SISTEMA MULTIPROCESADOR CON BUSES MULTIPLES

MEMORIA PRESENTADA POR DOLORES ISABEL REXACHS DEL ROSARIO, PARA OPTAR AL GRADO DE DOCTOR EN INFORMATICA.

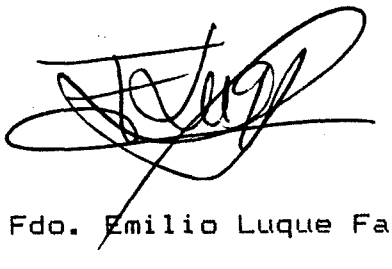
BARCELONA NOVIEMBRE 1987

SISTEMA MULTIPROCESADOR CON BUSES MULTIPLES

Memoria presentada por DOLORES ISABEL REXACHS DEL ROSARIO, para optar al grado de doctor en Informática por la Universidad Autónoma de Barcelona. Trabajo realizado en el Departamento de Informática de la Facultad de Ciencias de la Universidad Autónoma de Barcelona, bajo la dirección del Dr. Emilio Luque Fadón.

Bellaterra, Noviembre 1987.

Vº. Bº. Director Tesis



Fdo. Emilio Luque Fadón

Isabel Rexachs del Rosario

INDICE

Quiero expresar mi agradecimiento a todas aquellas personas que de una forma u otra han colaborado en este trabajo, en especial a:

Al Dr. D. Emilio Luque Fadón, al que deseo expresar mi agradecimiento más efusivo, ya que sin su valiosa dirección y ayuda no hubiera sido posible la realización de este trabajo.

A la Dra. Dña. Ana Ripoll Aracil por sus acertadas sugerencias y constante estímulo que tanto han infuído en la realización de este trabajo.

A los miembros del Departamento de Informática muy especialmente a Carles Siscart, Joan Sorribes, Porfidio Hernández, Lorenzo Moreno e Ignacio Serra, por el constante apoyo recibido.

CAPITULO 2

2.- ESTRUCTURA DE INTERCONEXION DE BUSES MULTIPLES.

CAPITULO 3

3.-REALIZACION DE UN SISTEMA MULTIPROCESADOR.

PROLOGO

La presente memoria describe el trabajo desarrollado para la concepción de un sistema multiprocesador, utilizando una red de interconexión de buses multiples. Esta memoria ha sido estructurada en tres capítulos cuyos contenidos resumimos a continuación.

En el capítulo 1 se presenta una introducción sobre sistemas multiprocesadores, indicando sus características principales, comentando algunos de los parametros más importantes que intervienen en su diseño tales como redes de interconexión, requerimientos de software y árbitros. Siendo un aspecto clave en los sistemas multiprocesadores la red de interconexión, se describen las redes de interconexión más utilizadas en los sistemas multiprocesadores. Seguidamente se comentan brevemente diferentes aspectos del software. Para resolver los conflictos que se producen en los sistemas multiprocesadores en la utilización de los recursos comunes se hace un estudio sobre mecanismos de arbitraje. Finalmente se indican los objetivos de este trabajo.

El capítulo 2 se centra en la red de interconexión de buses múltiples, puesto que es la red propuesta para el sistema multiprocesador. Se resumen sus características principales y se hace una breve recopilación de diferentes trabajos publicados sobre buses múltiples y su evaluación. Se presentan dos formas de utlizar los buses múltiples para reducir su costo, los esquemas de interconexión reducidos y la multiplexación en los buses múltiples. Seguidamente se propone un metodo de arbitraje para la gestión de buses múltiples analizandose dicho sistema y se expone como utilizar este tipo de arbitraje en esquemas de interconexión completos, en esquemas de interconexión reducidos y en un sistema con buses múltiples en el que se realiza multiplexación. Por último se comentan dos formas de realización del sistema de arbitraje, una centralizada y otra distribuida.

El capítulo 3 se ha dedicado a la descripción del sistema multiprocesador con buses múltiples realizado, se exponen las diferentes decisiones de diseño estudiadas, sin hacer especial incapié en la red de interconexión ya que ésta ha sido estudiada en el capítulo anterior. A continuación y por último se describe el prototipo realizado, en el que se utiliza

un sistema de arbitraje centralizado para la gestión de buses multiples y se detallan los árbitros utilizados en el prototipo.

CAPITULO 1

SISTEMAS MULTIPROCESADORES

1.1 Introducción

La existencia de microprocesadores y otros módulos VLSI, han promovido al desarrollo y diseño de sistemas multiprocesadores. Los sistemas multiprocesadores representan una opción de diseño para acercarse, en la construcción de sistemas computadores, a la medida de los requerimientos de una aplicación inicialmente dedicada y tambien para adaptarse a cambios durante el ciclo de vida del sistema.

Los sistemas multiprocesadores estan definidos como una subclase de sistemas computadores MIMD (Múltiple Instrucción Múltiples Datos), según la clasificación hecha por Flyn (Fly-72), en el que los procesadores tienen acceso común a la memoria principal y canales de estrada/salida y existe un único sistema operativo controlando todo el sistema (Ens-77).

En la definición clásica dada por Enslow (Ens-77) una arquitectura multiprocesador debe tener las siguientes características:

- Dos o más procesadores de capacidades aproximadamente comparables.
- Memoria compartida. Todos los procesadores comparten accesos a un conjunto de módulos de memoria común.
- Periféricos compartidos.
- Sistema operativo único.
- Interacción hardware/software, a todos los niveles.

En un sistema multiprocesador, además de las memorias y dispositivos compartidos, cada procesador puede tener su propia memoria local y dispositivos privados.

Los multiprocesadores pueden ser caracterizados por dos atributos (Hwa-84):

1.- Un multiprocesador es un único computador que incluye múltiples procesadores.

2.- Los procesadores se pueden comunicar y cooperar a diferentes niveles para soluccionar un problema dado. La comunicación puede ser enviando mensajes de un procesador a otro o compartiendo una

memoria común

La organización hardware del sistema multiprocesador está determinada principalmente por la estructura de la red de interconexión usada entre procesadores y memoria.

Una red de interconexión es una conexión a través de conmutadores y enlaces que permite la comunicación de datos entre procesadores y memorias en un sistema multiprocesador. Algunas redes de interconexión han sido propuestas para estos sistemas, tales como el bus único, el crossbar, memorias multientrada (multiport), multietapa (MIN), bus múltiple y otras.

1.2 Clasificación de los sistemas multiprocesadores.

Los sistemas multiprocesadores pueden ser clasificados por el grado de acoplamiento entre procesadores (capacidad de los diferentes elementos para compartir recursos) y la forma de la intercomunicación entre los procesadores, en dos tipos: Sistemas Debilmente Acoplados y Sistemas Fuertemente Acoplados.

-Sistemas multiprocesadores debilmente

acoplados: Un sistema multiprocesador debilmente acoplado está formado por procesadores autónomos con sus memorias locales y un conjunto de dispositivos de entrada/salida. Los procesadores son interconectados via circuitos de entrada/salida. La comunicación entre procesadores es realizada a través de una red de interconexión, requiriendo las transferencias de información operaciones de entrada/salida. Se utilizan cuando los procesos gastan la mayoría de su tiempo en sus estructuras de datos locales y muy poco tiempo en intercambio de datos.

- **Sistemas multiprocesadores fuertemente acoplados:** Son los sistemas en los que todos los procesadores tienen acceso al espacio de memoria común, en cada procesador puede existir una memoria local o buffer de alta velocidad (Cache). Los procesadores se comunican a traves de una memoria principal compartida. La memoria común proporciona un medio muy rápido de transferencia de datos, haciendo posible compartir un código común. Los sistemas fuertemente acoplados deben utilizarse si existen requerimientos de alta velocidad o procesamiento en tiempo real.

Los sistemas debilmente y fuertemente acoplados estan representados en las figuras 1.1a y 1.1b respectivamente.

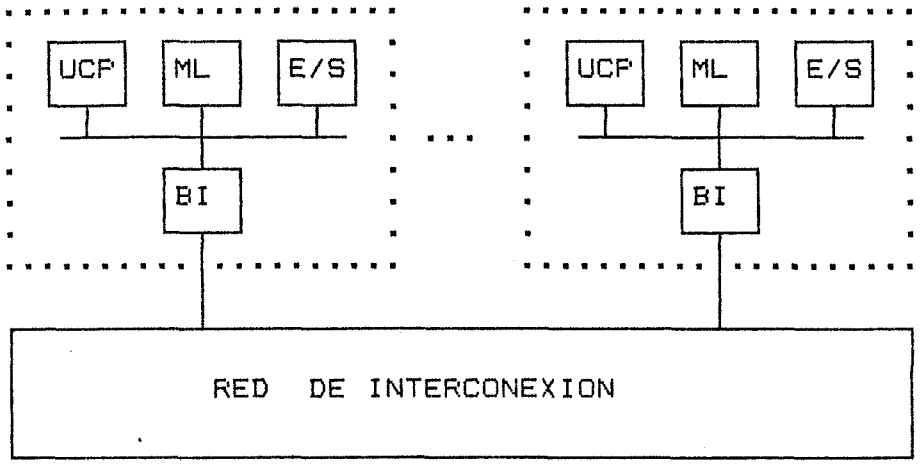


Fig. 1.1a Sistema multiprocesador debilmente acoplado

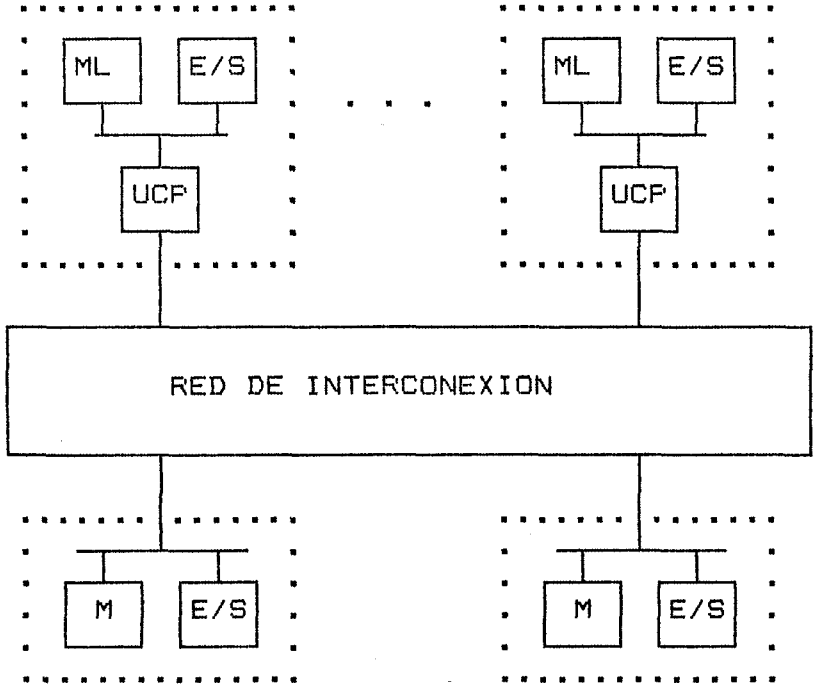


Fig. 1.1b Sistema multiprocesador fuertemente acoplado

Los sistemas fuertemente acoplados estan clasificados en (Bow-80) en cuatro configuraciones mostradas en la figura 1.2:

-Sólo Recursos Globales: En esta configuración todos los periféricos y memoria son comunes. Todos los procesadores ejecutan código desde esta memoria. El sistema es modular y homogéneo física y funcionalmente.

-Memoria Global y Privada: En esta configuración el tráfico a través de la red de interconexión se reduce al existir código en la memoria privada de cada procesador.

-Sistema con Recursos Distribuídos: En esta categoría existen periféricos asignados permanentemente a un procesador.

-Memorias de Doble Entrada: Esta configuración permite que las memorias privadas sean accedidas desde el bus. Puede crearse un mapa de memoria en el cual cada procesador puede ver las memorias privadas de otros procesadores como extensiones de la suya propia.

El conjunto de procesadores utilizados en un sistema multiprocesador pueden ser **homogéneos** o **heterogéneos.** Es homogéneo si los procesadores son funcionalmente identicos. Aunque los procesadores sean homogéneos, pueden ser asimétricos, es decir, dos componentes funcionalmente idénticos, pueden

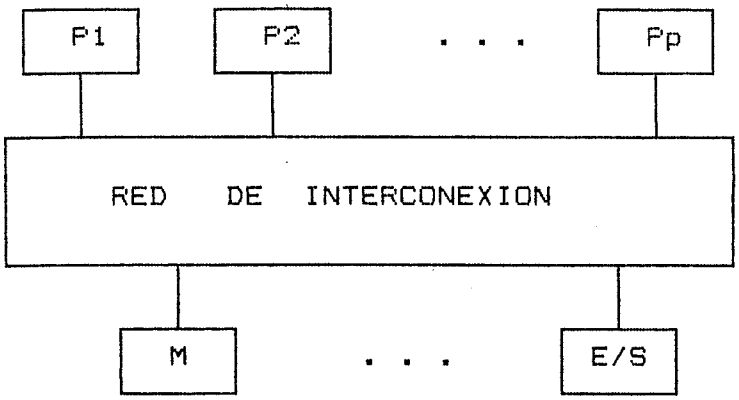


Fig. 1.2a Solo Recursos Globales.

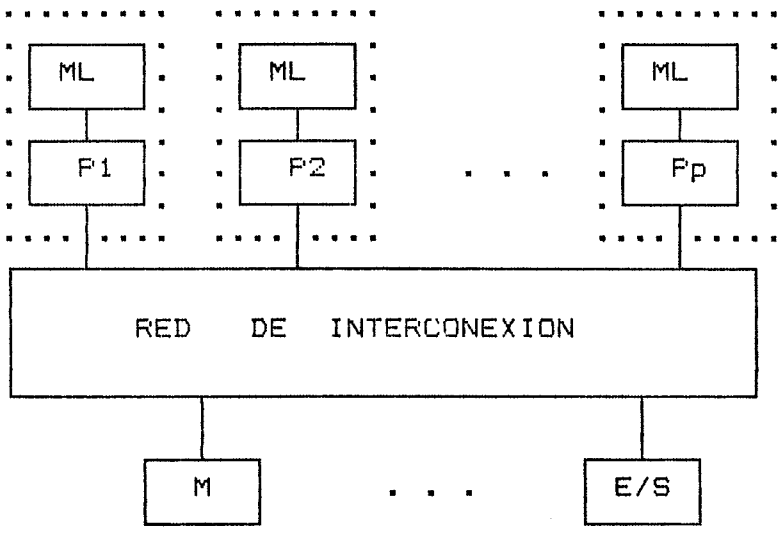


Fig. 1.2b Memoria Global y Privada.

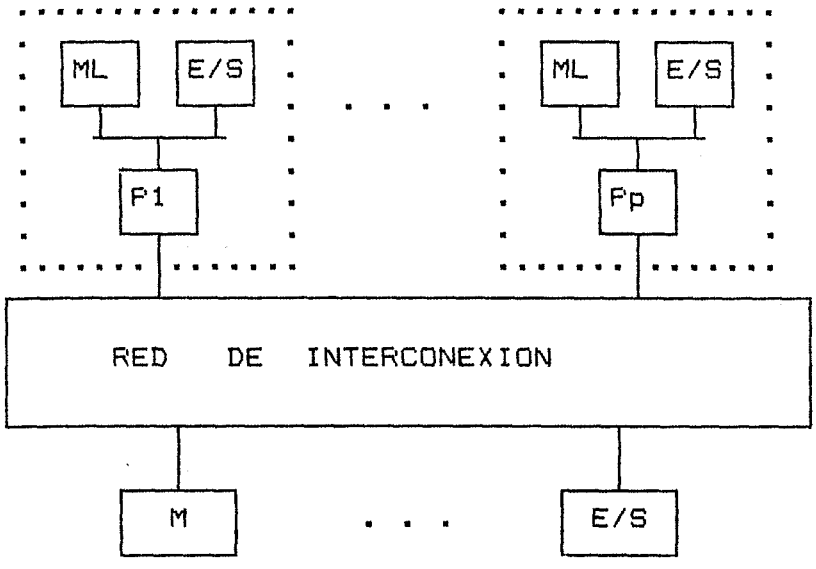


Fig 1.2c Recursos Distribuídos.

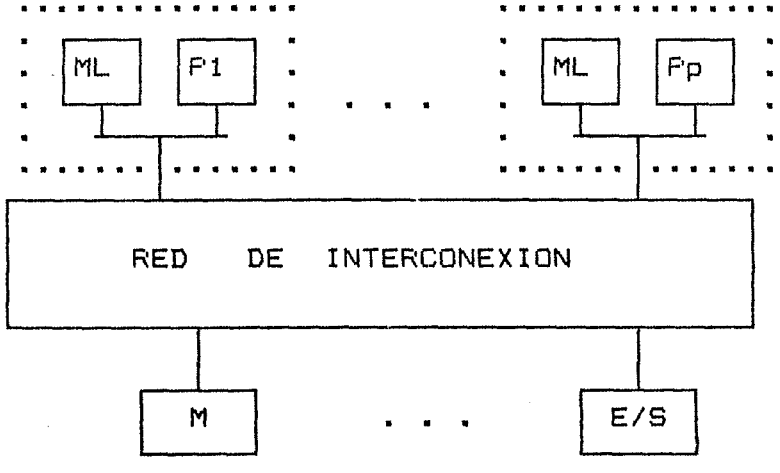


Fig 1.2d Memorias de doble entrada.

diferenciarse en otras dimensiones, tales como accesibilidad de Entrada/Salida (conectividad de los dispositivos de entrada-salida a los procesadores) rendimiento o fiabilidad. Los sistemas simétricos usualmente puede facilitar una mejor recuperación en caso de fallo. En general, un sistema homogéneo es más fácil para programar y elimina problemas al conectar, estos problemas surgen con procesadores diferentes para una comunicación efectiva. En un sistema heterogéneo todos los procesos no pueden ser ejecutados en todos los procesadores, por tanto, no permiten una fácil degradación y existe una pérdida de capacidad de reconfiguración.

1.3 Características de los sistemas multiprocesadores.

Un objetivo de los sistemas multiprocesadores es el de aumentar la velocidad de cálculo en los computadores que ejecutan los algoritmos de forma paralela. Tambien existe el requerimiento de aumentar la velocidad cuando son diseñados sistemas en tiempo real en los que el tiempo de ejecución es un factor crítico.Si la velocidad de un único procesador no satisface estos requerimientos, es necesaria una

solucción multiprocesador.

Las ventajas de mayor velocidad y potencia del multiprocesador, sobre el ordenador monoprocesador, pueden ser explotadas si se cumplen dos condiciones básicas:

a.- El problema a resolver es descompuesto de acuerdo a la naturaleza distribuida del multiprocesador, para aprovechar el paralelismo del sistema.

b.- El tiempo extra empleado en el sistema (overhead) debido a la cooperación de los procesadores es bajo.

La fiabialidad (tolerancia a fallos) puede ser definida segun Paker (Pak-83), como la capacidad para funcionar satisfactoriamente en presencia de fallos. Los sistemas multiprocesadores aumentan la fiabilidad, debido a la duplicación del hardware, el fallo de un procesador no es crucial, puesto que existen otros para realizar el trabajo, aunque exista una perdida de rendimiento, el sistema puede continuar funcionando.

Una de las ventajas de un sistema multiprocesador es su capacidad potencial para recuperar errores. Si un procesador falla, deberia ser posible, que otro

procesador pueda recuperar el estado del proceso interrumpido, para que la ejecución del proceso pueda continuar. Sin esta característica, la fiabilidad potencial queda sustancialmente reducida.

Das y Bhuyan indican en (Das-85), que la fiabilidad de los multiprocesadores se incrementa si una tarea puede ser ejecutada con menos procesadores y memorias que los de la configuración incial del sistema, puesto que existe una mayor probabilidad de que el sistema pueda recuperar con éxito cuando existe un fallo.

El número de interconexiones y la multiplicidad de unidades de procesamiento y otros módulos del sistema pueden y de hecho causan una mayor probabilidad de fallos en un multiprocesador que en un uniprocesador. Sin embargo, la redundancia inherente en un sistema multiprocesador incrementa probablemente su disponibilidad a ser tolerante a fallos.

La modularidad ha sido citada como una de las principales ventajas del sistema multimicroprocesador. La estructura Hardware hace posible añadir más módulos, para incrementar la carga de procesamiento o los requerimientos de cambio de la aplicación. La

modularidad está impuesta considerando la necesidad de adaptar el sistema propuesto a requerimientos del usuario, tales como potencia de procesamiento, restricciones en el tiempo de respuesta en situaciones en tiempo real, fácil adaptación del sistema en el caso de degradación por fallo de alguno de sus componentes, la inclusión de nuevas tareas, diferentes grados de cooperación entre tareas...

En los sistemas monoprocesadores la mayoría de los recursos están desocupados durante un alto porcentaje de tiempo. Una de las ventajas de compartir los recursos en un sistema multiprocesadores, es que permite la optimización en el uso de estos recursos.

1.4 Redes de Interconexión

En la arquitectura multiprocesador, mostrada en la figura 1.1b, la red de interconexión es la responsable de la conexión de los procesadores a los módulos de memoria compartida. La memoria compartida está dividida en módulos independientes, tales que cada uno de estos módulos permite un acceso por ciclo, es decir, asumimos que todos los módulos de memoria común son memorias uni-port, de forma que sólo un procesador puede acceder

a un módulo en un tiempo dado. Si dos o más procesadores intentan acceder al mismo módulo de memoria se produce un conflicto que debe ser resuelto o arbitrado por la red de interconexión. Para evitar excesivos conflictos, el número de módulos de memoria, es usualmente mayor que el número de procesadores . Otro método usado para reducir el conflictos es asociar una memoria local con cada procesador. . El rendimiento de un multiprocesador depende principalmente de su red de interconexión.

La interconexión de procesadores, memorias y periféricos distingue las diferentes realizaciones de un sistema multiprocesador.

1.5 Clasificación de las redes de interconexión

Una red de interconexión puede ser clasificada segun Laxmi (Lax-87), por las características operacionales de tiempo, conmutación y control.

Los tiempos de control de una red de interconexión pueden ser **síncronos** o **asíncronos.** Los sistemas síncronos están caracterizados porque existe un reloj global que emite una señal de reloj a todos los

dispositivos en la red de interconexión para que funcionen en modo síncrono. Los sistemas asíncronos, soportan operaciones independientes de los dipositivos sin un reloj global.

Una red de interconexión transfiere datos mediante **conmutación de circuitos, conmutación de paquetes o conmutación de mensajes.** En conmutación de circuitos, una vez un dispositivo tiene concedido un camino en la red de interconexión, ocupará el camino durante la transferencia de un dato. En conmutación de paquetes, la información es fragmentada en pequeños paquetes de tamaño fijo que compiten individualmente por un camino en la red. La conmutación de mensajes, es similar a la de paquetes, con la diferencia de que los mensajes tiene una longitud variable.

Basandonos en el esquema de control utilizado en la red, una red de interconexión puede ser clasificada como **centralizada o descentralizada.** Cuando el control es centralizado, un controlador central recibe todas las peticiones y transmite los mensajes en la red. En un sistema descentralizado, las peticiones son manejadas independientemente por los diferentes dispositivos en la red.

Estas tres características operacionales junto con la topología definen una red de interconexión.

1.6 Topologías de las redes de interconexión.

La principal característica de un sistema multiprocesador es la capacidad de cada procesador para compartir un conjunto de módulos de memoria principal y dispositivos de Entrada/Salida. Esta capacidad para compartir está proporcionada por la red de inrterconexión. Existen diferentes formas fisicas para las redes de interconexión. Discutiremos en esta sección algunas organizaciones básicas de redes de interconexión

1.6.1 Bus único.

La interconexión más simple para sistemas multiprocesadores es un camino de comunicación común conectando todas las unidades funcionales. Un ejemplo de un sistema multiprocesador utilizando el Bus único es mostrado en la figura 1.3.

La red de interconexión es a menudo una unidad

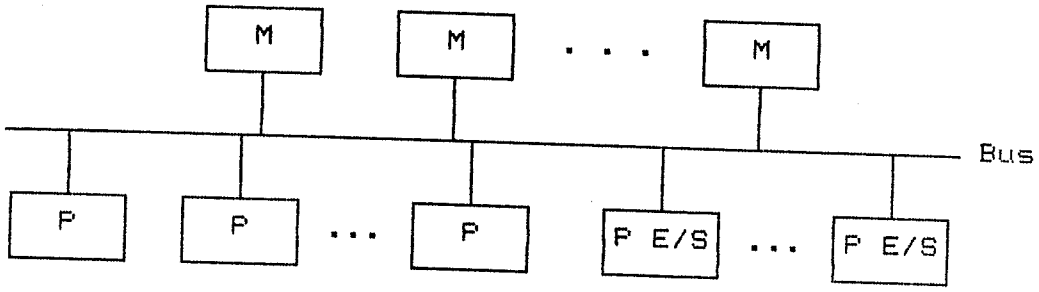


Figura 1.3 Bus único

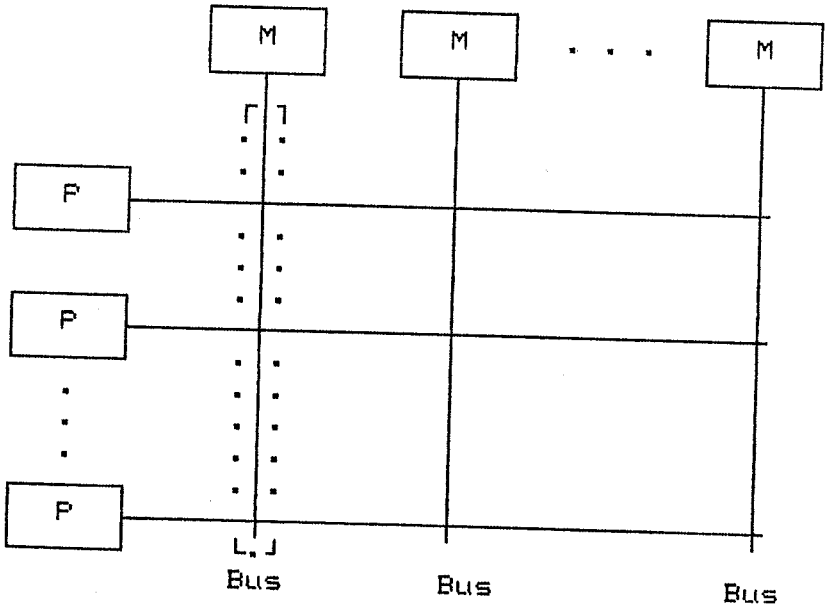


Figura 1.4 Crossbar

totalmente pasiva. Las operaciones de transferencia están controladas por las unidades emisoras y receptoras. Puesto que el bus es un recurso compartido, debe existir un mecanismo para resolver la contención. Los métodos de resolución de conflictos incluyen prioridades fijas o estáticas. Un árbitro o controlador de bus centralizado simplifica la resolución de conflictos, pero puede tener efectos negativos en la fiabilidad y flexibilidad del sistema.

Esta organización es la menos compleja y tiene un coste relativamente bajo. Es fácil modificar fisicamente la configuración hardware del sistema añadiendo o quitando unidades funcionales.

El bus es un componente crítico en el sistema que puede causar un fallo catastrófico del sistema como resultado de un malfuncionamiento de cualquier circuito de interconexión al bus. Además la expansión del sistema, añadiendo más procesadores o memoria, incrementa la contención del bus, lo cual degrada la productividad del sistema e incrementa la lógica de arbitraje. La velocidad total de transferencia dentro del sistema, está limitada por el ancho de banda y velocidad de este único camino, por esa razón memorias y dispositivos de entrada/salida privados son muy

ventajosos.

Esta organización es usualmente apropiada sólo para sistemas pequeños y con poca comunicación entre procesadores.

Existen varios ejemplos de sistemas que utilizan este tipo de organización, el PDP-11 Unibus (Hwa-84), los multiprocesadores Minerva (Wid-76), entre otros y los actuales sistemas basados sobre buses estandard (Bor-81), tales como el VMEbus (Pri-86), Unibus II (Bor-85), etc.

1.6.2 Crossbar

El Crossbar es una organización de buses compartidos en la que existe un bus separado asociado con cada módulo de memoria, estando cada procesador conectado a todos los buses. La topología de esta red está representada en la figura 1.4.

Las importantes características de un sistema utilizando una matriz crossbar de interconexión son la simplicidad de los circuitos de interconexión para la conmutación de unidades funcionales y la capacidad para

soportar simultaneamente transferencias a todos los módulos de memoria. Para proporcionar estas características requiere mayor capacidades harware en la conmutación. Cada punto de cruce, no sólo debe ser capaz de la conmutación para permitir transmisiones paralelas, sino que además debe ser capaz, cuando llegan en un único ciclo múltiples peticiones para el acceso al mismo módulo de memoria, de decidir que procesador debe acceder al módulo de memoria.

Teoricamente, la expansión del sistema está limitada sólo por el tamaño de la matriz de conmutación. Añadir un nuevo procesador implica añadir las conexiones a todos los módulos de memoria, suponiendo que los árbitros estaban preparados. Añadir un módulo de memoria, significará añadir las conexiones a todos los procesadores y un árbitro. Existe la posibilidad de diseñar las matrices crossbar para una mayor capacidad que la inicialmente requerida y equipando sólo para los requerimientos presentes. Así se facilitará la expansión.

En el crossbar solamente existe un camino que puede conectar a cada par de unidades funcionales, por lo tanto, el fallo de un punto de conmutación no deja un camino alternativo a las correspondientes unidades.

El coste depende fundamentalmente del número de puntos de cruce del sistema, para un sistema con muchos procesadores, el coste de esta red de interconexión es muy alto.

Ejemplos de sistemas que utilizan esta red son el C.mmp (Wul-72) y el S-1 (Hwa-84).

1.6.3 Memorias multientrada

En esta organización el control, conmutación y lógica de arbitraje están distribuidos en los módulos de memoria, como el ejemplo mostrado en la figura 1.5a.

Cada procesador tiene acceso a través de un bus a todos los módulos de memoria. El método más utilizado para resolver los conflictos de accesos a las memorias es el de asignar permanentemente prioridades prefijadas a cada entrada de la memoria. Excepto por la prioridad asociada con cada una, todas las entradas son idénticas.

En la configuración del sistema es posible designar partes de la memoria como memoria privada para ciertos procesadores como se muestra en la figura 1.5b. Este tipo de organización del sistema tiene la ventaja

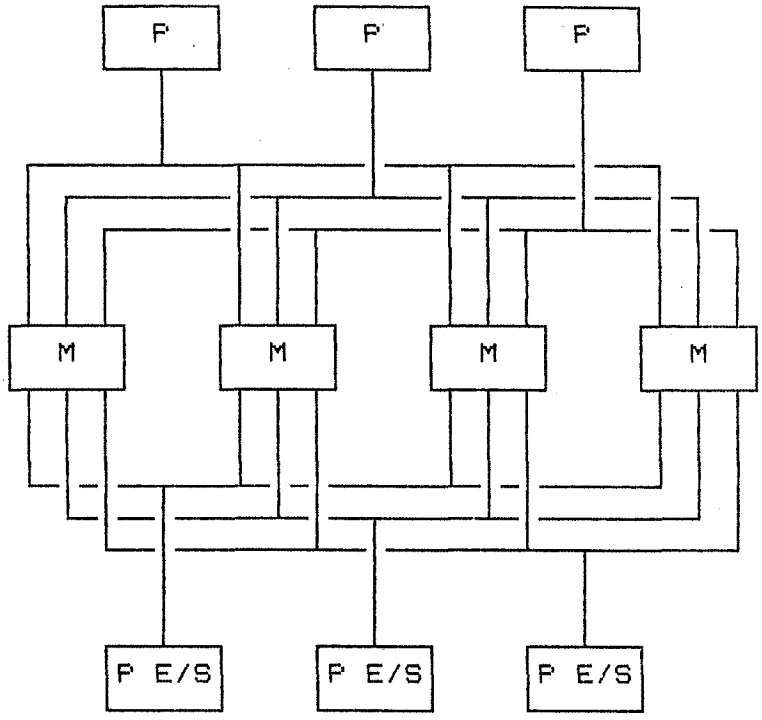


Figura 1.5a Memorias multientrada.

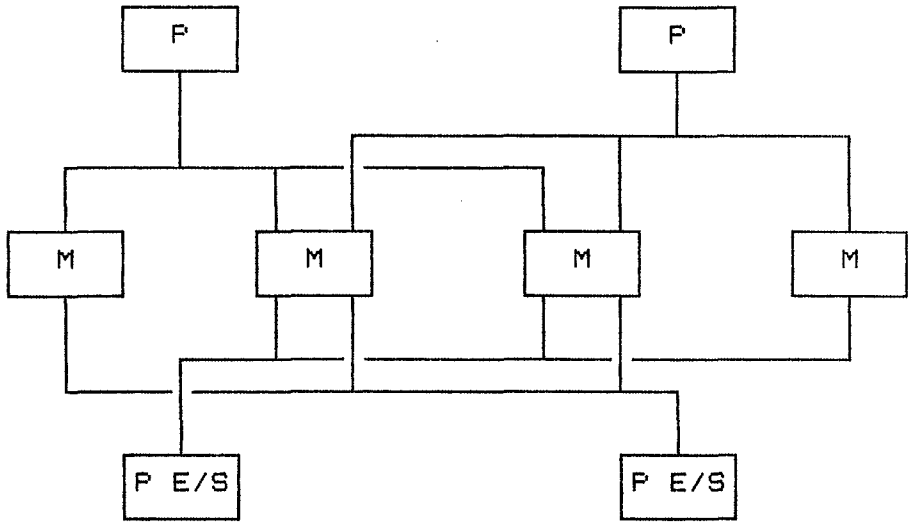


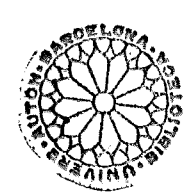
Fig 1.5b Organización memorias multientrada con memorias privadas.

de incrementar la protección contra accesos no autorizados y permite almacenar rutinas en áreas de memoria que no pueden ser modificadas por otros procesadores; sin embargo, existe la desventaja en el sistema de recuperación, cuando un procesador falla, si otros procesadorees no son capaces de acceder a la información del estado y control en un bloque de memoria asociado con el procesador que falla.

La anchura y posibles opciones de configuración están limitadas por el número y tipos de entradas de memoria disponibles, esta decisión se realiza en el proceso de diseño y es difícil modificarla. Además este sistema requiere un gran número de cables y conectores.

Un ejemplo de sistemas con memorias multientrada son el IBM sistema 370/168 (Sat-80) y el Univac 1100/90 (Hwa-84).

Es muy difícil justificar el uso de organizaciiones crossbar o memorias multientrada para grandes sistemas multiprocesadores.



1.6.4 Redes multietapa

Estas redes estan formadas por diferentes etapas, cada una de ellas está constituída por elementos básicos de conmutación, siendo necesario el paso a través de diferentes elememtos, dependiendo de la red, para que se establezca la conexión procesador-memoria.

Una red de interconexión multietapa M*N conecta M procesadores a N módulos de memoria. La red puede conectar cualquier $M = m_1, m_2, \ldots, m_r$ a $N = n_1, n_2, \ldots, n_r$ salidas a través de r etapas de conmutadores. El i-ésimo estado emplea $m_i * n_i$ conmutadores crossbar y está precedida por una interconexión barajada generalizada que es esencialmente una clase de las interconexiones Delta (Pat-79) (Pat-81) y Omega (Law-75). Esta es la versión más generalizada de una MIN, en la que se permite diferentes tamaños de entradas y salidas (Lax-87).

Una red Delta (Pat-81), mostrada en la figura 1.6 puede conectar $M = a^n$ entradas a $N = b^n$ salidas a través de n etapas de a*b conmutadores crossbar. El modelo de enlace entre etapas es tal que existe un único camino de longitud constante de un fuente a cualquier destino. El camino es controlado por dígitos

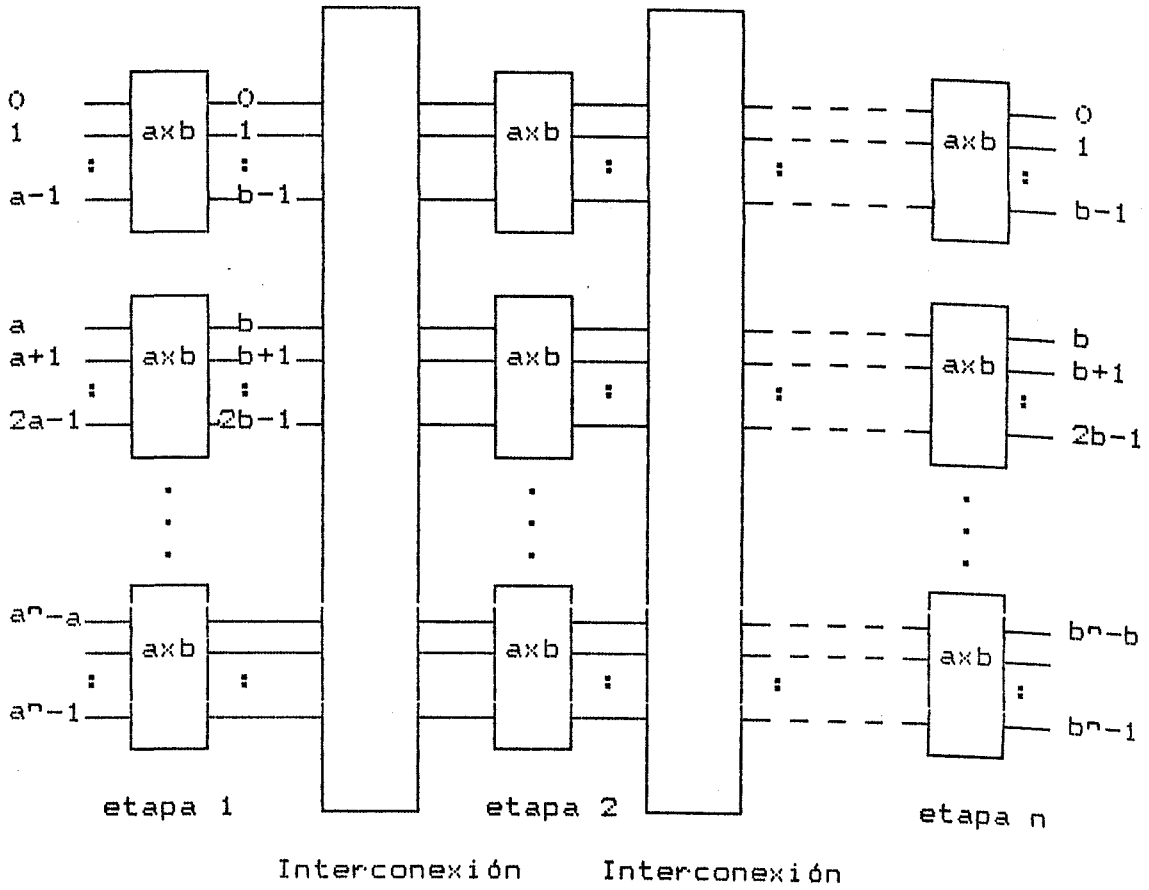


Figura 1.6 Red Delta.

tal que un módulo crossbar conecta una entrada a cualquiera de sus b salidas dependiendo de un único dígito base-b tomado de la dirección destino. En una red delta, ningún terminal de entrada o salida de cualquier módulo crossbar está inconectado por la izquierda.

Una red Omega (Law-75) mostrada en la figura 1.7, es una subclase de redes Delta con a=b=2, está caracterizada por una perfecta interconexión barajada precediendo cada estado de conmutadores. Los procesadores peticionarios generan una etiqueta (tag) que es la representación binaria del destino. La conexión de un conmutador en el i-ésimo etapa es efectuada por el i-ésimo bit de esta etiqueta binaria a partir del bit más significativo. Esta auto-ruta propia de una MIN, evita la necesidad de un controlador central.

En estas redes existe un único camino de una entrada a una salida, si existe un fallo, existirán módulos de memoria que no podrán ser accedidos por algunos procesadores. Es necesario incorporar alguna tolerancia a fallos en estas redes para que al menos un único fallo en un conmutador o un enlace pueda ser

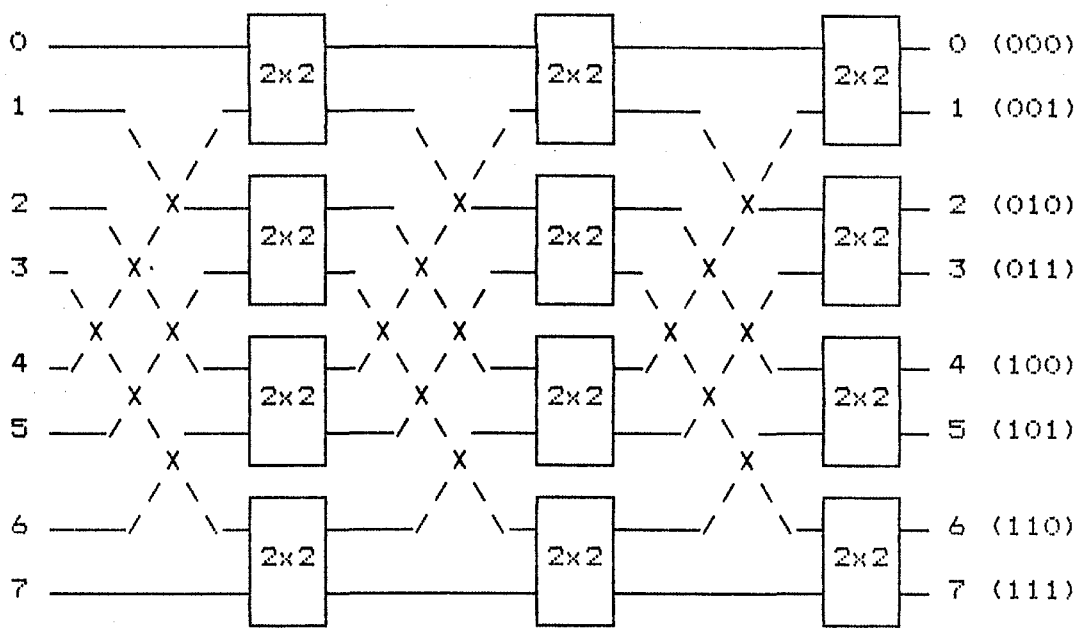


Fig. 1.7 Red Omega 8x8.

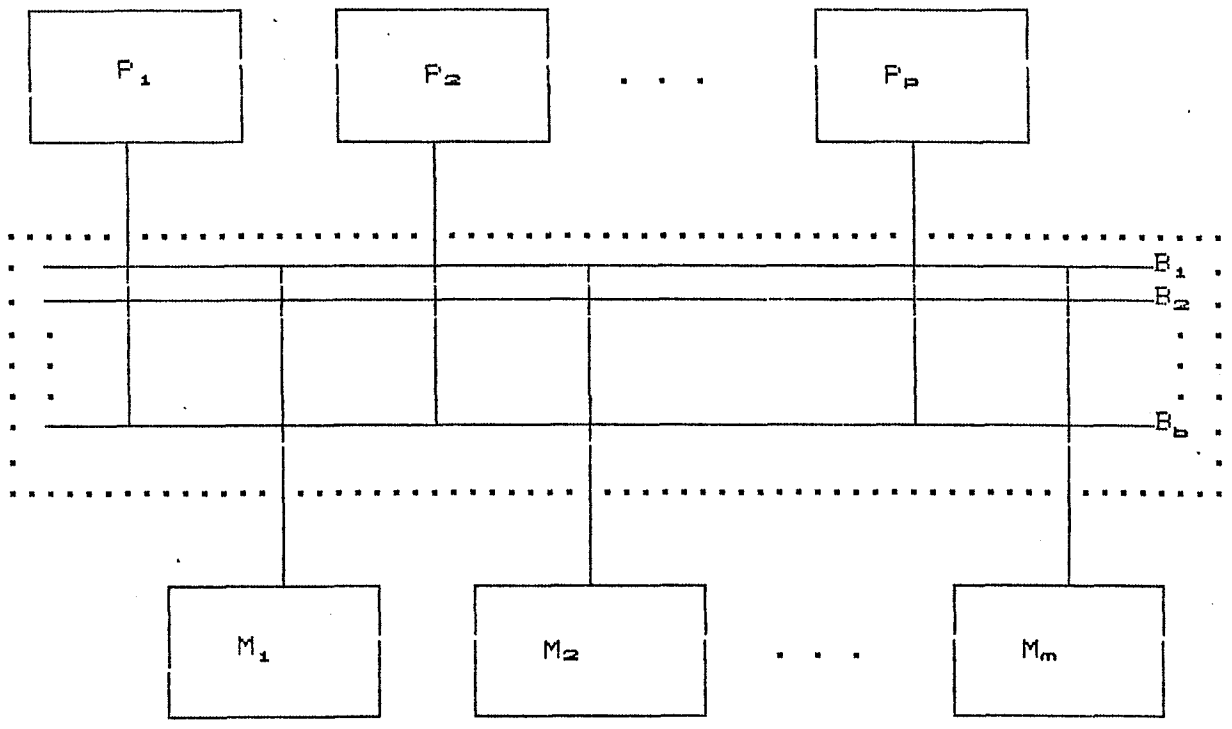
tolerado. Además la inclusión de nuevos procesadores o módulos de memoria, requieren un rediseño de la nueva red.

1.6.5 Buses Múltiples

En la red de conexión de buses múltiples mostrada en la figura 1.8, los P Procesadores y los M Módulos de Memoria, están conectados a los B Buses (B <= min (P,M)). Cada procesador está conectado a todos los buses y cada bus a todos los módulos de memoria, así un procesador puede acceder a un módulo de memoria via un bus en tiempo compartido, si el módulo direccionado está libre y un bus disponible.

El sistema de interconexión no puede ser una unidad totalmente pasiva, cada bus necesitará de un dispositivo que controle cada uno de los puntos de conexión de las unidades al bus. Este circuito debe ser modular y rápido al ejecutar el algoritmo de asignación de buses.

La asignación de buses debe cumplir, para aprovechar el ancho de banda, las siguientes condiciones (Lan-82):



P = Procesadores

M = Módulos de Memoria común

B = Buses

p = número de procesadores

m = número de módulos

b = número de buses

Figura 1.8 Multiprocesador con una estructura de buses múltiples.

1.- Sólo un bus puede ser asignado al conjunto de procesadores que piden acceso al mismo módulo.

2.- Sólo un bus puede ser asignado a un procesador.

3.- La política de asignación debe ser imparcial.

La red de interconexión de buses multiples tiene las siguientes características (Mud-85):

- Es compatible con la filosofía predominante que siguen la mayoría de las familias de componentes de los microprocesadores de utilizar un "Bus Centralizado".

- El sistema de buses múltiples es modular, permitiendo incrementar facilmente el número de elementos de procesamiento, el número de módulos de memoria y el número de buses.

- Finalmente, el sistema de buses múltiples es tolerante a fallos, si un bus falla el sistema trabajará aunque con un rendimiento menor, en lo que calificariamos de modo degradado.

El sistema de buses múltiples con B buses permite como máximo B conexiones procesador-memoria por ciclo. El sistema queda saturado cuando se generan más de B peticiones y permite sólo B conexiones

simultaneamente. La degradación que introduce esta red depende, del número de procesadres (P), módulos de memoria (M) y buses (B), así como de la distrribución de peticiones de los procesadores.

La fiabilidad de estas redes es buena, dado que aún cuando fallen B-1 buses, el sistema puede seguir funcionando.

En un sistema de buses múltiples si B = min(P,M), tenemos un crossbar, y si B = 1 tenemos un bus único.

El coste y rendimiento de las redes MIN y buses múltiples tiene un balance rezonable entre los de un bus compartido y un crossbar.

Diversos autores han estudiado y evaluado esta red, entre otros citaremos, (Ira-84), (Lan-82), (Val-83), (Mud-85). Sistemas que utilizan este esquema de interconexión estan descritos en (Mae-79), (Bow-80) y (Mud-87).

El número de conexiones en un sistemas de buses múltiples es B*(M+P) lo cual puede ser un número grande, en (Lan-83), se demuestra que esta conexión es redundante y se introducen los esquemas reducidos que

producen el mismo rendimiento con un coste menor. La reducción es especialmente significativa cuando el número de buses es grande. por ejemplo el número de conexiones requerido es reducido aproximadamente el 25%, cuando el número de buses es la mitad del número de procesadores (para M=P). En (Lan-83) tambien se proponen diferentes esquemas con el mínimo número de conexiones.

1.7 Requerimientos software

Es necesario un software básico para realizar las tareas de cooperación entre todos los procesadores, tales como, tareas de comunicación, sincronización, políticas de asignación,...

Existen tres organizaciones básicas que han sido utilizadas en el diseño de sistemas operativos para multiprocesadores, llamadas configuración Master-Slave, Supervisor separado para cada procesador y Control en modo flotante (Dei-84).

En modo Master-Slave el supervisor siempre se ejecuta en el mismo procesador y es el más simple de realizar. El procesador llamado master, mantiene el

estado de todos los procesadores en el sistema y reparte el trabajo a los otros procesadores. Un procesador "slave" sólo puede ejecutar programas de usuario. Como el supervisor es siempre ejecutado en el mismo procesador, si otro procesador necesita una operación que debe ser ejecutada por el supervisor, debe enviar al master una petición, via un trap o una llamada al supervisor y esperar hasta que el programa actual en el procesador master sea interrumpido, cuando el master reconoce la petición ejecuta la operación apropiada. El supervisor y sus procedimientos no necesitan ser reentrantes, puesto que puede ser ejecutado sólo por un procesador y sólo a favor de un usuario en un momento dado.

Otra característica es que se simplifican los conflictos en tablas y problemas de "lock-out" para el control de tablas. Sin embargo, este modo de operación puede causar un fallo catatrófico cuando el procesador designado como master tiene un fallo o un error irrecuperable. Además, la utilización de los otros procesadores puede llegar a ser apreciablemente baja si el master no puede atender los procesos bastante rápido para mantener a los demas procesadores ocupados.

Este tipo de sistama operativo es más efectivo

para aplicaciones de propósito especial, donde la carga de trabajo está definida o para sistemas asimétricos en los que el procesador master tiene más capacidad que el resto de los procesadores.

En la organización de **Supervisor** separado cada procesador tiene su propio sistema operativo y responde a las interrupciones de los usuarios ejecutando en dicho procesador, por lo tanto cada procesador sirve sus propias necesidades. Puesto que existe interacción entre los procesadores, es necesario, para una parte del código supervisor, ser reentrante o replicada para proporcionar copias separadas para cada procesador. Aunque cada procesador tiene su propio conjunto de tablas privadas, algunas tablas son comunes y compartidas por todo el sistema, esto crea problemas de acceso a tablas, estando el acceso a estas tablas cuidadosamente controlado usando técnicas de exclusión mútua. El método usado para acceder a los recursos compartidos, depende del grado de acoplamiento entre los procesadores.

Este sistema operativo, no es tan sensible a un fallo catastrófico como un sistema Master-Slave. Tambien, cada procesador tiene su propio conjunto de dispositivos de entrada/salida y archivos, y cualquier

reconfiguraciónn de entrada/salida, requiere usualmente intervención manual. La réplica del supervisor en los procesadores puede requerir mucha memoria que puede ser poco utilizada. En esta organización los procesadores no cooperan en la ejecución de un proceso individual, un proceso corre hasta terminar en el procesador al que ha sido asignado, siendo posible que algún procesador quede desocupado, mientras otro procesador ejecuta un proceso lentamente.

. En el supervisor en modo de **Control flotante** o multiprocsamiento simétrico, cada procesador puede por si mismo, tener acceso al sistema operativo para ejecutar aquella parte del mismo que precise. Las rutinas del supervisor "flotan" de un procesador a otro, si bien algunos de los procesadores pueden estar ejecutando rutinas del supervisor simultaneamente. Este tipo de sistema puede obtener mejor equilibrio sobre todos los tipos de recursos. Los conflictos en peticiones de servicio son resueltos por prioridades que son activadas estaticamente o bajo control dinámico. La mayoría del código del supervisor debe ser reentrante, puesto que varios procesadores pueden ejecutar la misma rutina al mismo tiempo, tambien son necesarias técnicas de exclusión mútua. En este sistema los conflictos de acceso a tablas y retardos de

"Lock-out" de tablas no pueden ser evitados y es importante controlar estos accesos de modo que la integridad del sistema esté protegida.

1.7.1 Procesos. Caracterización de un proceso

Un proceso es un programa secuencial independiente, con sus propios datos. Un proceso consta de dos partes: El programa (un código secuencial que no cambia y las tablas fijas) y el vector de estado (la parte que va cambiando a lo largo del tiempo) (Tan-76).

Cuando un proceso es suspendido por cualquier razón, debe ser guardada la información sobre su estado, para permitir que sea recomenzado en el mismo estado en el que fué parado.

Un proceso opera normalmente con sus propios datos privados, no puede operar en datos pertenecientes a otros procesos y además ningún otro proceso puede tener acceso a sus datos privados. Para capacitar estructuras de datos compartidos se definen estructuras de comunicación entre procesos.

Un proceso concurrente interactúa de alguna manera

con otros procesos. Los procesos son los principales elementos activos de un software concurrente.

1.7.2 Comunicación entre procesos

Los procesos necesitan comunicarse con otros procesos por varias razones. Una de las más importantes es para compartir información. Otra razón de la comunicación entre procesos es la necesidad de permitir a los procesos sincronizar sus actividades.

Cuatro problemas básicos son identificados para realizar ordenadamente la cooperación entre los procesos: exclusión mutua, sincronización, deadlock y determinación (Hwa-84).

- **Exclusión Mutua:** Es la necesidad de forzar el uso secuencial estricto de un recurso por los procesos que compiten hasta que la tarea termine.

- **Sincronización:** Los procesos cooperando pueden requerir alguna forma de sincronización para permitir coordinar una secuencia de operaciones en recursos compartidos.

- **Deadlock:** Este término describe una situación que ocurrre cuando los procesos estan esperando mutuamente acontecimientos causados por otros procesos.

- **Determinación:** Este problema se refiere a la

posibilidad de que procesos que pueden acceder concurrentemente a datos comunes, puedan alterar estos datos por una secuencia entremezclada de operaciones, de forma que esto los pueda invalidar. Son necesarias precauciones para asegurar que las alteraciones de datos comunes están coordinadas.

1.7.3 Regiones Críticas. Semáforos

Si los procesos de un ordenador comparten recursos o cooperan en tareas comunes, deben poder compartir los datos de un modo conrolado. Dijkstra (Dij-68), estudiando este problema llegó a la conclusión de que era esencial, que se ejecute estrictamente una operación en cada momento sobre las variables comunes, entonces la máquina debe retardar futuras operaciones sobre las mismas variables, hasta que la operación presente haya terminado. Dijkstra (Dij-68) tambien introduce el nombre de "Región Crítica" para las operaciones en variables comunes que sólo pueden efectuarse una cada vez.

Una **Región Crítica** es una parte de código en un sistema concurrente que está controlada para que sólo un proceso pueda entrar en cada momento. El concepto de región crítica es la clave para la interacción ordenada

de procesos concurrentes. Los procesos estan obligados a entrar a estas regiones críticas de acuerdo a alguna disciplina preestablecida. La entrada y salida de estas regiones críticas están controladas por un "Semáforo".

Las operaciones de manipulación de un semáforo deben ser indivisibles. Una operación indivisible (formada por una secuencia de código), no puede ser interrumpida por ninguna característica, hardware o software, del sistema.

Un **Semáforo** es una construcción lógica que puede ser usada para controlar la entrada y salida de una región crítica sin espera ocupada (Busy Wait). Incluye las colas y procedimientos apropiados para suspender un proceso (Bow-80).

En su forma más simple, un semáforo está compuesto de una cola y una variable entera no negativa.

El semáforo es usualmente inicializado a 1 y llevado a cero cuando la región crítica está ocupada. Cuando la variable puede tomar valores 0 ó 1, es llamado "semáforo binario", actúa como un bit de cierre (lock), permitiendo sólo la entrada de un proceso, durante un intervalo de tiempo, en la región crítica

asociada. Si el semáforo toma cualquier valor entero, es llamado "semáforo contador". El contador puede ser tambien usado para recordar el número de procesos que han sido suspendidos (y por lo tanto deben ser recomenzados), permitiendole tomar valores negativos.

Dijkstra introdujo las dos operaciones P (wait) y V (signal) para operar sobre un semáforo (Dij-68). Las operaciones Wait y Signal son operaciones indivisibles llamadas primitivas de sincronización.

La primitiva P (wait) actúa para adquirir permiso para entrar. La primitiva V (signal) indica la terminación de una región crítica.

Las reglas de sincronización de semáforos son las siguientes (Ben-82):

1.- Si la operación P, cuya función es la de decrementar el valor de su semáforo, es comenzada cuando el semáforo tiene un valor positivo, el proceso continúa; pero si un proceso va a decrementar un semáforo con valor cero, es bloqueado y llevado a una cola de procesos asociada con el semáforo.

2.- La operación V incrementa el valor del semáforo en uno; si uno o más procesos están esperando

en la cola asociada con el semáforo, entonces uno de ellos es seleccionado y capacitado para continuar (signal), los otros procesos quedan bloqueados. El método para escoger uno de los procesos esperando debe ser imparcial.

1.7.4 Monitores

Un Monitor es una región crítica generalizada en la cual la comunicación estre procesos y la asignación ordenada es efectuada utilizando condiciones variables.

Un **monitor** define una estructura de datos compartida y todas las operaciones que pueden ser ejecutadas en él (procedimientos). Consta de los datos locales y uno o más procedimientos que los procesos pueden ejecutar (Bri-73).

Un monitor es una entidad estática y puede ser activada por procesos, llamando a los procedimientos del monitor. Un monitor tambien incluye un procedimiento de inicialización (operación inicial), que es ejecutada una vez cuando la estructura de datos es creada.

Un monitor puede ser usado como una esrtructura de

sincronización para procesos concurrentes y transmitir datos entre ellos. Los procesos que no pueden operar directamente sobre datos compartidos, pueden llamar a los procedimientos del monitor que realizan los accesos a los datos compartidos. Para asegurar la consistencia, la estructura del monitor fuerza accesos exclusivos ejecutando durante un intervalo de tiempo un procedimiento. Si otra llamada llega mientras un procedimiento está siendo ejecutado, el proceso que efectúa la llamada es retardado.

1.8 Compartición de recursos. Arbitros.

Cuando en un sistema digital existen muchos procesadores operando independientes y que comparten el uso de recursos comunes, es necesario un circuito para arbitrar su uso y resolver los conflictos que pueden ocurrir. Una solución válida para este problema es diseñar un mecanismo de control (árbitro de P-usuarios), que cumpla las siguientes condiciones generales (Cal-86):

- 1.- Sólo un procesador puede usar un recurso compartido en un instante de tiempo.

2.-Cualquier petición para usar un recurso común debe ser servida en un tiempo finito.

Un árbitro es un dispositivo dedicado a secuenciar peticiones simultáneas a un recurso común.

Un árbitro no es un circuito secuencial operando en modo fundamental , debido al caracter asíncrono de sus entradas variables (señales de petición), que pueden cambiar en momentos arbitrarios, independientes unas de otras, en los que el árbitro no está en un estado estable, y entonces es necesario describir las operaciones propias del circuito bajo los supuestos de cambios en las entradas no restringidos. No puede ser un circuito secuencial síncrono, porque el valor de una variable de entrada puede cambiar cuando el pulso de reloj está presente y por lo tanto hay una posibilidad estática de errores lógicos que surjan de los elementos de memoria del árbitro, entrando en sus estados metaestables (Len-78) (Pea-75).

La organización lógica de un árbitro es mostrada en la figura 1.9. El árbitro recibe las peticiones al recurso por parte de los usuarios aspirantes, y asigna el recurso a uno de ellos de acuerdo a una política predefinida.

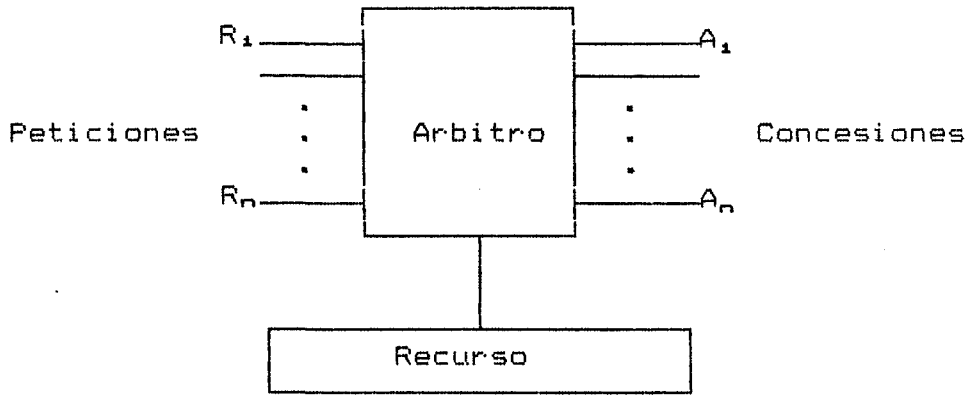
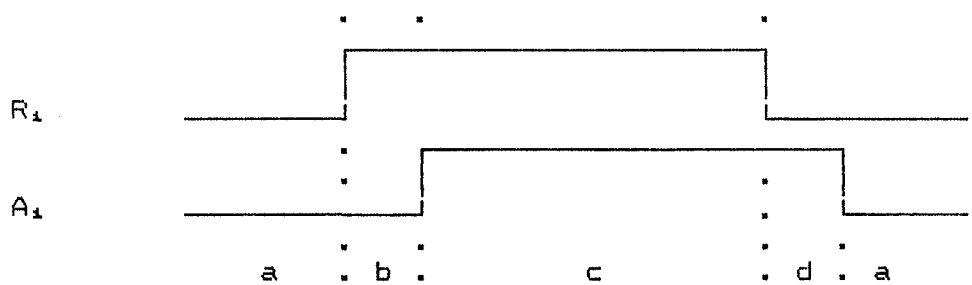


Figura 1.9 Organización típica de un árbitro.



a: Inactivo.
b: Petición y no concesión.
c: Petición concedida. Utilización del recurso.
d: Liberación.
R_i Petición del procesador i al árbitro.
A_i Concesión del árbitro al procesador i.

Fig 1.10 Convenio de señales para árbitros.

La comunicación entre cada procesador P_i, i=1,...,n y el árbitro se establece por medio de dos líneas:

- la línea de petición R_i y
- la línea de reconocimiento A_i.

el procesador P_i comunica que quiere usar el recurso común y envía una señal de petición conmutando su línea R_i a 1 (activando R_i). El árbitro, de acuerdo a su esquema de prioridad, dará permiso para usar el recurso común al procesador P_i con una señal de reconocimiento, conmutando su línea A_i a 1, que le indica la concesión del recurso. Cuando el procesador P_i ha terminado de usar el recurso común, hace un "reset" en su línea de petición (R_i=0) y esto causa que el árbitro haga un reset en la correspondiente línea de reconocimiento (A_i=0). Al procesador no se le permite hacer una segunda petición para usar el recurso común hasta que el árbitro ha restaurado A_i=0. Esto es, adoptamos la convención de señales mostrada en la figura 1.10 y de acuerdo con la condición:

1) $A_i.A_j=0$ Para todo i,j i<>j.

Por lo tanto, en un instante determinado sólo una señal de concesión puede estar activa.

1.8.1 Clasificación

La estructura lógica mostrada en la figura 1.9 puede ser implementada de diferentes modos.

Los árbitros pueden ser clasificados dependiendo de la distribución de la función de arbitraje entre los procesadores. Puede ser **centralizado** en una única unidad o **distribuído** a través del sistema (Thu-72). Aunque se distinga entre realizaciones centralizadas y distribuídas, se debe tener en cuenta que el árbitro es una única unidad lógica asociada con el recurso a ser arbitrado o compartido (Civ-8).

Otra nueva clasificación, que identifica las diferentes estructuras de arbitraje, esta basada en la política de arbitraje adoptada. El nivel de prioridad asignado a cualquier petición puede ser fijo o variable, de acuerdo a diferentes reglas. **Prioridad fija, rotante** y **algorítmica** (Cio-83) pueden ser usadas para decir que petición debe ser la primera en obtener el acceso al recurso común en presencia de peticiones simultáneas.

Cuando la prioridad es fija, si las peticiones de mayor prioridad estan continuamente llegando, pueden

dejar fuera de servicio a los petionarios de baja prioridad. Algunas técnicas han sido desarrolladas para evitar esto y hacer al árbitro "imparcial", estos métodos estan basados en cambios dinámicos de los niveles de prioridad, o en una imparcialidad forzada.

Finalmente, los árbitros pueden ser clasificados dependiendo de la técnica empleada para recibir las peticiones y conceder accesos, pueden hacerse análisis simultáneo o un escrutinio secuencial de las peticiones, estas técnicas son llamadas **daisy chain, polling** y **peticiones independientes** (Thu-72).

1.9 Objetivos del trabajo descrito en esta memoria

Describimos un sistema multimicroprocesador con memoria común y buses múltiples que ha sido concebido teniendo en cuenta los siguientes objetivos: Estructura modular del sistema y ofrecer mecanismos que faciliten el desarrollo de un software de aplicación siendo el sistema transparente al usuario. La transparencia está definida como la capacidad para escribir progamas de aplicaciones sin considerar la configuración actual del sistema.

El sistema multimicroprocesador modular con buses

miltiples que hemos desarrollado, está organizado a partir de 3 módulos principales: Procesadores, Memoria Común y Buses. El sistema puede ser dimensionado para los requerimientos de aplicaciones específicas cambiando el número de procesadores módulos de memoria y buses.

En este trabajo se propone un sistema de arbitraje , realizado por hardware y que puede ser facilmente expandible, para resolver los siguientes tres tipos de conflictos que pueden presentarse en los sitemas multiprocesadores:

1 - Cuando diferentes procesadores intentan acceder al mismo módulo de memoria simultaneamente.

2 - Cuando un procesador intenta acceder a un módulo de memoria ya ocupado.

3 - Cuando uno o más procesadores intentan acceder a módulos de memoria y no existen buses disponibles.

El acceso simultáneo a los módulos de memoria está resuelto utilizando M árbitros del tipo P usuarios 1 servidor (P a 1). Cada uno de estos árbitros selecciona equiprobablemente uno de los procesadores que tiene petición pendiente al módulo de memoria al que está asociado dicho árbitro. La asignación de buses para

acceder a los módulos de memoria, está realizada utilizando un árbitro tipo M usuarios B servidores (M a B). Este árbitro, asigna a las peticiones para acceder a los módulos de memoria común seleccionados, los buses disponibles.

Si un procesador quiere acceder a un módulo de memoria y dicho módulo ya está ocupado, o no existen buses disponibles, tiene que esperar hasta que haya terminado la condición de conflicto. Los árbitros utilizados son "nonpreentives", por lo tanto un usuario no puede ser obligado a dejar el recurso por la llegada de uno más prioritario.

Una petición puede no ser concedida en caso de conflicto, pero si tiene concesión, la conexión se realiza sin añadir tiempo extra al ciclo.La red de interconexión no proporciona retardo adicional para afectar a los ciclos de procesador y memoria.

El sistema de arbitraje propuesto para realizar la asignación de buses puede soportar la asignación de buses tanto en sistemas de buses múltiples con un esquema de interconexión completo como en sitemas de buses múltiples con esquemas de interconexión reducidos.

Las funciones de sincronización básicas para accesos a módulos de memoria común estan soportadas en su mayoría por hardware, permitiendo accesos ciclo a ciclo a un recurso común y accesos multiciclo necesarios para realizar operaciones indivisibles.

CAPITULO 2

ESTRUCTURA DE INTERCONEXION DE BUSES MULTIPLES

2.1.- Buses Múltiples

Uno de los aspectos principales a considerar en el diseño de sistemas multiprocesadores, es la estructura de la red de interconexión de los procesadores a los módulos de memoria compartida (Figura 2.1)

Algunos de los parámetros que influyen para realizar esta elección son: tolerancia a fallos, costo, modularidad, ancho de banda y capacidad de expansión del sistema o red de interconexión.

Para los sistemas multiprocesadores, han sido propuestas algunas redes de interconexión, tales como el crossbar (Wul-72), bus único (Swa-77), buses múltiples (Bow-80) (Lan-82) (Mud-87), shuffe-exchange (Tha-81)y otras redes de interconexión especiales (Sie-79) (Wu -81).

El rendimiento del crossbar, que ha sido muy analizado, proporciona el mayor potencial de ancho de banda, porque no existen conflictos en la red, sólo

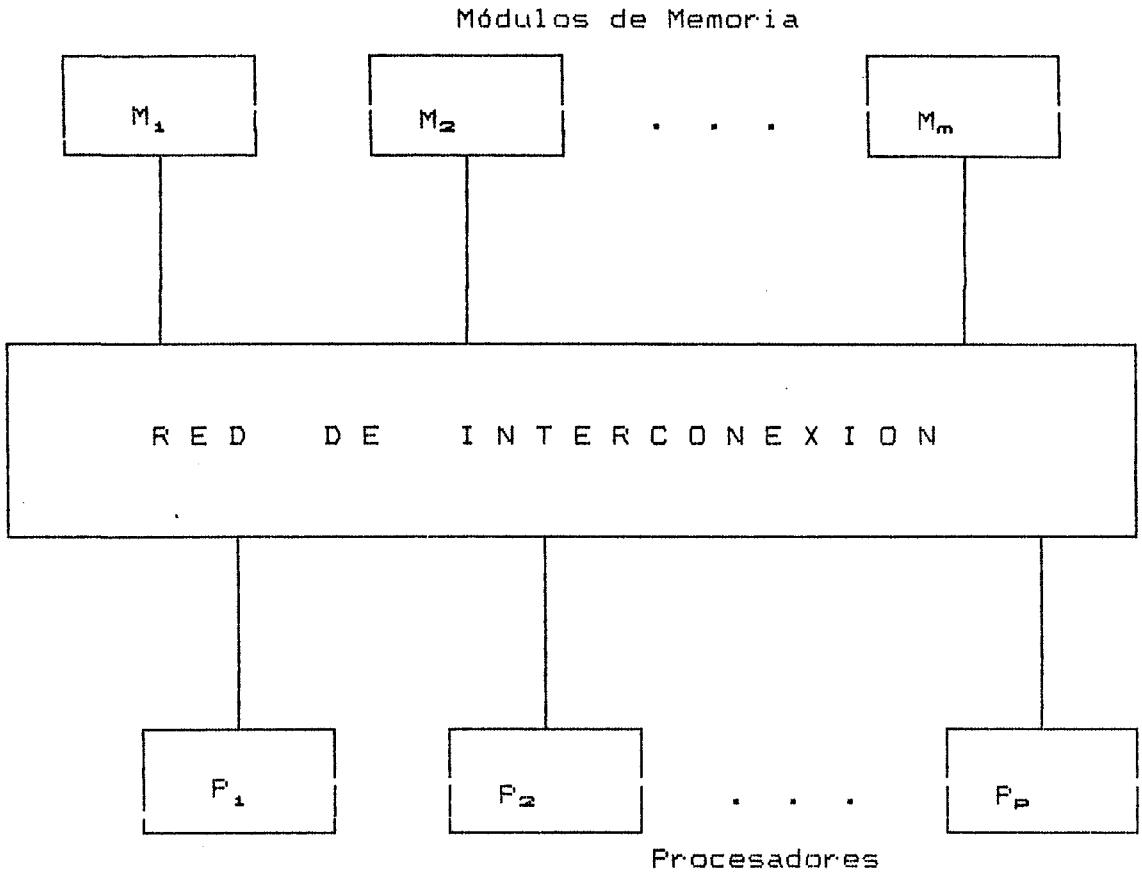


Figura 2.1 Sistema Multiprocesador

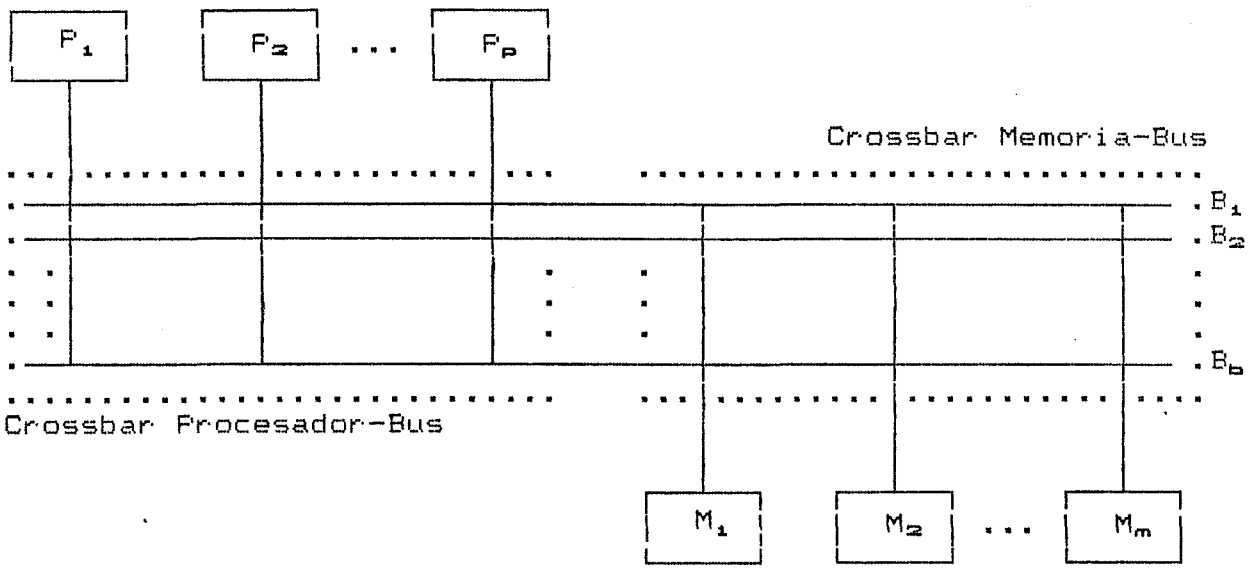
existen conflictos, en la memoria común, cuando más de un procesador desea acceder al mismo módulo, puesto que un bus está conectado a todos los procesadores pero sólo a un módulo de memoria. En (Lan-82a) concluyen que el ancho de banda que proporciona el crossbar no es totalmente utilizado, debido a los conflictos en memoria. Además, tiene un alto coste, que es prohibitivo para un gran número de procesadores y es poco tolerante a fallos, un fallo en uno de los M buses desconecta completamente un módulo de memoria y no permite utilizar dicho módulo con prestaciones reducidas, dejandolo aislado. Consecuentemente, otras redes de interconexión llegan a ser atractivas, sin proporcionar una degradación significativa, en el rendimiento del sistema.

Nosotros hemos centrado nuestra atención en la red de interconexión de buses múltiples, para un sistema multiprocesador en el que cada procesador posee memoria privada y existen ademas unos módulos de memoria compartida (Luq-85). Esta estructura de interconexión dispone de un conjunto de buses, que pueden ser utilizados indistintamente, para interconectar cualquier procesador con cualquier módulo de memoria compartido. Por lo tanto, en cada instante podrán establecerse tantas conexiones Procesador-Módulo de

memoria compartida como buses existan.

La red de interconexión de buses múltiples es tolerante a fallos, ya que puede operar en modo degradado. Despues del fallo de un subconjunto de buses, bajan las prestaciones, pero sigue funcionando, ya que existen caminos alternativos para comunicar un procesador con un módulo de memoria, hast un máximo de B-1 fallos. El fallo de funcionamiento de un bus, solamente provocará, al reducirse el número de posibles caminos de conexión, una degradación en la velocidad y simultaneidad del sistema, pero a diferencia del crossbar, no dejará a ningún procesador o módulo de memoria aislado.

La configuración de este tipo de sistemas es usualmente representada por 3 índices (P,M,B), donde P, M y B son el número de procesadores, el número de módulos de memoria y el número de buses respectivamente. El esquema de conexión estandard para los buses múltiples está mostrado en la figura 2.2. Cada procesador está conectado a todos los buses y cada bus a todos los módulos de memoria compartida. Es interesante notar como el esquema de buses múltiples empleado para conectar estos dos tipos de unidades, equivaldría a un doble crossbar.



P = Procesadores con memomoria local privada

M = Módulos de Memoria común

B = Buses

p = número de procesadores

m = número de módulos

b = número de buses

Figura 2.2 Multiprocesador con una estructura de buses múltiples.

En (Ira-84) al estudiar los sistemas multiprocesadores con interconexión por buses múltiples, los autores establecen la siguiente clasificación: Si B >= min (P,M) entonces el sistema será llamado **"Sistema Bus Suficiente"** (BS); de otra manera será llamado **"Sistema Bus Deficiente"** (BD). Está claro que no existen conflictos por el bus en un sistema Bus Suficiente. Cuando B = min (P,M), la red de interconexión entre procesadores y la memoria común funciona, desde el punto de vista del ancho de banda,como un crossbar, pero su fiabilidad y su coste (número de puntos de cruce y árbitros) son mayores que los de un crossbar. Sus resultados indican que si, en un sistema Bus Suficiente, se reduce el número de buses,a la mitad del valor mínimo original:

$$B = \min (P,M)/2$$

la degradación en el rendimiento llega a ser casi despreciable.

El número de transferencias simultáneas que pueden realizarse, es igual al número de buses. Para B<min(P,M), la red produce una degradación en el ancho de banda con respecto al crossbar, debido a conflictos que ocurren en la red, cuando el número de peticiones para módulos de memoria diferentes es mayor que B. En (Lan-82a) los autores han evaluado esta degradación,

realizando simulaciones, concluyendo que la red de buses múltiples con un número de buses B aproximadamente igual que P/2 para P=M , produce una degradación en el ancho de banda de menos del 5% con respecto al ancho de banda obtenido con el crossbar.

En (Val-83) se presenta un modelo analítico para evaluar la red de buses múltiples, cuando procesadores y módulos de memoria estan sincronizados, es decir, los procesadores pueden pedir acceso a los diferentes módulos de memoria sólo en instantes de tiempoos específicos. Muestran que cuando:

$$B = {}^{1}/_{2} \min (p,m) + 1$$

la degradación es menor que el 5% con respecto al crossbar.

En (Das-85) concluyen que los buses múltiples son más fiables que el crossbar, ya que los buses son independientes y sólo un bus es suficiente para mantener funcionanado el sistema, sin embargo en un crossbar la estipulación de la configuración memoria-bus es completamente rígida.

Otra característica del sistema de buses múltiples, es que el sistema puede ser facilmente expandido, añadiendo módulos de memoria, procesadores,

o buses. Al realizar cualquier expansión, no es necesaria ninguna reconfiguración, es decir, no es necesario realizar ningún cambio en la conexiones entre los otros módulos de memoria o procesadores y buses. La inclusión de un nuevo módulo de memoria o procesador al sistema, implica dotarle de las B conexiones corespodientes a los buses del sistema. Cuando se incluye un nuevo bus, este es conectado a todos los módulos de memoria y a todos los procesadores.

.En la literatura aparecen otros trabajos de evaluación de los buses mutiples, entre los que citaremos, (Mar-82) donde Marsan presenta modelos de cadenas de Markov para buses múltiples. Mudge y otros presentan en (Mud-84) un modelo analítico para estudiar el ancho de banda de sistemas con buses múltiples. Towsley presenta en (Tow-86) dos modelos aproximados para multiprocesadores con buses múltiples.

2.2 Reducción del costo de realización

La estructura de buses múltiples podría ser demasiado costosa, en sistemas donde, el número de procesadores, módulos de memoria y buses fuese elevado. Esto nos lleva a considerar otras redes basadas

directamente en los buses múltiples pero con un coste menor.

A continuación describiremos dos esquemas de interconexión propuestos para minimizar el coste de los buses múltiples: los "esquemas de interconexión reducidos" para buses múltiples y los "buses multiples con multiplexación".

2.2.1 Buses múltiples, esquemas de interconexión reducidos.

El esquema de conexion estandar para buses multiples mostrado en la figura 2.2 en el que P procesadores y M módulos de memoria estan conectados a B buses, es un modelo de interconexión completo . El número de conexiones en este esquema es B*(M+P), este número de conexiones, puede ser un número grande y el resultado una red costosa. En (Lan-83) se demuestra que esta conexión es redundante y se introducen los esquemas reducidos.

La reducción es especialmente significativa cuando el número de buses es relativamente grande. Por ejemplo el número de conexiones requerido se reduce

aproximadamente el 25%, cuando el número de buses es la mitad del número de procesadores, siendo el número de procesadores igual al número de módulos de memoria.

Los esquemas reducidos producen el mismo rendimiento con un coste menor. Este coste ha sido estudiado en (Fio-83) (medido por el número de conexiones y la carga a memoria y bus).

Para la determinación del número necesario de interconexiones podemos referirnos a (Lan-83) donde se enuncia el siguiente teorema, que limita el número de conexiones mínimas:

Teorema 1; "El limite inferior del número de módulos de memoria conectados a un bus es M-B+1 asumiendo que todos los procesadores estan conectados a todos los buses.".

Estas conexiones mínimas tienen que cumplir unas ciertas reglas en cuanto a su distribución sobre los posibles puntos de conexión memoria-bus para que todos los buses puedan ser asignados a cualquier posible combinación de peticiones a módulos de memoria, ya que si estos huecos se distribuyen aleatoriamente pueden producir una perdida de prestaciones en el sistema consecuencia de una reducción puntual del número de

conexiones memoria-bus, no pudiendose acceder simultaneamente al min(B,M) módulos de memoria. Dentro de estas reglas debemos mencionar, que una condición necesaria, pero no suficiente es que cualquier conjunto de B módulos de memoria este conectado a B buses. Esta condición no es suficiente, puede ocurrir que se activen en un mismo instante peticiones a módulos determinados, que no puedan ser asignadas simultaneamente quedando buses libres.

Algunos esquemas mínimos que cumplen estas condiciones han sido descritos en (Lan-83): el rómbico,la escalera y el balanceado. Tambien ha sido objeto de estudio el esquema de interconexión cíclico, éste es un esquema de interconexión reducido no mínimo propuesto en (Lan-83), pero que tiene la importante característica de que cada módulo de memoria está conectado al mismo número de buses, y de que la carga está regularmente distribuída. Este esquema ha sido tambien estudiado debido a sus propiedades ya citadas.

En la figura 2.3a se muestran las interconexiónes que existen entre 8 procesadores, 8 módulos de memoria y 4 buses para el esquema rómbico y en la figura 2.3b se utiliza una representación mediante una matriz que indica la interconexión del crossbar entre los buses y

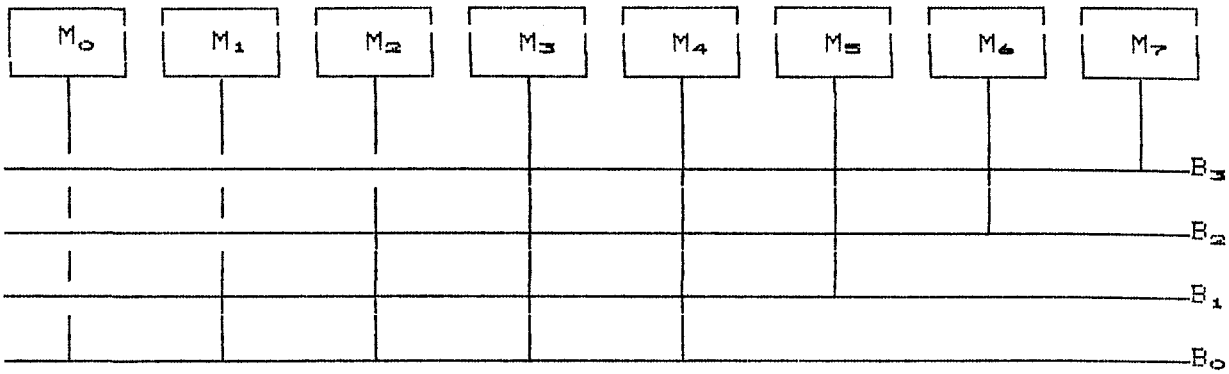


Fig 3.3a Crossbar módulos de memoria-buses en el esquema rómbico

Módulos de memoria

Buses	0	1	2	3	4	5	6	7
3				X	X	X	X	X
2			X	X	X	X	X	
1		X	X	X	X	X		
0	X	X	X	X	X			

Figura 2.3b Representación del esquema de interconexión rómbico

los módulos de memoria.

Aplicando el teorema 1 enunciado en (Lan-83), para 8 módulos de memoria y 4 buses:

$$M - B + 1 = 8 - 4 + 1 = 5$$

vemos que cada bus tiene que estar conectado como mínimo a 5 módulos de memoria.

En la figura 2.4 se muestran las matrices de interconexión del crossbar módulos de memoria-buses para los esquemas reducidos escalera, balanceado y cíclico. En el esquema cíclico hay 6 módulos de memoria conectados a un bus y cada módulo de memoria está siempre conectado a 3 buses, estando por tanto la carga regularmente distribuída.

Los esquemas mínimos para la conexión bus-memoria, se basan en que el crossbar procesador-bus está completo. Este mismo tipo de reducción se puede hacer en el crossbar procesador-bus, si el crossbar memoria-bus está completo.

Deben considerarse así mismo los teoremas enunciados en (Lan-83) y (Fio-83) para la reducción en el número de conexiones entre memorias-buses y entre buses-procesadores.

Buses	0	1	2	3	4	5	6	7
3				X	X	X	X	X
2			X		X	X	X	X
1		X			X	X	X	X
0	X				X	X	X	X

Figura 2.4a Escalera

Buses	0	1	2	3	4	5	6	7
3			X	X	X		X	X
2			X	X		X	X	X
1	X	X	X		X	X		
0	X	X		X	X	X		

Figura 2.4b Balanceado

Buses	0	1	2	3	4	5	6	7
3	X		X	X	X		X	X
2		X	X	X		X	X	X
1	X	X	X		X	X	X	
0	X	X		X	X	X		X

Figura 2.4c Ciclico

Figura 2.4 Esquemas de interconexion reducidos

Considerar el bus "i", sean "m_i" y "p_i" respectivamente, el número de procesadores y módulos de memoria conectados a él. Analogamente, p^-_i y m^-_i, serán los números correspondientes de desconexiones. Evidentemente:

$$p_i + p^-_i = P \qquad y \qquad m_i + m^-_i = M$$

Teorema 2: Para que no existan degradaciones "m_i" y "p_i" deben satisfacer las siguientes restricciones:

$$1)\ M - B + 1 <= m_i <= M$$

$$2)\ (P + M + 1) - (B + m_i) <= p_i <= P$$

Corolario : Si cada bus está conectado a "m" módulos de memoria, el número total mínimo de conexiones es :

$$B*m + B*(P + M + 1 - (B + m)) = B*(P + M - B + 1)$$

Este número es independiente de "m".

Teorema 3: En un sistema no degradado, cada bus i puede estar desconectado de no más que B-1 procesadores o módulos de memoria, esto es:

$$p^-_i + m^-_i <= B-1 \qquad \text{para } 1<=i<=B-1$$

Por lo tanto cada bus debe tener más de P+M-B conexiones.

$$p_i + m_i > P+M-B \qquad \text{para} \quad 1<=i<=B-1$$

Una desventaja que presentan los esquemas reducidos, frente al esquema completo de buses múltiples, en lo referente a su materialización física mediante placas de procesador y memorias unidas por un backplane, es que si conectamos los diferentes procesadores y módulos de memoria del sistema, a través del backplane de buses, no todas las tarjetas de procesadores ni de módulos de memoria estarán conectadas a todos los buses, desaprovechando por lo tanto su potencial capacidad de conexión, puesto que todos los buses existen a lo largo de todo el backplane. Además, puesto que los circuitos de salida de cada placa estarán conectados a diferentes buses, y en algunos esquemas a un número de buses diferentes, existirán placas tanto de procesadores como de módulos de memoria diferentes.

Una alternativa para aprovechar las conexiones al backplane de buses, es la de incluir más de un procesador, o más de un módulo de memoria, que estén

conectados a diferentes buses en una misma placa, esta solucción tambien nos disminuye el número de slots en el bakplane de buses.

El hecho de tener placas diferentes al utilizar esquemas reducidos, presenta además el problema de que estas placas no son intercambiables. Al ser cada placa específica, si existe un fallo, hay que sustituirla por otra con los mismos puntos de conexión al bus, ya que estos serán los que estaran previstos que controle el circuito de gestión de buses.

2.2.2 Formas reducidas incorporando redundancia.

Si consideramos la tolerancia a fallos de los esquemas de conexión reducidos, vemos que hay que modificar las conexiones, para que el sistema pueda operar en una forma degradada, cuando un bus falla, es decir cuando el sistema tenga que funcionar con B-1 buses.

Puede verse, que si existen B-1 buses, el teorema 1 requiere, para que no se produzcan degradaciones, que cada bus este conectado, al menos a M-B+2 módulos de memoria:

$$m_i >= M - (B - 1) + 1 = M - B + 2$$

si existen B-n buses tenemos:

$$m_i >= M - (B - n) + 1 = M - B + n + 1$$

esto no tiene por qué ocurrir siempre, si se han utilizado modelos de interconexión, con el mínimo número de conexiones. Desgraciadamente, sólo para valores particulares de M y de B, resultan esquemas que no se degradan.

Además, puede ocurrir, en esquemas como la escalera, que si un bus falla quede un módulo completamente desconectado, por tanto estos esquemas deben ser modificados si se quiere que funcionen sin degradación al fallar "n" buses, estos esquemas deben ser modificados, para satisfacer el teorema 1 (para B-n buses) y además asegurar, que un módulo de memoria esté conectado al menos a n+1 buses.

El sistema de arbitraje debe estar diseñado para que garantice un funcionamiento satisfactorio frente a fallos, de forma que si existen B-n buses se realicen B-n tansacciones y no ocurra que un bus quede libre existiendo una petición pendiente.

2.2.3 Buses múltiples utilizando multiplexación

Algunas veces, un problema que existe al utilizar un backplane de buses para los buses múltiples, es la limitación en el número de líneas físicas del backplane, insuficientes, cuando el número de buses es grande,

Para aliviar este problema, que se presenta en el momento de la realización física, se pueden utilizar los buses en forma multiplexada.

Cuando un procesador desea acceder a un recurso común, retiene normalmente el bus como mínimo durante un ciclo, utilizando propiamente el bus en los momento de enviar o recoger información. Para realizar la multiplexación,nos fijamos en que el tiempo de transmisión por el bus (tiempo de uso del bus), suele ser menor que el tiempo de acceso a un recurso común, por lo tanto, otros procesadores pueden usar el bus en tiempo compartido durante un ciclo de acceso a la memoria, para enviar información a otros recursos compartidos.

Como mostramos en la figura 2.5, cuando un

procesador desea acceder a un recurso común, emitirá una petición para realizar dicho acceso, transcurrido el tiempo de arbitraje, se le concederá un bus, si es el procesador seleccionado para acceder a dicho recurso y exite algún bus libre, a partir de este momento se enviará la información al módulo de memoria, transcurrido el tiempo de acceso necesitará, en el caso de una lectura, nuevamente el bus para recoger los datos.

Vamos a analizar como se utilizarían los buses multiples con multiplexación y para ello distinguiremos dos casos dependiendo de que el acceso sea para realizae una lectura o una escritura:

-Acceso para realizar una escritura en un módulo de memoria común:Una vez realizada la petición y obtenida la señal de concesión, el procesador enviará las direcciones, datos y señales de control necesarias al módulo de memoria, estas señales serán almacenadas en unos latches, cuyo tiempo de carga es menor que el tiempo de acceso de la memoria, por lo tanto no es necesario tener el bus asignado a este módulo de memoria durante el resto del ciclo, (a memos que el protocolo de comunicación lo exija), pudiendo ser usado dicho bus por otros procesadores.

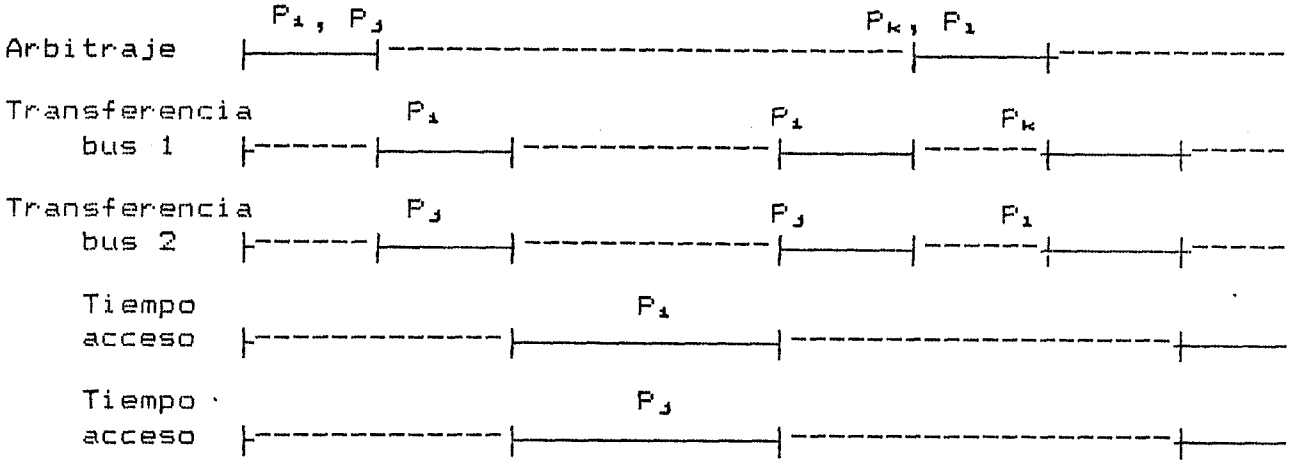


Fig 2.5 Utilización de dos buses sin multiplexar

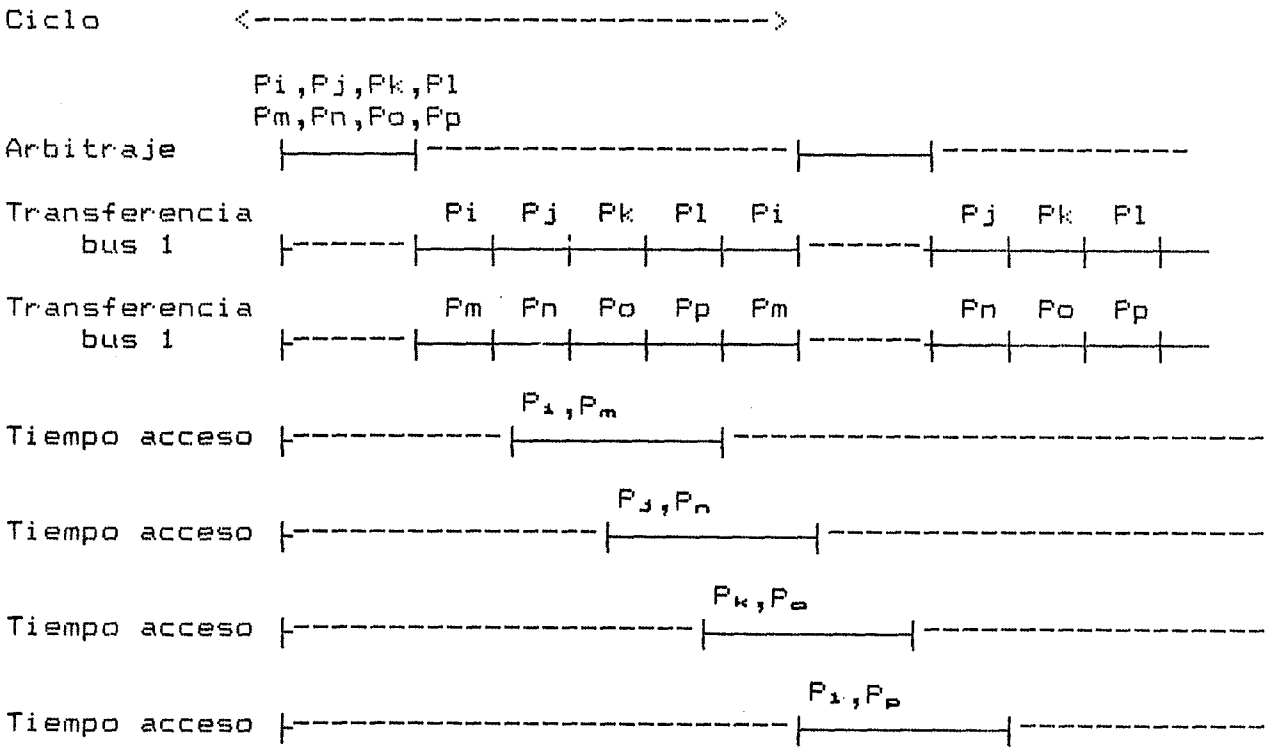


Fig 2.6 Utilización de un bus multiplexando

-Acceso para realizar una lectura en un módulo de memoria: Una vez realizada la petición y obtenida la señal de concesión, el procesador enviará las direcciones de la posición de memoria a la que desea acceder y las señales de control necesarias, estas señales serán almacenada en un latch. Durante el tiempo de acceso de la memoria, en el que los datos no son útiles, el bus puede ser asignado a otro procesador, asegurando que transcurrido un cierto intervalo de tiempo, aproximadamente igual al tiempo de acceso de la memoria, un bus será asignado nuevamente, para que el el módulo de memoria, que ya tendrá los datos disponibles, pueda enviarlos al procesador. Una vez realizada esta transmisión el bus quedará nuevamente libre. Si por requerimientos específicos del procesador, éste necesitase tener los datos estables más tiempo, estos datos pueden ser tambien almacenados en latches en la tarjeta del módulo del procesador correspondiente.

Vemos que para poder realizar la multiplexación, es necesario añadir unos latches de entrada, en los módulos de memoria, para almacenar las direcciones datos y señales de control. El número de transmisiones que se puede multiplexar en el bus, dependerá del tiempo de ciclo del procesador, del tiempo de