The Space Shuttle Launch Computer Control System at NASA Kennedy Space Center

Borko Furht
MODCOMP, an AEG company
Fort Lauderdale, Florida 33340

Robert Luken
NASA
Kennedy Space Center, FL 32899

Abstract

This paper illustrates how a complex application, the Data Acquisition and Space Shuttle Launch Control System at NASA Kennedy Space Center, which requires both intensive communications and high processing power, can be mapped into an advanced open system architecture. This is achieved by employing a symmetric, tightly-coupled multiprocessor system which includes a high bandwidth common memory bus, an efficient interrupt distribution scheme, dedicated high-performance I/O processors, and runs a multiprocessor real-time UNIX operating system.

1. Introduction

The Data Acquisition and Space Shuttle Launch Control System at NASA Kennedy Space Center performs launch control functions for the Space Shuttle from T-minus-72-hours until T-minus-31-seconds. At T-minus-31-seconds control is handed over to the space shuttle's on-board computers, but the launch control system continues to monitor all elements of the shuttle.

The current control system consists of 400 MODCOMP 16-bit proprietary computers, however, there is an on-going project with the goal of replacing the system with state-of-the-art real-time open multiprocessor systems. This paper presents a study, which shows how a symmetric, tightly-coupled multiprocessor system can be used to upgrade this data acquisition and launch control system.

The configuration of the core system for data acquisition and launch control is shown in Figure 1. [1].

The core system is composed of five subsystems: Data Acquisition System, Application Processing System, Archival and Data Base System, External Interfaces System, and Display Processing System.

In this study we will concentrate on the most complicated subsystem: the Data Acquisition System. First, the communication and processing requirements for the system will be specified. Then, on the basis of the model of the system, the multiprocessor architecture will be defined to meet these requirements.

2. The Data Acquisition System

The Data Acquisition System (DAS) performs data collection and distribution services. The core of the DAS is the Data Acquisition Processor (DAP), which receives data from the Data Acquisition Modules (DAMs) and passes this data to the Application Processing System (APS). The functional diagram of the DAP is shown in Figure 2.

As the number of DAMs in a particular set increases, the number of data packets to be distributed increases. This increase reaches the limit of a typical machine well before it reaches the bandwidth limit of the network. In order to reduce this traffic the DAP does a packet concentration function. Then, the DAP distributes data to the Application Processing System or just broadcasts all of the data for use by any Application Processor. This analysis assumes the later broadcast mode.

Front-end Data Traffic. The DAP system will pass out only significant changes of the front-end data. About 60,000 significant changes per second are expected, and they are evenly distributed across several data types, as shown in Table 1.

The total front-end traffic is 577,000 bytes/sec, which requires 400 Ethernet packets/sec, with 1500 bytes per packet, as indicated in Figure 2.

Command Traffic. The application processor sends commands to the DAP system in order to change the value of an effector and/or to update front-end tables. An acknowledgement from the DAP system is expected for each command. These packets are small,

0-8186-2212-1/91/0000/0184$01.00 © 1991 IEEE

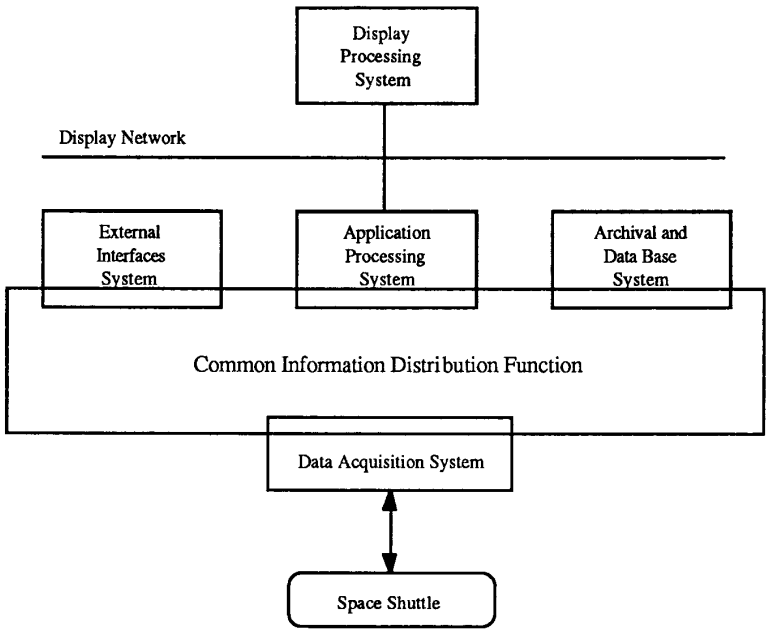

Figure 1 - The configuration of the NASA core system for data acquisition and launch control

Table 1
Significant Changes Distributed Across Data Types

Data type	Change/sec	Kbytes/sec
Time Homogeneous	120	3
Analog Measurement	19980	280
Digital Pattern	8160	65
Discrete Measurement	29760	208
Single Precision FP	1680	17
Multi-Word Double Precision FP	240	4
Total	≈ 60,000	577

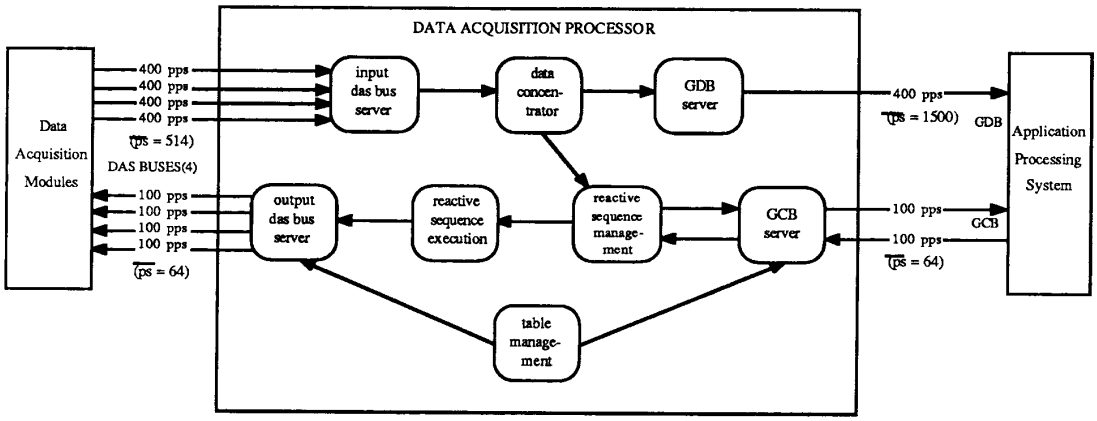

Figure 2 - The functional diagram of the Data Acquisition Processor

10 to 100 bytes (on average 64 bytes). The application processor generates 100 commands/sec and receives 100 acknowledgements/sec, as indicated in Figure 2.

3. The Data Acquisition Processor Model

The Data Acquisition Processor (DAP) model is derived [1] in order to describe the worst case computational requirements for the DAP. The DAP system from Figure 2 is partitioned into eleven processing modules, as shown in Figure 3.

The eleven processing modules have the following functions:

- P_1 - Data Bus Handler
- P_2 - Reactive Control Logic
- P_3 - Command Bus Handler
- P_4 - Data Concentrator
- P_5 - Selector Module
- P_6 - Distributor Module
- P_7 - Security Module
- P_8 - DAS Bus Handler

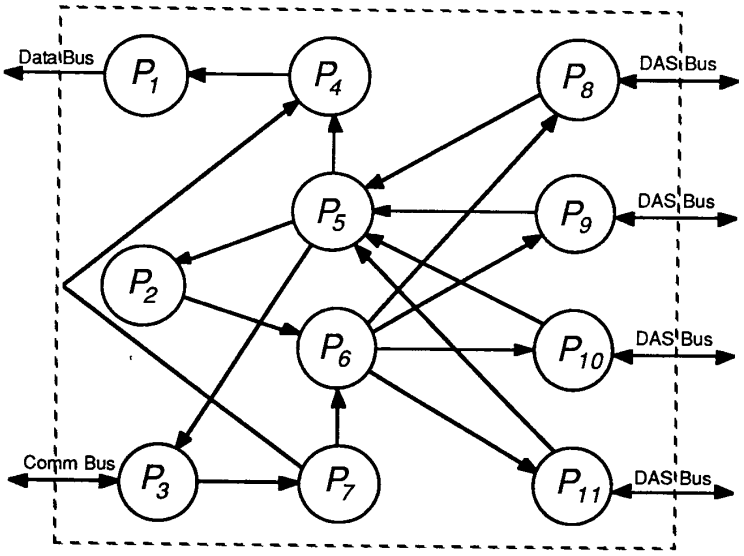

Figure 3 - The Data Acquisition Processor (DAP) Model

P_9 - DAS Bus Handler
P_{10} - DAS Bus Handler
P_{11} - DAS Bus Handler

In order to handle the worst case data rates, the Data Acquisition Processor must support the following:

a. Network Interfaces

- 1800 total packets received per second
- 500 packets received per second per DAS bus
- 100 packets sent per second per DAS bus
- 400 total packets broadcast per second on the Data Bus
- 200 total packets per second sent and received on the Command Bus

b. Processing Power

The required processing power of eleven processing modules can be decomposed into: (1) application instruction rate, (2) system call overhead, and (3) context switch overhead.

The total required instruction rate for processing module *i* is:

$$T_i = P_i + S_i + C_i$$

where:

T_i - total instruction rate for module *i*
P_i - application instruction rate for module *i*
S_i - system call overhead for module *i*
C_i - context switch overhead for module *i*

The application instruction rates (P_i) for the eleven processing elements are computed as follows:

$$P_1 = P_R \times N_I = 400 \times 30 = 12{,}000 \ inst/sec$$

$$P_2 = E_R \times N_I = 20 \times 500 = 10{,}000 \ inst/sec$$

$$P_3 = P_R \times N_I = 200 \times 50 = 10{,}000 \ inst/sec$$

$$P_4 = \frac{D_{PR}}{M_{length}} \times N_I = \frac{141{,}719}{8} \times 300 = 5{,}314{,}500 \ inst/sec$$

$$P_5 = N_B \times N_D \times P_R \times N_I = 4 \times 4 \times 125 \times 50 = 100{,}000 \ inst/sec$$

$$P_6 = N_B \times P_R \times N_I = 4 \times 100 \times 50 = 20{,}000 \ inst/sec$$

$$P_7 = \left(\frac{D_{PR}}{M_{length}}\right) \times N_I = \left(\frac{24{,}663}{8}\right) \times 100 = 308{,}288 \ inst/sec$$

$$P_8 = (N_D + 1) \times P_R \times N_I = (4 + 1) \times 100 \times 25 = 12{,}500 \ inst/sec$$

$$P_9 = (N_D + 1) \times P_R \times N_I = (4 + 1) \times 100 \times 25 = 12{,}500 \ inst/sec$$

$$P_{10} = (N_D + 1) \times P_R \times N_I = (4 + 1) \times 100 \times 25 = 12{,}500 \ inst/sec$$

$$P_{11} = (N_D + 1) \times P_R \times N_I = (4 + 1) \times 100 \times 25 = 12{,}500 \ inst/sec$$

where:

P_R = Packet Rate
D_{PR} = Practical Data Rate
M_{length} = Measurement length
N_I = Number of Instructions
N_D = Number of DAMs per DAS Bus
N_B = Number of DAS Buses
E_R = Reactive sequence exception rate

All the data is obtained experimentally [1].

The total application instruction rate (P) is then given as:

$$P = \sum_{i=1}^{n} P_i = 5{,}824{,}788 \ inst/sec$$

The system call overheads (S_i) for eleven modules can be calculated as:

$$S_1 = P_R \times I_S = 400 \times 1200 = 480{,}000 \ inst/sec$$

$$S_2 = E_R \times I_S = 80 \times 1200 = 96{,}000 \ inst/sec$$

$$S_3 = P_R \times I_S = 500 \times 1200 = 240{,}000 \ inst/sec$$

$$S_4 = P_R \times I_S = 500 \times 1200 = 600{,}000 \ inst/sec$$

$$S_5 = N_B \times N_D \times P_R \times I_S = 4 \times 4 \times 200 \times 1200 = 3{,}840{,}000 \ inst/sec$$

$$S_6 = P_R \times I_S = 100 \times 1200 = 120{,}000 \ inst/sec$$

$$S_7 = P_R \times I_S = 100 \times 1200 = 120{,}000 \ inst/sec$$

$$S_8 = (N_D + 1) \times P_R \times I_S = (4 + 1) \times 100 \times 1200 = 600{,}000 \ inst/sec$$

$$S_9 = (N_D + 1) \times P_R \times I_S = (4 + 1) \times 100 \times 1200 = 600{,}000 \ inst/sec$$

$$S_{10} = (N_D + 1) \times P_R \times I_S = (4 + 1) \times 100 \times 1200 = 600{,}000 \ inst/sec$$

$$S_{11} = (N_D + 1) \times P_R \times I_S = (4 + 1) \times 100 \times 1200$$
$$= 600{,}000 \ inst/sec$$

where:

P_R	=	Packet Rate
D_{PR}	=	Practical Data Rate
M_{length}	=	Measurement length
I_S	=	Number of system instructions
N_D	=	Number of DAMs per DAS Bus
N_B	=	Number of DAS Buses

The total system call overhead (S) is then:

$$S = \sum_{i=1}^{n} S_i = 7{,}896{,}000 \ inst/sec$$

The context switch overheads (C_i) are calculated in the modules that perform communications with the DAMs and Application Processors. For these modules, (P_1, P_3, P_8, P_9, P_{10}, and P_{11} in Figure 3) the context switch overheads are computed as follows:

$$C_1 = P_R \times I_S = 400 \times 4000 = 1{,}600{,}000 \ inst/sec$$
$$C_3 = P_R \times I_S = 200 \times 4000 = 800{,}000 \ inst/sec$$
$$C_8 = P_R \times I_S = 400 \times 4000 = 1{,}600{,}000 \ inst/sec$$
$$C_9 = P_R \times I_S = 400 \times 4000 = 1{,}600{,}000 \ inst/sec$$
$$C_{10} = P_R \times I_S = 400 \times 4000 = 1{,}600{,}000 \ inst/sec$$
$$C_{11} = P_R \times I_S = 400 \times 4000 = 1{,}600{,}000 \ inst/sec$$

The total context switch overhead (C) is then calculated as:

$$C = \sum_{i=1}^{n} C_i = 8{,}800{,}000 \ inst/sec$$

The total processing power required by the DAP is summarized in Table 2.

4. The Tightly Coupled Multiprocessor as the DAP

We will use the DAP model from Figure 3 to map the data acquisition system into a symmetric, tightly coupled architecture.

A tightly coupled multiprocessor architecture with a common bus, shown in Figure 4, is the most popular real-time multiprocessing architecture today. All processors share global memory, and there is a single copy of the operating system and data structures.

Table 2
Processing Power Required by the DAP

Process	Inst/sec
Application	5,824,788
System Calls	7,896,000
Context Switches	8,800,000
Total	22,520,788

Besides improving the speed of computation, the symmetric, tightly coupled multiprocessor provides some other features needed for real-time applications [2,3]:

- allows the application to be broken into independently schedulable tasks,
- provides mechanisms for intertask communication and synchronization,
- provides high reliability,
- provides predictable execution time and low interrupt latency, and
- supplies high performance.

Using the data for P_i, S_i, and C_i obtained from the DAP model, the total instruction rate (T_i) for eleven processing modules becomes:

T_1 = 12,000 + 480,000 + 1,600,000
= 2,092,000 *inst/sec* ≈ 2.1 MIPS

T_2 = 10,000 + 96,000 + 0
= 106,000 *inst/sec* ≈ 0.1 MIPS

T_3 = 10,000 + 240,000 + 800,000
= 1,050,000 *inst/sec* ≈ 1.1 MIPS

T_4 = 5,314,500 + 600,000 + 0
= 5,914,500 *inst/sec* ≈ 6 MIPS

T_5 = 100,000 + 3,840,000 + 0
= 3,940,000 *inst/sec* ≈ 4 MIPS

T_6 = 20,000 + 120,000 + 0
= 140,000 *inst/sec* ≈ 0.2 MIPS

T_7 = 308,000 + 120,000 + 0
= 428,000 *inst/sec* ≈ 0.5 MIPS

T_8 = 12,000 + 600,000 + 1,600,000
= 2,212,000 *inst/sec* ≈ 2.25 MIPS

T_9 = 12,000 + 600,000 + 1,600,000
= 2,212,000 *inst/sec* ≈ 2.25 MIPS

T_{10} = 12,000 + 600,000 + 1,600,000
= 2,212,000 *inst/sec* ≈ 2.25 MIPS

T_{11} = 12,000 + 600,000 + 1,600,000
= 2,212,000 *inst/sec* ≈ 2.25 MIPS

Total ≈ 23 MIPS

Assuming the symmetric, tightly coupled multiprocessor from Figure 4, as the DAP system,

with a 33 MHz MC68030 as the CPUs or I/O processors each with 6.5 to 9.5 native machine inst/sec, the DAP model can be partitioned as shown in Figure 5.

The symmetric, tightly coupled multiprocessor system consists of five processing elements: three I/O processors and two CPUs. These five processing elements require the following instruction rate:

$IOP_1 = T_8 + T_9 = 2.25 + 2.25 = 4.5$ Minst/sec

$IOP_2 = T_{10} + T_{11} = 2.25 + 2.25 = 4.5$ Minst/sec

$IOP_3 = T_1 + T_3 = 2.1 + 1.1 = 3.2$ Minst/sec

$CPU_1 = T_4 = 6$ Minst/sec

$CPU_2 = T_2 + T_5 + T_6 + T_7 = 0.1 + 4 + 0.2 + 0.5 = 4.8$ Minst/sec

The tightly coupled multiprocessor system, shown in Figure 6, consisting of five processing elements, three configured as I/O processors and two as CPU processors, meets these requirements.

4.1 Communication Requirements Analysis

The proposed tightly coupled multiprocessor supports the TCP/IP network protocol. These communications functions are provided by the Ethernet controllers connected via the VMEbus™. The maximum transfer rate is 1.2 Mbytes/sec. Performance of the network depends on the layer at which the communication is performed and the packet size. Table 3 shows experimental results obtained for the VLAN-E Ethernet controller.

This data was obtained from a dedicated, two-node network. Each node was a MODCOMP model-9730 processor with 8 megabytes of main memory. The model-9730 is based on a Motorola model-147 CPU board with a 25 MHz MC68030 processor.

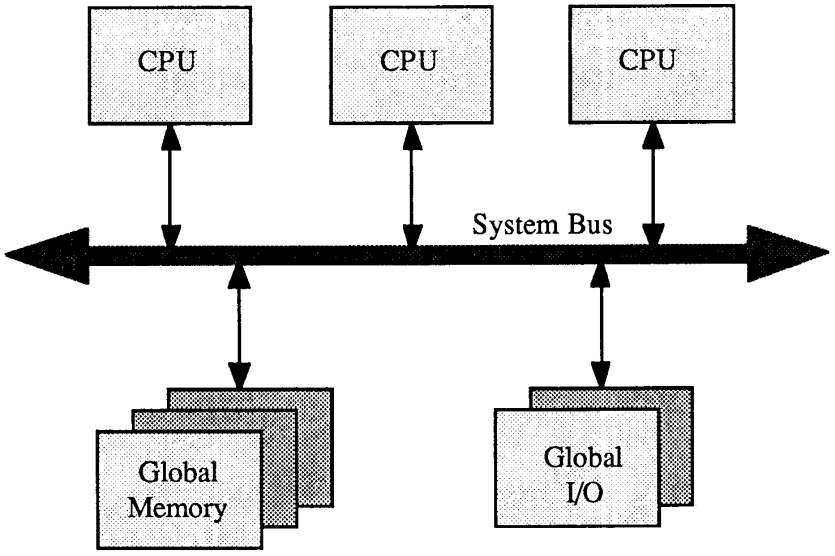

Figure 4 - The tightly-coupled multiprocessor architecture with common memory

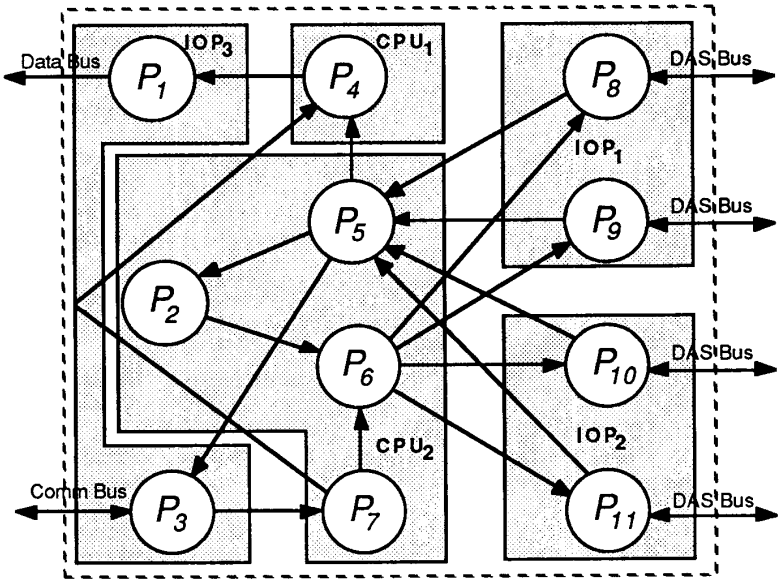

Figure 5 - Partitioning function of the DAP system into five processing elements: three I/O processors and two CPUs

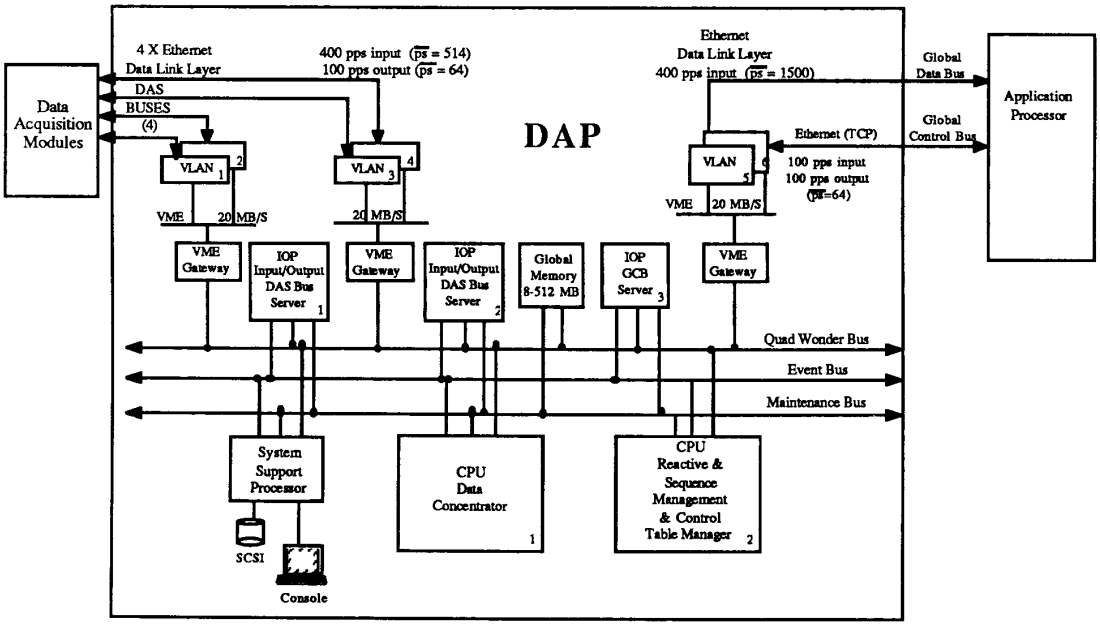

Figure 6 - The tightly coupled multiprocessor system configured as the Data Acquisition Processor

Table 3
Performance Evaluation of the VLAN-E Ethernet Controller

Layer	Packet size [bytes]	Transfer rate	
		Packets/sec	Kbytes/sec
Data Link Layer	64	1937	121
	512	1486	743
	1024	1169	1169
UDP	64	390	24
	512	536	268
	1024	497	497
TCP	64	548	34
	512	361	180
	1024	263	263
	1500	133	195

Front-end Communications. The front-end communications are handled by two I/O processors (#1 and #2) which are used as the I/O DAS bus servers and four related VLAN ethernet controllers (#1 to #4), as shown in Figure 6.

The VLAN controllers receive the data from DAMs via DAS buses, form a packet, and when the packet is ready, they send an interrupt to the corresponding I/O DAS bus data server. In response to the interrupt, the I/O DAS bus data server receives the data via the VME bus and stores it in the global memory. The solution proposed here is based on four independent Ethernet lines where communication is performed at the Data Link layer. According to Table 3, the transfer rate of the VLAN controller is 1486 packets/sec (or 734 Kbytes/sec), when the packet size is 512 bytes. This meets the DAP requirements, of 400 packets/sec with an average packet size of 514 bytes.

The proposed solution to handle the interface to the Application Processor via the General Data Bus consists of another I/O processor (#3), which serves as the GCB Server, and two related VLAN Ethernet controllers (#5 and #6). The VLAN controller (#5) interrupts the GCB Server, whenever it is ready to send the next packet. The GCB Server then passes the packet from the global memory to the VLAN controller, which then sends the packet to the Application Processor. The throughput requirement of 400 pps (ps=1500 bytes), is met at the Data Link layer. As shown in Table 3, the transfer rate at the Data Link layer is 1169 pps, with ps=1024 bytes, which exceeds the requirements.

Command Communications. Commands are handled by the I/O processor (#3), the GCB Server, and the VLAN Ethernet controller (#6). The VLAN controller receives the commands from the Application Processor and when the packet is ready, sends an interrupt to the GCB server. This server receives the packet and stores it in global memory. It also sends the corresponding acknowledgement packet, via the VLAN controller, to the Application Processor. The throughput requirements are 100 pps per input and 100 pps per output (average packet size is 64 bytes), which gives a total of 200 pps. The proposed solution is the communication at the TCP level, at which the transfer rate is 548 pps (ps=64 bytes), as indicated in Table 3.

5. Summary and Conclusions

This study has illustrated how a complex application, the Data Acquisition and Space Shuttle Launch Control System at NASA Kennedy Space Center, which requires both intensive communications and high processing power, can be mapped into an open system architecture. This is achieved by employing a symmetric, tightly-coupled multiprocessor system which includes a high bandwidth common memory bus, an efficient interrupt distribution scheme, and dedicated high-performance I/O processors.

The prototype of this symmetric, tightly coupled multiprocessor system using 2 to 10 MC68030 processors, has been developed at MODCOMP. The final system, referred to as the REAL/STAR System, will consists of 2 to 20 MC88100 RISC processors and will run the Multiprocessor REAL/IX Operating System, MODCOMP's real-time implementation of the AT&T System V [4].

References

[1] D. Luken, "GCS Performance Study", Report, NASA Kennedy Space Center, August 1989.

[2] B. Furht, G. Rabbat, R. Kibler, J. Parker and D. Gluch, "The Design of Tri-D Real-Time Computer Systems", *Proc. of the Euromicro Workshop on Real-Time*, Como, Italy, June 1989, pp. 84-92.

[3] B. Furht, G. Rabbat, J. Parker, R. Kibler, W. Earnshaw, H. Ohel, and P. Ripy, "Tightly-Coupled Multiprocessor Structure for Real-Time Applications", European Patent No. 89112824.1, 1989.

[4] B. Furht, D. Grostick, D. Gluch, G. Rabbat, J. Parker, and M. McRoberts, "Real-Time UNIX Systems: Design and Application Guide", Kluwer Academic Publishers, Norwell, Massachusetts, 1991.