FTCS-11

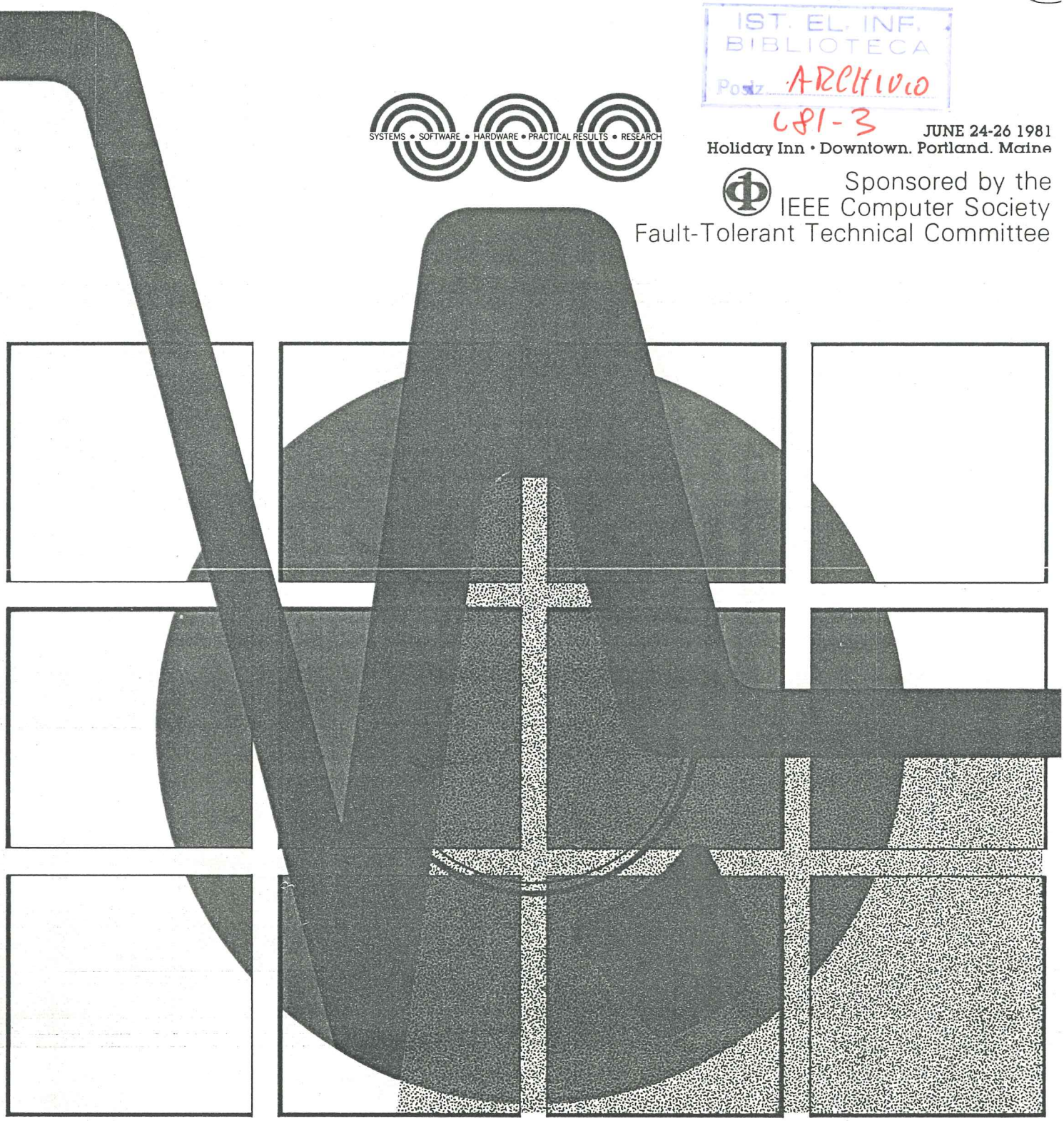
DIGEST of PAPERS

The Eleventh Annual International Symposium on Fault-Tolerant Computing

SYSTEMS • SOFTWARE • HARDWARE • PRACTICAL RESULTS • RESEARCH

JUNE 24-26 1981
Holiday Inn • Downtown. Portland. Maine

Sponsored by the
IEEE Computer Society
Fault-Tolerant Technical Committee



IEEE Catalog No. 81CH1600-6
Library of Congress No. 79-613257
Computer Society Order No. 350

IEEE COMPUTER SOCIETY PRESS

MuTEAM: ARCHITECTURAL INSIGHTS OF A DISTRIBUTED MULTIMICROPROCESSOR SYSTEM

G. Cioffi*, P. Corsini**, G. Frosini***, L. Lopriore****

*Istituto di Automatica, Università di Roma, Roma, Italy
**Istituto di Elettronica e Telecomunicazioni, Università di Pisa, Pisa, Italy
***Istituto di Informatica, Università di Ancona, Ancona, Italy
****Istituto di Elaborazione dell'Informazione, Consiglio Nazionale delle Ricerche, Pisa, Italy

ABSTRACT

A multimicroprocessor system prototype is presented, intended for embedded real-time applications. Main goals are modularity, expandability, error confinement and robustness. Such goals have been achieved by the definition of a system architecture which unifies hardware configuration, fault tolerance and operating system kernel requirements.

INTRODUCTION

In this paper, the architecture of the MuTEAM multimicroprocessor system is described in some detail /1/. The system is constituted of a set of clusters, loosely connected via serial lines. Each cluster consists of a set of autonomous Computer Elements tightly coupled through a parallel bus. In a cluster the shared memory is partitioned into blocks, one block for each Computer Element. A memory block in a given Computer Element can be accessed by the processors belonging to the other Computer Elements via the parallel bus, and by the processor in the same Computer Element through a privileged access port. Each Computer Element is provided with private memory and private I/O.

The address generated by a processor is segmented, and the segment name is mapped by an address translator either in the private address space or in the global cluster address space.

A protection mechanism is provided for controlling the accesses to shared memory segments. This protection mechanism is of the access control list type /2/, and the accessors are the processors of the cluster. This kind of protection can be simply implemented and assures the required degree of error confinement in both cases of hardware and logical failures. Moreover, this protection, integrated with a software protection among processes performed at the kernel level, assures a correct implementation of self-diagnostic procedures.

SYSTEM CONFIGURATION

The overall system is constituted of clusters, interconnected by means of a set of serial links. Each cluster consists of up to 16 Computer Elements (or nodes), that communicate throughout a parallel bus (cluster bus). The partition of the system into different clusters does not transpare over the kernel level, so that the system appears as a set of identical virtual nodes. Other multiprocessor systems constituted of clusters have been described in the literature. One of the best known is Cm* /3/, other examples are presented in /4/ and /5/.

In the following we will examine the architecture of a single cluster in some detail, while the problems related to the interconnections among clusters are under investigation. The cluster configuration is shown in Fig. 1. Nodes

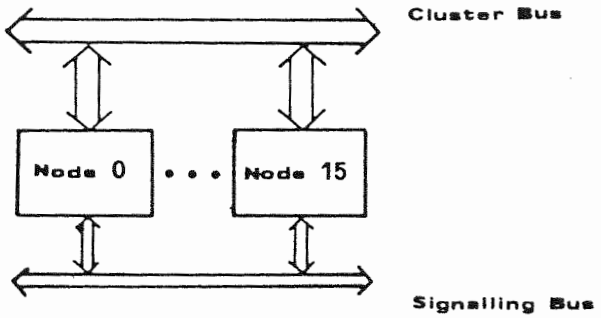


Fig. 1 Cluster configuration

are connected, besides the cluster bus, also by means of a secondary bus (constituted of few wires), called signalling bus and mainly used for interprocessor interrupt. The structure of a node is given in Fig. 2. Its main parts are: 1) a CPU with segmentation facilities, that actually is a microprocessor Zilog Z8001; 2) an Address Translator (AT); 3) a Shared Memory Subsystem; 4) a Private Memory and I/O Subsystem; 5) a Communication Controller (CC) for the transmission (to other nodes via the signalling bus) of

This work has been supported by the Computer Science Program of the Italian National Research Council

interrupts and short urgent messages; and 6) an arbitration logic (not shown in the figure) for accessing the cluster bus.

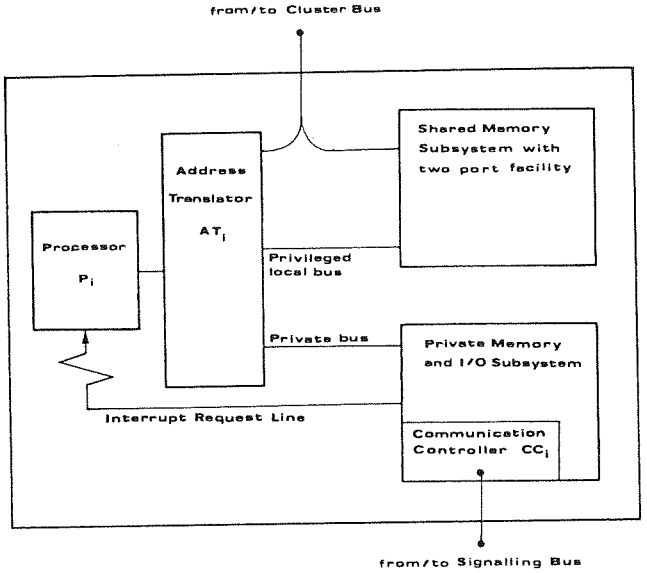


Fig. 2 Block diagram of the i-th node

VIRTUAL MEMORY ORGANIZATION

The address generated by the CPU consists of a segment field (7 bits), specifying the segment name local to the running process, and an offset field (16 bits). The AT logic translates the pair {running process name, segment local name} into a segment virtual name. The virtual space contains 2^{14} shared segments and 2^{14} private segments. Such a space is partitioned into 32 blocks, 16 blocks of shared segments (shared blocks) and 16 blocks of private segments (private blocks) where each block contains up to 2^{10} segments. Each node (Fig. 3) has associated a shared block and a private block. The segment virtual name generated in a node can refer to any shared block, and to the private block associated with the node itself. The number of the registers of the AT logic must be sufficient to store the information for translating the segment local names of at least a kernel process and a user process at the same time. Actually AT has 2^{10} registers, that are sufficient to store the information for 8 processes. Each register has 15 bits whose contents is interpreted according to the following criteria:

bit 0: if set, specifies that the segment belongs to a private block; otherwise, specifies that the segment belongs to a shared block;

bits 1-4: specify the node to which the segment belongs;

bits 5-14: specify a segment address in the pertinent block.

It should be noted that the AT logic, when it has generated the virtual address of a segment, accesses the cluster bus only if the segment belongs to the shared block associated to a different node. If the segment belongs to the

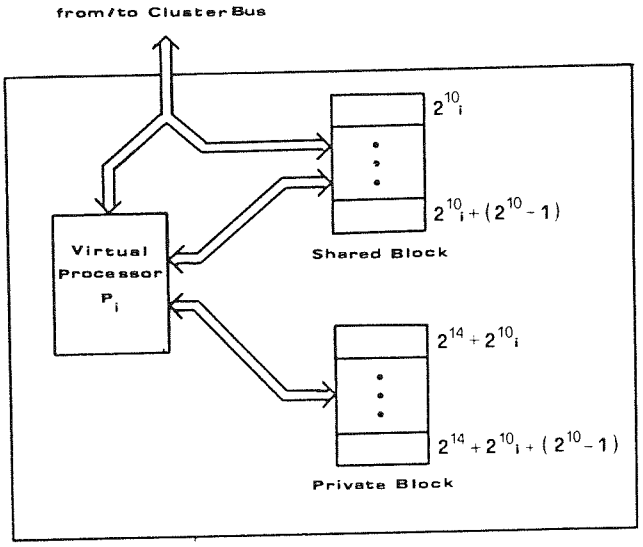


Fig. 3 Memory blocks associated to the i-th node

shared block associated to the node, the privileged local bus is involved in the memory operation, without occupying the cluster bus. Contentions due to simultaneous attempts to access a shared block via the cluster bus (from a different node), and via the privileged local bus (from the same node) are resolved by a proper local arbiter. Finally, if the segment belongs to the private block, the private memory subsystem is accessed via a private dedicated bus.

PHYSICAL ADDRESS GENERATION AND PROTECTION MECHANISMS

The Shared Memory Subsystem of a node receives a complete virtual address, which is constituted of a 10-bit segment virtual name and a 16-bit offset. A proper Relocation Unit translates the complete virtual address into a 20-bit physical address relative to the shared physical memory bank, whose maximal capacity is 1M bytes. The Relocation Unit mainly consists of 2^{10} registers, a register for each segment. Every register is 32 bit long, and is organized in two 16-bit fields: a field specifies the limit of the

segment, while the other contains an integer that multiplied by 16 represents the segment base in the physical memory. Thus a minimal segment length of 16 bytes is allowed.

A similar Relocation Unit is included also in the Private Memory and I/O Subsystem.

The Shared Memory Subsystem is also provided with a Protection Unit, which implements an access control list technique for protecting shared segments. Accessors are the processors of the cluster, and so the complete virtual address, transmitted on the cluster bus, is always paired with the 4-bit name of the processor generating the address. At present, the provided access rights on segments are Read/Only, Read/Write and No Right.

The Protection Unit mainly consists of 2^{10} registers, each being 32 bit long. A register implements the access list for a segment belonging to the shared block of the node. More precisely, the {i-th, (i+16)-th} bits specify the access rights of the i-th processor of the cluster, i=0,1,...,15. A dedicated status line of the cluster bus specifies the type of the access attempt on the actually addressed segment. A proper access violation checker compares the access type with the rights of the accessing processor on the specific segment. The block diagram of the Protection Unit is shown in Fig. 4.

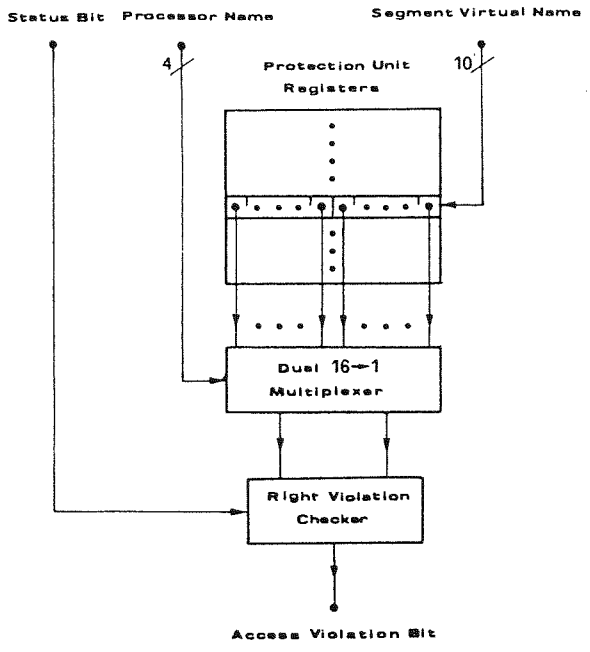


Fig. 4 Block diagram of the Protection Unit

The protection environment is completed at the hardware level by a feature of the AT logic that, in case of a write operation, requests the cluster bus if and only if the pertinent processor is in the supervisor state: this is consistent with the fact that write operations in shared non-local segments are not provided at the kernel level if the running process is a user process.

This protection mechanism assures a satisfactory degree of error confinement, in both cases of hardware and logic failures.

INTERPROCESSOR INTERRUPT SUBSYSTEM

An interprocessor interrupt subsystem is provided at the cluster level. The subsystem is mainly constituted of a set of Communication Controllers (CC's), a Communication Controller for each node, and of a 16-wire signalling bus. Each CC is actually a 8-bit microcomputer and is connected both to the signalling bus and to the private bus of the pertinent node. When a process p', running on the processor Pa, needs to interrupt a process p", running on the processor Pb, a≠b, the kernel of Pa sends an information packet to CCa. The CCa logic requests the signalling bus, and when it obtains the bus mastership, it sends the name of Pb. CCb identifies this name, so that a communication link is established between CCa and CCb. When the complete message has been transmitted via the link, CCb executes a proper kernel routine (according to the received message) whose effect is to test the state of process p" and, eventually, to send an interrupt to processor Pb. Moreover CCb, in case of messages coming from different Communication Controllers, enqueues them according to the priority of the processes to be interrupted. Besides interprocessor interrupts, the described subsystem implements a fast communication channel among processors, which will be utilized for the transmission of short, urgent messages.

REFERENCES

/1/ "The MuTEAM: Preliminary Presentation", Technical Report, MUMICRO Series, National Computer Science Program, 1980 (In Italian).

/2/ J.H. Saltzer, M.D. Schroeder, "The Protection of Information in Computer Systems", Proc. IEEE, Sept. 1975, pp. 1278-1308.

/3/ R.J. Swan et al. "Cm*-A Modular, Multimicroprocessor", Proc. 1977 NCC, AFIPS, Vol. 46, pp. 637-644.

/4/ G. Mazaré, " A Few Example of How to Use a Symmetrical Multimicroprocessor", Proc. 4th Annual Symp. Computer Architecture, Vol. 5, No. 7, pp. 57-62.

/5/ M. Ajmone et al., "Architecture, Communication Procedures and Performance Evaluation of the μ^* Multimicroprocessor System", Proc. 1st Int. Conf. Distributed Computing Systems, Huntsville, Oct. 1979, pp. 106-115.