



MULTI-PROCESSING SYSTEMS TECHNOLOGY REVIEW AND STANDARDISATION ISSUES *

0 INTRODUCTION

There are five main issues associated with the use of technology in military avionics systems from the perspective of the procurement body. These are performance, through life costs, operability, availability, reliability and maintainability (generally considered together) and timescales. Typically reduced cost and higher performance are in conflict; higher performance invariably costs more.

High performance processing engines tend to be expensive to design, develop, use and support throughout the lifecycle. Processor/memory architecture has developed significantly to arrive at the sophisticated microprocessors available today. This architecture, despite improvements such as feature size reduction, load/store architectures, pipelining and Reduced Instruction Sets, is now approaching the limit of its capabilities with respect to processing performance. Further enhancements of this type of processing architecture are becoming increasingly more complicated and expensive.

The lowering of life cycle cost for processors has arisen from a number of developments. Device processing and density improvements have led to enhanced capability and hence lower cost per unit processor. The development of test methodologies and techniques have improved reliability through the use of built in tests. Life cycle costs have, however,

* **This report is published by the Avionic Systems Standardisation Committee (ASSC) to advance the role of standardisation in avionics. The use of this is entirely voluntary, and its applicability and suitability for any particular use, including any patent infringement arising therefrom is the sole responsibility of the user.** Copies of this paper are obtainable from the ASSC Agency as an Avionic Systems Standardisation Committee publication.

been increased by the proliferation of devices and systems available and the corresponding requirement for through life support.

For the purposes of this document a multiprocessor system is one which has the ability to run more than one task within the same time period. See the definitions in paragraph 2. This is generally understood to be at least two CPUs communicating using a data bus, shared memory or other means. Traditional sequential microprocessors can be accommodated by the definition of a multiprocessor by switching between tasks. However, this is very inefficient and represents a significant overhead when compared with true multiprocessor systems.

It has become apparent that the processing requirements for the next generation of computers generally, and avionics specifically, will impose a requirement for some form of parallel processing capability. It will be essential to further reduce life cycle costs whilst moving towards the adoption of parallel systems in avionics systems.

Blank page

Table of contents		Page no.
0	Introduction	1
1	Scope	5
2	Definitions	5
3	Referenced documents	5
4	Current multi-processing technology	6
4.1	Classification Schemes	6
4.2	Hardware	7
4.3	Operating Systems	8
4.4	Software	9
4.5	System aspects	10
4.6	Evaluation	11
5	Standardisation issues	12
5.1	Hardware	12
5.2	Inter-processor communication	12
5.3	Operating System	13
Figures		
Figure 1 :	Simple performance increase analysis in parallel processing systems.	11

1 SCOPE

1.1 The purpose of this document is to provide the readership with a background knowledge of the issues involved in the area of multi-processing, particularly when applied in military avionics applications. It also includes guidance on the analysis and classification of multi-processing systems

1.2 It is hoped that the use of this document will reduce procurement or development costs for multi-processing systems and encourage common approaches allowing ease of information interchange.

1.3 For the purposes of this document multi-processing systems do not include those where the same hardware and tasks are duplicated for the purpose of improved reliability. Systems which have unassigned processing potential and the ability to reconfigure under fault conditions are, however, considered.

1.4 The target readership includes, but is not limited to, procurement managers, procurement advisory staff, project managers and systems implementors.

2 DEFINITIONS

The definitions contained in IEEE 610.12 have been used in this document. Where this is not the case the definitions are included in the text.

3 REFERENCED DOCUMENTS

IEEE 610.12 Glossary of Software Engineering Terminology
(update to ANSI/IEEE Std 729-1983), Nov, 1989.

ANSI/Mil-Std 1815A-1983 Reference Manual for the Ada¹ Programming Language, .

1 Ada is a registered trademark of the U.S. Government, Ada Joint Program Office

4 CURRENT MULTI-PROCESSING TECHNOLOGY

4.1 Classification Schemes

4.1.1 Attempts at classifying multi-processing architectures have sought to encompass aspects as diverse as instruction pipelines in RISC microprocessors and supercomputers. However, all multi-processing systems have three fundamental elements in common. These are:

- a. Processing Elements (PEs). These are the entities which provide the data processing capabilities of the system. These may be single function or multi-functional.
- b. Inter-PE Communication. Many PE inter-connection topologies are theoretically possible although most of these are impractical using current technology. Communication may be via shared memory or through the use of message passing. Common practicable topologies for message passing systems include linear, ring, grid, net and hypercube structures. Communication may be synchronous or asynchronous and can be data driven or demand driven. The topology adopted can affect the performance of some algorithms.
- c. Control strategies. Many schemes have been proposed for the classification of control strategies. One classifies the processing architecture in terms of that which induces processing.
 - i A control driven scheme is one where instructions are executed in a programmed sequence.
 - ii Data driven control is where an instruction is executed when the arguments required for that instruction are available.
 - iii Demand driven control is where instructions are executed as a result of the requirements of other active instructions.
 - iv Pattern driven control is where instructions are executed as a result of the existence of some pattern or condition.

Another commonly used classification scheme is provided by Flynn's Taxonomy. This scheme categorises the structures in terms of single (S) and multiple (M) instructions (I) and data (D). This leads to four possibilities;

SISD	- von Neumann architecture
SIMD	- array processors

- MISD - pipeline processors
- MIMD - multi-processors.

4.2 Hardware

The provision of efficient hardware support for the transfer of data between processing elements is fundamental in order to achieve maximum performance of multi-processor systems. This is available in a number of ways including shared memory, message passing and crossbar switches, etc.

4.2.1 Shared Memory The use of shared memory involves a number of processors using the same physical location for the transfer or sharing of data. In this method the data itself is not moved, access to the data is changed. Shared memory systems of this type can be implemented in a number of ways, the following are for multi-port systems:

- a. Pre-arranged memory locations are used to store data for access by other processors. When there is no other contact between the communicating processes transfer is hardware controlled and can therefore be made to be as fast as the hardware will allow. Whilst this technique can be made to be fast it is completely non-deterministic in systems where there is no provision for data freshness signalling. The provision of such signalling will present a processing overhead in the system. In systems without an indication of the currency of the data there is no guarantee that the data being taken from memory will be up to date. This can present a problem in systems where the correctly timed transfer of data is required (safety critical systems such as process or flight control, etc.). This problem is similar to coherency problems in microprocessors using caching techniques.
- b. The use of signalling techniques to indicate the freshness of data. The use of semaphores is one such technique. In this case the semaphore is set to indicate current data and reset to indicate that current data has been read. More sophisticated data coherency schemes using the operating system may be used at the expense of speed of transfer.
- c. The use of First In First Out (FIFO) memories. Such devices constitute a temporary storage location where data can be written to and read from both independently and asynchronously. This type of system is inherently unidirectional and can be used to interface between two processors operating at different speeds. This technique is sometimes used to implement the 'pipe' in Unix based systems.

4.2.2 Message passing. In this technique a protocol for the passing of the data around the system is established. This may be a protocol that is implicit in the hardware architecture, for example in systolic systems where the data is passed to a near neighbour. Alternatively there may be addressing and other information added to the basic data in order to route the data to a particular processor. The protocols used will depend, primarily, on the granularity of the system. Greater amounts of protocol overheads will be incurred in more loosely coupled systems.

4.2.3 Crossbar Switches. These devices are basically programmable multiport routing devices allowing n processors to be connected to m independent memory banks in a programmable manner.

4.3 Operating Systems

4.3.1 In multi-processing systems the operating system controls hardware, allocation of tasks to processing resources, inter-process communication and other features by means of the embedded firmware. Those operating systems designed for single user machines are usually single tasking systems, however multitasking versions of these operating systems may be available in some cases.

4.3.2 Operating systems in multi-processing applications will depend on the system architecture. Tightly coupled, closely coupled and loosely coupled systems architectures will impose different bounds and requirements on the particular operating system.

- a. Tightly coupled multi-processing architectures have a high degree of co-operation and the PEs usually have copies of the same operating environment, (e.g. communication and task allocation facilities) included as part of the system.
- b. Individual processing elements in closely coupled systems have the same operating systems but task allocation and communication are controlled by a supervisor processor.
- c. Loosely coupled multi-processing environments are those which use local or wide area networks running different applications and/or different operating systems. The common element in such an architecture is usually the communications interface. In closely coupled and loosely coupled system architectures overall system control may be distributed among the processing elements using distributed data base techniques and control passing algorithms.

4.3.3 Operating systems used in military avionics systems have been specific to the particular application. This has arisen from the performance requirements of such systems. The use of a standard operating system can impose a significant overhead which could adversely affect performance. In past systems there has been little to be gained by the use of a standard operating system for avionics systems across a number of disparate platforms. A standardised approach to the use of operating systems for multi-processing systems is, however, appropriate where system wide fault tolerance and reconfigurability are issues. The actual implementation details of the operating system could be left undefined but the entry and exit conditions are defined. This could cover hardware configuration tables, allocated task list, task waiting list priority scheme, etc. The benefits would include the development of interoperability in hierarchical multi-processing systems.

4.3.4 Examples of where standardisation activities are taking in this area include:

- a. The Ada Real Time Working Group (ARTWG) is part of the Special Interest Group for Ada (SIGAda) and is concerned with real time aspects of Ada.
- b. The Distributed Ada Real-time Kernel (DARK) is a software prototype intended to support real-time applications on distributed targets and was developed by the Software Engineering Institute in the US. DARK provides a set of Ada specifications which may be used by applications programs to perform parallel processing, inter-process message communication, device and interrupt handling, and other actions useful in real time programming. This kernel is implemented on each processing node in the system and contains information regarding process allocation and state, message queue state and scheduling parameters.

4.4 Software

4.4.1 Software in this context refers to both the application software and the tools used for its development. Compilers which allow the exploitation of algorithmic parallelism are available for most modern programming languages. Standardisation at the high order language (HOL) level is instituted with Ada as the required high level language for UK military systems. Since other languages need thorough justification for their use in military systems they will not be considered any further.

4.4.2 Ada was designed for use in embedded systems and hence incorporates support for multi-processing environments whether real (two or more processors and processes) or perceived (a single processor switching between tasks). This support is provided by a language entity called the task. Tasks are entities which proceed in parallel in that each task is allocated to a logical processor. Ada 9X is an updated version of Ada implementing multi-processing features.

4.4.3 Software development are an important factor in the life cycle costs of software. The requirement for standardisation of the interface to Ada tools is also reflected in the current development of the Portable Common Interface Set (PCIS).

4.5 System aspects

4.5.1 As discussed in paragraph 4.3 the operating system and overall system architecture are directly linked. In order to accrue the benefits of increased reliability, military avionics systems are moving towards a distributed architecture based on common processing modules and incorporating techniques such as on-line fault tolerance and reconfiguration. To support these techniques a distributed operating system is required. This has a local operating system for use by the processing element and also a version of the overall operating systems concerned with communications, system wide task allocation tables, task priority identifiers, processing resource identifiers, etc. Such an operating system, along with the use of configurable communication paths, especially in tightly coupled systems allows the on-line switching of tasks and facilitates system wide reconfigurability.

4.5.2 Multi-processors use several processing elements (logical or physical) to perform a single task. The ability of a task to be split into constituent sub-tasks which can operate in parallel and the mapping of these sub-tasks onto the processing architecture is an important issue in the design of systems. Using simple analysis it can be shown that the ratio of speeds of a single processor system to a multi-processing system is

$$\frac{\text{Single processor system}}{\text{multi-processor system}} = \frac{1}{(f/n) + (1-f)}$$

where n = number of processors

f = fraction of task which can make use of processors in parallel.

This is shown graphically in figure 1 where n has the values 1 to 10 (shown by the lines in the graph) and f is increased from 0.1 to 1.0 in each case.

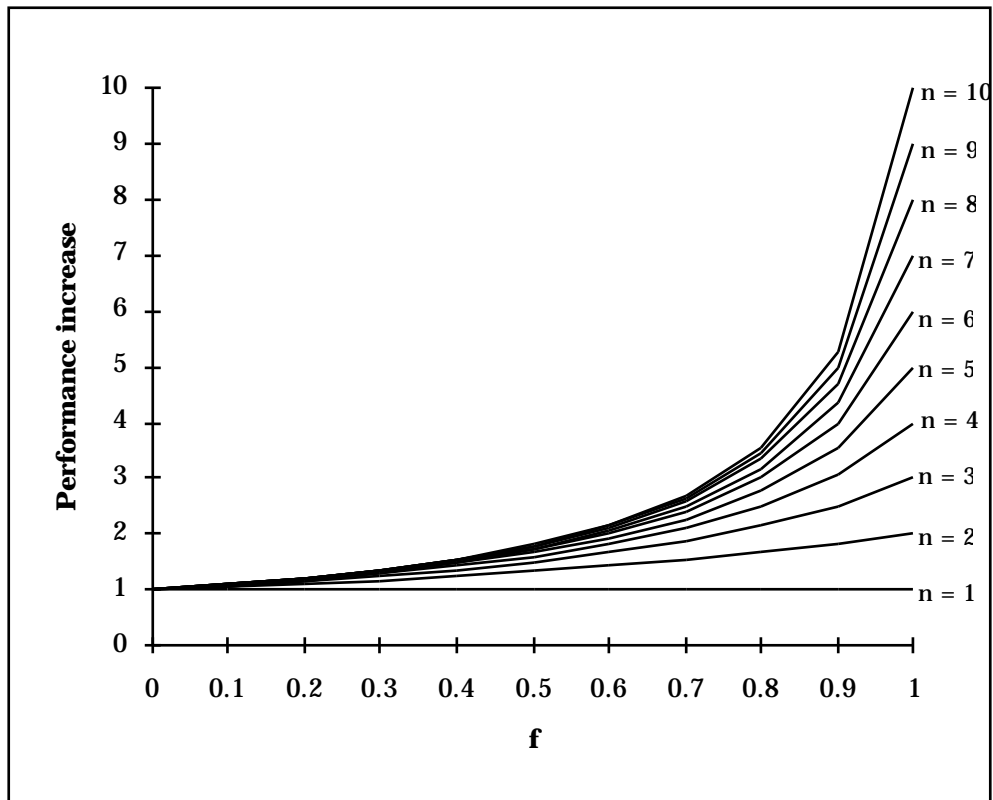


Figure 1 : Simple performance increase analysis in parallel processing systems.

This is a simple analysis taking no account of the effect of the communications and other operating system overheads. However, it shows that the fraction of the task that can be performed in parallel is the most important factor in performance increase.

The proportion of task which can be performed in parallel is fixed for a particular algorithm; a change in algorithm is required to improve the mapability, and hence performance, of the task.

4.6 Evaluation

The evaluation of multi-processing systems is difficult for a number of reasons e.g. number of processors, communication and operating system overheads, architecture/task mapping, etc. Comparative evaluation of a number of multi-processing systems, as with other processing systems, is best realised by implementing the end application on the candidate systems under consideration. For many systems, however, this will be impossible as the end application is not available. In this case system modelling can

provide the best indication of the viability of the proposed systems without recourse to implementation.

5 STANDARDISATION ISSUES

5.1 Hardware

5.1.1 From the above discussions the scope for standardisation in hardware aspects of multiple processing systems is limited and has similar constraints to those of standard avionics processing resources. Standardisation at the processing element (microprocessor) level is entirely appropriate on a project by project basis or where commonality of processing capabilities is required across a number of platforms. In this way it is possible to take advantage of developing technological capabilities and decrease development and through life support costs. This approach has been adopted in the US Pave Pillar programme and the UK Advanced Avionics Architectures and Packaging (A3P) programme. In these architectures some flexibility is allowed for systems where the processing requirements outstrip the capabilities of the standard processors.

5.1.2 In a multi-processing system, assuming the processing element chosen as standard has those features required to facilitate many configurations (local communications, reprogrammability, etc), it may be possible to standardise on a single processing element, a number of which can be used to implement a scalable processing system. This could then meet the needs of many processing requirements with a single processor. Such a processor would need to be configurable to different applications such as graphics processing, signal processing and general purpose processing. A reduced instruction set processor would be appropriate in this case with particular requirements facilitated by configurable ASICs.

5.2 Inter-processor communication

Data communication standards exist for application in closely coupled and loosely coupled multi-processing architectures. These are:

- a. backplane type standards e.g. PI-Bus, VME, Multibus, etc
- b. global type data communication standards e.g. Def Stan 00-18 (Part 2), prEN 3910, SAE HSDBs

None currently exist which are applicable at the tightly coupled processor level. Such a standard would allow different manufacturers' processors to exchange data simply and

easily. Standardisation at the protocol level would allow different schemes (shared memory, message passing) to be accommodated.

5.3 Operating System

Real time operating systems for avionics systems would appear to offer the most scope for standardisation in disparate application areas. Standardisation would have to be at a high level and written in Ada. The actual implementation details would not be standardised as these would vary with implementation. Examples of the areas where operating system calls are applicable include local memory management, global memory management, communications control, task control and allocation and configuration mapping.

R. P. Gray
Electronic Systems Division
ERA Technology Ltd.