicates that it is idle. The lowest address line on the transputer's memory interface is connected to a monostable multivibrator, which controls a 24-bit counter. On a rising edge on the bus line, the counter is incremented. If the counter is continuously enabled, it overflows after 3.36 s. For each sync. interrupt, the 16 most-significant bits of the counter are stored into a CPU load register, which can be read by the T212 transputer.

Monitoring data is gathered from the T800 transputers by the on-board T212, which reads data from the FIFO buffers and the CPU load register. The monitoring data is then forwarded along the T212 ring to the control system host, where it can be stored, analysed and presented to the user.

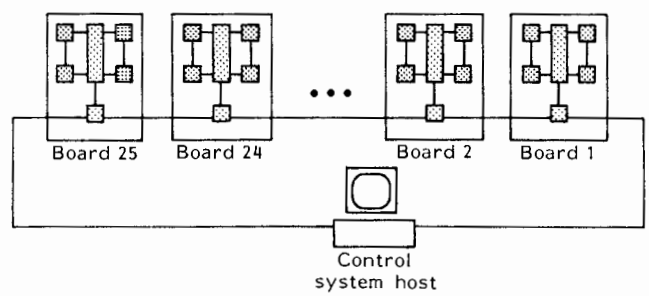


Figure 6. Control system

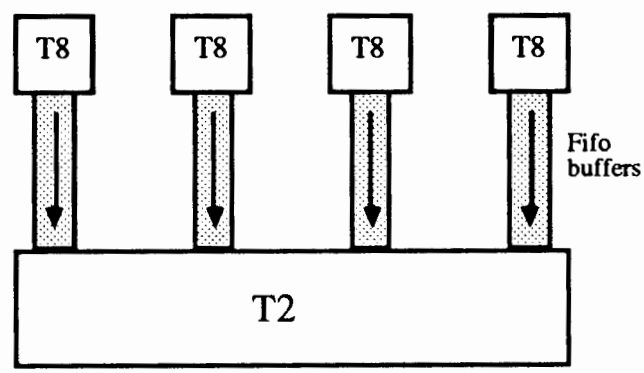


Figure 7. FIFO buffers

Interrupt system

The transputer architecture only supports one type of external interrupt, called an EVENT. The interrupt must be served by a process waiting on input from the EVENT channel. If there is no process waiting on an EVENT, the interrupt will have no effect on the processor. Hathi-2 has an extended interrupt system which is implemented using the transputer's EVENT pin. On Hathi-2, a T800 transputer can receive three different interrupts — *Full Flag (FF), T2int* and *Sync.* Upon all interrupts an EVENT signal is generated to the transputer, and the process waiting for an event is scheduled. This process must then read an interrupt status register to identify the cause of the interrupt.

The FF interrupt is raised by hardware when the FIFO buffer on a transputer becomes full. Additionally, the full flag is also set on the T800 transputer. The process handling the interrupt can then take measures to avoid loss of data in the FIFOs. The T2int interrupt is an interrupt from the T212 transputer on the same board. The T212 can send interrupts to any of the four T800 transputers on the same board. The Sync. interrupt is generated by the master T212 in a partition and reaches all transputers in that partition. It is used by the monitoring software (described below) to generate interval synchronization signals.

HATHI-2 HARDWARE ENVIRONMENT

The Hathi-2 system is located in the Department of Computer Science at Åbo Akademi. It is installed in a standard instrument cabinet, containing three 19 in racks, each with room for 14 boards. The cabinet contains the multiprocessor system, which consists of 25 boards, power supply and fans for cooling. The system is operated in a normal laboratory environment.

This section presents the host computers that can be used in the Hathi-2 system and the peripheral units that can be accessed from it. All connections between Hathi-2 and other units are via transputer links. The equipment connected to Hathi-2 must be located in the same room as the Hathi-2 system, as the maximum length of a link cable running at 10 Mbit s^{-1} is about 6 m, and for links running at 20 Mbit s^{-1} about 3 m.

Host computers

Hathi-2 is used as a back-end computing resource. The user edits, compiles and links programs on a host computer. The user's host must be equipped with at least one transputer to be able to run software which loads programs onto, and controls, Hathi-2.

The following host computers are connected to the Hathi-2 system:

- Sun-3/160M: this is equipped with a Niche NT1000 transputer board, which contains four T800 transputers, each with 2 Mbyte of memory. The Sun is connected to the local Ethernet network in Åbo Akademi, which can in turn be accessed from the national network connecting the Finnish universities, as well as from other international networks. This means that the system can be accessed from any terminal, regardless of where the user is physically located.
- IBM-PC AT: two IBM-PC AT computers with Inmos B004 transputer boards are connected to Hathi-2. One is used as the host computer in the control system (see above); the other is used as a user host.
- Apple Macintosh II: a Macintosh II with a Levco transputer board containing one T800 transputer and 4 Mbyte of memory is also available and can be used as a host computer.

Normally, Hathi-2 is partitioned into five partitions, of which four are connected to the host transputers in the Sun workstation and one is connected to an IBM PC host.

Peripheral units

The following peripheral units are attatched to Hathi-2 and can be used in programs running on the system.

- Graphics system: the graphics controller is an Inmos B007 graphics board containing one T414 transputer with 512 kbyte of dual-ported video RAM, 512 kbyte of program memory and an IMS G170 colour palette. The resolution of the graphics controller is 512 by 512 8-bit pixels. A 20 in Salora 445A RGB monitor is used as a graphics display.
- Mass storage system: An Inmos B005 20 Mbyte hard disc can be directly connected to any of the I/O links in Hathi-2. The disc can be used for storing intermediate data or results from a computation. The hard discs in the host computers (Sun, PC and Mac) can also be used as mass storage systems.

HATHI-2 SOFTWARE ENVIRONMENT

Hathi-2 is normally programmed in Occam, using the TDS[14] or the stand-alone Occam toolset. It can also be programmed using C or FORTRAN. Also available are the distributed operating systems Helios[15] and Trollius[16]. The operating systems can be executed on a partition of Hathi-2, using either a PC or a Sun workstation as a host processor.

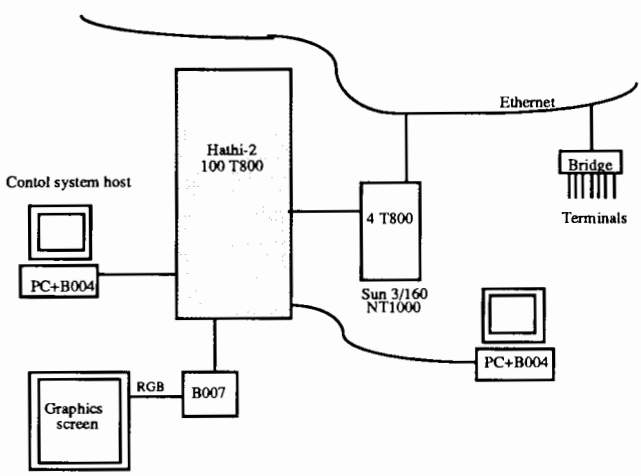


Figure 8. Hathi-2 hosts and peripheral units

Control system software

The control system in Hathi-2 is responsible for carrying out system administration and service routines. The user can request service from the control system by calling service request procedures, which communicate with the control system by sending messages over the link connecting the user's host to the control system. The system service functions do not impose any overhead on the computation in the multiprocessor system, as they are executed by the T212 transputers in the control system.

The user is connected to the control system through a link from the user's host transputer to the master T212 transputer in the partition. Communication between the user and the control system follows a specific protocol and should only be accessed through the service request procedures.

The control system software is based on a general message-passing system, which handles transparent forwarding of messages between processes in the control system. This is implemented by a communication kernel executing on each transputer in the control system. The communication kernel supports any type of messages and any number of processes.

The kernel forwards messages from a source process either to one destination process or to a group of processes using broadcasting or multicasting. Messages can be of any type and length. A message consists of an address header and the actual message data. Addresses consist of two fields: a processor number identifying a processor in the control system, and a process number identifying one of the system service processes executing on the control system.

Messages flow simultaneously in both directions in the ring. The communication kernel calculates the shortest route to the destination node and forwards the message in that direction. The kernel prevents starvation of any single process by alternating the priority for communication between messages to be forwarded from a neighbouring process and messages from a service process on the same processor. The message-passing scheme is deadlock free[17].

Currently, there are three types of service processes executing on each T212 transputer in the control system:

- User host interface: handles communication between the user's host and the control system. It listens continuously to messages from the user or responses from the control system and forwards these either to the user's host or to the communication kernel. The process checks that all communication over the link between the user's host and the control system conforms to the defined protocol.
- Switch controller: controls the setting of the C004 switch on the board by sending commands to it over the switch control link. The process does not perform any extensive checks on the switch commands, but merely passes on the received commands to the C004 switch. The switch controller process is a part of the Transputer Network Topology administrator software.
- Monitor controller: handles generation of interval synchronization signals and gathering of monitoring data from the FIFO queues and from the load meter. The process reads the FIFO registers and the load meter register and collects monitoring data from one time interval to a packet, which is sent via the message-passing system to a node where the data can be stored in a file. The monitor control process is part of the Transputer Network Monitoring system.

TRANSPUTER NETWORK TOPOLOGY ADMINISTRATOR

The Transputer Network Topology (TNT) administrator is a utility that supports the reconfiguration capabilities of the distributed switching network in Hathi-2. The TNT administrator allows the user to change the interconnection of the processors in the partition. Only a static reconfiguration capability is supported, i.e. the network configuration can be changed only between separate program executions, not during run time.

Program structure

The TNT administrator software consists of three different parts: the user interface which is executed on the user's host transputer, a switch controller process executing on each transputer in the control system and the central topology administrator which is executed on the control system's host (see Figure 9).

- User interface: executes on the user's host transputer. It presents a menu of topology alternatives to the user, and sends a description of the chosen topology to the central topology administrator via the message-passing system. The user interface process checks the size of the partition it is connected to and allows the users to select only those alternatives that can be established on this partition. The descriptions of the topologies are stored in files on the user's host.
- Switch controller: one of the system service processes in the control system. It receives switch control commands from the central topology administrator and forwards them to the C004 switch.
- Central topology administrator: this process is executed on the control system host and is responsible for keeping record of all established link connections in Hathi-2. It receives topology descriptions from the message-passing system and translates these to switch commands, using a shortest-path routing algorithm. It also takes into consideration the limitations imposed by the limited number of links between neighbouring boards. If it is not able to establish a topology, it send an error message back to the user.

The user selects a processor interconnection topology from a library of predefined topologies. The user interface

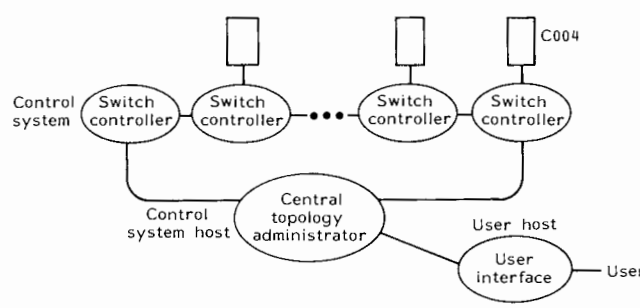


Figure 9. Structure of the TNT administrator

process sends a reconfiguration request to the central topology administrator and waits for an acknowledgement. When the acknowledgement is received, the process sends the topology description via the message-passing system to the central topology administrator process on the control systems host. The description of the selected topology is then interpreted and translated to switch commands. These are sent via the message-passing system to the switch controller processes, which in their turn pass them to the C004 crossbar switches. The tool does not allow a user to change the interconnection of processors other than those belonging to his own partition. The central topology administrator queues the incoming reconfiguration requests and serves one request at a time, thus allowing several users to request reconfiguration simultaneously.

The TNT administrator is menu driven. The tool is designed to have minimal interaction with the user. Therefore, it automatically checks the size of the user's partition and displays only those topologies that can be established on the partition.

The user can extend the topology library with new topology descriptions. These consist of a number of connection statements, where a connection statement specifies that two processors should be connected to each other with a link. A connection statements has the form

Connect processor 2 link 1 to processor 5 link 3

The processor numbering scheme used in the connection statements is obtained from a partition map, which assigns a unique logical number to all processors in a partition. The link numbers correspond to the transputer link numbers 0 - 3.

Link routing

The routing mechanism used in the TNT administrator is a simple routing algorithm based on Djikstra's shortest-path algorithm[18]. The algorithm takes as input a number of node pair connections, described by connection statements, which are to be established on the Hathi-2 switching network. For each link connection the routing algorithm finds the shortest path through the switching network. The algorithm is sequential, i.e. the connections are established sequentially.

The distributed switching network allows every processor to be connected to any other processor in the network. However, there are limitations on which connections can be established simultaneously, due to the limited number of links available between neighbouring boards. It is possible that processors on different boards cannot be connected because there is no path available between the respective boards, i.e. all paths are blocked by other connections using these paths.

The algorithm uses no backtracking or heuristics to find the optimal path through the network. It will either find a connection between a pair of nodes or be unable to connect the nodes, depending on which connections are already established in the switching network. The connections are initially sorted in ascending order on the minimum distance between the nodes, to allow the shorter connections to be established first. The TNT administrator checks that all connections are established within a user's partition and thus prevents a user from corrupting the topology of another partition.

TRANSPUTER NETWORK MONITORING SYSTEM

The Transputer Network Monitoring system is a utility with which the user can monitor the performance of a parallel program executing on Hathi-2. Monitoring is done by collecting information about the utilization of the CPU and the communication links from the processor load meter and the FIFO buffers described above.

During monitoring, time is divided into equally long time intervals by a global synchronization signal, which is generated by the master T212 transputer in the user's partition. Monitoring data is sampled once every time interval, and sent via the message-passing system to a hard disc for storage. This creates a trace of the execution, which describes the dynamic behaviour of the executed program. The monitoring data is presented to the user in graphical form.

Information about processor load is generated automatically by the load meter hardware and does not affect the computation on the T800 transputers. Information about link communication has to be written by the T800 transputers to the FIFO buffers, and thus causes some overhead on the computation. The user has to insert some additional code into the program to write monitoring information into the FIFO buffers. This information does not necessarily have to concern link communication; it can also report on other events the user wishes to monitor during program execution. However, the amount of additional computation introduced by these monitoring statements is very small and can be neglected.

Monitoring can be done either off-line or on-line. In off-line monitoring, data is gathered during the program execution and the data generated is examined after execution. This allows the user to examine the generated data at his own pace by stepping through the monitoring data. In on-line monitoring, data is presented graphically or numerically during program execution without any intermediate storage. On-line and off-line monitoring can also be combined, so that monitoring data is both presented and stored in a file simultaneously.

Program structure

The Transputer Network Monitoring system consists of four different parts: a monitor data multiplexer process executing on each T800 transputer being monitored, a monitor control process executing on each T212 transputer in the control system, a collector process executing on the control system host, and a presentation process executing on the user's host transputer (see Figure 10).

- Monitor data multiplexer: runs in parallel with the monitored user program on the T800 transputers. The process acts as a multiplexer, which collects raw monitoring data, e.g. messages about link usage, from the user processes and outputs the data as packets to the FIFO buffer once every time interval.
- Monitor controller: one of the system service processes in the control system. The process reads the FIFO buffers and the processor load meter register every time interval. The monitoring data is sent via the

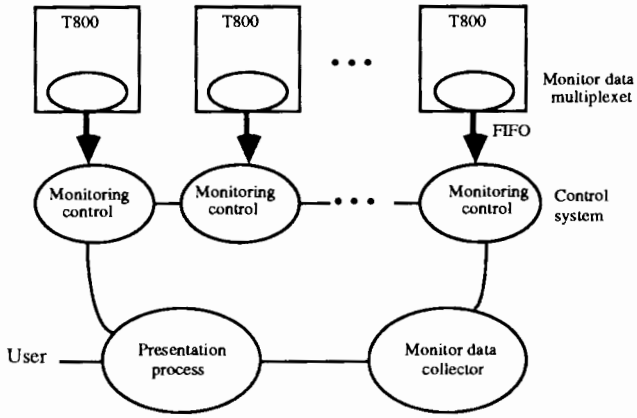


Figure 10. Transputer Network Monitoring system

message-passing system to the collector process. The monitor control process executing on the master T212 transputer in the partition is also responsible for generating a synchronization signal every time interval. The length of the interval can be varied from 50 ms to 3 s.

- Collector: runs on the control system host, or on some other processor connected to a hard disc unit. The process collects the monitoring data sent from the monitor control process and saves the data onto disc. The data file can be retrieved by the presentation process on the user's host transputer for presentation to the user.
- Presentation process: presents the monitoring data to the user (see Figure 11). The monitoring data is read from the hard disc and the necessary calculations are performed on the data. The user may, for instance, wish to view the results of the monitoring with a longer time interval than the sampling interval. In this case, data from a number of sampling intervals are read from the disc and the mean values for the utilization are calculated. The presentation is based on the processor structure of the monitored program, i.e. the utilization of the CPUs and links are shown on a graphical representation of the processor topology.

Monitoring a program

To monitor the resource utilization of a parallel program executed on Hathi-2, the user has to modify his program by adding some monitoring code to it. First, the monitor data multiplexer process has to be executed in parallel with the other processes on each monitored T800 transputer. Second, the user has to call a pair of predefined procedures for every communication statement that is to be reported to the monitoring system. The procedures are always called pairwise, one before the communication statement and one after the communication statement. To monitor a communication statement *C ! X* (or equivalently for an input statement *C ? X*) the following code has to be inserted into the user's program:

StartCommunication (linknumber)
C ! X
EndCommunication (linknumber, SIZE(X))

The call to the procedure *StartCommunication* will start a timer for the channel specified by the parameter *link-*

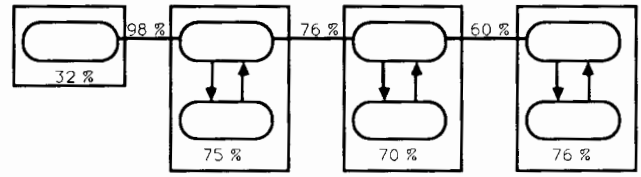


Figure 11. Presentation of monitoring data

number. The value of this timer is read by the call to *EndCommunication*. The time for the communication and the number of bytes sent over the channel is sent as a message to the multiplexer process. The total communication time and number of transferred bytes are accumulated for each time interval, and then sent through the FIFO buffer to the control system when the next interval interrupt signal is received.

CONCLUSIONS AND FUTURE WORK

The system has been in use for over a year, and no severe hardware fault has yet occurred. This indicates that the transputer technology used in Hathi-2 is reliable. The system has proved to be flexible to use, as it is possible for the user to adapt the system to the demands. The flexibility of the system is based on the following properties of Hathi-2:

- Hathi-2 can be used from different types of host systems (e.g. Sun, IBM-PC, Macintosh). This gives the user the ability to choose the environment best suited to the demands. It also gives the user access to a large number of different programming environments and languages for transputer-based systems.
- Hathi-2 can be partitioned into a number of smaller independent subsystems and used by several users at the same time. This makes it possible to give a larger number of users access to the system and to make more efficient use of it. This is an important feature, as Hathi-2 is mainly used for program development and the users are normally satisfied with a small number of transputers.
- The size of the partitions can be varied. Normally, the system is partitioned into five partitions, of which the largest partition has 32 T800 transputers. However, for larger test runs a partition with up to 96 T800 transputers can be formed.
- The Hathi-2 system can easily be extended. More processors can be added to the system just by adding new boards. To maintain the torus connection between the boards, the system should be incremented in steps of four boards, i.e. 16 T800 transputers. However, if new boards are added as separate partitions (like the 25th board) the system can be enhanced in steps of one board (four T800 transputers). Also, commercially-available transputer hardware, like graphics systems and hard discs, can be connected to the Hathi-2 system via the I/O links.
- Most commercially-available software for transputer-based systems can be executed on Hathi-2. Basically, the system consists of a number of transputers connected to each other via links. The additional features in the Hathi-2 architecture, like the distributed switching network and the control system, do not have to be taken into consideration by a program running on Hathi-2.

The Hathi-2 system is presently being upgraded with more memory for the T800 transputers. Each T800 transputer will have 1.25 Mbyte of memory, raising the total memory capacity of the system from 25 Mbyte to 125 Mbyte. Furthermore, a number of hard discs will be added to Hathi-2. The discs will be directly connected to an I/O link and will not be dependent on a host computer, thus allowing much faster disc access.

A number of program development tools are currently being developed for the Hathi-2 system. Among them is a tool for mapping parallel programs automatically to the processor network from a graphical description. Development of the user interface to the monitoring tool will allow the user to browse through monitoring information, and animated graphical representation of parallel program execution is also planned. A single graphical user interface will run on a Sun workstation.

ACKNOWLEDGEMENTS

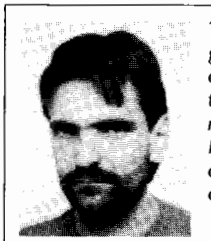
The Hathi-2 multiprocessor system was designed and built in the Hathi project, which was financed by The Technology Development Center (TEKES), The Academy of Finland, Åbo Akademi University and The Technical Research Centre of Finland (VTT). Part of the work has been done in the FINSOFT III research programme, financed by TEKES. The authors wish to thank Kari Leppälä, Kari Pehkonen and Kyösti Rautiola at VTT/TKO in Oulu and Tapani Äijänen at Jyväskylä Institute of Technology, for their central contribution in developing the Hathi-2 hardware system. The authors also wish to thank everybody who has participated in the software development for the Hathi-2 system: Tom Björkholm, Jonny Boman, Edward Eriksson, Jens Granlund, Pekka Kuusela, Stefan Levander, Yngve Nyman, Aarne Rantala, Antti Raunio, Kaisa Sere, Ulla Solin, Lena Ståhl, Dan-Johan Still, Marina Walldén, Patrik Waxlax, Sami Viitanen and Göran Öhman.

REFERENCES

1 **Äijänen, T** 'Distributed interconnection of a reconfigurable multicomputer system' *Microprocessing Microprog.* Vol 23 (1988) pp 243–246
2 **Pehkonen, K** 'A dynamically reconfigurable parallel computer Hathi-2' *Licentiate thesis, University of Oulu,* Finland (1989)
3 **Aspnäs, M and Malén, T-E** 'Hathi-2 users guide' *Reports on Computer Science, Ser B, No. 6, Åbo Akademi* (1989)
4 **Walldén, M and Sere, K** 'Free-text retrieval on transputer networks' *Microprocessors Microsyst.* Vol 13 No 3 (April 1989) pp 179–187
5 **Öhman, G A, Malén, T-E and Kuusela, P** 'Numerical fluid flow and heat transfer calculations on multiprocessor systems' *Heat Engineering Laboratory report 88-3, Department of Chemical Engineering, Åbo Akademi* (1988)
6 **Rantala, A, Raunio, A and Still, D-J** 'Some parallel implementations of a geometric image transformation' *Proc. 6th Scandinavian Conf. Image Analysis, Oulu, Finland* (1989)
7 **Inmos Limited** *Transputer Reference Manual* Prentice-Hall, UK (1988)
8 **Inmos Limited** *Transputer Instruction Set: a compiler writer's guide* Prentice-Hall, UK (1988)
9 **Inmos Limited** *Occam 2 Reference Manual* Prentice-Hall, UK (1988)
10 **Jones, G and Goldsmith, M** *Programming in Occam 2* Prentice-Hall, UK (1988)
11 **Bowler, K C and Kenway, R D** 'Physics on parallel computers, part 1: the new technolgy' *Contemp. Phys.* Vol 28 No 6 (1987) pp 573–598
12 **Bowler, K C, Bruce, A D, Kenway, R D, Pawley, G S and Wallace, D J** 'Exploiting highly concurrent computers for physics' *Physics Today* (October 1987)
13 **Nicole, D A** 'Esprit Project 1085, reconfigurable transputer processor architecture' *Proc. CONPAR 88, UMIST, Manchester, UK* (1988)
14 **Inmos Limited** *Transputer Development System* Prentice-Hall, UK (1988)
15 **Perihelion Software Limited** *The Helios Operating System* Prentice-Hall, UK (1989)
16 **Burns, G, et al** 'Trillium operating system' *Proc. 3rd Conf. Hypercube Concurrent Computers and Applications* ACM (1988) pp 374–376
17 **Roebbers, H and Vlot, M** 'A communication processor on the transputer' in **Bakkers, A (Ed.)** *Applying Transputer Based Parallel Machines (Proc. 10th OUG Technical Meeting)* IOS, The Netherlands (1989)
18 **Schwartz, M** *Telecommunication Networks Protocols, Modelling and Analysis* Addison-Wesley (1987)

Mats Aspnäs received an MSc in computer science in 1987 from ÅBO Akademi. He worked in the Hathi research group from 1986 to 1988, and is now working in the Millipede project at Åbo Akademi. His major research interests are distributed computing and monitoring of parallel systems.

Ralph-Johan Back received a PhD in computer science in 1978 from the University of Helsinki, Finland. He worked as a research assistant at the University of Helsinki Computing Centre from 1973 to 1979. He was a visiting scientist at the Mathematical Center in Amsterdam, The Netherlands, during 1979-80, and a senior researcher at the Academy of Finland from 1981 to 1984. He has been professor of computer science at Åbo Akademi University since 1983. His research interests include program methodolgy, program verification, distributed and parallel programs, programming and specification language design and semantics.



Tor-Erik Malén worked in the Hathi research group at Åbo Akademi from 1986 to 1988, during which he wrote his Masters thesis, and in the Millipede project until autumn 1989. He is now at the Computing Centre of Åbo Akademi. His major research interests are distributed computing and data aquisition and process control.