OBJECTS FOR REAL TIME EXECUTIVES IN MESSAGE SWITCHING EXCHANGES WITH DISTRIBUTED ARCHITECTURE

J.A. Grompone, F. Simini

Interfase Ltda., Montevideo, Uruguay

ABSTRACT

The design of message exchanges with a redundant and multimaster architecture leads to the implementation of efficient real time kernels. We suggest the definition of new attributes for such objects as tasks, mailboxes and semaphores. These attributes (TIME EFFICIENT vs. RECOVERY EFFICIENT, PRIVATE vs. PUBLIC, INTERNAL vs. EXTERNAL, INTERRUPT vs. POLLED) allow the system software to deal efficiently with the typical aspects of distributed and real time operation. Design data based on telex switching exchanges show that the new concepts apply to over a third of the objects used.

OBJECTS FOR REAL TIME EXECUTIVES IN MESSAGE SWITCHING EXCHANGES WITH DISTRIBUTED ARCHITECTURE

J. A. Grompone, F. Simini

Interfase Ltda., Montevideo, Uruguay

1. INTRODUCTION

The design of telex exchanges implemented in a Multimaster and Redundant Processor environment (1, 2, 3) suggests a few comments on distributed system software. The inadequacy of existing real time executives (operating system kernels) for multiple processor systems has led to the design of efficient software/hardware structures and to the present definitions of necessary primitives.

In designing message electronic exchanges the time requirements are very demanding. In such applications the general purpose real time executives are useless because overhead times are not specified. This is felt particularly when synchronization of redundant processors is needed and when time critical communication between processors are designed. If a real time executive is to take care of these functions, it must include primitives with known overhead times. Time slices as short as 100 microseconds are implemented in the illustrated exchange.

The concepts of semaphore and mailbox, usually implemented for inter-task transfers, need to be further defined to encompass communications between masters belonging to the same bus and between masters in unfrequent but dense interaction (like redundant or front-end processors).

Real time executives should offer primitives to create, receive and modify semaphores in a multimaster environment. Semaphores should be dynamically definable by the user (much like programmable ports) to be either software or linked to a hardware line (incoming, outgoing or part of the bus). These flexible semaphores should therefore range

----

This work stems from a revision of the project of Telex Exchanges designed and produced in a joint effort by Interfase Ltda., Zabala 1372, Montevideo, Uruguay and Controles Ltda., Montevideo, Uruguay. The Model CTA Exchanges were first designed in 1976 and operate the Uruguayan public network since 1980.

from a mere software flag to a bit on an interrupt line.

Similarly, mailboxes should be managed by real time executives so as to interchange data not only between tasks, but also between distinct processors. In the illustrated example transfer rates of 30 Kbytes per second between processors are found.

In a multimaster environment the real time executive should also allow a master to alter a task running on another master.

2. DISTRIBUTED ARCHITECTURE

In order to discuss the primitives on an example of distributed architecture we present here the description of a switching system (up to 4096 lines) implemented with TMR (triple modular redundancy) (1).

The basic system architecture (TMR central processors and peripheral processors) can handle comfortably the control and switching of 32 peripheral processors (Fig. 1). A larger system is obtained with several slice processors working in parallel on the same Multimaster Bus (7).

The functions of the processors shown as an array in Fig. 1, can be divided in the following way:

1. Coordinating Processors and Common Memory M(0,0), M(0,1) and M(0,2). They execute all clock synchronization, reset a faulty redundant module and try to recover it; they also take care of a special high speed processor. The special peripheral units (statistical and billing logger, discs, operator's console) are connected to one of the Coordinating Processors.

2. Line Slice Processors M(i,j), i=1...N, j=0,1,2. They execute all switching tasks corresponding to 32 N peripheral processors (N slices of 512 lines each). They have access to the Multimaster Bus only when necessary and they usually process independently, reducing the effect of the von Neumann bottleneck between processors and common memory (6).

3. High Speed Peripheral Processor for special trunk connections. In case of a telex exchange it could handle teletext links.

4. Peripheral Processors P(i,j), i=1...N, j=1...32. They consist of interfaces between the lines and the slice processors and also act as majority voters.

It must be pointed out that all processors shown in Fig. 1, as well as the peripheral processors, are implemented with the same standard microcomputer board (INTEL Multibus computer series).

3. PRIMITIVES

The minimum resources of a real time executive include three types of objects: tasks, mailboxes and semaphores.

a) The design experience of the telex exchanges suggests that semaphores and mailboxes should be definable for the following attributes in addition to the usual ones:

- INTERNAL OR EXTERNAL, referring to accessibility by distinct processors connected by datapaths different from a Multimaster Bus (e.g. redundant processors).
- PRIVATE OR PUBLIC, referring to accessibility by distinct processors on the same bus (e.g. multimaster architecture).

b) Semaphores should be of a very flexible nature to meet the needs of efficient real time programming. The attribute INTERRUPT or POLLED to be assigned by program enables a semaphore to act either as a processor INTERRUPT or a FLAG and change from one to the other.

c) Timing characteristics should be well defined so as to cover distinct programming necessity, namely rare events that trigger programs of great priority and frequent events that require very fast and simple actions with as little overhead as possible. Our experience suggests the definition of additional task attributes:

- TIME-EFFICIENT, for tasks with a minimum overhead time but little capacity to save and retrieve parameters.
- RECOVERY-EFFICIENT, for tasks with extensive identification of parameters and system status but not well defined overhead.

d) Tasks can be managed within a master or triggered through the bus in other master processors. This leads to the following attribute distinction:

- PRIVATE, a task activated or terminated within a processor with no access to the bus.
- PUBLIC, a task activated or terminated on a processor by a task running on perhaps another processor of the bus.

Some of these attributes should be available simultaneously, for a given object.

4. EXAMPLES OF OUTSTANDING OBJECTS

Very few real time operating systems have been engineered for distributed architecture (15). The lack of time specifications and dedicated efficient structures turns them useless for switching systems design. Complex communication systems implemented with a distributed architecture use objects of a great variety. We present a review of objects that feature interesting combinations of attributes amongst the ones we have defined.

Semaphores

Semaphores are normally used for mutual exclusion and synchronization of tasks (5).

The clock semaphore usually acts as an INTERRUPT to all processors both through the bus (PUBLIC) and to redundant processors (EXTERNAL).

The synchronization of redundant processors is accomplished using EXTERNAL semaphores that are POLLED by the tasks running on the three Coordinating Processors.

Consider the case of an unlikely event that is taken care of by a sequence of tasks when it happens: temperature alarms, addressing error or any external event are possible examples. The event generates an INTERRUPT which is changed into a FLAG for later polling, possibly by other masters. This semaphore is essentially EXTERNAL and can be either PUBLIC or PRIVATE.

The reset of a discrepant TMR module is based on semaphores that originate in separate tasks as FLAGS; this causes an INTERRUPT to the Coordinating Processor of the discrepant module.

Mailboxes

In existing real time executives, mailboxes act as communication links between tasks running on a single machine (8, 9, 10, 14); they are said to be PRIVATE and INTERNAL according to our definitions.

Consider the procedure that allows a discrepant central processor to be reset and re-initialized by the other two: a memory dump is to be transferred without stopping the system operation. This mailbox can be implemented with a high speed link between processors that do not belong to the same bus (EXTERNAL and PRIVATE mailbox). This implementation is similar to the concept of a data path.

Another example of EXTERNAL and PRIVATE mailbox is the one that links each triplet of slice processors with the peripheral processors through the peripheral bus in Fig. 1.

We find INTERNAL and PUBLIC mailboxes when tasks running on different slices interchange information with the Coordinating Processors through the Multimaster Bus. This information includes statistics, time and date and, of course, the exchange table.

Billing data use a typical EXTERNAL and PUBLIC mailbox. EXTERNAL because they are to be transferred to an external device, like a tape logger or disc; PUBLIC because it is used by all the slice masters and therefore has access to the Multimaster Bus. The operator's console mailbox is another example of EXTERNAL and PUBLIC object.

Tasks

A task is a piece of software that includes its own code, priority level and data areas. The activation and termination of tasks is the responsibility of the executive. The tasks as they are usually implemented (8, 9, 10) are RECOVERY EFFICIENT. Moreover, the activation time is generally not constant, not known to the user and is usually of the order of hundreds of microseconds (4). Only occasionally are overhead time figures given (13) but then they concern executives that run on a single processor.

Consider on the other hand a task that polls the status of a line waiting for a call. This task must be TIME EFFICIENT since it must be executed once during each character time for all the lines of the slice. It is of paramount importance that this task be quick since nearly all the lines of communications systems are idle waiting for a call. There is no need for data recovery except for line identification. In the illustrated example the activation and execution of this elementary task takes 12 microseconds. The remaining 80 tasks or so that perform all the steps of the call are also TIME EFFICIENT tasks because they must be activated in the shortest possible time. Our executive features an overhead time of 7.5 microseconds for TIME EFFICIENT task activation.

The distinction between PRIVATE and PUBLIC tasks becomes clear if one considers for example the end of a call. The task that manages the end of a call for a line must activate the "end of call" task for the other line to run perhaps on another slice of the system. Therefore, the real time operating system must distinguish between PRIVATE and PUBLIC tasks because an access to the Multimaster Bus may have to be performed. This kind of task is common place in communications systems and was a major reason for considering existing executives as inadequate.

5. COMMENTS ON THE USE OF OBJECTS

Table 1 shows the distribution of task attributes in a typical multimaster and redundant processor telex exchange. About half of the program area is occupied by the conventional RECOVERY EFFICIENT and PRIVATE tasks. The other half includes mostly PUBLIC tasks, typical exchange functions that have access to other slice masters for activation or termination of tasks. Most of these tasks are TIME EFFICIENT while a small proportion (14%) includes postponable tasks that need extensive parameter extraction.

Consider the distribution of mailboxes in Table 2. A conventional executive could handle only about half of the mailboxes implemented in the illustrated example. The remaining mailboxes need either a dedicated hardware (data paths) or a Multimaster Bus conflict resolution structure.

Table 2 also shows the distribution of semaphores.

These tables show that at least one third of the objects found in a distributed system bear the attributes we are suggesting.

TABLE 1 - Distribution of tasks according to their attributes in a multimaster and redundant processor telex exchange

	Number of tasks	Percentage of program memory
Time-Efficient and Private Tasks	2	1%
Time-Efficient and Public Tasks	90	35%
Recovery-Efficient and Private Tasks	58	50%
Recovery-Efficient and Public Tasks	13	14%
	----	-----
Total	163	100%

TABLE 2 - Distribution of the suggested attributes amongst the mailboxes and semaphores of the same telex exchange

	Number of semaphores	Number of mailboxes	Percentage of mailbox area
Internal and Private	19	71	61%
Internal and Public	2	52	31%
External and Private	2	7	0.1%
External and Public	6	6	8%
	----	----	------
Total	29	136	100%

6. COMMENTS ON THE DESIGN OF THE NUCLEUS

The basic trade-off to be made is between a flexible nucleus and a dedicated, efficient nucleus. In the design of communication system we found that a flexible nucleus is inadequate, and that highly specialized functions (like TMR voting, excluding and recovery) call for dedicated primitives. It must be stressed that all external semaphores and mailboxes need a hardware implementation which is not a trivial matter. This aspect of real time executive design turns software and hardware issues impossible to separate. The nucleus of a distributed real time system is strongly dependent upon the processor used and the overall architecture. Some processors turn their use in communications systems very cumbersome (11), mainly due to the absence of support for real time actions and for parallel processing.

The operation of the switching system requires that all masters have access to the Common Memory as well as to their own on-board memory. This has two immediate consequences on the design of the real time executive:

- The executive must implement a memory mapping mechanism to allow access to both local objects and to objects located in Common Memory (PUBLIC/PRIVATE).

- The executive must run on all masters to coordinate all multimaster interactions.

Moreover, the real time executive of a distributed system must distinguish between the type of memory accessed (static memory or dynamic memory) because they present different timing characteristics:

- The refresh cycles of dynamic memory steal machine cycles to the normal sequence of accesses.

- Dynamic memory may have different read and write times.

The execution time of a task using dynamic memory is therefore not constant and the real time executive must control this aspect in a stringent timing specification environment. We suggest the nucleus should respect the following policy when assigning memory locations to mailboxes and working areas:

- STATIC MEMORY: for areas used by tasks that must have a known and specified execution time (e.g. for high speed inter-processor data buffers).

- DYNAMIC MEMORY: for non time-critical and internal processing.

- PRIVATE MEMORY: for areas frequently used to minimize access to the Multimaster Bus enhancing thus the overall throughput.

- PUBLIC MEMORY: for the control sections of public tasks and all public semaphores and mailboxes.

The creation and deletion of objects are the responsibility of real time executives. In a single master architecture (8, 10, 12, 13, 14) a suitable queuing mechanism manages the activation of objects. In a multimaster architecture the creation and killing of EXTERNAL or PUBLIC objects in real time applications is not a trivial matter because all concerned masters must be aware of the modification of an object that will concern them. Let us revisit two examples that will illustrate these difficulties.

Recall first, the killing of the "communication in progress" PUBLIC task at the end of a call. The line that terminates the call attempts to kill this task that runs for the other line on another slice processor. This killing can only be executed after receiving a clearence, for example, from the slice processor responsible for on-line monitoring of a subscriber.

The next example will illustrate the difficulties encountered when an EXTERNAL and PUBLIC mailbox is deleted. All slice processors generate billing information to a mailbox to an output device. Suppose the receiver of this mailbox is changed from a tape logger to a disk; before the tape logger mailbox is killed, all masters must acknowledge the change.

7. CONCLUSION

The main result presented in this paper is a set of definitions of objects whose implementation is useful for the design of efficient, distributed, real time systems. Recovery efficient tasks must be activated differently from time efficient tasks by the nucleus; memory area is to be assigned to requesting objects depending on Multimaster Bus access and memory technology. The implementation of mailboxes and semaphores includes the design of hardware links and the solution of bus conflicts. We feel that, although the definition of object attributes can be general, the nucleus of a real time distributed and dedicated system must be taylored for each particular application.

It must be stressed that, although multiprocessors have been commercially available for years, to our knowledge there are no real time executives that feature objects with the attributes we are suggesting.

8. REFERENCES

1. Grompone, J.A., Jerusalmi, J., Simini, F., 1983, "Telex Electronic Switching Exchanges Designed as Networks of Microcomputers", Comm. Int., accepted for publication.

2. Grompone J.A., 1982, "Viability of Domestic Projects: An Automatic Telex Exchange Produced in Uruguay", ITU Telecom. J., 49:

280-284.

3. Grompone, J.A., Mace, N.M., 1981, "Una central telex con procesadores triplicados", VIII Conf. Latinoamericana de Informatica (CLEI), Buenos Aires, 1: E-2, E-15.

4. Bonfiglio, J., Grompone, J.A., Jerusalmi, J., 1982. "Implementacion de un Tablero Electronico en el Uruguay", IX Conf. Latinoamericana de Informatica (CLEI), Lima, Peru.

5. Dijkstra, E.W., 1965, "Solution of a Problem in Concurrent Programming Control", Comm. ACM, 8: 569.

6. Backus, J. 1978, "Can programming be liberated from the von Neumann style? A functional style and its algebra of programs", Comm. ACM, 21: 613-641.

7. Adams, G., Rolander, T., 1978, "Design Motivations for Multiple Processor Microcomputer Systems", Comp. Des., 17: 81-88.

8. "iRMX 80 Real-time multi-tasking executive", and "iRMX 86 operating system", INTEL Corp., Sta. Clara, CA, 1981.

9. Galindo, J., 1982, "El software de base del sistema TESYS", SECOINSA, Madrid, Espana.

10. "TX5 Operating System", "DX10 Operating System", Texas Instruments.

11. Middelburg, C.A., 1982, "The Effect of the PDP-11 Architecture on Code Generations for CHILL", ACM Comp. Arch. News, 10 No. 2: 149-157.

12. Wicklund, T.L., 1982, "MINI-EXEC: A Portable Executive for 8-Bit Microcomputers", Comm. of the ACM, 25: 772-780.

13. Cheriton, D., Malcom, M., Melen, L., Sager, G., 1979, "THOTH, a Portable Real-Time Operating System", Comm. ACM, 22: 105-115.

14. Chien, J.P., 1980, "Multitasking Executive Simplifies Realtime Microprocessor System Design", Computer Design, 19, No. 1: 109-117.

15. Civera, P., Conte, G., Del Corso, D., Gregoretti, F., and Pasero, E., 1982, "The u* Project: an Experience with a Multimicroprocessor System", IEEE Micro, 2: No. 2, 38-50.

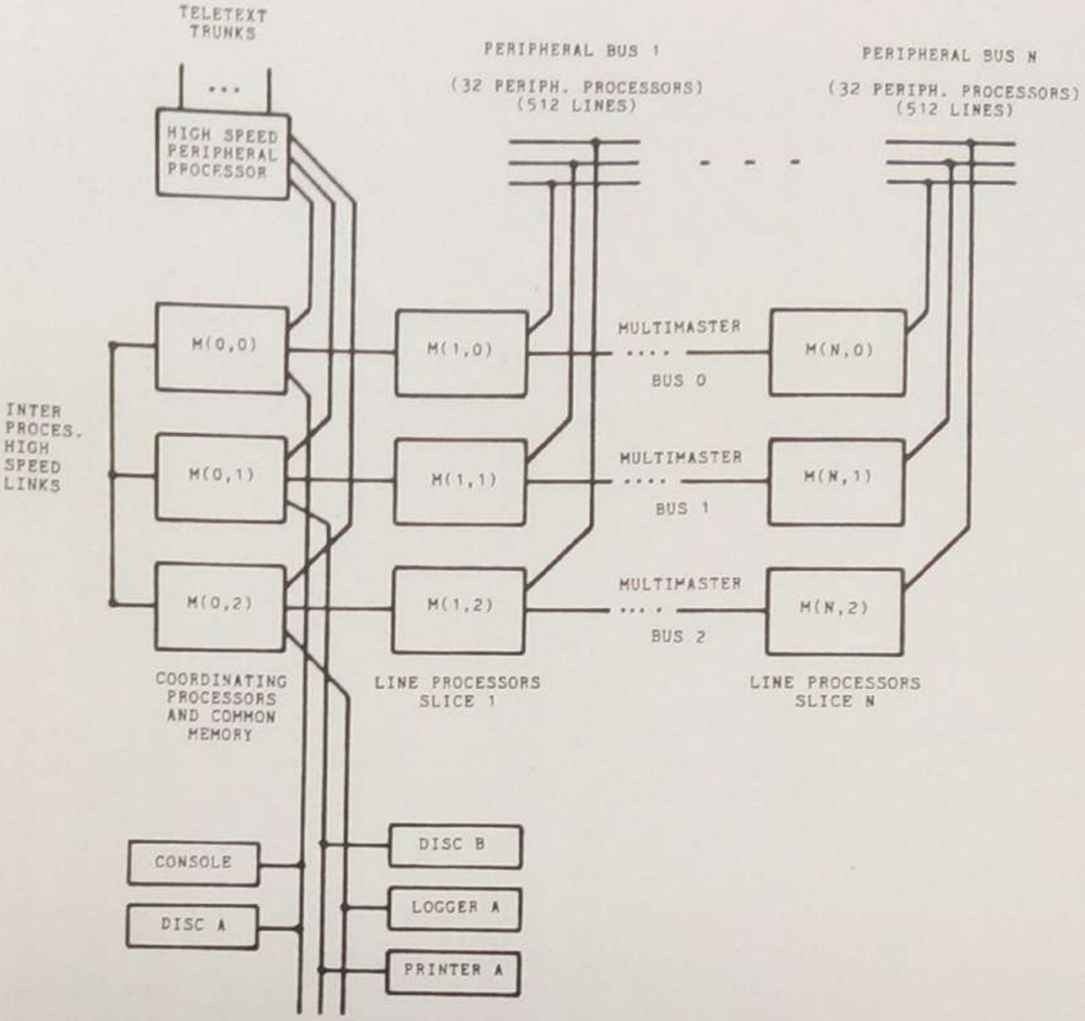

Fig. 1: Multimaster and TMR (triple modular redundancy) architecture of a telex exchange. All processors (coordinating, slice and peripheral processors) are implemented with the same standard single board computer. Note that in this representation a PUBLIC object has access to several processors in the HORIZONTAL direction while an EXTERNAL object has access to several masters in the VERTICAL direction.