RSX-11M - PDP-11/70

MULTIPROCESSING

SYSTEM FUNCTIONAL SPECIFICATION

COMPANY CONFIDENTIAL

COPY NO. E212

DIGITAL EQUIPMENT CORPORATION . MAYNARD, MASSACHUSETTS

ABSTRACT

This document describes a multiprocessor system product based on the PDP-11/70 processor and RSX-11M operating system software. The perspective is that of describing an integrated system with emphasis on availability and performance issues. Hardware and software components are discussed with regard to the functions they provide in the multiprocessor system. The purpose of this specification is to define the interactive requirements of the hardware and software components for the implementers and to describe the system product to those interested.

RUSSELL L. MOORE

SYSTEM PROJECT ENGINEER

JULY 22, 1977

1.0 Project Goals

Near Term (to end FY'78)

Introduce tightly coupled (shared memory) multiprocessor system technology to our standard corporate product family. Enhanced availability and performance extensibility is provided thru replicated CPU's, data paths, and peripherals and the ability to reconfigure around failed system elements. (Q4 FY'78 FCS for initial 11/70mP.) The system also features reconfigurability to a loosely coupled (non-shared memory) mode. Initial RSX-11M support will be complimented by TPS support within 2 qtrs.

Initial 11-based implementations will be used for technical and marketing experience.

This is an advanced system tool, utilizing off-the-shelf elements to the greatest extent practical, and requiring sophisticated customers to build on this initial foundation. The complex recovery techniques are postponed to later versions. However, as a standard Corporate product with a software base, even this first modest step is significant.

Longer Term (FY'79 +)

Develop improved degraded mode capabilities and end-user functionality, e.g., improved fault detection and identification, greater data integrity, faster response time, automatic reconfiguration; employ multiprocessing to span greater range of performance with fewer CPU designs.

2.0 Market Description

Market can be divided into availability and performance oriented segments. The current emphasis appears to be availability with performance enhancement through load sharing as a secondary goal. Powered-up repairability is mandatory.

The system will be marketed as a standard DEC product. Primary use will be in message switching, transaction processing, industrial control, telephone systems, banking, business, hospitals, etc.

Currently DEC supplies only certain multiprocessor options via CSS, and offers a system integration service to help customers install their systems. All programming is done by the customers, including development of multiprocessor device handlers.

To date DEC has supplied options to customers for approximately 300 multiprocessor systems, at a current rate of about 100 per year. High availability systems constitute some 75% of the total, high throughput 20% and others (e.g., resource sharing) 5%.

At least initially, the 11/70mP systems should sell to a relatively cost insensitive market. A dual system will sell in the range of $250K-$400K.

3.0 Product Description

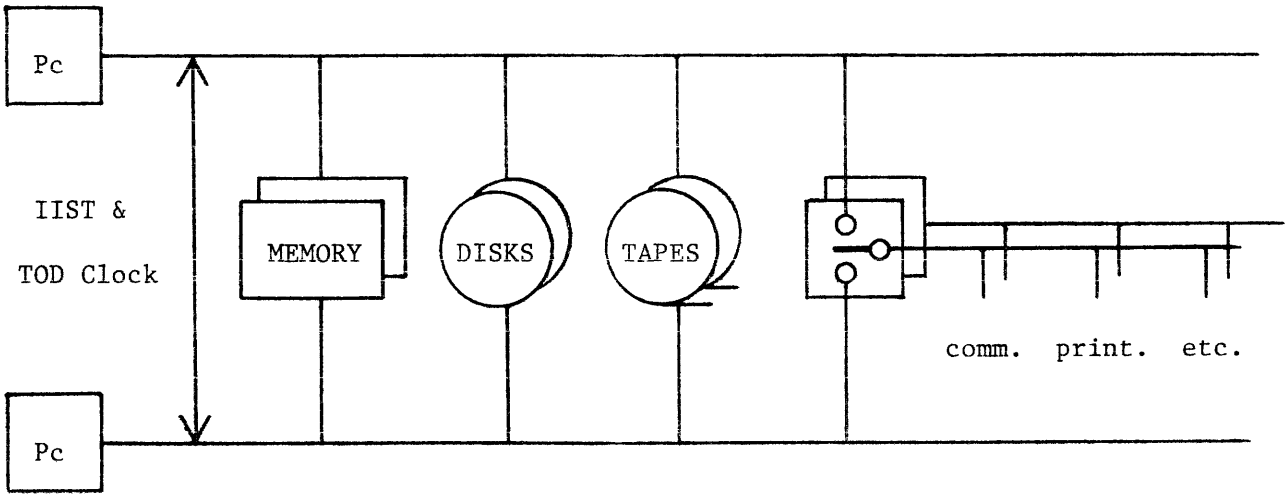
3.1 System Organization

This section describes the organization of the hardware in the multiprocessor systems with a brief description of the hardware functions that are important for multiprocessor operation. The description of how this hardware along with the RSX-11mP software function as a system is described in section 3.2, "System Functional Characterization".

A Multiprocessor

These multiprocessor systems have two to four processor units. All processors have access to several common central memory units (i.e. they are tightly coupled) and at least a portion of the I/O devices. All processors are being controlled by a single operating system that provides interaction between the processors, the programs they are executing, and the I/O device interrupts. The topology is symmetrical (i.e. any processor can perform any task as opposed to a master/slave configuration).

Figure 3.1 shows a simplified typical two processor configuration. The figures of section 3.3.1, "Typical Two & Four Processor Configurations" show more detailed system configurations. Examination of these figures should help understanding of the following sections.



Note: This drawing has been simlified for illustration. See figures 3.2 and 3.3 in Section 3.3, "Hardware Configurations", for more detailed illustrations.

Figure 3.1

Two Processor Configuration

(simplified drawing)

Distributed I/O Devices

The RSX-11mP Executive has been designed to permit a program, running on any processor to use any I/O device in the system, even if the I/O device is not physically connected to the processor on which the program is running. Special programming is not required to make use of this distributed I/O feature.

These systems may be configured with I/O devices that do not have port connections to all processors (i.e. dual port devices in a four CPU system and single port devices). Unlike main memory, external page (I/O) addresses are local to each processor. Thus, I/O initiation and interrupt service can only occur on a processor having access to the device.

Application programs will not be required to know how the I/O devices are distributed in the system. The "Interprocessor Interrupt (IIST)" is used by the RSX-11mP software to pre-empt task execution on a CPU whenever access to a device via that processor is required by the Executive. Thus a task can request I/O service on any device regardless of which CPU is executing the task. The ability to distribute the I/O while retaining symmetry at the task level offers the following advantages:

- Redundant data paths to dual ported devices
- Interrupt processing is distributed among all CPU's
- System bandwidth is better utilized because fewer devices are contending for each bus.
- I/O load levelling across ports of multiport devices (the effectiveness of these algorithms are being reviewed).

The Interprocessor Interrupt

The primary function of the Interprocessor Interrupt (IIST) is to provide a mechanism for the RSX-11mP Executive to interrupt processors for the purpose of rescheduling system activity. Pre-empting task execution for I/O service (as described in the previous section) is a case of rescheduling system activity.

The IIST is also capable of causing an interprocessor boot. This capability is used by the M9301 bootstrap module to bring up the multiprocessor system.

The IIST contains a Sanity Timer for each processor. This timer is periodically refreshed by the Executive software. If the refresh commands do not occur within a programmed period, the IIST will issue an interprocess interrupt to inform the other processors in the on-line system of the faulty operation.

The interconnecting IIST bus is "T" connected to permit disconnection of an IIST interface without disturbing the remaining IIST system. This feature along with programmable and manual port isolation capabilities permit the interconnected IIST interfaces to be reconfigured for establishing off-line maintenance systems or multiple, independent on-line systems. (Many configurations are possible e.g., a four CPU arrangement consisting of two independent systems each of which is a tightly coupled dual processor. On-line reconfiguration is discussed further in section 3.2.2, "Availability Functionality".) Individual IIST interfaces may be powered down and removed for service without affecting the remaining IIST system.

Switchable UNIBUSES

Non-multiport UNIBUS devices can be placed on the switchable bus(es). In the event of a failure in a CPU or on the UNIBUS of a CPU, the devices on the switchable UNIBUS(es) can be transferred to another processor for I/O service. Devices critical to the system operation can be replicated on multiple switched buses using additional UNIBUS switches.

Dual Port/Dual Access Mass Storage Devices

The MASSBUS storage devices in the system are both dual ported and dual accessed. There are two paths over which data can be transferred between the mass storage device and main memory (dual porting). Each mass storage device can be controlled by either of two RH7Ø controllers (dual access). Failures along one of the paths (CPU, MASSBUS, RH7Ø controller, or one port of the main memory system) may be tolerated without loss of the mass storage device as a resource to the system.

Single port/access MASSBUS devices (TU16/TE16) are given dual port/access functionality via the RHØ1 Dual Port MASSBUS Adapter.

Multiport Main Memory System

The memory system consists of several boxes of multiported main memory and a number of other functional elements within each CPU (Cache, UNIBUS Map, Memory Management Unit).

The main memory is contained in at least two boxes of multiport MOS memory. The actual number of boxes of memory (and all other devices critical to system operation) will depend on the availability required. Each box contains:

- Up to 512K bytes of MOS memory with ECC. (The maximum box capacity will be 2M bytes when 16K memory chips become available.)
- A multiplexor that automatically arbitrates and grants use of the memory to the various ports without programming.
- Port control logic at each port that permit manual or programmable reconfiguration of the memory.
- Power supplies for the box.

Each box of memory is connected to independent A.C. power (power that is not switched off with any other device). The memory may be two-way interleaved within each box as well as two-way or four-way interleaved across memory boxes.

The Cache, Memory Management Unit, and UNIBUS Map elements of the memory system perform the same basic functions in the multiprocessor system as they do in the PDP-11/70. The functionality of these elements that has been changed or added is for the purpose of:

- Providing the hardware functionality necessary for the support of software locks (i.e. synchronize utilization of the Executive & the common data base).
- Provide for efficient, transparent operation of the local cache memories in the multiprocessor system (i.e. avoid stale data).

The details of this new functionality are described later in section 4.1, "Device Functional Descriptions".

Replicated System Elements

All system elements that are critical to system operation (processors, memory boxes, I/O devices, controllers, and access paths) are replicated. The on-line reconfiguration feature of the RSX-11mP software and special port/bus isolation features of the hardware permit reconfiguration of the on-line system around failed elements. This reconfiguration is via operator control and does not necessitate physical disconnection. One processor, some memory, and the failed device (which may be the processor or memory) may be taken off-line to perform maintenance while the remaining healthy hardware performs the critical on-line functions. (This is discussed in greater detail in section 3.2.2, "Availability Functionality" and section 3.2.3, "Maintenance Functionality".

Time of Day Clock

The multiprocessor configurations include a multiported Time of Day (TOD) clock. The TOD consists of a master time keeping unit, slave interface units, and an interconnecting cable system. The internally redundant TOD master has an on-board battery backup power supply which permits the time keeping function to be maintained through power outages and most single failures. The slave interfaces permit the processors to read the time value that is broadcasted from the TOD master.

Many features have been incorporated in the TOD design to make the TOD subsystem reliable including automatic error detection and correction in the time keeping and broadcast functions. For applications demanding even greater reliability of the TOD functions, the multiprocessor system may be configured with replicated TOD subsystems.

The primary function of the TOD subsystem is to provide application level software with a time value that is independent of discontinuities in operating system service (i.e. power failures, hardware failures, system faults, etc.). This information could be used by application level recovery software to implement time dependent error recovery techniques.

Another function of the TOD subsystem is to provide a synchronized, high resolution time source that does not require high interrupt overhead. The time value read through the slave interface has a resolution of 100 microseconds. This time value is maintained in the TOD hardware and does not require interrupt service to update the time.

The RSX-11mP Executive does not require a TOD clock to function but will use the TOD clock if it is present for setting the time after a re-boot and for validating the time value that is kept internal to the Executive.

3.2 System Functional Characterization

This section describes how the hardware along with the RSX-11mP Software function as a system with regard to programming, availability, maintenance, and performance functionality.

3.2.1 Operating and Programming Functionality

mP System Transparent To Tasks

The Multiprocessor system will be transparent to all non-privileged and most privileged tasks running in the system. The RSX-11mP Executive will take care of all multiprocessor system interactions. The Executive has been designed to efficiently exploit the multiprocessor hardware configuration to provide availability and performance enhancements without requiring special programming at the application level.

Non-privileged tasks can be transported between single processor (RSX-11M V4) and multiprocessor (RSX-11mP) systems without source code modification or relinking. Privileged tasks may be similarly transported between systems without modification or relinking provided that the task adheres to the established protocols for referencing the Executive data base and does not reference any data structures specific to multiprocessing.

The capabilities of the RSX-11mP Executive make it possible to write application level software that is aware of the multiprocessor configuration allowing further exploitation of the hardware configuration. For example, tasks can be written that run concurrently in different processors and communicate through common data areas and through the intertask communication & control facilities in RSX-11mP. Such tasks can realize increased processing speed in performing the desired application through concurrent execution and by exploiting multiple data paths for I/O activity.

Multiprocessor Bootstrap

A multiprocessor bootstrap module M93Ø1-mP is provided for initial system booting and rebooting at power-ups. The bootstrap allows hand system booting from any mixed massbus devices and from any RH7Ø controller in the system through CSR specification.

System load media supported by the bootstrap are all MASSBUS devices (i.e., RPØ4, RPØ5, RPØ6, TE16, RMØ3). Bootstrap routine code for the following non-supported devices are provided in the documentation: TU10, TC-11, RPØ3, TA11, RX11, PC11, RKØ5, RKØ6.

The multiprocessor bootstrap is operationally flexible in that it also does properly operate in single processor systems and in multiprocessor systems that have been decomposed into two or more independent systems (e.g. a two CPU system that has been decomposed into two single processor systems or even a four CPU system that has been decomposed into two dual processor systems).

3.2.2 Availability Functionality

Design Approach To Enhanced Availability

The availability of a system may, in general, be improved by either making the components of the system more reliable or by reducing the impact of failures on the system. These multiprocessor systems have been designed to reduce the impact of failures on system availability by providing multiple system elements and the capability of bypassing failed elements while they are being repaired.

In conventional, single processor systems, a failure in the system potentially impacts the operations of that system in three ways:

1. One or more of the devices are no longer functional.

2. The entire hardware/software system is often unavailable during isolation and repair of the failed unit.

3. Applications level data bases may have been corrupted as a result of the failure.

The new features that have been incorporated in these multiprocessor systems attack the first two of these effects. The third effect is being addressed in current DEC application product designs which will run in the multiprocessor configurations. These include journaling and backup capabilities in TPS-11 and RMS-11. The current multiprocess software is being designed with the goal of easy transition of these products from the single processor to the multiprocessor versions of the RSX-11M software.

The design of the multiprocessor systems permit additional devices to be included in the configuration for the purpose of increasing the availability of the system to perform a desired function (the "critical function"). Capabilities in the hardware and RSX11-mP software permit failing devices to be isolated from the system. The loss of devices will not prevent the system from performing the critical function as long as sufficient functional hardware remains.

The replicated hardware and the ability to reconfigure on-line permits the system to service the on-line application while a portion of the hardware is being used to repair failing units. Thus, the repair time of most failures is not part of the system down time.

The multiprocessor system may be unavailable (less than 10-15 minutes for most failures) during the isolation of a failure. The functionality of on-line diagnostics, improved by error log, and on-line reconfiguration have been incorporated in the multiprocessor system to reduce the time to isolate and reconfigure around failures.

Multiport/Multibus Structures

Loss of a device in the multiprocessor system may mean the loss of one or more of the interconnecting buses in the system. For example, if one of the processors is reconfigured out of the multiprocessor system, the system also loses the access to I/O devices over the UNIBUS and MASSBUSes that are connected to that processor. Loss of an interconnecting bus does not, however, mean the loss of all devices connected to that bus. This is because all devices are either dual ported (dual accessed) or reside on switchable UNIBUSes.

Critical devices that reside on switched UNIBUSes are replicated on other switched UNIBUSes elsewhere in the system. Similarly, dual access MASSBUS devices (RpØ6, RHØ1/TU16, etc.) are replicated using a second pair of MASSBUSes (and RH7Ø controllers). This arrangement of the hardware, along with the symmetry effected through the RSX-11mP Executive, greatly reduces the impact that hardware failures have on the system availability. If there is at least one redundant unit for each critical device type in this hardware configuration, all known single failures, with the exception of catastrophic failures such as fire, can be reconfigured out of the system to permit the remaining functional hardware to service the critical application. (It is assumed that uninterruptable AC power sources will be installed in systems that cannot tolerate power source failures.)

Mixed MASSBUS Configurations

The single processor and multiprocessor versions of the RSX-11M software support configurations with different types of devices connected to the same MASSBUS. This feature is important to the multiprocessor systems because it reduces the required number of MASSBUSes (and RH7Ø controllers).

In the multiprocessor hardware configurations, the critical MASSBUS devices are connected to each processor via two MASSBUSes. That is, normally half of all disks making connection to each processor do so through one RH7Ø and the other half through another RH7Ø. This arrangement reduces the impact that a failure of an RH7Ø or a MASSBUS would have on the system availability. (Distributing the mass storage devices across many RH7Ø controllers also allows better utility of system I/O bandwidth.)

Normally, two RH7Ø controllers are required per processor. Having the ability to put different device types on the same MASSBUS eliminates the necessity to repeat this scheme for each type of MASSBUS device.

On-Line Diagnostics for Mass Storage Devices

On-line diagnostics will be available for those mass storage devices supported by this system. These tools can be run compatibly with the application software. They permit the operation of a suspected faulty device to be verified, on-line, without reconfiguring the system to run stand-alone diagnostics on the device. The on-line diagnostics will primarily be used by the system operator.

Stand alone diagnostics are also included in the diagnostic tools for the multiprocessor system. The use of stand alone diagnostics is discussed in section 3.2.3, "Maintenance Functionality".

Improved Error Logging

Another tool for locating faulty devices is the error log. The error logging facility of the multiprocessor software has been upgraded to include information on:

- System configuration at time of error
- Processor I.D. through which the error occurred
- A snapshot of system activity at the time of failure
- Volume label reporting for all errors from mounted media
- Power fail/recovery reports
- System Service Messages
- DMA communication Device (DMC11) Support
- IIST and TOD support

The error log reports give an indication of a failing device or the subset of hardware that probably contains a failing device.

The error log facility will be used by both the system operator and Field Service (see 3.2.4, "Maintenance Functionality").

On-Line Reconfiguration

It will be desireable to reconfigure the system for the following cases:

- To isolate the system that is performing the critical function (the on-line system) from one or more failing devices or from a subset of hardware containing a failing element.
- To remove enough hardware from the on-line system so that an off-line system is established for the purpose of performing maintenance while the on-line system continues to function.
- To divide the hardware into multiple, independent on-line systems. (I.e. loosely coupled configurations with separate operating systems).

In all of these cases it is necessary to remove hardware resources from the on-line system (primarily a software function) and to isolate the on-line system from the effects of these hardware elements (primarily a hardware function).

The ability to reconfigure the multiprocessor system around hardware elements and then isolate the system from the effects of these elements is provided through the on-line reconfiguration function's in the RSX-11mP software and the port isolation capabilities of the hardware. These combined capabilities permit the system to be reconfigured without requiring physical disconnection of devices.

A set of Monitor Console Routine (MCR) commands provide the operator with a means of logically removing or adding devices from the Executive's pool of available system resources. Removing a device may mean the loss of other devices or access paths. The RSX-11mP Executive is knowledgeable of the hardware arrangement and will make the appropriate adjustments of hardware resources whenever a device is removed or added. In general, devices may be routinely removed or added without preventing the system from being able to perform the critical function, however, removing a memory box will require rebooting the system.

The reconfiguration MCR commands make use of the programmable port control capability of the multiport memory boxes. When the operator adds one or more boxes of memory, a reconfiguration task (part of the supplied system software) will program the address specification on all ports of each memory box that will be used by the system. The added memory will be assigned physical addresses just above the previous upper limit of memory. The reconfiguration commands permit the operator to specify that memory boxes be non-interleaved as well as two-way or four-way interleaved (across two or four boxes, respectively) as they are added.

The reconfiguration MCR commands also permit the operator to control the programmable UNIBUS switches. The operator may instruct a switched UNIBUS segment to switch to another processor or disconnect. He may also instruct the Executive that the UNIBUS switch will be manually operated.

The port isolation capabilities of the multiport devices provide considerable isolation via program control. For devices on a switched UNIBUS, much isolation is provided by switching the UNIBUS switch. When the Executive is instructed to remove a processor, the IIST interfaces on the remaining processors are programmed to mask any interrupts or boot signals from the IIST on the processor that was removed.

A second level of isolation is provided via manually operated port control switches. The IIST interfaces have interface disable switches that, when thrown, prevent that interface from asserting signals on the interconnecting IIST bus. The multiport memory boxes have port disable switches that prevent the port from asserting signals on the memory bus. UNIBUS switches may be placed in manual control, forcing the switched bus to one port or to disconnect. The multiport mass storage devices have manual controls that inhibit port switching.

Isolation is also facilitated by special power fail characteristics of the multiport devices. The multiport memories, UNIBUS switches, and multiport MASSBUS devices, permit the buses that are connected to their ports to be operational while these devices are without power. The IIST and TOD Slave interfaces permit the interconnecting buses to which they are respectively connected to be operational when they are without power.

Availability Metrics

Multiprocessor systems used in enhanced-availability applications are characterized by the presence of redundant components. A subset of the multiprocessor system made up only of components required by the application is referred to as the minimum system. As such, the minimum system could also be a multiprocessor system, e.g. the critical function requires greater than one CPU's throughput.

The following three metrics provide measures describing the availability enhancement in multiprocessor systems:

- Unavailability Reduction Factor (URF):

  Is a measure of the relative decrease in system total down time of multiprocessor systems over the corresponding minimum systems.

  In most applications the design goal will be a factor greater than 3-5 (although much greater factors are theoretically possible).

- MTBF improvement factor:

  Is a measure of the relative improvement in average system MTBF of multiprocessor systems over the corresponding minimum system.

  In most applications the design goal will be a factor greater than 3-5.

- Average system reconfiguration time:

  Is a measure of the amount of time required to reconfigure the system around a faulty component.

  The design goal is that reconfiguration time will be less than 15 minutes for most failures.

Appendix A, "Multiprocessor System Availability Analysis", includes more details on the availability analysis used to date.

3.2.3 Maintenance Functionality

Powered-Up Repairability

These multiprocessor systems have the capability of removing hardware from the on-line system for the purpose of establishing an off-line, maintenance system. The off-line system is used to service hardware elements (or de-bugg software) while the on-line system continues to service the on-line application. Possible erroneous interaction between the two systems is reduced due to the isolation characteristics built into the hardware and software elements. Elements of the off-line system may be powered down or uncabled without adversely affecting the on-line system.

The minimum hardware required for the off-line maintenance system is one processor, one box of memory, and the failing device(s) (which may be the processor or memory box). This type of off-line system configuration permits the use of the same stand alone diagnostics that are used in conventional single processor systems. This also permits Field Service personnel to utilize their experience with single processor systems.

Improved Error Logging

In most instances, when the Field Service representative arrives on site, the system operator (or a privileged user) will have already reconfigured the on-line system around the failure. (Note: A second reconfiguration may also be necessary to establish an off-line system.) The Field Service representative will not, in general, be able to observe system operation in the exact configuration prior to reconfiguration. He will, however, have a detailed record of system activity at the time of errors before and after reconfiguration. This information is contained in the error log.

The information in the error log includes detailed information on how devices were using the system (starting address, number of words transferred, transfer type, over which bus, etc.). When an error occurs, this information is recorded for the device reporting the error as well as all other error log supported devices that were active at the time of the error. This information will be particularly useful in analyzing "bus problems" or errors that are related to device interaction.

System Loadable Diagnostics

The RSX-11mP software includes a diagnostic loader, the function of which is to load stand-alone diagnostics into a memory box that will be used as part of an off-line system for stand-alone maintenance. A memory box that is used for stand-alone maintenance is not used by the on-line system (except to load the diagnostic). The on-line system may then be protected from undesireable interaction with the maintenance system via the port/bus isolation features of the memory box. The ability to load diagnostics in this fashion eliminates the requirement for dedicating an operable mass storage device for stand-alone diagnostic operation.

Off-Line System Exerciser

While reconfiguring the on-line system around a failure, it may not have been expedient to locate the failure down to the device level. Instead, a collection of hardware containing the failure may have been removed from the on-line system. (This would probably be the case when the fault is related to the operation of one of the buses.) In this situation, locating the failure within the off-line system is facilitated by a DEC-x11 type exerciser included in the system diagnostic package.

This exerciser provides a means of loading the off-line system with activity in a controlled manner. This exerciser will typically be used for:

- Investigating interactive problems so that a device diagnostic can be suggested.
- Quickly bringing out problems that would normally occur infrequently at normal system loading.
- Hardware validation in an interactive environment.

This exerciser is also capable of exercising multiprocessor off-line configurations. This is of particular importance when an interactive problem involves more than one processor.

Maintenance Metrics

Multiprocessor systems used in enhanced availability applications will exhibit a higher rate of component failure than the associated minimum system because of the additional components introduced.

The following metric provides a measure of this change:

- Components' failure rate factor:

  This is a measure of the relative increase in expected Field Service call rate of the multiprocessor system over the minimum system.

3.2.4 Performance Functionality

Concurrent Task Execution

Both the single processor and multiprocessor versions of RSX-11M use multiprogramming to better utilize system resources. In both operating system versions, system activity is, in general, being stimulated by more than one task at a time. For example, one task may be in execution on the processor of a single processor system while I/O transfers are concurrently taking place to service other tasks that are waiting for the I/O to complete.

The operation of the multiprocessor system (RSX-11mP) differs from the single processor version in that there may be more than one task concurrently in execution. In this system, processors, like other devices in the system, are resources for use by the RSX-11mP Executive. In general, there will be as many tasks concurrently in execution as there are processors (up to four processors).

Automatic Utilization of Redundant Resources

These multiprocessor systems have been designed to have redundant devices (including processors) for the purpose of increasing the availability of the system to perform a desired function (the "critical function"). For example, if the critical application requires one processor and two disks, a multiprocessor with at least two processors and three disks is used. In this manner, redundant hardware (processors, memory, disks, I/O devices, etc.) is configured to maintain the critical function in the event of a failure. The multiprocessor system does, however, make use of any redundant hardware that is available with a resulting increase in performance.

CPU And I/O Load Leveling

A task, in general, may execute on any of the processors in this symmetrical multiprocessor system. The processors appear as anonymous resources for the RSX-11mP Executive to allocate to these tasks. As processors become available they will be used to execute new tasks. The Executive will attempt to keep all available processors busy. CPU load leveling is implicit in this scheme.

An explicit mechanism within the RSX-11mP Executive provides load leveling across the ports of dual ported devices. If, while trying to service an I/O request, the Executive finds the path to one port of a dual port device busy and the path to the other port is available, the I/O request will be serviced over the available port. This has the effect of equalizing the data transfer load at each port and reducing bus contention.

Improved Disk Seek Operation

In many systems using moveable head disks, the useable rate at which information may be stored or retrieved is limited by the rate at which the heads can be positioned. Two features have been incorporated in both the single processor (RSX-11M V4) and multiprocessor (RSX-11mP) versions of the Executive to reduce the effective disk seek time:

(1) Disk seek operations via the same controller may now be overlapped.

(2) A disk seek optimization algorithm has been implemented in the Executive.

Performance Metrics

The following metrics will be of importance to system designers using the multiprocessor systems:

Lock Wait Overhead: This is a measure of the overhead due to contention for the use of the Executive in the multiprocessor system.

Interrupt Service Overhead: This is a measure of the time added to the service of I/O interrupts in the multiprocessor Executive.

I/O Device Bandwidth: These metrics indicate the load that each of the I/O devices present when connected to the multiprocessor system. Accompanying the device metrics are system metrics that indicate the maximum I/O load that can be tolerated by the system.

Performance per CPU: These metrics indicate a relative measure of effective processing power as the number of CPU's is increased from one to four. The base figure of "1" is representative of a single processor system running the single processor version of RSX-11M (V4).

The performance of a two processor system is expected to be 140% to 170% of a single processor system.

3.3 Hardware Configurations

3.3.1 Typical Two & Four Processor Systems

Figures 3.3 and 3.4 show typical two and four processor systems. Additional disk and magnetic tape drives may be added at the same points for those device types. Each MASSBUS can support up to eight device connections. Each processor can have up to four MASSBUSes. As devices are added to the system configuration, they are distributed across processors and buses as evenly as possible.

3.3.2 Supported Devices

The following devices will be supported in multiprocessor systems for enhanced availability applications:

1. Processors:

   11/70

2. Memories:

   MKA11

3. Massbus devices:

   RPØ4, RPØ5, RPØ6, RSØ3, RSØ4, RMØ1, RMØ3, TE16, TU45

4. Communication devices:

   DZ-11, DH-11, DMC-11, KMC-11

5. Switches:

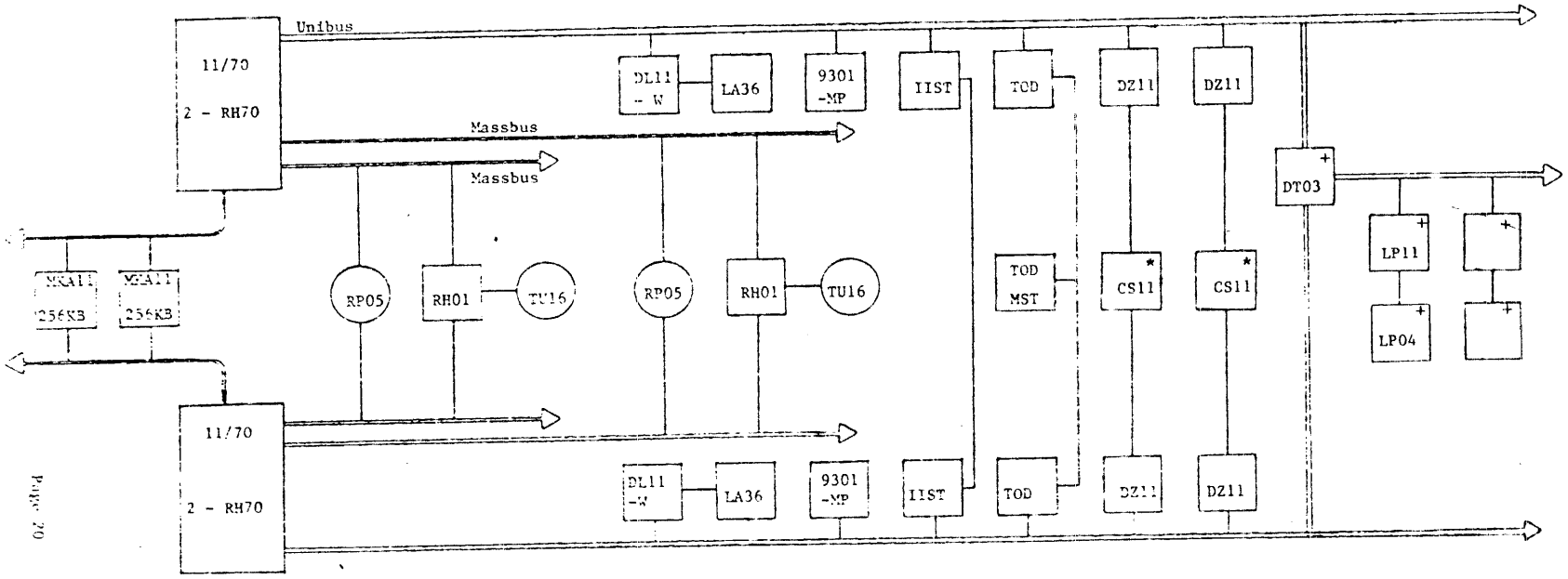
   DT03-FP, CS11-MC

Any other RSX-11M(V4)-supported device can be configured in multiprocessor systems but will not be supported for availability enhancement unless it is connected through a supported switch.

MULTIPROCESSOR SYSTEM

(2-Processor Configuration)

Figure 3.2



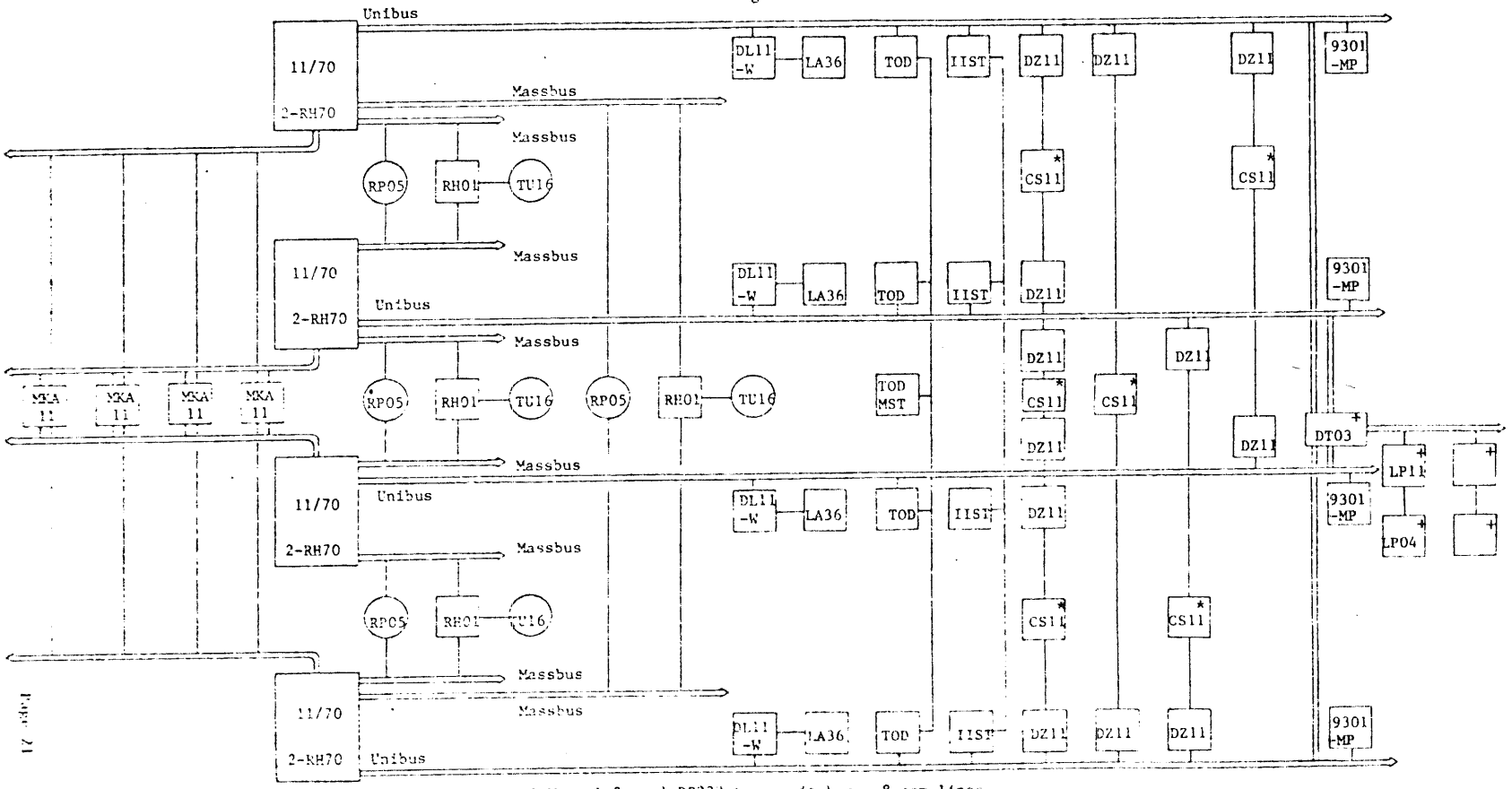
* Manual 2 to 1 RS232 type switches - 3 com. lines.
\+ Non-critical devices on a single DT03.

4.14.77 K.P.

MULTIPROCESSOR SYSTEM

(4-Processor Configuration)

Figure 3.3



* Manual 2 to 1 RS232 type switches - 8 com.lines.

\+ Non-critical devices on a single DT03.

4.14.77 K.P.

4. Component Functional Requirements

4.1 Hardware Component Requirements

This section describes the basic functional extensions that are required in the hardware for implementation of these multiprocessor systems. It is assumed that additional extensions (over what is described here) will be necessary in order to meet the requirements of diagnostic programming, serviceability, and reasonable device design.

More detailed information can be found in the following hardware specifications:

- 11/70-mP Processor Specification
- MKA11 MOS Multiport Memory Specification
- Interprocessor Interrupt and Sanity Timer (IIST) Specification
- Time-Of-Day (TOD) Clock Specification

4.1.1 Processor Modifications

The implementation of these multiprocessor systems will require that some new capabilities be added to the existing PDP-11/70 design. These modifications are necessary to:

- maintain cached data that is consistent with data elsewhere in the system,
- provide the throughput advantages of cache memories local to each Pc while operating in the multiprocessor environment,
- implement semaphores (the ASRB used as a "lock" instruction),
- maintain single processor performance level

The functions added to the existing PDP-11/70 design are:

- Cache Flush
- Cache Bypass on Virtual Page
- Force Bypass
- Bypass and DATIP-DATO Operation for ASRB
- Cache Bypass on Map Page

These functions provide mechanisms for dealing with multiple cache memories in the multiprocessor system. A discussion of how these functions are used to prevent inconsistent cache data from being a problem can be found in section 4.2.1, "Executive Extensions" (see "Local Cache Memory" discussion).

The multiprocessor 11/70 is fully compatible with the single processor 11/70. A multiprocessor 11/70 configured as a single processor system will run existing software.* The performance of the processor is not degraded with the exception of the ASRB instruction (which will not use the cache for operand references ....see below).

Cache Flush

This function provides the ability to clear the local cache under software control. The effect of the flush is to mark all of the words that may have been in the cache as invalid.

The cache flush is used by the RSX-11M software to rid the cache of any incorrect or inappropriate data each time a process is started or resumed.

* New control functions have been added to the control registers of the processor. Software expecting these bits to be non-functional may not run properly.

This "stale data" might have been caused by:

- Process migration *
- Indirect I/O **

It is important that the flush function take as little time as possible so as to minimize its impact on performance. If the flush requires more than a few processor cycles, the processor (and peripherals) must be permitted to access the main memory (i.e., effectively run without a cache) for the duration of the flush.

Considering the proposed design of the RSX-11mP Executive, the time between flush operations will infrequently be less than 150 microseconds, but the hardware must be capable of handling back to back flush commands without preventing access to main memory.

The cache flush should be implemented in a manner consistent with the degraded mode features of the existing 11/70 cache design. That is, the flush function must work properly if either one of the two groups within the cache are disabled. Error indications associated with the valid stores must indicate which group was in use at the time of error.

* Process execution may shift from one processor to another as process/system activity dictates.

** The execution of a process that has requested an I/O transfer and the service of the I/O request may occur on different processors. This is referred to as indirect I/O.

Cache Bypass on Virtual Page

This function adds an attribute to each virtual page of memory. A bit is added to each Page Description Register (PDR) in the Memory Management Unit so that when a PDR is programmed to map a virtual page into a physical page, it will be possible to specify that all memory references to the page will bypass the cache.

When a memory reference is being mapped through a PDR with the Bypass On Page bit set, the following will happen:

- If a write operation, data will be written into main memory and not into the cache. If a hit occurs, the cache location is invalidated.

- If a read operation, data will be read from main memory and not placed in the cache. If a hit occurs, the cache location is invalidated.

Note: As explained above, this function is really a bypass with invalidation on hits. This mode of operation preserves the contents of the cache (as much as possible) so that maximum cache advantage can be realized for subsequent cached operations.

The Bypass on Virtual Page mechanism is used by the RSX-11M software to avoid stale cache data for shared common data regions. (See discussion of "Local Cache Memories" in section 4.2.1). The bypass on virtual page function will be used to map to data regions when:

- the data region is shared by more than one task (i.e., declared as a common region), and

- the region has been declared writeable by at least one task.

The bypass on virtual page function permits shared, read-write data to be non-cached while still maintaining the advantages of the cache for all other data and instruction regions.

Force Bypass

This function permits the software to unconditionally force the bypass mode of operation for all memory references from the processor or Unibus. A bit is added to the Cache Control Register (CCR) that, when set, will force the same operation (bypass with invalidation on hits) to occur as is described in the preceding section.

The force bypass function is used for extremely short sections of code that must have access to the "current" contents of the shared memory in situations where a Cache Flush would impact processing speed. (See discussion of "local Cache Memories" in section 4.2.1).

Bypass and DATIP-DATO Operation for ASRB

The ASRB instruction is used in the multiprocessor system to implement semaphores ("locks"). Semaphores are used to synchronize and serialize access to critical regions of shared memory. There are two basic requirements for proper operation of the ASRB as a lock-type instruction:

- The read-modify-write operation of the ASRB instruction must read and modify operand data in the main memory.

- All other references to the target operand data must be locked out during the read-modify-write of an ASRB instruction.

There is a third requirement that is necessary for proper operation of the ASRB when used as a conventional instruction:

- If the operand data does exist in the cache, this data must be either invalidated or updated before the completion of the instruction.

In order to meet all of the above requirements, the appropriate logic will be modified so that:

- The bypass mode (with invalidation on hits) will be invoked when accessing the operand data of the ASRB instruction.

- An indication of the read-modify-write (DATIP-DATO) operation for the operand of the ASRB instruction will be sent out over the main memory bus. (This will be used by the "PAUSE" logic of the multiport memory to lock out access by any other port for the duration of the DATIP-DATO cycle.)

Note: The above operation is to occur only for operand operations of the ASRB instructions. The "Fetch" portions of the ASRB instruction as well as all operation of all other instructions are to remain as they are in the existing PDP-11/70.

Cache Bypass on Map Page

This function adds an attribute to each page of memory mapped through the UNIBUS Map. A bit is added to the high word of each UNIBUS mapping register. This allows the software to specify that all UNIBUS transfers through a page of the UNIBUS Map bypass the cache (with invalidation on hits). The operation invoked by setting these bits is the same as previously described for the Cache Bypass on Virtual Page operation.

Map/Cache Modifications For MK11 CSRs

The MK11 ECC MOS Memory has control and status registers located on the main memory bus. The present 11/70 has made no provisions for accessing such registers. The UNIBUS Map and Cache will be modified to map I/O page references to the Memory Parity CSR Addresses (772100 thru 772136 to addresses on the main memory bus (160772100 thru 160772136).

In order to keep the design compatible with the existing PDP-11/70, the map will contain switches (or jumpers) that permit the map to respond to a subset of the 16 possible CSR addresses. This will permit use of the processor in other (special) systems that have both MK11/MJ11 memory and UNIBUS memory. (These configurations are not supported by Central Engineering but this feature has been included for use by C.S.S..)

4.1.2 Multiport Memory

The multiport memory (MKA11), has been designed in conjunction with the MK11 MOS single port memory for the PDP-11/70. The MKA11 uses the same module types as the MK11 plus one new module, the multiplexer control module, and a new backplane.

Modular Design

The total multiport memory for the system will be contained in from two to eight MKA11 "boxes" *. Each box will typically contain 256K bytes of memory for systems having up to 1 megabyte (larger systems will typically have more than 256K bytes per box). The design of the MKA11 is such that failed boxes may be isolated from the system & repaired (without necessitating recabling) while the remaining memory is used for the on-line system.

Each MKA11 has a prewired backplane that can accept up to four port module sets. A two port memory can be upgraded to three or four ports by adding additional port module sets.

Up To Four Symmetrical Ports

Each MKA11 memory can have up to four ports through which the processors make connection. The multiplex control logic within the MKA11 arbitrates use of the memory on a per transfer basis (except DATIP-DATO cycles...see below). All ports have equal priority.

DATIP-DATO Cycle

The MKA11 has the same cycle types as the MK11 and MJ11, (READ, WRITE, & EXCHANGE cycles) plus one new cycle type, READ PAUSE. The READ PAUSE cycle will be used for the read portion of a read-modify-write (DATIP-DATO) operation from a CPU. A READ PAUSE is a read which is always followed by a write from the same port, with no other port allowed to access the shared memory (until the write operation occurs). A READ PAUSE will lock the multiplexer steering logic to a particular port until a write operation from the same port occurs.

* The actual packaging of the memory units will be in card cages. "Box" is used here to mean memory and power supplies.

Port Time-Out Logic

Faulty operation of a device using the multiport memory through one port should not prevent access to the memory through the other ports. Time-out logic within the MKA11 will sense if the multiplexer has remained switched to a port for too long. If the time-out logic indicates faulty operation, the port operation is terminated and the arbitration logic will allow the other ports to use the memory. This logic times the duration of individual READ, WRITE, or READ-PAUSE-WRITE cycles. Only hardware malfunctions will trigger a port time-out. It is not possible to cause a port time-out because of heavy or improper use by software.

Programmable Port Control

Each port of the memory contains logic that allows software manipulation of the memory parameters (address assignment, interleaving, etc.) for the port. These parameter assignments are on a per port basis and can be dissimilar for ports of the same memory. One of these parameters allows the software to select that the parameter assignments be made from the Manual Control Panel.

Manual Control Panel

A cable from each port interface makes connection to a manual control panel. There is one control panel per box of memory. This panel permits manual selection of the port parameters. Switches on the control panel allow a forced manual override of parameter selection on a per port basis.

A set of "ON-LINE/OFF-LINE" switches (one for each port) provide a mechanism for isolating a memory box from the system. When one of these switches is placed in the "OFF-LINE" position, the following will be true for the corresponding port:

- The port is disabled from driving any signals onto the connecting memory bus. The inputs of all the driver gates are disabled (grounded). All failure modes within the memory (except for failures of the driver gates and disable signal) are prevented from affecting the memory bus of the port.

- The connecting memory bus remains operational even during power up/down transients of the memory box.

- The address recognition logic for the port is disabled so that activity on the connecting memory bus cannot access the memory within the box.

The Manual Control Panel also has controls or indicators for:

- AC/DC Power ON/OFF switch (per box)
- Memory Power & Battery Status indicators (per box)
- ECC Disable/Enable switch (per port)
- Address/Interleave Switches In Use indicator (per port)

Independent Power Supplies

Each MKA11 will have its own power supplies and battery back-up. AC and DC power controls will be on a per box basis and independent from any other box or device in the system.

Power for the memory bus terminators is not supplied by the memory box and is routed to a set of connectors on the backplane (a separate power connection for each port). The last memory box on the memory buses will have power supplied to these connections from the respective processor power supplies. This is done so that the last memory box is not a single point of failure of the system.

Power Fail Operation

The power fail operation of the MKA11 is port specific. The action taken at each port is dependent upon the position of the ON-LINE/OFF-LINE switch on the memory control panel for the port.

If a port has not been disabled (ie., the ON-LINE/OFF-LINE switch is in the ON-LINE position), the following will be true for the port:

- Loss of AC power will cause the properly sequenced assertions of MAIN AC LO and MAIN DC LO on the port. Both signals will remain asserted until either the box has regained AC power and is again operational or the port is manually switched OFF-LINE.
- When AC power is regained, MAIN DC LO and MAIN AC LO on the port will be unasserted in the proper sequence. The unassertion of MAIN AC LO indicates that the memory is again able to service port requests. In the event of a DC failure, the unassertion of MAIN AC LO on the port will be delayed until the memory has completed all internal power-up cycling (writing of the initial ECC codes throughout memory).

- If during a power failure the memory box experiences a DC power failure (the battery back up has been exhausted), the memory will assert the Boot Enable signal* on the port along with MAIN AC LO. If selected to respond (via the BOOT/IIST Control Panel...see section 4.1.4, "BOOT/IIST Control Panel"), the assertion of Boot Enable will cause a hardware boot of the corresponding processor when MAIN AC LO is negated.

If a port has been switched OFF-LINE:

- MAIN AC LO, MAIN DC LO, or BOOT/ENABLE will not be asserted on the port under any conditions. The MAIN MEMORY BUS connected to the port remains operational and is not affected by the power-up/down transients of the MKA11.

4.1.3 Interprocessor Interrupt and Sanity Timer (IIST)

The Interprocessor Interrupt and Sanity Timer (IIST) is a new device designed for use in multiprocessor (and multicomputer) applications. The primary functions of this device are:

- Programmable Interrupt Device
- Programmable Boot Device
- Sanity Timer For System Activity

Programmable Interrupt Device

The IIST provides a mechanism for software running in a CPU to interrupt any subset of the CPUs in the system. This is the primary function of the IIST and is used by system software to cause rescheduling of system activity.

Programmable Boot Device

Software running in a CPU may cause any of the CPUs to begin executing the code of the M93Ø1-mP Bootstrap module that is local to each processor. (An electrical connection is made between the IIST interface and the M93Ø1-mP bootstrap module.) This function is used by the multiprocessor boot code to permit the first CPU that has been booted to bring up the other CPUs in the system.

* This signal uses a previously unused line on the MAIN MEMORY BUS to "wire-OR" the Boot Enable signals for all memory boxes on a per processor basis. This signal is not used by the processor but is routed via memory backplane connections through the BOOT/IIST Control Panel to the M93Ø1-mP module for the processor.

Sanity Timer For System Activity

The IIST contains a Sanity Timer for each processor. This timer is periodically refreshed by the Executive software. If the refresh commands do not occur within a programmed period, the IIST will issue an interprocess interrupt (or an interprocessor boot) to inform the other processors in the on-line system of the faulty operation. The software running in each processor can select which of the processors are to be interrupted (or booted) when the sanity timer for the processor times out.

Programmable Interrupt & Boot Mask

Registers within each IIST interface allow the software to selectively mask interrupt or boot action initiating from any subset of processors. This is used by the reconfiguration software to isolate one (or more) of the processors that have been established as an independent system (e.g., when an off-line system is established for maintenance, when a 4 CPU system is split into two dual processor systems, etc.).

Modular Design

The IIST consists of an interface for each processor. These interfaces are interconnected by the IIST bus. Each interface is a single PC board which mounts in a Small Peripheral Controller slot of the CPU backplane (or DD11 Peripheral Mounting Panel). The interconnecting IIST bus is "T" connected to permit disconnection of an IIST interface without disturbing the remaining IIST system.

Maintenance Operation/Power Fail

Each IIST interface has a connection to the BOOT/IIST control panel. This connection accepts an IIST interface disable signal from a set of switches on the panel. There is one set of these switches for each IIST interface. The corresponding IIST interface can be prevented from asserting and/or receiving selected signals on the IIST bus. These interface disable switches are used to isolate the on-line system from one (or more) of the processors that are being used for maintenance or as an independent system.

The IIST modules receive power from the processor (or expander) backplane and will lose power when the CPU is powered down. The IIST interface modules are designed so that loss of power on a module will not disturb the remaining IIST system.

4.1.4 BOOT/IIST Control Panel

The BOOT/IIST Control Panel provides a mechanism for:

- Manually initiating the M93Ø1-mP boot sequence on a processor.
- Manually selecting which processors are to be booted after recovery from a DC power failure in the memory system.
- Manually disabling (isolating) an IIST interface from the rest of the IIST system.

Manual Initiation of the M9301-mP Boot

A set of momentary action switches (one for each processor) allow the M9301-mP bootstrap modules of the respective processors to be activated.

Selection of Power-Up Re-Boot Action

The Boot Enable signal from each Main Memory Box is brought from the backplane of one of the memory boxes to the BOOT/IIST Panel. The Boot Enable signals indicates whether or not any of the memory boxes that have been enabled for use by a given processor are recovering from a DC power failure. Switches on the BOOT/IIST Control Panel gate the connection of these signals to the respective M93Ø1-mP bootstrap modules.

IIST Interface Disable

Switches on the BOOT/IIST Control Panel allow disabling of any of the IIST interfaces from the remaining IIST System. (See section 4.1.3, "Interprocessor Interrupt and Sanity Timer".)

(To Be Supplied)

Figure 4.1

BOOT/IIST FUNCTIONAL INTERCONNECTIONS

4.1.5 Time Of Day Clock

The Time Of Day (TOD) Clock is a new device that is designed for use in multiprocessor (and multicomputer) applications. The primary functions of this device provide:

- Continuously operational common time source
- High resolution common time value
- Clock interrupt source for each processor

This device is primarily intended for use by application level software in real time applications (for example, implementation of "smart" error recovery techniques).

The RSX-11mP software will use the TOD clock to set the internal time kept by the software whenever the system is booted or when recovering from a power failure and to periodically check the internal software time value against the TOD clock. The RSX-11mP software will function without the TOD clock, requiring operator action to set the time after a system boot.

Modular Design

The TOD consists of a master time keeping unit which broadcasts time information to slave interface units (one for each processor) over an interconnecting cable system. The slave interface units and the cable system are designed such that a slave unit may be powered down or disconnected without affecting the remaining TOD system.

Continuously Operational Common Time Source

The time keeping mechanism of the TOD clock has on-board battery backup that permits the time keeping function to be maintained through power outages lasting as long as several weeks. The time keeping logic is replicated and employs voting to detect and correct single failures within the master time keeping unit. The logic to transmit data (in the TOD Master), and receive data (in the slave interfaces) as well as the signals on the interconnecting cables are replicated. Logic within each slave will detect and correct any single errors incurred during transmission and reception of the time values from the TOD Master.

High Resolution Common Time Value

The TOD system provides a synchronized, high resolution time value that does not require high interrupt overhead. The time value read through the slave interfaces has a resolution of 100 microseconds and an overflow period of greater than one year. This time value is maintained in the TOD hardware and does not require interrupt service to update the time value.

Clock Interrupt Source For Each Processor

Each of the TOD Slave interfaces can be programmably enabled to provide a source of clock interrupts for the respective processors. The period of clock interrupts is 10 milliseconds. (Each processor is equipped with an alternate clock interrupt source, a line clock. The operator can cause the alternate clock interrupt source to be used if the TOD system fails.)

4.1.6 Multiport/Multiaccess Devices

This section addresses the attributes of multiport/multiaccess devices that are particularly important to the multiprocessor system. They are:

- Port symmetry
- Bus operation not dependent on device power
- Device operation not dependent on bus power
- Port disable (isolation)

The new multiport/multiaccess device designs (MKA11, IIST, and TOD) address these considerations. The existing multiport/multiaccess devices selected for the multiprocessor system (RPØ4-Ø6, RHØ1, and DTØ3) have features that fit these requirements. The system development effort will include validation of these attributes for both new and existing devices.

Port Symmetry

All ports to the device must provide the same capabilities. The arbitration scheme for port usage should be as symmetrical as possible such that over large amounts of activity, no port has priority over another. Activity on a port or ports (including faulty use of the device) must be prevented, as much as possible, from blocking access to the device through the remaining ports.

Bus Operation Not Dependent on Device Power

There must be some method of permitting all of the busses that connect to the ports to be operational while the device is without power.

Device Operation Not Dependent on Bus Power

Loss of power on one or more of the busses connecting to the ports must not prohibit operation of the device through the remaining ports.

Port Disable (Isolation)

There must be some mechanism to isolate the device from a port. When disabled (isolated), the device must be prevented from asserting signals on the port (even during transients of powering up or down the device) and port activity must be prevented from causing activity within the device (even for power transients on the bus for the port).

4.2 Software Component Requirements

This section describes the basic functional extensions that are required in the software for implementation of these multiprocessor systems. More detailed information can be found in:

- RSX-11M Multiprocessing Software Functional Specification

4.2.1 Executive Extensions (RSX-11mP)

The following is a cursory description of the basic functional extensions necessary in the RSX-11M Executive.

Single Shared Executive

These systems function with a single Executive resident in the shared memory system. Access to impure areas of the Executive by processes running on different processors is serialized by semaphores ("locks").

Local Cache Memories

The multiprocessor system exhibits a propensity for inconsistent data in the cache memories that are local to each processor (Pc). This is due to effects of:

- Process Migration
- Indirect I/O
- Shared Data Regions

Process migration occurs because of the symmetry of the system. Execution of processes (user tasks and the Executive) may shift from one processor to another as process or system activity dictates. When a process moves to a new Pc, residual data left in the cache from previous activity may be inconsistent with data in main memory for the new process. The cache flush mechanism is used to rid the cache of any inconsistent ("stale") data prior to the migration of a process. The cache flush is also used prior to the initiation of a new process. User process migration and initiation are always preceded by and a result of Executive action. The cache flush mechanism is invoked by the Executive to purge stale data that might result from migration or initiation of processes.

The execution of a process that has requested an I/O transfer and the service of the I/O request may occur on different processors. This is referred to as indirect I/O. On completion of an indirect I/O request, the executive forces a flush of the cache on the processor on which the originating process is executing. This purging operation is always completed before the executive notifies the task of the end-of-I/O status. Note that this is true only for asynchronous I/O: if the task had elected to enter a wait state until notification of I/O completion, its processor's cache would have been flushed because of rescheduling.

When a data region is shared by more than one process and at least one process can write into the shared region, there is a potential for inconsistent cache data. For user processes (tasks) stale cache data is avoided via the Bypass On Virtual Page mechanism. When the Executive sets up the Memory Management Unit of a processor for a task that will use a shared read/write data region, the Bypass On Virtual Page function is used for the page (or pages) that map to the shared region. The cache will not be used for references to the shared read/write region. Should the cache contain a copy of data that is being referenced on the bypassed page, the cache shared data will be invalidated (i.e., invalidation on hits).

A special case of shared read/write data exists for a portion of the Executive. Sections of I/O driver code are shareable and can be concurrently executing on different processors, however, the I/O drivers manipulate a shared data region. Stale data is avoided by the Force Bypass mechanism. The Force Bypass function causes direct access to main memory without modification of cache data (except for invalidation on hits). Maximum cache advantage is maintained for subsequent cached operations. Using the Flush or Bypass On Virtual Page functions in this instance would degrade processing speed.

Distributed I/O Devices

The multiprocessor system permits configuration in which I/O devices are not accessible to all processors. Extensions to the Executive data base allow the multiprocessor Executive to route I/O requests to the appropriate processor.

Dual Access Mass Storage Device Support

Each mass storage device can be accessed via either of two controllers. Extensions of the executive data base permit the association of more than one controller for a device. If while attempting to initiate I/O activity the Executive finds one of the access paths to a dual access device busy, the I/O activity will be attempted through the other access path for the device. Should one of the controllers be marked "off-line" by the reconfiguration facility, the device will remain accessable through the other controller.

On-Line Reconfiguration

The On-Line Reconfiguration facility consists of a set of Monitor Console Routines providing program-assisted reconfiguration of functional units under operator control, i.e.:

- Add memory boxes
- Add or remove processors, devices, or controllers from the pool of on-line system resources.
- Display the status of all system resources.
- Manipulation of Switched UNIBUSES

The reconfiguration commands also provide the capability of specifying controller-device combinations for on-line maintenance and troubleshooting.

DTØ3 Support

The RSX-11mP Executive software includes a driver for the DTØ3 UNIBUS Switch. This driver will be used by the reconfiguration code of the Executive.

Mixed MASSBUS Device Support

The multiprocessor system permits dissimilar devices to be connected to the same MASSBUS (e.g., disk and tape drives connected to the same MASSBUS). Controllers for MASSBUSes with mixed device types are allocated by the Executive.

Use of The TOD Clock

The software invoked upon booting of the system or when recovering from a power failure will check to see if the TOD clock is available and will set the internal software time value using the time value supplied by the TOD clock if possible.

Diagnostic Loader

A loader is included in the system software that will load a stand-alone diagnostic into a memory box from a Files-11 device. The files for the loader input can be created using FILEX to read the diagnostics from magnetic tape volumes in xxDP format.

4.2.2 Bootstrap Module Program

The M93Ø1-mP multiprocessor bootstrap provides an initial load mechanism for the multiprocessor system. There is one M93Ø1-mP module for each processor which, when activated, will boot the operating system via a single processor. Once booted, the Executive, via execution of a startup command file, will bring other processors and devices on line.

Console Boot Routine

The M93Ø1-mP allows specification of the boot device, unit number, and controller CSR address via console input. The unit number and controller have defaults of "Ø" and the "standard" CSR address for the specified device type. Operatorless rebooting is made possible by a software timer that will cause a boot from a hardware specified (via switches on the module) device if operator input does not occur following a prompt for input on the terminal.

Diagnostic Routines

The M93Ø1-mP tests all basic instructions for proper CPU operation and the memory box CSR's for proper configuration as seen by the processor.

Bootstrap Loader Routines

The M93Ø1-mP contains bootstrap loader routines for RPØ4,RPØ5, RPØ6,TU16/TE16,RMØ3 MASSBUS devices. Bootstrap loader code for all non-supported devices will be provided in the M93Ø1-mP documentation.

4.2.3 Diagnostics

The components of the diagnostic software for the multi-processor system are:

- On-line error logging
- User mode (on-line) diagnostics
- Stand-alone system and device diagnostics

4.2.3.1 On-Line Error Logging

The primary function of error logging is gathering information about errors that occur during normal operation of the system. A secondary function is the presentation of this information in readable form for operator analysis.

Summary of Error Logging features:

- Error Logging support is automatically included whenever multiprocessing is generated into the system.

- The system configuration will be entered into the Error Log whenever the configuration is changed, the system is bootstrapped, or Error Logging is turned back on.

- The logical and physical configuration of the system can be displayed at any time by the operator via a Monitor Console Routine (MCR) command. System configuration entries in the Error Log will be displayed, chronologically, in the Error Log Report indicating the configuration history.

- The system configuration associated with each error is referenced in each error report.

- The operator can enter system service messages into the Error Log. The messages are time stamped.

- Volume labels will be reported for all errors from mounted media (disk and magnetic tape).

- At time of error, a snapshot of all devices interfaced with the error log, for which I/O was pending, will be provided. All mass storage devices are included in the snapshot. The error report will include the function in progress and the address of the memory involved in the data transfer for each device at the time of the error.

- Power fail/recovery reports are time stamped and entered into the Error Log.

- Processor identification is included in device, spurious interrupt, and parity error reports.

- IIST, TOD clock, and DTØ3 UNIBUS Switch are supported in the Error Log.

- A Quick Summary Report is provided that indicates the total number of hardware and software errors for each device.

4.2.3.2 User Mode (On-Line) Diagnostics

The system software includes diagnostics that can be run by a user or field service representative to test MASSBUS mass storage devices and terminals. These diagnostics are tasks that run under the RSX-11M operating system and can execute concurrently with other system and user tasks. Normal operations of the system do not have to be disturbed except, possibly, to mount a scratch pack for the diagnostic.

User mode diagnostic and data reliability programs are available for:

- RPØ4, RPØ5, RPØ6
- TU16 *
- Terminals

* User mode diagnostic support for the TE16 is being investigated.

4.2.3.3 Stand-Alone System and Device Diagnostics

Once an off-line maintenance system has been established, any of the standard stand-alone diagnostics can be used for maintenance. Stand-alone diagnostics are supplied as part of the XXDP Diagnostic System package.

If the off-line maintenance system has been established with a load device, the XXDP Diagnostic Package may be used in the same manner as in conventional single processor systems to perform maintenance.

Alternately, system software provides a mechanism for loading diagnostics (via FILEX) from the XXDP media into a FILES-11 volume of the on-line system. The diagnostic can then be loaded (via a loader utility) into the memory box that will be used by the off-line system for maintenance. This makes it unnecessary to dedicate a load device for maintenance. This software also makes it possible to store diagnostic programs on the system disk to facilitate rapid service.

APPENDIX A

Multiprocessor System Availability Analysis

MULTIPROCESSOR SYSTEM AVAILABILITY ANALYSIS

I. Background

The availability of a repairable system is the ratio of the total up-time (i.e. sum of all working time intervals) over the total mission time. In a system made up of interconnected components, the total system availability can be obtained, using classical methods, from the components' availabilities and interconnections. Repairable systems can be classified either as minimum (i.e. systems with no redundant components and in which every component's failure implies a total system failure) or as non-minimum (i.e. systems with replicated elements). Non-minimum systems have higher availabilities than the associated minimum ones because every component's failure does not necessarily imply a system failure. A tool has been developed for calculating the availability of any system whose basic components' availability characteristics and interconnections are known.

The (2-Pc) multiprocessor system shown in figure 3.2 of the system functional specification is a non-minimum system. The associated minimum system is defined in section II-2 below. Non-critical components are not used in the calculations since the basic multiprocessor system is defined as consisting only of critical components.

II. Availability Computation

The basic component's availability values (MTBF's and MTTR's) used in computing systems' availabilities, are approximated values. Because the confidence level in these numbers is quite low, the obtained results should not be taken as absolute values, but rather as relative ones (we are pushing to get better values). Components for which the above metrics do not exist as yet (e.g. IIST, TOD, MKA11), were assigned estimated values.

1. Assumptions:

The following assumptions were made in calculating systems' availabilities:

a) Components' coverage with respect to failure detection and subsequent repair is as good as in the uniprocessor environment.

b) The MTBF values used are calculated ones, based on MIL-HBK-217B in fixed-ground environment.

c) Software related failures and externally induced failures, such as power glitches, are not taken into account.

2. Minimum system:

The minimum system, as defined above, is a single processor system containing the following components:

Processor	- (1)	11/70
Disk	- (1)	RP05 } + (1) RH70
Tape	- (1)	TU16 }
Console term.	- (1)	LA36 + (1) DL11W
Bootstrap	- (1)	M9301
Clock	- (1)	DL11W
Memory Box	- (1)	MKA11
16 com. lines	- (2)	DZ11

3. Metrics Definitions:

The following metrics are shown in the next section with corresponding values for the minimum system and/or the multiprocessor system where applicable.

- System availability - is the availability of the system calculated as explained above.
  (zero - mean time to reconfigure)

- System unavailability reduction factor - is the ratio of the minimum system unavailability over that of the multiprocessor system (i.e. availability improvement).

- System MTBF - is the MTBF of the system calculated from the corresponding system availability with the assumption that the system's MTTR is equal to the weighted average of its components' MTTRs.

- System MTBF improvement - is the ratio of the multiprocessor system MTBF over that of the minimum system.

- System component failure rate - is the rate at which components are failing in the system. Note that in the multiprocessor system a component's failure does not necessarily imply a system failure.

- System price - is the price of the system as computed from the January '77 end-users price list.

- System price factor - is the ratio of the multiprocessor system price over that of the minimum system.

- Effective mP system availability - is the resulting mP system availability when the mean time to reconfigure is as indicated.

- Effective unavailability reduction - is the ratio of the minimum system unavailability over the effective mP system unavailability.

4. Results:

Metrics	Min. System	MP System	Units
System availability:	.98562	.99992	
System unavailability reduction factor (i.e. availability improvement):	-	171.2	
System MTBF:	441.	76540.	Hours
System MTBF improvement factor:	-	173.7	
System components failure rate (i.e. expected Field Service call rate):	1.84	3.89	per 1000 hrs.
System price:	149K	323.4K	dollars
System price factor:	-	2.16	

Mean time to Reconfigure (minutes)	Effective mP System Availability	Effective mP System U.R.F.
0	.99992	171.2
5	.99959	35.2
10	.99926	19.6
15	.99894	13.6

Note: The absolute mean-time to reconfigure will generally fall in the 1 to 10 minutes interval. Thus, the 15 minute mean-time to reconfigure, can be considered as worst case.

Appendix B Loose vs. Tight Coupling

The fact that the loose vs. tight coupling controversy continues in high availability applications indicates that no one has yet established beyond doubt that one scheme is vastly superior to the other (except in very well-bounded cases). Since the multiprocessor system described in this specification is not tightly bounded by a priori knowledge of target application, the selection of tight coupling has evolved through a combination of technical and marketing factors, on a backdrop of economic reality.

Since the Eckhouse/Cutler/Miller investigations of multiprocessing were based on shared memory, a certain amount of momentum was given to a close coupling/RSX-11M approach before the formal product/project notion occurred. This approach was a manageable way for Eckhouse to achieve his objectives, since available software (RSX-11M) lent itself well to the hardware available.

Since one of the principal objectives of the project was to gain engineering and marketing experience with multiple CPU 11-based products, utilizing existing hardware and software wherever possible, the decision to select the tightly coupled/RSX-11M path was de facto.

Aside from the fact that such a product is feasible in a timeframe consistent with the 16 bit PDP-11 era, an extremely important consideration for marketing, engineering, and support aspects, there are a number of other supporting arguments, enumerated below:

- Reduces the number of failed states the system is subject to. Because of the configuration restrictions imposed (i.e., N + 1 of all components which comprise a "critical configuration", one which will support the customer's application), any single failure causes loss of one functional unit, but seldom loss of the ability to perform the critical function.

- The software complexity inherent in a super-supervisor or network operating system, providing graceful degradation, general fault tolerance, load balancing, shared volume access, and transparent system access to the user is perhaps an order of magnitude greater than that of the closely-coupled approach, which still provides all those features.

This complexity goes against loose coupling for a first implementation, not only because of cost in manpower and time, but also because increased complexity produces an increase in possible software failure states.

- A tight coupling scheme normally requires less hardware, in that non-critical components need not be duplicated.

- Since the next generation of PDP-11's are currently directed at close coupling when multiprocessor configurations are implemented, the use of close coupling in this project will provide valuable engineering, marketing, and service data to those implementers.

Tight coupling does have the following disadvantages:

- The single shared operating system is more prone to total system failure, due to logic error or hardware-induced corruption, than multiple independent operating systems. Furthermore, shared devices may serve as conduits to propagate errors.

- There are physical limitations to the number of independent ports to a given memory, thus limiting the number of processors sharing it. Moreover, although a shared memory bus could seemingly solve the problem, bus contention would negate this advantage for a high-performance processing element, such as the 11/70.

- As the number of processors in the multiprocessor system is increased, contention for the single, shared executive increases. Executive contention limitations probably make the tightly coupled approach inappropriate for systems containing large numbers of processors.

Appendix C Choice of RSX-11M

RSX-11M was chosen as the operating system component of the multiprocessor system in part because feasibility of its use had already been demonstrated (Eckhouse/Cutler/Miller), but also because of the following:

- Manageable internal serialization of access to system critical regions, implemented by software interlock, rather than elevation of BR level.

- Full functionality, capable of real-time sensor-based or communications use, as well as providing full multi-language program development facilities.

- Broad current or committed use as an internal kernel product for TPS and MCB.

- Compatible with emulator mode on next generation PDP-11's.

- Mixed massbus device support previously demonstrated.

- Applicable to broadest range of configurations of any PDP-11 operating system, providing flexibility in follow-on product direction, as well as flexibility in backup hardware contingency choices for this product.

- Excellent supportability performance, important for a high availability system. Made Datapro's Honor Roll, a mark of high customer acceptance.

Among our other major PDP-11 operating systems, RSTS/E is too narrow in capability focus; RSX-11D and IAS use elevated BR levels to achieve synchronization, and leave synchronization to each handler or process, rather than centralized as in RSX-11M; and RT11 is not a multiprogramming system, nor does it have the wide range of device and language support.

Appendix D Choice of 11/7Ø

Originally, the PDP-11 multiprocessor program was to be initiated with multiple 11/34's, since this was a very low risk hardware approach: no cache, and MA11 multiport memory already available. However, strong product line preference dictated the change to 11/70. The following were the reasons for selection of 11/70, and the implications resulting from that selection:

- The 11/70 provides for large main memory, whereas the other PDP-11s are restricted to 124KW. Physical address extension seems almost mandatory for multiple 11's executing from a single shared main memory.

- Multiple bus structure of the 11/70 provides greater flexibility in fault isolation and reconfiguration.

- The cache of the 11/70 provides both advantages and disadvantages, the disadvantages being related to the additional hardware modifications required to solve stale cache data problems, and the advantages being enhanced performance, and understanding on reliability and performance of cached multiprocessors.

Implications of the choice are increased risk due to expansion of scope of effort, an increase in cost of about $500K, and deferral of development of lower priced multiprocessors, which may have broader market appeal. In fact, a non-cached 11/34 breadboard will still be implemented for subsystem checkout and project insurance.

Appendix E Design of Multiprocessor With Cached Memory

Multiprocessing With Caches

INTRODUCTION

The addition of a cache to a processor increases the effective memory utilization through data buffering in a small fast memory. The cache can be utilized at the intrinsic cycle time of the CPU (assuming proper design and hit ratio goals) thereby achieving the best possible processing rate. For a UNIBUS architecture, a cached processor offers a lower bus utilization; the optimum performance occuring when the memory on the UNIBUS is requested only when a main store reference is actually required.

A cache may be organized in many different ways, utilizing several different cache update schemes (write-back, write-thru, etc.). Regardless of the update scheme, the goal is to maintain the contents of the cache in a consistent state so that it accurately represents the contents of main memory (the backing store for the cache). If the cache contents are inconsistent, then the software may not execute properly.

Our 11/70 uniproessor systems exhibit the propensity for inconsistent cache contents because they allow simultaneous operation of the CPU and I/O devices on a cycle stealing basis. The problem occurs becuase the cache is between the CPU and main memory so that I/O data modifying main memory is not reflected in terms of the cache contents. To avoid this potential problem, I/O traffic to main memory forces an invalidation of the cache contents should a match occur between the I/O memory address and the cache tag address.

Multiprocessor configurations where two cached CPU's are addressing the same main memory, exhibit the same propensity for generating inconsistent cache contents. The difference found in the multiprocessor system is that not one, but several caches have to be maintained in a common, consistent state. The reason for multiple caches in a multiprocessor system is that multiprocessor performance (in a strict computational sense) is a direct fucntion of the utilization of the memory bus by a particular processor [1]. The effect of a cache is to increase the computational performance of the system since it dramatically reduces the fraction of the time the memory bus is utilized. In addition, the effective I/O bandwidth is higher because a greater fraction of bus cycles are available.

---

[1] R. Eckhouse & D. Nelson, "Closely Coupled Multiprocessor Systems", Oct 75

STALE DATA IN A MULTIPROCESSOR SYSTEM

In tightly coupled multiprocessor systems there will be "N" independent, unsynchronized caches connected to a common memory with either shared or independent I/O busses. Given these indepedent caches, it is concievable that they are in different states so that a process which migrates from one processor to another, while accessing a shared data structure, will in fact see different copies of the same data. In effect, the cached data has become "stale" and this is referred to as the "stale data" problem. A particular scenario exhibiting this characteristic might be described as follows.

We assume that there are two processes, A and B, which are executing simultaneously on a multiprocessor system. Each process has occasion to reference a shared data item, D. Let us look at three instances in time where at time t1 process A reads D, at time t2 process B reads and alters D, and at time t3 process A again reads D. We may represent these operations as shown below:

Time	Action	Result
t1	A reads D	D moved to Pc D in backing store D in cache A
t2	B reads D	D moved to Pc D in backing store D in cache B
	B alters D to D'	D' in cache D' in backing store
t3	A reads D	D in cache B D moved to Pc D' in backing store

Clearly we have a case of stale data in cache B.

In a uniprocessor system with only one cache it is possible to solve this problem by having the hardware invalidate the cache contents whenever something is written to the backing store. In a multiprocessor system the proper solution to the problem is to insure that any writes to the cache (either NPR or CPU) invalidate any other caches in the system which may have the same tag. As shown by Nelson [2], this "cross invalidation" of cache data causes an effective performance degradation for more than two processors. In essence the argument goes that if writes are only 10% of all memory references, the cross invalination will force the cache to

---

[2] D. Nelson, "Cache Interference Calculation", 15 Jan 76

spend 40% of its time checking cache tags for a four processor system. In addition, fairly sophisticated techniques are required to queue up addresses while invalidation or an instruction are in progress. In the worst case, a processor doing a write will stall until all other caches have completed tag checking. Thus although it is a means for solving the problem, cache invalidation is rejected because it is a poor performer.

SOLVING THE STALE DATA PROBLEM

The stale data problem can occur either a) when a single process is scheduled to run on different processors at various times during its execution, or b) when processes share a common data area. In the latter case, a simple expedient is to not cache the shared data so as to remove the problem (although the problem of cooperation between sequential processes generally requires additional serialization through the use of locks).

Avoiding The Use Of The Cache

We may carry this expedient out by associating an explicit attribute with the shared area. This attributed is often loosely referred to by several names including: "don't cache", "cache bypass", "forced miss" and "read thru". Actually all of these names are not equivalent so that it is useful to differentiate them as follows:

DON'T CACHE: Do not make a copy of the value in backing store and place it in the cache. This attribute is generally associated with memory management and implies that a word or segment of main memory is never cached. Indeed, if a cache tag hit occured on either a read or write operation, what happens to the cache is undefined.

CACHE BYPASS: Treat the system as if there wasn't a cache present. What happens if there is one and a tag hit occurs is undefined. Note that this case is essentially the same as the don't cache case.

FORCED MISS: Whether or not a cache hit occurs, force a miss to happen. Depending on the read/write stragegy used on a cache hit, update the cache accordingly.

READ THRU: Same as a forced miss.

Using the cache bypass (or don't cache) attribute, it is possible to share a common data area. Since such shared areas must be explicitly declared to the linker, in order that a mapping register be allocated to it, the existence of the cache bypass implies that a bit be set in the corresponding mapping control register. Thus, when a memory reference is made to the shared data

area, the backing store will be accessed directly. This insures that an accurate image of the current data in backing store has been obtained. Using the previous example, we may describe the operations as follows:

Time	Action	Result
t1	A reads D	D moved to Pc
	(cache bypassed)	D in backing store
t2	B reads D	D moved to Pc
		D in backing store
		D in cache B
	B alters D to D'	D' in cache if tag match
		D' in backing store
t3	A reads D	D' moved to Pc
	(cache bypassed)	D' in backing store

Since the absolute serialization of each process's access is not guaranteed by the cache bypass, time dependencies in data access among concurrent processes must be carefully avoided.

By combining the cache bypass capability with the notion of ownership, it is possible to allow a user-level process to "LOCK" a shared data region, while still achieving the overall performance advantage of cache bypass [3]. A A software algorithm, implemented as a library subroutine, is used to create a "LOCK" on the shared data region. The cache bypass capability insures the validity of the lock. If a process fails in an attempt to set a lock, an executive call to queue the lock request is required. Since both the lock and the shared data are normally co-resident in the same segment, neither would be cached.

A secondary advantage which may be obtained from a cache bypass capability is to reduce the total number of cache data replacements when executing Interrupt Service Routines (ISRs). Whenever the ISR makes reference to the common executive data, it is necessary to avoid stale data between ISR invocations. Should the ISR code be sufficently short so as to not benefit greatly from the caching, then the loss of time to not cache its code and data access will be offset by not having to completely invalidate and refill the cache each time a running process is interruptec.

Performing A Cache Flush

Sometimes it is not sufficient to simply bypass the use of the

---

[3] Miller, Hardware Requirements of RSX-11Mp on the 11/70", 29 Jan 76

cache to execute a short piece of code, or to share some ommon data. For instance, different invocations of the executive may lead to stale data being left in the system common area. While it might seem desirable to simply set the cache bypass bit for this area, it generally is unfeasible to a) force this data to be contiguous and reside in an area mapped by one memory management register, and b) give up the use of a memory management register exclusively for the use of not caching one piece of the executive. Indeed, should the system common area require two (or more) memory management registers to map it, or should the code contain some "impure" data areas, there is no alternative by to completely bypass the use of the cache for the entire time the executive is in execution.

To handle this problem, a "cache flush" function is added. Thus whenever the executive is entered, a cache flush is performed before accessing any system common information or impure areas. While there is an initial performance loss while the cache is being refilled, thereafter performance will be on a par with the exectuion of non-executive code.

Conclusions

The cache bypass facility is primarily a performance improvement over the cache flush. The improvement is a result of reducing the number of cache invalidations or the total number of cache data replacements. The performance improvement can be split into two parts:

1) on a single PDR basis which essentially improves performance for processes simultaneously accessing common data, and

2) on a global basis which is used to minimize the disruption of the steady state cache condition by allowing short segments of system code to execute directly from the backing store. The stale data problem for shared data may be overcome either by forcing a process to "own" the shared area while it is being used, or by bypassing the cache on memory references to the shared area.

Glossary Of Terms:

AVAILABLE A system is said to be available when it is able to perform the critical function.

AVAILABILITY A ratio of the time that a system is available to perform the critical function over a given period. Important aspects of system availability are redundancy of components, expected mean time between failures (MTBF) of all critical components, and mean time to repair (MTTR) of all critical components.

BUS RUN A path for UNIBUS data and control signals that cannot be separated under program control.

CONFIGURATION The selection of the physically connected set of operational components of a computing system which can be used by the operating system currently in control, as established by initial system generation and on-line modification of software device tables. Normally, configuration is the process of selecting hardware components which, when physically interconnected, will provide the basic set from which a particular hardware environment may be selected by the operating system.

COVERAGE Probability of detecting, isolating, and recovering from a fault/failure.

CRITICAL Pertinent to that central application for which a particular installation has been established, and for which high availability is mandatory. In particular, all those components of software and hardware without which the central mission of the system cannot be performed are said to be critical.

CRITICAL COMPONENT Any part of a system required to perform the central mission of the system.

CRITICAL FUNCTION The central mission of a particular computer installation which is its reason for being established and which requires bounding of non-availability.

ERROR "An item of information which, when processed by the normal algorithms of the system, will produce a failure." (Melliar-Smith/Denning).

FAILURE "An event at which a system violates its specifications." (Melliar-Smith/Denning).

FAULT "A mechanical or algorithmic defect which may generate an error. The failure of a component may generate an error which will appear as a fault elsewhere in the system." (Melliar-Smith/Denning).

FAULT-TOLERANT A system may be said to be fault-tolerant if it may continue to perform the critical function despite the occurrence of faults.

FUNCTIONAL UNITS The major operational units of a digital computer; for example, central memory, arithmetic and logic, control, input/output controller or processor.

INTERPROCESSOR INTERFERENCE A reduction in the effective speed of a processor due to delays in accesses to hardware or software elements caused by the elements being in use by other processors.

LOCK The protection of a data set by inhibiting access to it by all processes except the single current user.

MULTI-ACCESS The ability to control a functional unit from more than one controller.

MULTICOMPUTER SYSTEM A system having more than one processor but not meeting all of the criteria of multiprocessors.

MULTIPORT The existence of more than one port used for the transfer of data to or from a functional unit.

MULTIPROCESSOR (The definition given in this specification is quite explicit.) Generally, a multiprocessor computer contains two or more processor units with the following attributes:

- All units are as nearly identical in capability as practical.
- Each unit has access to shared common central memory.
- Each unit has common access to at least a portion of the I/O devices.
- All units are controlled by one operating system that provides interaction as required between the processors and the programs they are executing at the task and hardware (interrupt) levels.

MULTIPROGRAMMING A system providing the capability for more than one program to seem to be executing concurrently, compete for resources, and share the facilities of a common operating system. Normally, additional attributes assumed of a multiprogramming system are protection of users from damage inflicted by other users, protection of the executive from all users, management of all shared resources by the executive, and prevention of unwarranted access to the resources of any one user (or the executive) by any other. It is important to note that multiprogramming produces more work per unit time than serial programming in only two cases:

1.) Sufficient input/output requests requiring completion (synchronous I/O) are issued by the programs in some stage of execution in a system that other programs may use the processor during wait time (overlap); and/or

2.) Additional processing elements are made available to the operating system (multiprocessing), allowing more than one program to actually execute at the same time.

OFFLINE The resource is known to the operating system and one or more of the following are true:

- The resource is not physically present.
- A path for the transfer of control and data signals does not exist from the online system.
- The resource has not been marked for online status by the reconfiguration software.

ONLINE The resource is known to the operating system and all of the following are true:

- The resource is physically present.
- A path for the transfer of data and control signals exists from the online system.
- The resource has been marked for online status by the reconfiguration software.

OPERATING SYSTEM All those software components which provide an operating environment for the development and execution of software programs. Central to the operating system is a stored program called the executive, part of which always resides in main memory, and which acts as a manager for all resources of the system.

PORT An interconnection point or interface to a functional unit for the transfer of data.

PRIVATE MEMORY That portion of central memory usually accessible only to a single processor.

RECONFIGURATION Changes in the system organization that may be achieved by either manual or automatic means.

RELIABILITY A measure of the ability to function without failure.

SWITCHED BUS RUN A UNIBUS run that can be switched by a UNIBUS switch.

THREAD Set of functional units necessary to form an operable system.