El resto de los procesos, pueden ser almacenados en memorias locales pero con menor nivel de prioridad, o en módulos de memoria común junto con todos los monitores (procesos globales). Hay que tener en cuenta que los procesos no prioritarios almacenados en las memorias locales de algún procesador sólo pueden ser ejecutados por dicho procesador, mientras que un proceso global puede ser ejecutado en diferentes momentos por cualquiera de los procesadores del sistema.

Cuando un procesador está libre, tomará alguno de los "procesos globales", almacenados en la memoria común y lo ejecuta, hasta que un proceso local de mayor prioridad, almacenado en la memoria del procesador que lo está ejecutando u otro proceso global más prioritario, lo interrumpa.

La inclusión de procesadores libres, es decir, procesadores sin procesos locales, mejora la ejecución de los procesos globales en terminos de disponibilidad de tiempo de ejecución.

En este sistema, no existe un procesador master encargado de la comunicación entre procesos. Nosotros proponemos utilizar un software básico residente en

cada procesador (Bri-73). Cada procesador estará controlado por una copia privada de este software que permite la cooperación entre tareas.

Las necesidades originadas por la compartición de recursos, así como la limitación de los microprocesadores incorporados en el sistema, hacen preciso la inclusión de ciertos circuitos específicos, a cuyo análisis vamos a proceder.

3.8 Arbitros.

En los sistemas multiprocesadores, la mayoría de los recursos compartidos por los procesadores, no permiten una utilización simultánea. En el caso de que dos o más procesadores deseen utilizar un recurso simultaneamente, se produce un conflicto que tiene que ser resuelto por un árbitro. El recurso debe ser asignado, de acuerdo a un esquema de prioridades, a uno de los procesadores peticionarios durante un intervalo de tiempo.

En un sistema multiprocesador, los árbitros son los elementos encargados de decidir en cada momento que conexiones procesador-memoria se establecen y que buses

se asignan para dichas conexiones.

3.8.1 Tipos de árbitros

Los árbitros pueden ser clasificados, como hemos visto en el capítulo 1, dependiendo de la distribución que de la función de arbitraje se realice entre los procesadores. Puede ser **centralizado** en una única unidad o **distribuído** a través del sistema (Thu-72).

-Arbitro centralizado: un árbitro es llamado centralizado cuando el hardware usado para pasar el control de un dispositivo a otro está basicamente concentrado en una única unidad física. La estructura del árbitro centralizado es mostrada en la figura 3.4. Es el método más simple de arbitraje (Gus-84), usa líneas especiales en el "bakplane" que forman una conexión en "estrella". Las señales de petición están conectadas desde cada dispositivo al árbitro. Una segunda conección en estrella lleva las señales de concesión a cada dispositivo. Este método permite cualquier esquema de asignación siendo muy rápido y eficiente, pero tiene algunas desventajas. Esta técnica no es la mas conveniente para sistemas modulares. Con esta estructura para añadir un nuevo

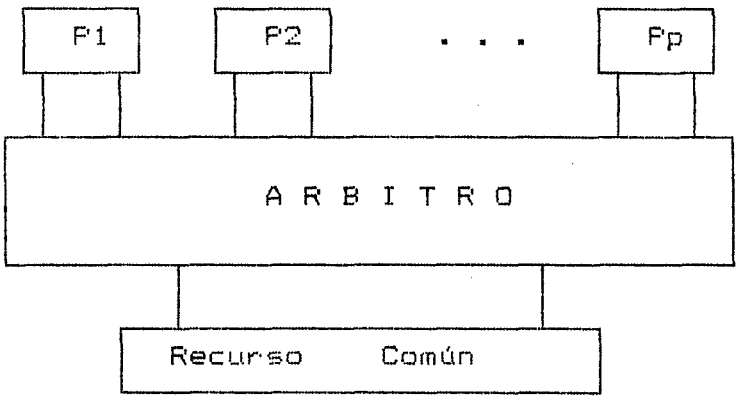


Fig. 3.4 Arbitro centralizado.

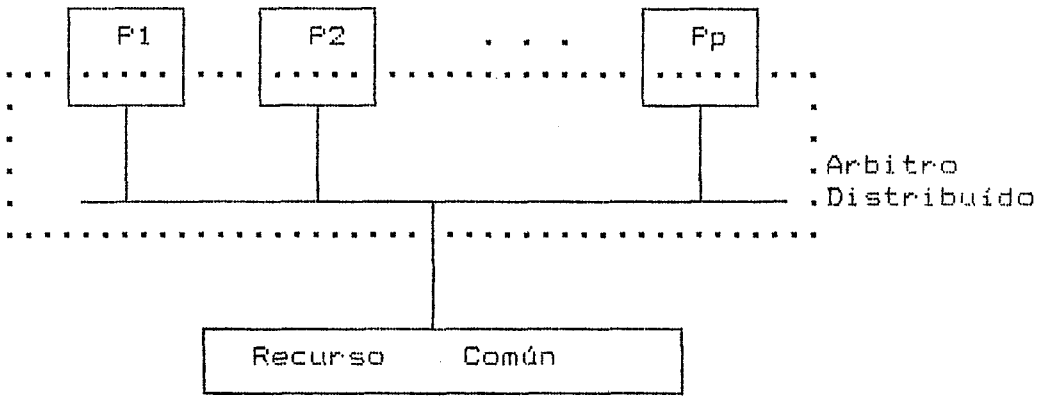


Fig. 3.5 Arbitro distribuido.

peticionario, es tambien necesario añadir líneas y modificar la estructura interna del árbitro. Por estas razones, arbitros centralizados son selecionados cuando son pocos usuarios o el número es fijo.

-**Arbitro distribuído:** el árbitro consta de un conjunto de unidades separadas fisicamente, que son partes de cada peticionario (Figura 3.5). Si el sistema de arbitraje completo esta formado por un conjunto de unidades idénticas, es completamente modular. Usa una topología de conección modular, tal como un bus.

Los árbitros pueden tambien ser clasificados dependiendo del algoritmo de arbitraje, examinaremos algunos algorítmos de arbitraje para controlar el acceso a los recursos comunes:

- **Prioridad fija:** cuando muchos dispositivos piden simultaneamente el uso de un recurso común, se le concede, el acceso a dicho recurso, al dispositivo con mayor prioridad. Usando prioridades estáticas, los dispositivos con prioridades más altas sufren tiempos de espera bajos. Sin embargo, cuando la prioridad es fija, si las peticiones de mayor prioridad estan continuamente llegando, pueden dejar fuera de servicio a los peticionarios de baja prioridad. No podría garantizarse para éstos una acotación en el tiempo de

respuesta.

-**Prioridad rotante** , cada peticionario tiene una prioridad que varía en el tiempo. Esta prioridad es incrementada, por la presencia de peticiones de dispositivos más prioritarios, hasta llegar al máximo y entonces una vez servida es reducida al mínimo para repetir nuevamente el ciclo .

-**Prioridad algorítmica.** Este tipo considera, en general políticas más sofisticadas, así se pueden realizar diferentes algoritmos tales como, el algoritmo "LRU", que da mayor prioridad al dispositivo pidiendo que no ha usado el recurso en un mayor intervalo de tiempo, o el "FCFS" (First-come. First-served) primero en llegar primero en ser servido, o un algoritmo de tiempo fijo que ofrece secuencialmente a cada dispositivo en un modo "round-robin" tiempo de acceso al recurso. Algoritmos que incluyan más de una política diferente asociada tan sólo a ciertos dispositivos podrían tener cabida en este tipo.

Finalmente, los árbitros pueden ser clasificados dependiendo de la técnica empleada para recibir las peticiones y conceder accesos (Thu-72):

-**Daisy chain,** (VME-85) la colocación física de los dispositivos peticionarios, determina su prioridad con respecto a sus vecinos. La señal de concesión se puede

propagar a través de los dispositivos, siendo el dispositivo con petición pendiente que recibe la señal de concesión, el que detiene su propagación y realiza el acceso al recurso común. Son requeridas pocas líneas todas ellas comunes y estas son independientes del número de dispositivos. Su mayor problema estriba en su lentitud en sistemas con un elevado número de dispositivos. Un punto crítico en su diseño radica en la velocidad que debe proporcionar cada dispositivo en su circuito de detención-transmisión de la señal de concesión. Esta velocidad se ve limitada por la resolución del problema de la metaestabilidad potencial originado por la actuación de dos señales asíncronas (la señal de concesión y la señal de petición del propio dispositivo).

-**Polling**, se realiza un escrutinio entre los procesadores conectados a un recurso. Si se utiliza un controlador centralizado, las líneas de concesión son reemplazadas por un conjunto de líneas de polling.

-**Peticiones independientes** (VME-85), en esta técnica, líneas separadas de petición de bus y concesión, están conectadas a cada uno de los dispositivos compartiendo el bus. Para arbitrar y seleccionar P usuarios son necesarias 2*P líneas. Esta técnica no es la mas conveniente para sistemas modulares, porque las líneas de petición y concesión

son conexiones punto a punto, organizadas con una topología de estrella. Con esta estructura para añadir un nuevo peticionario, es tambien necesario añadir una pareja de líneas y modificar la estructura interna del árbitro.

3.8.2 Arbitro binario.

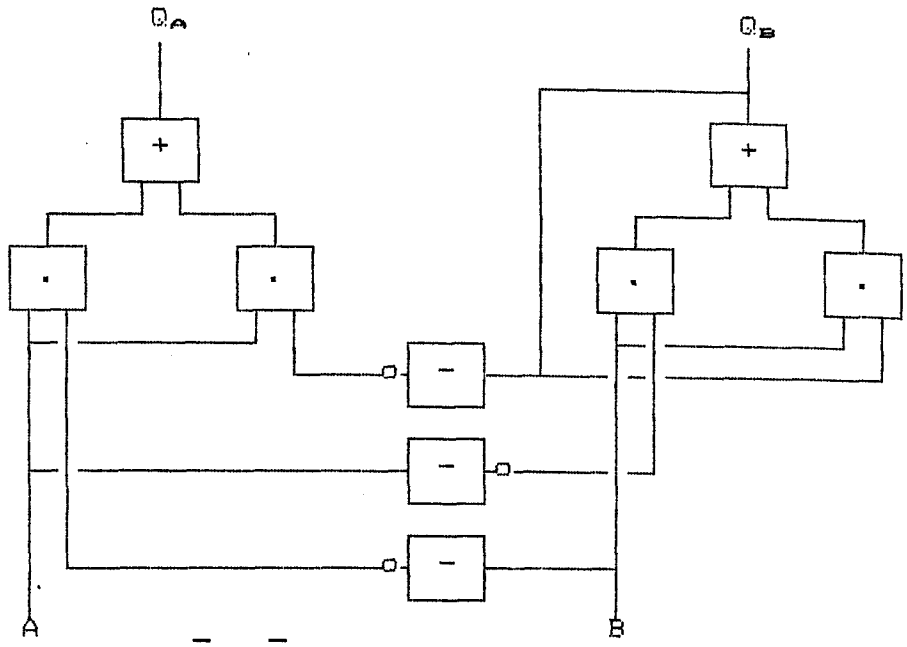
Como módulo básico en el sistema de arbitraje, hemos desarrollado el árbitro binario asíncrono con prioridad fija (Luq-83), mostrado en la figura 3.6a.

Las ideas que presiden el diseño del árbitro son las siguientes:

1.- El árbitro tiene en cuenta la prioridad en el caso de la simultaneidad. Si las dos peticiones llegan simultaneamente, $R_a=R_b=1$, se concederá el recurso al peticionario R_a. La petición R_a es por tanto más prioritaria que la R_b ($R_a>R_b$).

2.- El árbitro es nonpreemtive (Kle-76), por lo tanto un usuario "i" con el recurso concedido ($A_i=1$), no puede ser obligado a dejar libre el recurso por la llegada de otro más prioritario y mantendrá el recurso hasta que su señal de petición sea desactivada ($R_i=0$).

3.- Cuando un usuario cesa en su petición, si



$$Q_A = A.\overline{B} + \overline{A}.Q_B$$

$$Q_B = \overline{A}.B + B.Q_B$$

Figura 3.6a Arbitro binario asíncrono con prioridad fija.

Entradas t					Salidas t+ t	
A	B	Q_A	Q_B		Q_A	Q_B
0	0	0	0		0	0
0	0	0	1		0	0
0	0	1	0		0	0
0	0	1	1	*	X	X
0	1	0	0		0	1
0	1	0	1		0	1
0	1	1	0		0	1
0	1	1	1	*	X	X
1	0	0	0		1	0
1	0	0	1		1	0
1	0	1	0		1	0
1	0	1	1	*	X	X
1	1	0	0		1	0
1	1	0	1		0	1
1	1	1	0		1	0
1	1	1	1	*	X	X

* : Configuraciones de salida prohibidas.

Figura 3.6b Comportamiento lógico del árbitro binario con prioridad fija.

llega la del otro, será concedido el recurso al segundo.

4.- En ningún caso ambas señales de concesión (A_a y A_b), podrán ser simultaneamente igual a 1.

5.- Si no existen señales de petición, ambas señales de concesión serán iguales a cero.

En la tabla de la figura 3.6b, se describe el comportamiento lógico del circuito.

Para crear un sistema de arbitraje de "n" usuarios solicitando un acceso a un recurso común, hemos utilizado el árbitro binario descrito como módulo básico.

3.8.3 Arbitro Daisy Chain Paralelo

La estructura de árbitro Daisy Chain paralelo propuesto, ha sido concebida para que cumpla las siguientes especificaciones:

1.- Una completa modularidad en todas las secciones de arbitraje.

2.- Alta velocidad de arbitraje.

3.- Homogeneidad en el tratamiento de las peticiones.

4.- Puede ser utilizado tanto en sistemas síncronos como asíncronos.

5.- Permite un aislamiento lógico en el caso de fallo.

El arbitro daisy chain paralelo utiliza como módulos básicos un esquema simplificado del árbitro binario incluyendo tan sólo 2 entradas y 1 salida.

Este árbitro binario simplificado, mostrado en la figura 3.7, es un árbitro de prioridad fija, cuyo funcionamiento lógico es el siguiente:

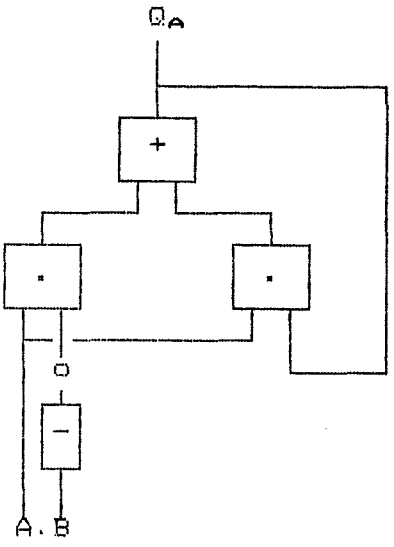
Si A=1 y B=0 ----> Q_A=1

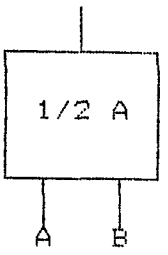
Si A=0 y B=1 ----> Q_A=0

Si A=B=1 simultaneamente ----> Q_A=1

Vemos que la entrada "A" es más prioritaria que la entrada "B". El árbitro es "nonpreentive", es decir, un usuario no puede ser obligado a dejar el recurso por la llegada de uno más prioritario.

El esquema Daisy-Chain paralelo consta de 2n-1 árbitros de 2 entradas 1 salida, tal y como aparece en la figura 3.8. En este esquema se evitan los retardos típicos introducidos por la propagación serie de las señales del Daisy-Chain, lo cual mejora





Entradas t | Salidas t+ t

A	B	Q_A	Q_A
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

$$Q_A = A.\overline{B} + A.Q_A$$

Figura 3.7 Comportamiento lógico y realización física del árbitro binario simplificado.

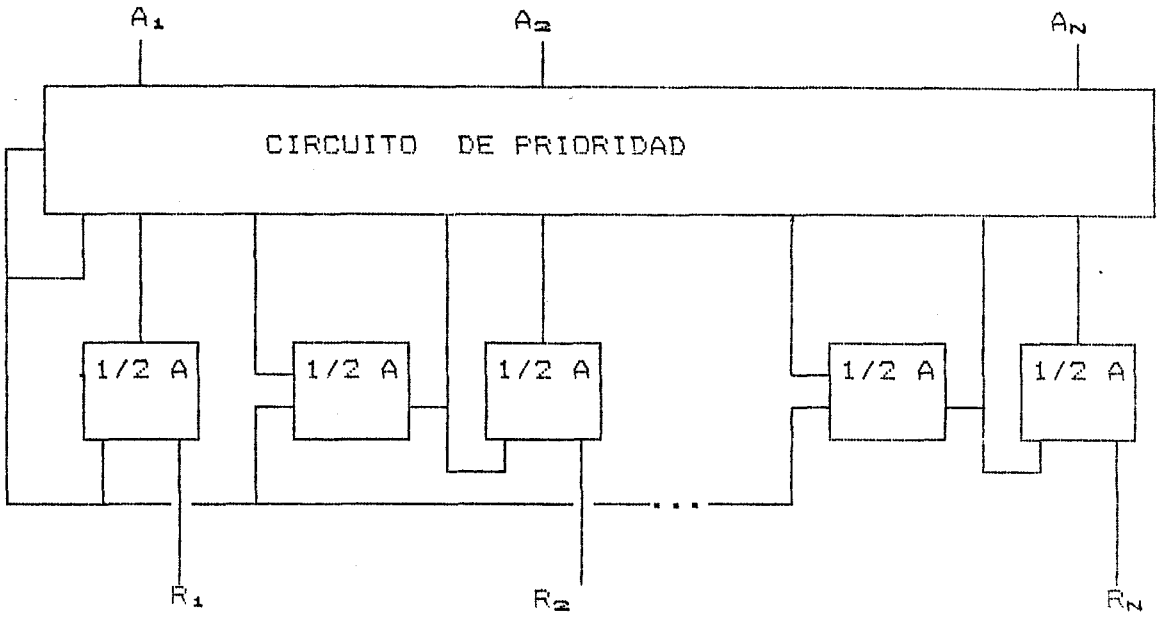


Figura 3.8 Arbitro Daisy-chain paralelo.

considerablemente el tiempo de respuesta, siendo el mismo para todos los peticionarios, independientemente de su situación física. Esta última característica es la que ha determinado su denominación de Daisy-Chain Paralelo.

3.8.4 Arbitro M a N

Como hemos visto, un sistema con buses multiples necesita dos tipos de árbitros, árbitros "P a 1", para seleccionar uno de entre los procesadores que intentan acceder simultaneamente a un mismo módulo de memoria y un árbitro "M a B", para asignar los "B" buses a "B" de aquellos procesadores que han sido seleccionados para acceder a los módulos de memoria.

El arbitro que proponemos es un árbitro de M entradas, y N salidas activas simultáneamente como máximo, siendo N<=M (el número total de salidas será tambien M). Es un árbitro modular basado en los módulos Daisy-Chain paralelos.

El número máximo de salidas activas N, puede ser programable por Hardware. Para ello se han incluído N entradas de inhibición I_j j=1,...,N.

Un diagrama de bloques del árbitro mostrado en la figura 3.9, permite 5 entradas (M=5) y como máximo 3 salidas (N=3).

Este árbitro lo hemos diseñado utilizando 2 niveles de Árbitros Daisy Chain paralelos. En el primer nivel existen N árbitros de M entradas; las entradas a los N árbitros son comunes, la velocidad del circuito frente a llegadas simultáneas dependerá del orden de conexión de las entradas a los diferentes árbitros.

En el segundo nivel existen M árbitros de N entradas cada uno. La salida de cada uno de los M árbitros del segundo nivel, son las M posibles señales de salidas, N de las cuales estarán activas como máximo en cualquiuer instante.

En los N árbitros del primer nivel, existe una señal de entrada "I_J", que sirve para inhibir el funcionamiento del árbitro correspondiente, siendo estas señales las que constituyen la programación del número máximo de salidas activas (N) permitido. La señal "I_J", cuando está a cero desactiva el funcionamiento del árbitro correspondiente, por cuanto "I_J=0", implica que todas las salidas del Daisy Chain

correspondiente estén a cero, no habiendo por lo tanto señales de petición al segundo nivel.

En cada uno de los N árbitros del primer nivel, existe una señal de entrada "I", que sirve para inhibir el funcionamiento de dicho árbitro, siendo esta señal la que permite la programación del número máximo de salidas activas N.

3.9 Prototipo realizado.

El sistema multimicroprocesador realizado consta de los siguientes módulos:

- Cuatro módulos del tipo procesador, incluyendo cada uno de ellos el microprocesadores R6502 con memoria privada y periféricos de entrada/salida locales. Estos módulos incluyen además la circuitería de interface a los buses y la lógica de actuación sobre el microprocesador por parte del gestionador de buses.
- Cuatro módulos de memoria común.
- Cuatro buses con su correspondiente sistema de gestión y asignación.

El multiprocesador ha sido construído utilizando microprocesadores no diseñados originalmente para

arquitecturas multiprocesador, existe por tanto una circuitería que permite la comunicación entre procesadores.

Hemos utilizado el sistema de arbitraje centralizado, para la gestión de buses. El número de buses que se desea utilizar puede ser programado por hardware mediante las señales de inhibición, tal y como ha sido descrito en 3.8.4.

. Debido a las características del microprocesador R6502, hemos utilizado el método de sincronización a nivel de instrucción, existiendo 2 modos de funcionamiento para indicar durante cuantos ciclos se va a mantener una petición:

- Durante todos los ciclos de una instrucción (Interleaving a nivel de instrucción).

- Durante la ejecución de varias instrucciones (Bloqueando el recurso). Esta forma se incorpora para permitir ejecutar primitivas de sincronización.

El bus asignado es fijo durante todos los ciclos que se mantenga activa una petición.

Para lograr estos modos de funcionamiento existe una circuitería de control de las peticiones. Este

circuito controla el paso de las peticiones de los procesadores al circuito de arbitraje.

Para la construcción del prototipo se han utilizado tarjetas de procesadores y memoria estandars, a los que les hemos añadido la circuiteria necesaria para la interconexión. En la figura 3.10 se muestran diferentes vistas del prototipo realizado.

Este prototipo ha sido realizado en wire-wrap, utilizando circuitos SSI. El circuito de arbitraje lo constituyen 2 tarjetas doble europa, en una tarjeta están todos los circuitos correspondientes a la selección del procesador (figura 3.10b) y en la otra los de la asignación del bus.

El bakplane mostrado en la figura 3.10c, está formado por 2 conectores estandar DIN 41612 de 96 patas.

En una tarjeta doble europa, están los circuitos de interconexión correspondientes a dos procesadores (figura 3.10d) o a 2 módulos de memoria.

Las señales en el bus son:

-Vcc.

-GND.

-2 Señales de reloj.

-4*16 líneas de dirección.

-4*8 líneas de datos.

-4*1 señales de lectura/escritura.

-16 señales de petición

-16 señales de concesión.

-16 señales para indicar la asignación de buses.

-Señales específicas de comunicación entre las 2 tarjetas de arbitraje.

Debido al elevado número de líneas en el backplane, cada tipo de tarjetas (tarjetas de procesadores, de módulos de memoria común y de gestión de buses) están asignadas a unos slots fijos, ya que las las líneas necesarias para la gestión de buses, son diferentes para cada tarjeta.

El prototipo tiene capacidad para funcionar sincrona y asincronamente. Cuando lo hemos utilizado funcionando asíncronamente, cada procesador con su propio reloj se generaban tiempos muertos en los buses, exitiendo peticiones activas, debido a que al microprocesador R6502 no se le puede reanudar su funcionamiento en cualquier instante del ciclo.

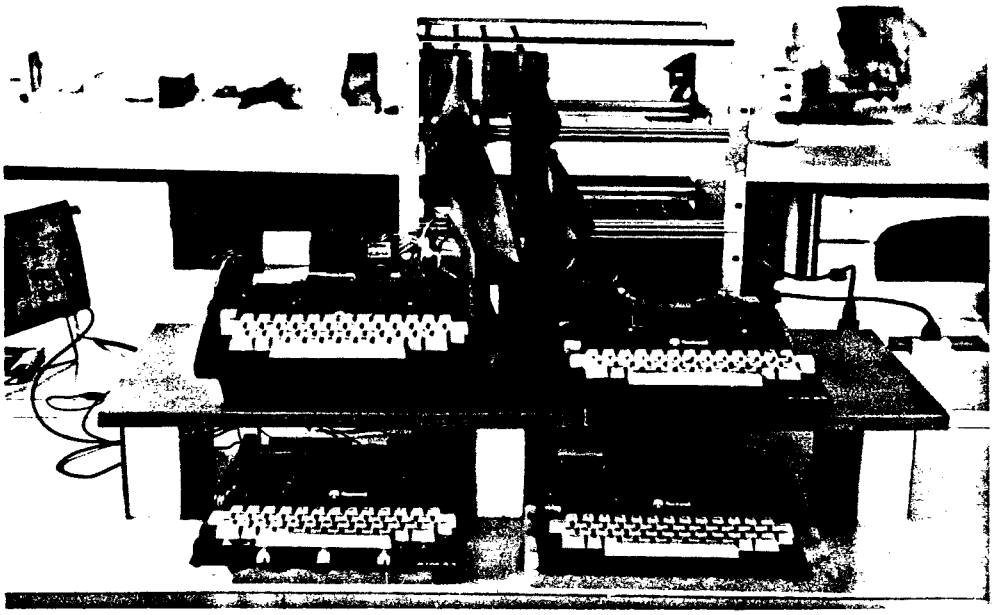


Figura 3.10a Prototipo realizado

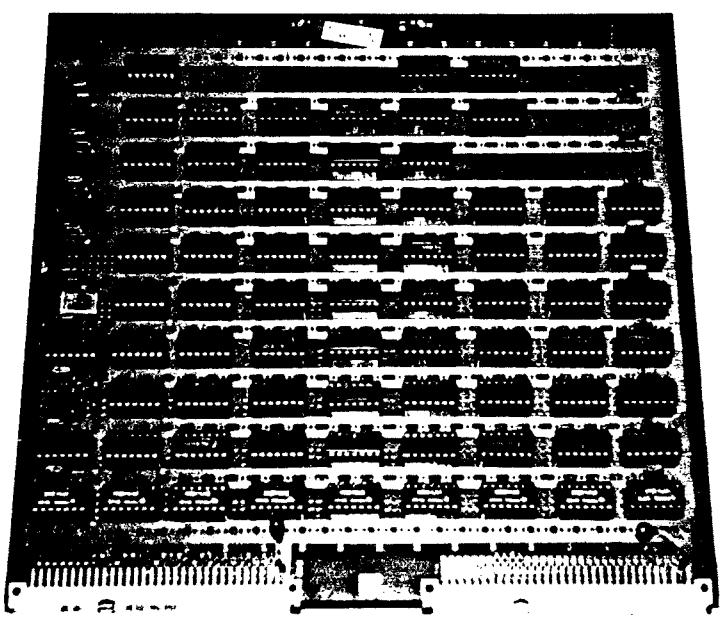


Figura 3.10b Circuitos para la selección del procesador.

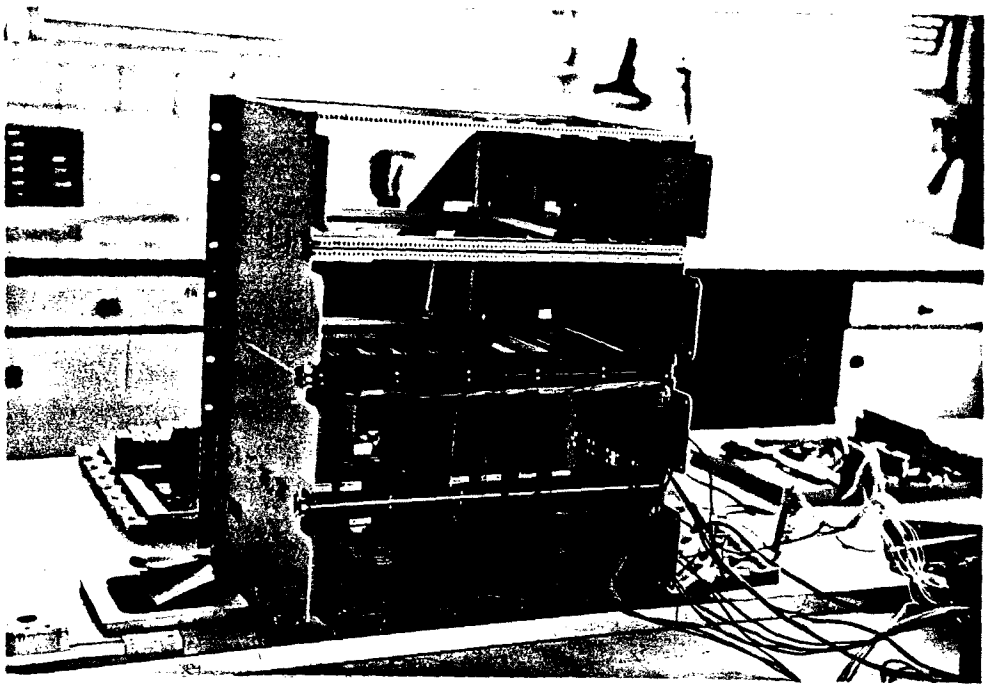


Figura 3.10c Bakplane

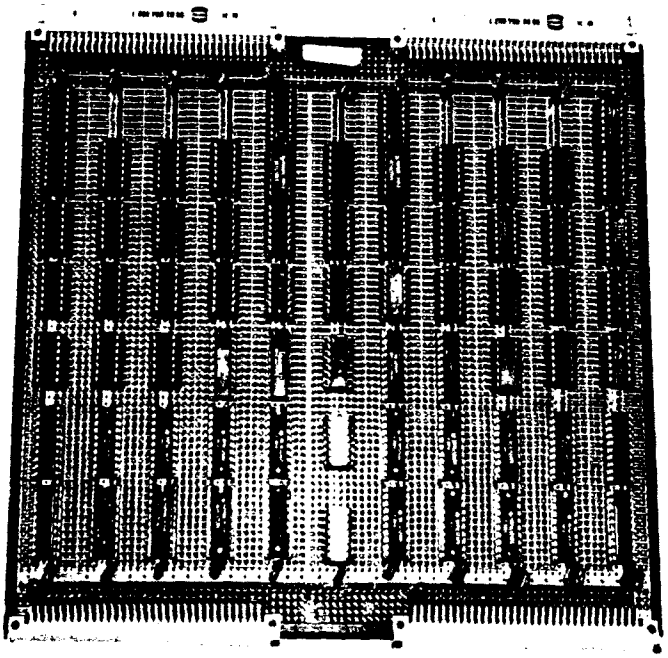


Figura 3.10d Circuitos de interconexión para 2 procesadores.

CONCLUSIONES Y LINEAS ABIERTAS

CONCLUSIONES

Se ha desarrollado un sistema de gestión de buses para redes de interconexión del tipo "buses-múltiples". Este sistema selecciona las peticiones a ser servidas y realiza la asignación de buses correspondiente. El número de buses a ser asignado puede ser programado.

El sistema propuesto posee una estructura regular, estando organizado a partir del árbitro m-1 como módulo básico.

El diseño propuesto para el sistema de gestión de buses, circuito combinacional realimentado, permite que pueda ser utilizado en sistemas multiprocesador, tanto de tipo síncrono como asíncrono.

Para utilizar la red de buses-múltiples en forma multiplexada, se propone una extensión al sistema de gestión de buses desarrollado. Dicha extensión elabora un secuenciamiento temporal de las señales que establecen la asignación.

La implementación del sistema de gestión de

buses, puede ser realizada mediante un esquema centralizado o distribuido.

El sistema de gestión de buses propuesto, puede ser utilizado, sin ninguna modificación, para elaborar la selección de peticiones y asignación de buses en las redes de interconexión de "buses múltiples" con esquemas de conexión reducidos.

Se ha desarrollado un programa de simulación funcional del sistema de gestión. Esta herramienta evalua las asignaciones de buses para todas las posibles configuraciones de peticiones. Los parámetros definibles son: número de buses, número de módulos de memoria y las matrices de interconexión del primero y segundo nivel. En los esquemas de conexión completos, los resultados que proporciona el programa, son las asignaciones realizadas y el tiempo empleado en cada asignación. En los esquemas de conexión reducidos, además de la información anterior, suministra todas las configuraciones de peticiones que no pueden recibir una asignación completa de buses.

A partir de los resultados proporcionados por el programa de simulación, se han encontrado matrices de interconexión que permiten gestionar los

esquemas de conexión reducidos, descritos en la literatura, mediante el sistema de gestión propuesto.

Un prototipo del sistema multiprocesador con buses múltiples incorporando el sistema de gestión propuesto, ha sido diseñado y realizado.

Entre las líneas de investigación que este trabajo deja abiertas, quisieramos señalar algunas de las que a nuestro juicio tendrían mayor interes:

Elaboración de un módelo del sistema de gestión de buses que permitiese obtener las matrices de interconexión óptimas y acotase el tiempo de asignación empleado.

Para los esquemas de interconexión reducidos y los reducidos incluyendo redundancia, la elaboración de un método heurístico para la elaboración de las matrices de interconexión correspondientes.

Estudio del sistema de gestión para elaborar un diseño del mismo, con capacidad de crecimiento incremental, a partir de módulos específicos.

APENDICE A

Programa de simulación

Este programa ha sido realizado para simular el funcionamiento del circuito de gestión de buses para una red de interconexión de buses múltiples con esquemas completos y reducidos.

Los parámetros de entrada al programa son:

-Las matrices de interconexión del nivel 1 y del nivel 2. Estas matrices pueden ser entradas desde teclado o estar almacenadas en disco.

-Las peticiones a los módulos de memoria.

Los datos salida del programa son:

-La asignación de buses, para una entrada dada de peticiones a los módulos

- El número de vueltas que ha realizado el circuito para obtener dicha asignación

Para comprobar el funcionamiento del árbitro, ante cualquier posibe configuración de peticiones de entrada, el programa tiene la opción de generar todas las posibles configuraciones de peticones de entrada, mostrando la asignación de buses para cada configuración, y el número de vueltas que necesita

realizar. El programa muestra, el número de vueltas máximo que ha realizado para obtener las asignaciones.

En los esquemas de buses múltiples reducidos, los huecos se indican , colocando en la posición correspondiente de la matriz, una prioridad mayor que el número de entradas al árbitro, lo cual se interpreta como una "no conexión".

En los esquemas reducidos, dependiendo de las prioridades en las matrices de interconexión, puede ocurrir ,para una configuración específica de entrada,que el sitema no le pueda asignar bus a todas las peticiones activas , quedando buses libres y peticiones sin bus asignado. En este caso el programa indica que no ha podido realizar una asignación correcta. Si el programa está comprobando todas las posibles configuraciones de entrada, muestra las configuraciones de entrada que no han podido ser asignadas, y el número total de configuraciones no asignables utilizando dichas matrices de interconexión.

Listado del programa

```
10 REM LINEA 30 MODIFICAR M1,B1
20 REM MODIFICAR PARA LA GENERACION DE  LAS  PETICIONES
4120-4190
30   REM   NOMBRE  DE  LSA  MATRICES  EN  DISCO  LINEAS
2510,2590,3350,3440
40 REM PROGRAMA DE ARBITROS
50 CLS
60 M1=16: B1 = 8:LPRINT "16 MODULOS Y 8 BUSES"
70  N1=12870  :  REM  N1  ES  EL  NUMERO  DE  TODAS LAS
PETICIONES POSIBLES
80 DIM NI%(16,8),NJ%(16,8),MI%(16,8),MJ%(16,8),S%(16,8)
90 DIM P%(8),PE%(8),MS%(16,8),MV%(16,8),SM%(8),SB%(8)
100 DD=0
110 INPUT "NIVEL 1 : CAMBIO DE PRIORIDAD (S/N) = ";A1$
120 NUM=1
130 IF A1$="S" THEN GOSUB 1550
140 CLS:INPUT  "NIVEL  2  :  CAMBIO  PRIORIDAD?(S/N)  =
";A2$:NUM=2
150 IF A2$="S" THEN GOSUB 1550
160 NUM=1: GOSUB 1240: NUM=2 : GOSUB 1240
170  CLS:INPUT    "¿ DESEA CAMBIAR ALGUN ELEMENTO DE LA
MATRIZ DE PRICRIDAD 1?S/N=";A8$:NUM=1
```

```
180 IF A8$<>"N" GOTO 2040
190 IF PP=1 THEN GOSUB 1670
200 CLS: INPUT "¿DESEA CAMBIAR ALGUN ELEMENTO DE LA
MATRIZ 2?S/N= ";A9$:NUM=2
210 IF A9$<>"N"GOTO 2040
220 IF PP=2 THEN GOSUB 1670
230 CLS
240 INPUT " IMPRIMIR NIVEL 1 ? (S/N) = ";A3$
250 NUM=1: IF A3$="S" THEN GOSUB 1400
260       INPUT "IMPRIMIR NIVEL 2 ? (S/N) = ";A4$ :CLS
270 NUM=2: IF A4$="S" THEN GOSUB 1400
280             INPUT " DESEA COMPROBAR TODAS LAS
CONFIGURACIONES? (S/N) = ";A5$
290 IF A5$="S" THEN 360
300  PRINT " ENTRE LA CONFIGURACION DE ";B1;"PETICIONES
"
310 FOR I=0 TO B1-1
320 PRINT I;" = ";
330 INPUT P%(I)
340 NEXT : A7%=999
350 FOR I=0 TO B1-1:PRINT P%(I) ",":NEXT:PRINT " ";:
GOTO 410
360 CLS: INPUT "IMPRESION DE ASIGNACIONES? (S/N) =
";A6$
370 INPUT "¿NUMERO MAXIMO DE VUELTAS?";A7%
380 AW=0
```

```
390 GOTO 1810
400 REM SUBRUTINA DE ASSIGNACION**************
410 REM
420 CO%=0
430 FOR B=0 TO B1-1
440 FOR PR=0 TO M1-1
450 FOR M=0 TO M1-1
460 IF NI%(M,B)=PR THEN 560
470 NEXT M
480 DD=DD+1
490 LPRINT " ERRORES PARCIAL :"DD
500 LPRINT " CONFIGURACION";
510 FOR I=0 TO B1-1
520 LPRINT P%(I) ",";
530 NEXT
540 LPRINT  " NO ASIGNABLE "
550 GOTO 1120
560 FOR I=0 TO B1-1
570 IF M=P%(I)THEN 620
580 NEXT I
590 NEXT PR
600 NEXT B
610 GOTO 660
620 IF MI%(M,B)<>0 THEN 590
630 MJ%(M,B)=1: REM 1 ES LA MARCA MODULO BUS
640 GOTO 600
```

```
650 REM **** NIVEL 2 ****
660 AX=0
670 FOR I=0 TO B1-1
680 MO=P%(I)
690 FOR PR=0 TO B1-1
700 FOR B=0 TO B1-1
710 IF NJ%(MO,B)=PR THEN 750
720 NEXT B
730 NEXT PR
740 GOTO 790
750 IF MJ%(MO,B)<>0 THEN MS%(MO,B)=1:GOTO 770
760 GOTO 720
770 SM%(AX)=MO:SB%(AX)=B
780 AX=AX+1
790 J=0
800 FOR K=0 TO B1-1
810 IF MS%(MO,K)<>0 THEN J=J+1
820 NEXT K
830 IF J=0 THEN 880
840 FOR K=0 TO B1-1
850 IF MS%(MO,K)=1 THEN MI%(MO,K)=0:GOTO 870
860 MI%(MO,K)=1
870 NEXT K
880 NEXT I
890 F=0
900 FOR I=0 TO M1-1
```

```
910 FOR K=0 TO B1-1
920 IF MS%(I,K)=MV%(I,K) THEN 950
930 F=1
940 MV%(I,K)=MS%(I,K)
950 NEXT K:NEXT I
960 CO%=CO%+1
970 IF F=1 THEN 430
980 FOR B=0 TO B1-1
990 FOR M=0 TO M1-1
1000 IF MS%(M,B)=1 THEN 1020
1010 NEXT M:GOTO 480
1020 NEXT B
1030  IF  A7%<(CO%-1)THEN LPRINT"HA SUPERADO LAS";A7%;"
VUELTAS":GOTO 170
1040 IF VM<CO%-1 THEN VM=CO%-1
1050 LPRINT CO%-1;"VUELTAS"
1060 IF A6$<>"S"THEN GOTO 1110
1070 FOR I=0 TO B1-1
1080 LPRINT "M:";SM%(I);"--B:";SB%(I);"    ";
1090 NEXT I
1100 LPRINT
1110 REM INICIALIZAR MATRICES
1120 FOR I=0 TO M1-1
1130 FOR K=0 TO B1-1
1140 MI%(I,K)=0:MJ%(I,K)=0:MS%(I,K)=0:MV%(I,K)=0
1150 NEXT K:NEXT I
```

```
1160 IF A5$="S" THEN 1180
1170 GOTO  100
1180 RETURN
1190 REM *******IMPRIMIR ERRORES****
1200 PRINT"ERRORES: ";DD
1210 PRINT "MAX VUELTAS";VM:PRINT " "
1220 STOP
1230 REM ******SUBRUTINA LEER DISCO MATRICES NI,NJ
1240 IF NUM=2 THEN 1310
1250 OPEN "I",#1,"PNI16"
1260 FOR I=0 TO M1-1
1270 FOR L=0 TO (B1-1)
1280 INPUT#1,NI%(I,L):NEXT L:NEXT I
1290 CLOSE#1
1300 RETURN
1310 OPEN "I",#2,"PNJ16"
1320 FOR I=0 TO M1-1
1330 FOR L=0 TO B1-1
1340 INPUT #2,NJ%(I,L)
1350 NEXT L
1360 NEXT I
1370 CLOSE#2
1380 RETURN
1390 LIST
1400    REM    *******SUBRUTINA    IMPRIMIR    MATRICES
PRIORIDAD********
```

```
1410 CLS
1420 LOCATE 3,10:LPRINT "PRIORIDAD NIVEL ";NUM
1430 LOCATE 5,5:LPRINT  "MODULOS   00    01  02  03  04
05  06  07  08  09  10  11  12  13  14  15"
1440 LOCATE 6,1
1450 FOR B=0 TO B1-1
1460 LPRINT  "BUS:";B;SPC(1)
1470 FOR M=0 TO M1-1:REM INT SPC(1);
1480 IF NUM=1 THEN 1500
1490 LPRINT " ";NJ%(M,B);:GOTO 1520
1500 IF NI%(M,B)<10 THEN  LPRINT  "  ";NI%(M,B);  :GOTO
1520
1510 LPRINT NI%(M,B);
1520 NEXT M
1530 LPRINT: NEXT B:LPRINT " "
1540 RETURN
1550 REM  *** SUBRUTINA DE ENTRADA DE NUEVA PRIORIDAD
1560 CLS : K=NUM
1570 PRINT "ENTRAR PRIORIDAD NIVEL ";K
1580 FOR L=0 TO (B1-1)
1590 FOR I=0 TO (M1-1)
1600 PRINT "M = ";I;" B = ";L;
1610 IF K=1 THEN GOTO 1640
1620 INPUT "=";NJ%(I,L)
1630 GOTO 1650
1640 INPUT "=";NI%(I,L)
```

```
1650 NEXT I
1660 NEXT L
1670 REM ENTRAR EN DISCO
1680 IF NUM = 2 THEN GOTO 1750
1690 OPEN "O",#1,"PNI16"
1700 FOR I=0 TO M1-1
1710 FOR L=0 TO  B1-1
1720 PRINT #1,NI%(I,L):NEXT L:NEXT I
1730 CLOSE#1
1740 RETURN
1750 OPEN "O",#2,"PNJ16"
1760 FOR I=0 TO(M1-1)
1770 FOR L=0 TO B1-1
1780 PRINT #2,NJ%(I,L):NEXT L:NEXT I
1790 CLOSE#2
1800 RETURN
1810    REM   **********GENERACION   PETICIONES   16*8
**********************
1820 Z0=0
1830 Z1=Z0+1
1840 Z2=Z1+1
1850 Z3=Z2+1
1860 Z4=Z3+1
1870 Z5=Z4+1
1880 Z6=Z5+1
1890 Z7=Z6+1
```

```
1900
P%(0)=Z0:P%(1)=Z1:P%(2)=Z2:P%(3)=Z3:P%(4)=Z4:P%(5)=Z5:P
%(6)=Z6:P%(7)=Z7
1910 AW=AW+1
1920 PRINT AW;" PETICIONES";
1930 FOR I=0 TO B1-1:PRINT P%(I) ",";:NEXT :PRINT " ";
1940 GOSUB 410
1950      IF Z7<15 THEN Z7=Z7+1:GOTO 1900
1960      IF Z6<14 THEN  Z6=Z6+1:GOTO 1890
1970      IF Z5<13 THEN  Z5=Z5+1:GOTO 1880
1980      IF Z4<12 THEN Z4=Z4+1:GOTO 1870
1990 IF Z3<11 THEN Z3=Z3+1:GOTO 1860
2000 IF Z2<10 THEN Z2=Z2+1:GOTO 1850
2010 IF Z1<9 THEN Z1=Z1+1:GOTO 1840
2020 IF Z0<8 THEN Z0=Z0+1:GOTO 1830
2030 GOTO 1190
2040 K=NUM:PP=NUM
2050 PRINT"NIVEL ";NUM;:INPUT " ¿MODULO?=";I
2060 PRINT"NIVEL ";NUM;:INPUT "¿BUS? =";L
2070 PRINT "M= ";I;"B="L;
2080 IF K=1 THEN 2110
2090 INPUT " =";NJ%(I,L)
2100 GOTO 200
2110 INPUT " =" NI%(I,L):GOTO 170
2120 INPUT "NUMERO DE LA CONFIGURACION=";AW:AW=AW-1
2130 PRINT"DEME LA CONFIGURACION"
```

BIBLIOGRAFIA

Ben-82 M.Ben-Ari."Principles of Concurrente Programming" Prentice-Hall International. 1982.

Bor-81 Paul L.Borrill."Microprocessor Bus Structures and Standards". IEEE Micro. Vol. 1,No.1, Febrero 1981 pp. 84-95.

Bor-85 Paul L. Borrill. "MicroStandards Special Feature: A Comparison of 32-Bit Buses".IEEE Micro. Diciembre 1985. pp. 71-79.

Bow-80 Bowen, Buhr. "The logical design of multiple microprocessor systems". Prentice Hall 1980.

Bri-73. Brinch Hansen . "Operating System Principles". Prentice Hall. 1983

Bri-78 Brinch Hansen. "Multiprocessor Arcchitecture for concurrent programs".CAN,Vol.7,No.4, 1978,pp.4-23

Cal-86 J. Calvo, J.I. Acha y M. Valencia. "Asynchronous Modular Arbiter". IEEE Transactions on Computers, Vol. C-35. No. 1. Enero 1986. pp. 67-70.

Cio-83 G.Cioffi y F.Velardi."A Fully Distributed Arbiter for Multiprocessor Systems" . North Holland Publishing Company . Microprocessing and Microprogramming 11. 1983. pp. 15-22.

Cor-79 Corsini P."Speed-independent asychronous arbiter" CDT. Vol. 2, No 5. 1979. pp221-222

Das-85 Chita R. Das y Laxmi N. Bhuyan. "Bandwidth Availability of Multiple-Bus Systems". IEEE Transaction on Computers, Vol C-34, n-10. Octubre 1985 pp. 918-926

Dei-84 Harvey M. Deitel. "An Introduction to Operating Systems". Addinson-Wesley Publishing Company.1984

Dij-68 E.W. Dijkstra. "Cooperating Sequential Processes" Programming Languages. F. Genuys, ed. Academic Press. 1968. pp. 43-112.

Ens-77 Philip Enslow. "Multiprocessor Organization-A Survey". Computing Surveys.Vol.9, No 1.Marzo 1979

pp. 103-129.

Fio-83 M. A. Fiol, M. Valero, J. L. A. Yebra y T. Lang "Reduced interconnection networks based in the multiple bus for multimicroprocessor systems". Int'l Symp. MIMI. Lugano. Junio 1983. pp 54-58.

Fly-72 M.J. Flynn."Some Computer Organizations and their Effectiveness".IEEE Transactio on Computers. C-21 No 9. Septiembre 1972. pp. 948-946.

Gus-84 David B. Gustavson. "Computer Buses-A Tutorial" IEEE Micro. Agosto 1984. pp. 7-22.

Hoj-78 Hojberg K. S. "One-Step programmable arbiters for Multiprocessors". Computer Design. Abril 1978. pp. 154-158.

Hoo-77 Cornelis H.Hoogendorn."A General Model for Memory Interference in Multiprocessors".IEEE Transaction on Computers.C-26,No.10.Octubre 1977. pp.998-1005

Hwa-84 Kai Hwang y Fayé A. Briggs."Computer Architecture and Parallel Processing" . McGraw-Hill Book Company. 1984.

Ira-84 Irani K. B. and Onyüksel I.H.. "A closed-form solution for the performance analysis of multiple-bus multiprocessor systems". IEEE Trans. on Comp. Nº 8. 1984, pags 1004-1012

Kle-76 L. Kleinrock. "Queing Systems" V. 2. Computer application. John Wiley. 1976

Lan-82a Tomas Lang y Mateo Valero. "M-Users B-Servers Arbiter for Multiple-Buses Multiprocessor". Multiprocessing and Microprograming.North-Holland Publishing Company. Octubre 1982. pp. 11-18.

Lan-82b Lang T., Valero M. y Alegre I.."Bandwidth of crossbar and multibple-bus contentions for multiprocessors". IEEE Trans. on Comp. C-21, Nº 12. 1982, pags 1227-1234.

Lan-83 Lang T., Valero M. y Fiol M.A.. "Reduction of connections for multibus organization". IEEE Trans. on Comp. C-33, Nº 8. 1983, pags 707-716

Lax-87 Laxmi N. BHuyan. "Interconnection Networks for Parallel and Distributed Processing" . Computer. Junio 1987. pp. 9-12

Law-75 Duncan H. Lawrie. "Acces and Alignment of Data in Array Processor". IEEE Transactions on Computers. Vol. C-24, No 12. Diciembre 1975. pp. 1145-1155.

Len-82 Lent B. "A variable priority arbiter for resource allocation in Asynchronous Multiprocessor Systems". Microprocessing and Microprogramming 9. 1982. pp. 229-307

Luq-82 E. Luque, L. Moreno, A.Ripoll, D.I.Rexáchs. "Multimicroprocesador para programación concurrente". 5º Congreso Informática y Automática. Tomo II. Madrid 1982. pp. 605-609.

Luq-83 E. Luque, L. Moreno, A.Ripoll, D.I.Rexáchs. "Multimicroprocesador para programación concurrente". Regulación y mando automático. No 129 Junio 1983. pp. 131-134.

Luq-84 E. Luque, D.I. Rexáchs y A. Ripoll. "Modular multibus multimicroprocessor". Mini and microcomputers and their applications. Bari. 1984. pp. 87-90.

Luq-84a E. Luque, D. Rexáchs y A. Ripoll. "Modular multiple microprocessor system for control applications". Computers in Communication and Control (EUROCON 84), pags 298-301

Luq-85 E. Luque, D.I.Rexáchs, J. Sorribes y A. Ripoll. "A modular arbitration for multiple buses multiprocessors". EUROMICRO. Bruselas 1985. pags 579-585.

Luq-85a E. Luque, D.I.Rexáchs, J. Sorribes y A. Ripoll. "Bus allocation in multibus-multiprocessors".Mini and microcomputers and their applications. 1985. pp.124-127.

Mae-79 Maekawa M. y otros. "Experimental Polyprocessor System (EPOS)-Architecture". CAN, Vol 7, No 6. 1979. pp. 188-195

Mar-82 M.A.Marsan y M.Gerla. "Markov Models for Multiple Bus Multiprocessor System". IEEE Transactions on Computers. C-31. No. 3. Marzo 1982. pp. 239-248.

Mud-84 T.N. Mudge, J.P. Hayes, G.D. Buzzard y D.C.Winsor "Analysis of Multiple Bus Interconnection Networks". Proc. of the 1984 Int'l Conference on Parallel Processing. 1984. pp. 228-232.

Mud-85 Trevor N. Mudge and Humoud B. Al-Sadoun "A Semi--Markov Model for Performance of Multiple-Bus Systems". IEEE Transactions on Computers.Vol C-34 n-10. Octubre 1985. pp. 934-942.

Mud-87 T.N. Mudge, J.P. Hayes y D.C. Winsor. "Multiple Bus Architectures".Computer. Junio 1987. pp.42-48

Muh-80 Muhlemann K."Method for reducing memory conflicts caused by busy waiting in multiple-processor synchronisation". IEE Proc. Vol. 127. Pt E, No 3. 1980. pp. 85-87

Pak-83 Y. Paker. "Multi-microprocessor Systems".Academic Press. 1983.

Pat-79 Janak H. Patel."Processor-Memory Interconnections for Multiprocessors". Computer Architecture News. Vol. 7, nº 6. 1979. pp. 168-177

Pat-81 Janak H. Patel "Performance of Processor-Memory Interconnections for Multiprocessors " . IEEE Transactions on Computers, Vol. C-30, No 10. Octubre 1981. pp. 771-780.

Pea-75 R.C.Pearce,J.A. Field y W.D. Little "Asynchronous Arbiter Module". IEEE Transactions on Computers, Septiembre 1975. pp. 931-932.

Pet-80 Petriu E. "N-Channel Asyncronous Arbiter resolves resource allocation conflicts". Computer Design. Agosto 1980. pp. 126-132.

Pri-86 Shlomo Pri-Tal."MicroStandards. The VME Subsystem Bus". IEEE Micro. Abril 1986. pp. 66-71.

Sat-80 M. Satyanarayanan. "Commercial Multiprocessing Systems". Computer. Mayo 1980. pp. 75-96.

Sie-79 H. Siegel y otros. "A survey of interconnection methods for reconfigurable parallel processing systems". Proc. NCC. Junio 1979. pp. 529-542.

Swa-77 R. J. Swan y otros. "CM* - A modular multimicro-processor". AFIPS, Conf. Proc., Vol. 46, 1977. pp. 637-644.

Tan-76 Andrew S. Tanembaum. "A Survey of Operating Systems". Informatie jaargang 18 nr.12. Amsterdam Diciembre 1976. pp.' 665-731.

Tau-84 D.M. Taub "Arbitration and Control Acquisition in

the proposed IEEE 896 Futurebus". IEEE Micro. Agosto 1984. pp 28-41.

Tha-81 S. Thanawastien and V. P. Nelson. "Interference analysis of shuffle-exchange networks". IEEE Trans. Comput. vol. C-30. Aug 1981. pp. 545-556.

Thu-72 Thurber K. J. y otros."A systematic approach to the design of digital bussing structures". AFIPS Procc. Vol. 41, 1972. pp. 399-420.

Tow-86 Don Towsley. "Aproximate Models of Multiple Bus Multiprocessor Systems". IEEE Transactions on Computer. C-35, No.3. Marzo 1986. pp. 220-228.

Val-83 M. Valero, J. M. Llabería, J. Labarta, E. Sanvicente, T. Lang. "A performance evaluation of the multiple bus network for multiprocessor systems" .ACM 1983. pp. 200-206.

VME-85 "VMEbus Specification Manual". Revisión C.1, Octubre 1985.

Wu -81 S. Wu and M. T. Liu . "A cluster structure as as interconnection network for large multimicro-computer systems". IEEE Trans. Comput. vol. C-30 Apr. 1981, pp. 254-264

Wid-76 L. C. Widdoes. "The Minerva Multiprocessor". in Proc. 3rd. Symp. Comp. Arch. 1976. pp. 34-39.

Wul-72 Wulf W. A. y Bell C. G. "C.mmp A multi-miniprocessor". Fall Joint Comp. Conf. AFIPS, 1972, pp. 756-777.