

(12) PATENT ABRIDGMENT (11) Document No. AU-B-75257/87
(19) AUSTRALIAN PATENT OFFICE (10) Acceptance No. 612582

(54) Title
COMPUTER BUS

International Patent Classification(s)
(51)⁴ G06F 013/40 G06F 013/42

(22) Application Date : 06.07.87

(30) Priority Data

(31)	Number	(32)	Date	(33)	Country
	882300		07.07.86		US UNITED STATES OF AMERICA

(43) Publication Date : 14.01.88

(44) Publication Date of Accepted Application : 18.07.91

(71) Applicant(s)
CONCURRENT COMPUTER CORPORATION

(72) Inventor(s)
KENNETH C. YEAGER

(74) Attorney or Agent
WATERMARK PATENT & TRADEMARK ATTORNEYS , Locked Bag 5, HAWTHORN VIC 3122

(56) Prior Art Documents
US 4459665

(57) Claim

1. A bus for transmitting data, function codes, module codes and parity information between modules connected thereto which comprises:

at least two paths which comprises:

- (a) 32 path lines for transmitting path data,
- (b) 5 path lines for transmitting path function codes.

(c) 8 path lines for transmitting path identification codes.

(d) 5 path lines for transmitting path parity information,

(e) 4 path acquisition lines for transmitting path acquisition signals, and

(f) 2 path acknowledge lines for transmitting path
acknowledge codes;

means for interfacing modules to the at least one path;

path acquisition means comprising:

(a) means, responsive to module signals generated on the at least one module, for generating the path acquisition signals and

(b) means, responsive to a path module acquisition request signal generated on the at least one module in response to the path acquisition

signals, for generating path module
acquisition grant signals;

path module acquisition means, disposed on the at
least one module, comprising:

- (a) means for generating the path module signals,
- (b) means for generating the path module
acquisition request signals in response to the
path acquisition signals, and
- (c) means for determining whether the at least one
module has acquired the path in response to
the path module acquisition grant signals.

COMPLETE SPECIFICATION

(ORIGINAL)

612582

Class

Int. Class

Application Number:
Lodged:

Complete Specification Lodged:

Accepted:

Published:

Priority:

Related Art:

Name of Applicant:

Concurrent Computer Corporation
~~THE PERKIN-ELMER CORPORATION~~

Address of Applicant:

15 Main Street, Holmdel, New Jersey 07733
~~761 Main Avenue, Norwalk, Connecticut 06859, United States~~
~~of America~~ U.S.A.

Actual Inventor:

KENNETH C. YEAGER

Address for Service:

EDWD. WATERS & SONS,
50 QUEEN STREET, MELBOURNE, AUSTRALIA, 3000.

Complete Specification for the invention entitled:

COMPUTER BUS

The following statement is a full description of this invention, including the best method of performing it known to:

US

COMPUTER BUS

Background of the Invention

5 The present invention pertains to a computer system and, in particular, a bus system for use therein.

Computer systems exist which use a bus system to route signals from a central processing unit to either a memory device or peripheral equipment and vice versa.

10 A need exists for a bus system which enables multiple central processing units to interface with each other and with memory devices, input/output devices and input/output device controllers. In addition, the need exists for such a bus system which can also interface several computer
15 systems, each using a bus system, to provide multitask-multisystem integrated computers for network and task coordination.

Summary of the Invention

20 The present invention pertains to a computer bus system which enables multiple central processing units to interface with each other and with memory devices, input/output devices and input/output device controllers.
25 In addition, the bus system also interfaces several computer systems, each using a separate inventive bus system, to provide multitask-multisystem integrated computers for network and task coordination.

30 The inventive bus comprises having two paths with each path being capable of transmitting 32 bits of data, 5 bits of function code, 8 bits of identification information, 5 bits of parity information, 2 bits of acknowledge codes, and four bus acquisition signals.

35

Brief Description of the Drawing

A complete understanding of the present invention may
5 be gained by considering the following detailed description
in conjunction with the accompanying drawing, in which:

FIG. 1 shows a block diagram of module interfaces to
paths on the inventive bus system;

FIG. 2 shows a block diagram of bus path acquisition
10 circuitry for the inventive bus system;

FIG. 3 shows a block diagram of a backplane of the
inventive bus system;

FIG. 4 shows a block diagram of clock distribution
circuitry for the inventive bus system; and

FIG. 5 shows a memory module for use with the
inventive bus system.

To facilitate reader understanding, identical
reference numerals are used to designate elements common to
the figures.

Detailed Description

The inventive, high-performance synchronous bus system
will be described in the context of its use in a Model
25 3280MPS computer system manufactured by Concurrent Computer
Corporation, hereinafter referred to as the disclosed
embodiment. This bus system, hereinafter referred to as the
S-bus, interconnects various modules of the computer
system, such modules being computer processor units (CPU),
30 input/output channels (I/O), and memories.

In general, the S-bus routes messages between modules
on two distinct 32-bit data paths, a "To" or T-path and a
"From" or F-path. The T-path transfers addresses and data
"to" memory modules from other modules. The F-path
35 transfers data read "from" memory modules to other modules.
The F-path is also used (1) to route inter-processor
messages; (2) to control direct input/output (I/O), i.e.

where an I/O device operates directly under the control of a CPU (this is contrasted with I/O that occurs under the control of a channel, otherwise known as DMA); and (3) to broadcast interrupts to modules other than memory.

5 FIG. 1 shows a block diagram of a system which uses an S-bus and the modules interconnected thereby. S-bus 10 comprises F-path 1 and T-path 2. For purposes of simplicity, FIG. 1 only shows an overview of the interconnections between the modules which use the S-bus
10 system and does not show the bus control and acquisition circuitry.

 F-path 1 and T-path 2 are connected to various modules of the computer system. CPUs 102_1 to 102_n are connected to S-bus 10 by means of connections which allow bi-directional
15 information transfer between the CPUs and F-path 1. The bi-directional connection is required because F-path 1, in addition to transferring data from memory modules to the CPUs, also transfers inter-processor messages, direct I/O and broadcast interrupts from the CPUs to other modules.

20 Composite Memory Modules (CMM) 101_1 to 101_n , to be described in detail hereinbelow, are connected to S-bus 10 by means of connections which only allow uni-directional information transfer to F-path 1 and T-path 2. Only a uni-directional connection is required because addresses and
25 data are transferred over T-path 2 to memory modules, whereas data read from memory modules is transferred to other modules over F-path 1.

 I/O channel 105, sometimes referred to as a DMA interface, is connected to S-bus 10 by means of connections
30 which allow bi-directional information transfer to F-path 1 and only uni-directional information transfer to T-path 2.

 Module 106 is an interface circuit for connecting "compatible" I/O devices to S-bus 10, in a similar manner to the connection of I/O channel 105. "Compatible" I/O
35 devices are devices which interfaced to prior computer systems, i.e., prior to the Model 3280MPS. Module 106

allows bi-directional information transfer to F-path 1 and uni-directional information transfer to T-path 2.

In addition to the above, S-bus Exchange circuits (SBX) 110 and 210 connect S-bus 10 to S-bus 20. SBX 110 and SBX 210 are each connected to paths on the respective S-buses by means of connections which allow unidirectional information transfer and are connected to each other by S-bus cable 107. This provides the means for modules connected to S-bus 10 to interact with modules connected to S-bus 20.

S-BUS OVERVIEW

T-Path (TO):

Processors and I/O channels initiate memory operations by sending addresses and data to memory modules on T-path 2. This is the only function of the T-path.

F-path (FROM):

Data read from any of the memory modules is returned to requesting processors and I/O channels on F-path 1. F-path 1 is also used for communications between processors and I/O channels.

F-path 1 and T-path 2 each contains 50 data lines, each data line carrying one bit of information. The bits, represented by the data lines, are divided, as listed below, into four groups or fields which comprise a bus data "item":

- 1) 5-bit function field,
- 2) 8-bit identification (ID) field,
- 3) 32-bit data field, and
- 4) 5-bit parity field.

In addition to the 50 data lines used to transmit information, F-path 1 and T-path 2 each contains six acquisition lines and two acknowledge lines. As will be explained in detail below, the acquisition lines for each

path are used to control path acquisition by the modules and the acknowledge lines for each path are used to control the interaction between modules by transmitting signals which indicate the receipt of data items.

5

Communication Among Modules Connected to the S-bus

ID:

10 1 - Modules such as processors and I/O channels make similar use of the S-bus and are identified by 8-bit unit ID numbers. On T-path 2, the unit ID number identifies the requesting module. For example, each module puts its unit ID number on T-path 2 when it initiates a memory operation
15 to identify itself as the requesting unit. Memory modules latch the unit ID number on T-path 2 and echo it on F-path 1 when responding with read data. The requesting module recognizes its unit ID number when it receives responses on F-path 1.

20 The ID appearing on F-path 1 addresses the destination module. Each module compares that ID with its own ID and latches all messages which match.

 An exception to the above for the disclosed embodiment occurs for a broadcast interrupt; here the sender ID
25 appears on F-path 1. The broadcast message is received by all processors, regardless of whether the ID matches. However, for any other F-path bus data item, only the addressed unit will respond.

 2 - Memory is referenced solely by memory address;
30 i.e. memory modules do not have unit ID numbers. Each memory module recognizes the addresses of the memory locations it contains.

35

Interaction Among Modules Connected to the S-bus

1 - Modules such as processors initiate memory operations on the T-path by specifying a memory operation
5 in the 5-bit function field of a bus data item and placing a memory address in the 32-bit data field of the bus data item. For data writes, the data to be written into the memory module is sent to the memory module during subsequent bus operation cycles. For memory reads, the
10 addressed memory module performs the read and then returns the data to the requesting module on the F-path. The disclosed embodiment provides four different memory reads and and four different writes for reading or writing from one to four words from memory.

15 2 - Modules such as processors and I/O channels communicate directly by means of the F-path. Inter-processor messages transfer interrupts and three data words from any processor or I/O channel to another designated module. Such messages are used (1) to coordinate processors
20 in multiprocessor systems; (2) to initiate I/O channel operations, and (3) to reschedule tasks when I/O operations are completed.

25 3 - Messages can be sent from one module, such as a processor, to interrupt another module, such as a different processor.

30 4 - An interrupt can be broadcast to all processors simultaneously. This allows an operating system to pre-empt lower priority processes. In the disclosed embodiment, processors receive and act upon all broadcasts regardless of their ID. Broadcasts are not acknowledged on the F-path.

5 - A processor can directly access an I/O channel to perform "Direct" Input/Output operations.

35 6 - Multiple S-buses may be interconnected to form larger computer systems by using two SBX circuits --one on each S-bus, e.g. SBXs 110 and 210 in FIG. 1-- and a "System Cable" --e.g. cable 107 in FIG. 1. Modules can thereby access memory in other computer systems by using real

addresses or by sending messages to other modules by unit ID numbers. An SBX monitors both the T-path and the F-path for messages addressed to the remote computer system. On the T-path, the SBX responds to selected memory addresses
5 which are stored as a bit map in a random access memory (RAM) contained therein --in the disclosed embodiment, a 64K x 4 bit RAM stores four bits for each 64K byte block in the S-bus 4G byte real address space. On the F-path, the SBX responds to selected unit ID numbers --in the disclosed
10 embodiment, these unit ID numbers are stored as a bit map in a 256 x 4 bit RAM. The messages intended for the remote computer system are acknowledged, buffered, and forwarded by the local SBX to the remote SBX. The remote SBX acquires the appropriate corresponding path in the remote computer
15 system and echos the message.

The module which is given access to a path on the S-bus is called the "master" and the module which responds to requests from the master is called the "slave." The master of each path, T or F, can send one bus data item during
20 each bus operation cycle. The master enables its bus drivers when it gains control of a path and disables them when it relinquishes control to another module. The bus operation cycle is long enough to allow time for:

- a. clock slew;
- 25 b. propagation delays of control flops (clock output);
- c. enable/disable times of bus path drivers (data propagation through the drivers is faster than enable/disable);
- d. bus path propagation and settling time; and
- 30 e. set-up time of bus path receivers and registers.

Every module receives every message. A bus data item is decoded during the cycle following its receipt; no logic delays are allowed during the transfer itself. Then, each module determines whether or not it is addressed: (1) if it
35 is addressed, it becomes a slave and (2) if it is not addressed, or if the function code of the bus data item is "idle," it takes no further action. A slave checks the

parity of each bus data item it receives, the parity of "idle" cycles is ignored.

5 A slave acknowledges each bus data item it receives after that item was sent. In the disclosed embodiment, the acknowledge occurs two cycles after the bus data item was sent, i.e. on the cycle after the slave decoded the information. This pipelining of acknowledges permits high transfer rates. However, a master contains logic circuitry which enables the master to repeat the entire command message if the "busy" acknowledge response is signalled by
10 a slave.

The acknowledge lines in the disclosed embodiment are driven with open-collector gates and the acknowledge signal is the logical-OR of all slaves responding. Normally only one slave will respond, but a parity fault or improper
15 configuration can cause two modules to respond to the same message. The fault code can be aliased if one of these boards is busy. The information conveyed by the acknowledge signals is given as follows:

20	<u>AK1</u>	<u>AK0</u>	<u>Meaning</u>
	0	0	No response - Idle cycle or module not present
	0	1	Here - Slave is present and accepted the item
25	1	0	Busy - Slave is present but did not accept item
	1	1	Fault - Slave detected a parity error

30 The acknowledge lines merely acknowledge the receipt of a bus data item. Thus, the acknowledge response "here" does not mean that the requested operation can be performed.

In the disclosed embodiment, when the slave sends a "busy" acknowledge response, this means that the slave's
35 input buffer is full. For proper system operation, the slave must empty its buffer expeditiously.

When the master receives a "busy" acknowledge response, it must relinquish the bus and repeat the entire operation later. In the disclosed embodiment, in order to prevent undue congestion, a module gets to keep the bus upon receiving two consecutive "busy" responses. Memory and other modules should assert "busy" only on address cycles, i.e. when a master is attempting to create a slave. Thus, a module should assert the "here" acknowledge response only when it can accept the entire operation. On both the T- and F-paths, one bit in the bus data item --the most significant bit of the function code-- is set to 1 for an address cycle and to 0 for a data cycle.

The "fault" acknowledge response indicates a hardware malfunction.

As described above, parity bits accompany the data, function and ID fields in the bus data item. In the disclosed embodiment, the parity bits are used to detect errors which alter an odd number of bits. The parity bits are generated by the master and checked by the slave. A parity error is detected when the parity check bit differs from the bus parity bit.

If a slave detects a parity error, it sends a "fault" acknowledge response to the master and the bus data item is ignored.

All operations on the S-bus are deferred response. This means that a master releases the bus after issuing a command and waits for an acknowledge response. The immediate acknowledge response assures the master that the command was received. However, a master must include time-out logic to handle missing responses due to, for example a hardware fault.

In the disclosed embodiment, memory operations should complete within 25 usec. Longer delays are faults.

Responses to channel I/O commands can be delayed indefinitely due to long mechanical delays, queuing, or human interaction. The software must determine and implement its own time-outs.

OVERVIEW: BUS ACQUISITION AND PRIORITY

The operation of the inventive S-bus is synchronous
5 with a system clock, shown in the disclosed embodiment to
be a 10-megahertz clock. A module must request access to a
path and then be given that access, i.e. become a master,
before the path can be used to transfer information. The
master has temporary control of the path. Addresses and
10 data are transferred on the path from the master to a slave
during one or more bus operation cycles.

Bus allocation operates in parallel with transfers on
the S-bus. The next master of a path is determined at the
end of each bus operation cycle.

15 The T-path and the F-path have identical, but
independent circuitry, which allocates a path on the S-bus
between the modules connected thereto. This circuitry
comprises a "bus acquisition circuit" within each module
and a "priority encoder" circuit on the backplane. Also, as
20 will be described hereinbelow, a priority scheme allows
more urgent requests to be given priority over less urgent
ones. In combination, the "bus acquisition circuit" and the
"priority encoder" circuit determine the sequence in which
requests for a path on the bus are granted.

25

Priority Schemes for the Disclosed Embodiment

1. Positional Priority

Each slot in the backplane of the system using the S-
30 bus has a physical location number. In the disclosed
embodiment, modules connected to slots having lower
location numbers have priority over modules connected to
slots having higher location numbers; with slot 0 having
the highest priority. This feature is referred to as
35 positional priority. When configuring a system which uses
an S-bus, processors should be given high positional
priority, i.e. be placed in low-numbered slots to avoid

system degradation which results when a processor must wait for the bus.

2. Overlays on the Positional Priority Scheme

5 When a module makes a request to acquire a path on the bus, its circuitry specifies the priority of the request as follows:

 a) high priority - the bus acquisition circuitry will grant each bus acquisition request which specifies a high
10 priority before granting lower priority requests;

 b) round-robin priority - modules in lower positional slots, i.e. those having higher positional priority, have priority over those in higher positional slots, but each may acquire a path only for one bus operation cycle until
15 all modules requesting round-robin priority have had access. This priority scheme is provided because a high-performance processor can monopolize the bus during periods of high memory usage and cause lower priority processors to stall if they cannot acquire the bus. Thus, round-robin
20 priority permits modules to share the bus more equally; and

 c) simple priority - I/O channels normally operate with low priority because they buffer data and can normally tolerate some delay without penalty. However, when extended delays cause the I/O channel buffer capacity to be
25 exceeded, an operation will be aborted. To avoid this, and the resulting lose of data, the I/O channel will make a high priority bus acquisition request to avoid losing data.

 Within any priority level, positional priority determines which request is granted first. But, within the
30 round-robin priority level, acquisition requests are selectively enabled to provide access to rotate through the connected modules.

BUS ACQUISITION CIRCUITRY

FIG. 2 shows bus acquisition circuitry for the disclosed embodiment. This circuitry is duplicated in the system for the F-path and the T-path. Module bus acquisition request circuit (MBARC) 30 is present within each module that interfaces to the S-bus. MBARC 30 generates appropriate bus acquisition request signals and keeps track of whether the module has acquired the bus.

Priority encoder circuit 40 is part of the system backplane. It responds to bus acquisition request signals generated by all the modules connected thereto by boards plugged into the backplane. It determines which module is to be given access to the path and then it distributes that access information to all the modules connected thereto.

Nomenclature:

Signals are labelled with a postfix minus to indicate the complement of a signal which is "active" when the condition named by its mnemonic is true.

<u>logic</u>	<u>true signal</u>	<u>complement</u>
	SIG	SIG-
signal active	logic 1	logic 0
signal inactive	logic 0	logic 1

In the disclosed embodiment, signal voltages are assigned positive-true logic levels:

<u>logic</u>	<u>voltage levels</u>
logic 0	low voltage (0.0 volts to 0.8 volts)
logic 1	high voltage (2.0 volts to 5.5 volts)

In FIG. 2:

(a) D flip-flop 309 within MBARC 30 generates bus request signal RQ to request acquisition of the path in response to a "set request" signal on lead 305 from the module upon which MBARC 30 is situated. For each path, every module is allotted one bus request signal.

(b) JK flip-flop 308 within MBARC 30 generates acquisition master signals MINE and MINE-. When MINE is true, the module has been given access to the path. The module whose request for a path acquisition has been granted is known as the path's master. During each bus operation cycle, there is only one master for each path, therefore the "MINE" flip-flop, i.e. flip-flop 308, must be reset in the MBARC of all modules which are not the master.

(c) JK flip-flop 310 within MBARC 30 generates round-robin priority path acquisition signals RREN and RREN-.

(d) D flip-flop 311 within MBARC 30 generates high-priority path acquisition signal HPEN in response to a "set high priority" signal on lead 306 from the module upon which MBARC 30 is situated.

Each of the above-described flip-flops is triggered by clock signal SCLK which is generated on a terminator card connected to the backplane. SCLK is subsequently transferred to all modules through the backplane in a manner to be described hereinbelow.

Note that MBARC 30 shown in FIG. 2 uses all the priority features describes hereinabove. However, a module need not use all the priority features generally available. For example, any module not requiring high priority bus access may delete the corresponding circuitry, starting with D flip-flop 311.

On each module: (a) MINE is applied to NAND 301; (b) MINE- is applied to NAND 302; (c) RQ is applied to NANDs 302-304 and to AND 312-314; (d) RREN is applied to NAND 303 and to AND 312; (e) RREN- is applied to AND 312; and (f) HPEN is applied to NAND 304 and to AND 314. The outputs of

NANDs 301-304 from all modules connected to the S-bus are wire-ored in the backplane, in priority encoder circuit 40, to produce signals KEEP-, REQ-, RREQ-, and HREQ- on leads 341-344, respectively. If no module has made a bus

5 acquisition request, the signal on a lead is high, whereas if one or more modules have made a bus acquisition request, the signal on the lead is low. In the disclosed embodiment, KEEP-, REQ-, RREQ-, and HREQ- are open collector signals, and each of the lines 341-344 is connected to a "pullup
10 resistor."

RREQ-, HREQ-, KEEP- and REQ- are each terminated by 150 ohm 2% resistors 361, 362, 351 and 352, respectively, to a +5 volt source (not shown). These resistors and their connections to the 5 volt source appear on a card called
15 the S-bus terminator card. The terminator card is connected to the backplane and is described in further detail hereinbelow. Since the open collector signals are not driven above the logic threshold, the rise time is determined by the RC constant of the circuit. The use of a
20 5 volt pullup improves the rise time of the open collector signal to a level above the logic threshold.

The wire-or of the output from NAND 303 on each module, i.e. round robin priority request signal RREQ-, is applied to AND 312 on each module. The wire-or of the
25 output from NAND 304 on each module, i.e. high priority request signal HREQ-, is applied to ANDs 312-313 on each module. Further, on each module, the outputs from ANDs 312-314 are applied to NOR 315. The output from NOR 315 is
30 module bus acquisition request signal RQn-, where n is the slot number on the backplane.

RQn- is applied to bus allocation positional priority circuit 316 which, in the disclosed embodiment, comprises a priority encoder designated 381 and a decoder designated 382. Bus acquisition request signal RQn- is active low and
35 pullup resistor 371, taken to be 1000 ohms in the disclosed embodiment, in circuit 40 assures that the request lines of

Bus acquisition request signal RQn is determined by the circuitry shown in FIG. 2 to be equal to:

```

RQn=(RQ*HPEN)                high priority
                                (AND 314)
+ (RQ*RREN*HREQ-)            round-robin, unless inhibited
                                (AND 313)
+ (RQ*RREQ-*HREQ-*RREN-)    simple, unless inhibited
                                (AND 312)

```

Positional priority encoder 316 has one input, i.e. RQn-, and one output, GRn-, for each S-bus slot. Positional
15 priority encoder 316 determines which module has the highest priority request pending and grants it control of the path for the next cycle by means of "bus path grant" signal GRn-. The "bus path grant" signals of all other modules remain inactive, i.e. high, and the other modules
20 must wait. As shown in FIG. 2, a "bus path grant" signal GRn- for each slot is sent back to the corresponding module on lead 391 to indicate whether the path was acquired for the next bus operation cycle. The number of slots in the disclosed embodiment is 22, hence, encoder 381 comprises
25 three 74F148 priority encoders and decoder 382 comprises three 74F138 decoders. Further, each RQn- signal presents one 74F unit load plus 1K ohms from pullup 371 at the encoder input, and each GRn- can drive 20ma at 0.4 volts.

In addition, REQ-, produced by the wire-or of the outputs from NAND 302 and KEEP-, produced by the wire-or of the outputs of NAND 302 are used to pass the path to another module for the next bus operation cycle or to keep it for the module which presently has access. A module may want to retain access to the path for more than one bus operation cycle because a particular operation may require more than one bus operation cycle to be completed. For

example, some memory reads or writes in the disclosed embodiment require more than one bus operation cycle.

When a module gains access to a path, it sets signal MINE which is output from JK flip-flop 308. If the module
 5 needs to maintain the path for another bus operation cycle it sets a "more data" signal on lead 382, which signal is applied to NAND 301 along with MINE. The output of NAND 301 is wire-ored on the backplane to form signal KEEP- which is then applied to NAND 321. The output of NAND 302, to which
 10 RQ, requesting a bus path acquisition, and MINE- have been applied, is wire-ored on the backplane to form signal REQ-. REQ- is inverted in inverter 320 and applied, along with KEEP-, to NAND 321. The output of NAND 321, signal PASS-, is applied, along with "bus path grant" signal GRn-, to NOR
 15 385. PASS- is also applied, along with the output of NOR 385, to NOR 386. When "more data" on lead 388 is set and MINE is set, the module retains control of the path, notwithstanding the fact that the "bus path grant" signal for another module has been made active.

Thus, in summary, the following signals are generated by the modules, in conjunction with the backplane, in order to acquire each path on the bus --The prefix for a signal
 20 indicates a signal generated for T-path or F-path acquisition:

25 Bus acquisition signals:

	<u>T-path</u>	<u>F-path</u>	
	TRQn-	FRQn-	bus path acquisition request signal;
30			-one signal per path for each module
	TGRn-	FGRn-	bus path grant signal;
			-one signal per path for each module

35

5	TKEEP- FKEEP-	keep bus path for the next cycle signal; -wire-or of signals from each module (open collector in disclosed embodiment)
10	TREQ- FREQ-	request bus path signal; -wire-or of signals from each module (open collector in disclosed embodiment)
15	TRREQ- FRREQ-	round-robin request signal; -wire-or of signals from each module (open collector in disclosed embodiment)
20	THREQ- FHREQ-	high priority request signal; -wire-or of signals from each module (open collector in disclosed embodiment)

Step 1: The module does not have the bus
and wishes to acquire it.

MINE is inactive, i.e. equal to 0 and MINE- is equal to 1

(a) A regular priority request for the bus is made by setting RQ, output from flip-flop 309, to active, i.e. 1.
Thus the output of gate 302, signal REQ-, will be 0, i.e. active.

$$REQ = RQ * MINE-$$

(b) In addition to setting RQ from flip-flop 309 to be active, a round-robin request for the bus is made by setting RREN from flip-flop 310 active.

$$RREQ = RQ * RREN$$

(c) In addition to setting RQ from flip-flop 309 to be active, a high priority request for the bus is made by setting HPEN from flip-flop 311 active.

$$\text{HREQ} = \text{RQ} * \text{HPEN}$$

5 HREQs have priority over RRENs and both have priority over other "simple" requests. These signals operate by inhibiting, and thus delaying, requests from modules with a lower level of priority. When HREQ is active, only high-priority requests are enabled. When RREN is active, simple requests are disabled. Since the "round-robin" control line RREN permits processors to share the bus more equally, processors typically request their first operation with both REQ and RREN. This gives them priority over simple requests. When several processors have pending requests, RREN stays active until each gets one operation. A processor which requests a second cycle while RREQ is still active must assert only REQ. This is a low priority request. When RREQ goes inactive, all processors may again use RREQ. Memory modules use RREQ to share F-path equally.

10 Processor messages and direct I/O should also use RREQ.

Step 2: The module has the bus and wishes to retain it for the next cycle.

25 MINE is active, i.e. equal to 1. When an operation requires more than one bus operation cycle, "more data" is set to be active, i.e. equal to 1.

$$\text{KEEP} = \text{MINE} * \text{"more data"}$$

30 The master of a path, i.e. the module having MINE=1, asserts KEEP during operations which require two or more cycles. This prevents loss of the bus in the middle of an operation. However, KEEP must not be used to retain the bus for consecutive operations. KEEP must be inactive during idle cycles and the last cycle of any operation.

35

Step 3: The bus is passed to a new master

KEEP is inactive, i.e. equals 0. REQ from this module is inactive, i.e. equal to 0, because RQ is inactive, i.e. equal to 0 and $REQ = RQ * MINE-$. However, since some other module is requesting the bus, and REQ is the wire-or of the REQ signal from all the boards, REQ for the bus is active, i.e. equal to 1. Thus, PASS is active, i.e. equal to 1.

PASS=KEEP-*REQ

As a result, the bus is lost and MINE is set to inactive.

LOSEBUS=PASS*GRn-

Here GRn- is inactive, i.e. equal to 1 because this module did not make a bus request, and therefore, no grant is made. Note, that unless another request is made, this module does not lose the bus. This is because REQ would be inactive, i.e. equal to 0 and hence, PASS would be equal to 0.

Step 4: The bus is obtained.

PASS is active, i.e. equal to 1, because KEEP is inactive and KEEP- is therefore equal to 1. REQ is active and equal to 1, and therefore, PASS is active, i.e. equal to 1. In addition, GRn- is active, i.e. equal to 0, because the bus is to be given to this unit or module.

GETBUS=PASS*GRn

GETBUS sets flip-flop 308 so that MINE is active and it sets flip-flop 310 so that RREN is active.

SUMMARY: Open Collector Bus Signals:

In the disclosed embodiment, each of the two bus paths uses 6 open collector signals: four for acquisition control and two for acknowledge. The acquisition control signals are driven by each module requesting a path on the bus. The outputs from each module are logically wire-ored, i.e., a

signal is active low if any board asserts it. The acknowledge signals are usually driven only by the slave, however, a parity error or an improper configuration can result in several modules responding at once.

5 The lines transmitting acknowledge signals TAK1, TAK0, FAK1, FAK0 and acquisition control signals TKEEP, FKEEP, TREQ and FREQ are terminated with 150 ohms to 5 volts on the backplane by the S-bus terminator card. The 5 volt pullup improves rise time to above the logic threshold, for
10 example 1.6 volts is a typical logic threshold for 74F logic obtained from Fairchild Camera and Instrument Corporation, Digital Products Division, hereinafter referred to as Fairchild. These lines need not be exactly matched to bus impedance, and a diode clamp to 0.6 volts
15 (not shown) reduces negative overshoots.

 Bus acquisition signals TRREQ, FRREQ, THREQ and FHREQ use 150 ohm 2% pullup resistors to provide fast rise times. In addition, Fairchild 74F38 buffer gates are required to drive these signals fast enough to overcome the extra delay
20 involved in driving inputs to priority encoder 316 and to Fairchild 74F64 NOR gate 315.

 FIG. 3 shows S-bus backplane 29 and a portion of terminator card 50 for T-path 2.

 Signals input to and output from backplane 29 are
25 generally collected above designation 1717. For purposes of discussion, let us focus on a module board installed at slot 0 of backplane 29. The 32 T-path data signals T31:00 are input to 32 pins on backplane 29, the pins all being represented in FIG. 3 by pin 501. The 32 lines on backplane
30 29 that connect T-path 2 to pin 501 are all represented in FIG. 3 by line 601. Each of the 32 lines represented by line 601 is terminated by a 150 ohm resistor, the resistors all being represented in FIG. 3 by resistor 651. Resistor 651 is connected to a +3 volt source (not shown) on
35 terminator card 50. In similar fashion, T-path signals TFN4:0, TID7:0, and TPAR4:050 are input to backplane 29 at pins represented by pins 502-504, respectively, and each of

the lines connecting T-path 2 to pins 502-504 is represented by lines 602-604, respectively. Each of the lines represented by lines 602-604 is terminated by a 150 ohm resistor represented by resistors 652-654, respectively, which resistors are connected to a +3 volt source (not shown).

In addition, the six open collector signals, TAK1:0, TKEEP-, TREQ-, TRREQ-, and THREQ- are applied as input signals to pins represented by pins 505-509, respectively. The lines on backplane 29 connecting T-path 2 to pins 505-509, respectively, are represented by lines 605-609, respectively. Each of the lines 605-609 is terminated by a 150 ohm resistor represented by resistors 655-659, respectively. Each of the resistors 655-659 is connected to a +5 volt source (not shown) on terminator card 29.

Bus path acquisition request signal TRQn- from the module board is applied as input to pin 510 and is terminated by a 1000 ohm pullup resistor (not shown) on terminator card 50. The TRQn- signals from all the modules are input into priority encoder 381. The outputs from priority decoder 382 are applied to the module boards as TGRn- signals. This is shown illustratively for slot 0 in FIG. 3 by the connection between the output of priority decoder 382 and pin 511.

Circuit 50 shown in FIG. 3 represents a portion of the S-bus terminator card referred to hereinabove. It provides the proper pull-up resistors for the open collector signals discussed above. In addition, as described hereinbelow with respect to Fig. 4, the terminator card contains circuitry for generating clock signals which are distributed to a clock distribution board.

FIG. 3 also shows the interface, through the backpanel, between clock distribution board 55 and the modules (clock distribution board 55 being responsible for distributing clock signals to the modules). As shown in FIG. 3, clock signals OSC- and CLK- are distributed to the module board in slot 0 at pins 498 and 499, respectively.

S-BUS CLOCKS

Bus timing is synchronous with a 10.00 megahertz (mHz) timing signal, i.e. clock SCLK. SCLK is generated on each module from two timing signals which are distributed radially on the S-bus backplane. The two timing signals are 20.00 megahertz oscillator clock signal OSC- and 2.000 megahertz oscillator clock signal CLK-. More specifically, in the disclosed embodiment, (1) OSC- is a 20.00 $\pm$ 0.001% mHz oscillator having a 50 nsec period with less than 2 nsec jitter and a 50% $\pm$ 10% duty cycle at its output connector and (2) CLK- is a 2.00 mHz signal, derived from OSC-, which is active for one OSC period (50 nsec) every 500 nsec.

In the disclosed embodiment, bus timing is taken relative to the falling edge of OSC- on the backplane --the rising edge of OSC- is imprecise and should not be used for any bus operation. Timing slew, defined to be the difference in timing between separate clock signals, from OSC- to SCLK must be minimized.

Clock Distribution:

All S-bus timing is derived from a single 20mhz crystal oscillator. Its output is buffered and radially distributed to each board in the S-bus by means of a terminator card. Radial cables minimize clock slew and waveform degradation.

The clock distribution is shown in FIG. 4. All signals are referenced to the falling edge of OSC- at the backplane pin 498. For the disclosed embodiment, it is preferable that slew be less than $\pm$ 6 nsec between boards in a chassis and less than $\pm$ 10 nsec between chassis.

In the disclosed embodiment, oscillator 401 on S-bus terminator card 50 provides clocks to its chassis using a coax cable and a 4-pin connector. Oscillator 401 can also

provide clocks to a second chassis via coax cable. To minimize slew, the electrical length of the cables should be equal.

Signal OSC, output from oscillator 401, is inverted in inverter 420, for example a Texas Instrument 74AS1000 buffer, to provide OSC-. OSC- is then applied to divide-by-2 circuit 403. One output from divide-by-2 circuit 403 is applied to divide-by-5 circuit 404, for example a Fairchild 74F163 synchronous presetable binary counter, to provide, in conjunction with NAND 421, 2 MHz signal CLK-. Signal CLK- comprises one 50 nsec pulse every 100 nsec and is used to synchronize the S-bus at 100 nsec cycles (10mhz). NAND 421 is repeated once for each chassis which needs CLK-.

OSC is inverted in inverter 408, for example a Texas Instrument 74AS1000 buffer, to provide OSC-. CLK- and OSC- are applied to clock distribution board 55 by coaxial cable 467.

Clock OSC- is received on clock distribution board 55 by buffer 410 which, in turn, drives buffer 411. One input to buffer 411 is ground, applied through inverter 409, for example a Fairchild 74F04 hex inverter. This reduces the electromagnetic interference if no module board is plugged into the slot driven by buffer 411. Buffer 411 drives one module board with OSC- at pin 498. Thus, buffer 411 is repeated, for example by buffer 412, for each module board that is driven.

Clock CLK- is input into JK flip-flop 477 on clock distribution board 55 along with the output of buffer 410. The output from JK flip-flop 477 is buffered in buffer 413, for example a Texas Instrument 74AS1000 buffer. Buffer 413 drives three module boards with CLK- at pins 499, 469 and 479, respectively. Thus, buffer 413 is repeated, for example by buffer 414, for each three module boards that are driven. In the disclosed embodiment, traces are less than 6 inches and are not terminated.

In the disclosed embodiment, each S-bus module board, like module board 89, may put one 74F load on CLK- with a

maximum of 1 inch trace stub length. Each S-bus module board, like module board 89, may put a maximum of 5 74F loads on OSC- with a maximum of 10 inches of trace stub length terminated by a 100 ohm resistor. Signal OSC- from backplane 29 is inverted in inverter 450 on module board 89 and applied, along with signal CLK- from backplane 29, to JK flip-flop 406 (or to logic that acts like a JK flip-flop) to produce module clock signals SCLK and SCLK- on lines 491 and 492.

TRANSFER OF INFORMATION USING THE S-BUS

Notation

Signals are represented by mnemonics composed of capital letters, the postfix numbers identify individual signals within a field. A field is labelled, along with its bit range, by using a colon, i.e., T31:00 represents the 32 bit signals T31 through T00. The most significant bit is listed first. A postfix minus indicates the complement of a signal which is "active" when the condition named by its mnemonic is true.

A "word" is 32-bits wide, a "halfword" is 16-bits wide, a "byte" is 8-bits wide and a "nibble" is 4-bits wide. Binary values are represented in hexadecimal notation, thus, each nibble has a value of 0 to 9 or A (=10) to F (=15).

"Power-of-two" notation is used to number bits within a field. Thus, T31 is the most significant bit in the data field, TFN4 is the most significant bit in the function field, TID7 is the most significant bit in the ID field, and so forth. Bit 0 is the least significant bit in a field. The weight of each bit "n" is 2^n , i.e. 2 to the power n. However, bytes and halfwords are numbered "left to right," i.e. byte 0 is the most significant and byte 3 is the least significant.

SIGNAL DEFINITIONS

T-path and F-path signals are identical. All T-path signals have "T" prefixes and all F-path signals have "F" prefixes.

5

T0-path	FROM-path	Signal Definition
---------	-----------	-------------------

Message fields:

T31:00-	F31:00-	32-bit data field
TFN4:0-	FFN4:0-	5-bit function select
10 TID7:0-	FID7:0-	8-bit unit identification number

Parity bits on Message:

TPAR4-	FPAR4-	even parity bit for ID and FN fields
15 TPAR3-	FPAR3-	even parity bit for data bits 31:24
TPAR2-	FPAR2-	even parity bit for data bits 23:16
TPAR1-	FPAR1-	even parity bit for data bits 15:8
20 TPAR0-	FPAR0-	even parity bit for data bits 07:00

DEFINITIONS OF THE CODES OF THE FUNCTION FIELD

On the T-path and the F-path, the 5-bit function field specifies what is on that path during each cycle. For a memory write or an I/O write, the data cycles must immediately follow the address cycle. Other data cycles, such as the memory read and I/O read, are separate.

Function "0" specifies an idle cycle during which there is no transfer. The other fields, including parity, are ignored. As a consequence, the master must drive the function code to zero during idle cycles and the other fields need not be driven.

T-path Function field Codes

The T-path is used solely to initiate memory operations. TFN4 is 1 for address cycles and 0 for data.

5 For the disclosed embodiment:

		TFN4:0	Mnemonic	Operation
	00	00000	IDL	Idle bus cycle
	01-03		*	(reserved)
10		-----Data-----		
	04	00100	DW0	Data word, end at byte 0
	05	00101	DW1	Data word, end at byte 1
	06	00110	DW2	Data word, end at byte 2
	07	00111	DW3	Data word, end at byte 3
15	08-0F			(reserved)
		-----Read-----		
	10	10000	MR1	Memory Read: 1 word
	11	10001	MR2	Memory Read: 2 words
	12	10010	MR3	Memory Read: 3 words
20	13	10011	MR4	Memory Read: 4 words
		-----Write-----		
	14	10100	MW1	Memory Write: 1 word
	15	10101	MW2	Memory Write: 2 words
	16	10110	MW3	Memory Write: 3 words
25	17	10111	MW4	Memory Write: 4 words
		-----Special-----		
	18	11000	MRS	Memory Read and Set
	19	11001	MRR	Memory Read and Reset
	1A	11010	MRI	Memory Read and
30	Increment			
	1B	11011	MRD	Memory Read and
	Decrement			
	1C	11100	MRW	Memory Read and Write
	Word			
35	1D	11101	MWD	Memory Write Diagnostic
	1E-1F		*	(reserved)

F-path Function field Codes

The F-path is used for memory read data, direct I/O, broadcast interrupts, and messages between units. The four "broadcast" functions are received and decoded by all units whereas the memory read data, direct I/O, and the four "message" functions are decoded only by the unit addressed by FID7:0.

10		FFN4:0	Mnemonic	Operation
	00	00000	IDL	Idle bus cycle
		-----Direct I/O Response-----		
	01	00001	ATN	Attention (compatible I/O interrupt lines
15	02	00010	IOK	I/O acknowledge, data okay
	03	00011	IER	I/O acknowledge, data error
		-----Memory Response-----		
	04	00100	MDC0	Memory Read, word 0, disable cache
20	05	00101	MDC1	Memory Read, word 1, disable cache
	06	00110	MDC2	Memory Read, word 2, disable cache
	07	00111	MDC3	Memory Read, word 3, disable cache
25	08	01000	MOK0	Memory Read, word 0, data correct
	09	01001	MOK1	Memory Read, word 1, data correct
30	0A	01010	MOK2	Memory Read, word 2, data correct
	0B	01011	MOK3	Memory Read, word 3, data correct
	0C	01100	MERO	Memory Read, word 0, data error
35	0D	01101	MER1	Memory Read, word 1, data error

OE	01110	MER2	Memory Read, word 2, data error
OF	01111	MER3	Memory Read, word 3, data error

5 -----Direct I/O-----

10	10000	IOR	Input/Output Read
11	10001	*	(reserved)
12	10010	IOW	Input/Output Write
13	10011	FDAT	F-path Data follows IOW or MSGn

-----Broadcasts-----

14	10100	PRE	"Pre-empt" broadcast interrupt
15	10101	DVA	"Delete Virtual Address" broadcast interrupt
16	10110	SYNC	"Start real time clock" broadcast interrupt
17	10111	*	(reserved) broadcast interrupt

20 -----Messages-----

18	11000	MSG0	Message 0
19	11001	MSG1	Message 1
1A	11010	MSG2	Message 2
1B	11011	MSG3	Message 3
25	1C-1F	*	(reserved)

IOW and FDAT must be sent as pairs of consecutive items, each MSGn must be followed by two FDAT items and the other items may be sent singly.

30 IOK and IER are responses to IOR and IOW commands.

MOKn, MERn and MDCn are responses for memory read operations. The responses for a multiple word read operation will usually occur on successive cycles, but this is not required.

35 The responses IOK, IER, MOK, MER, and MDC send one item to acknowledge completion of a direct I/O or memory

operation. All direct I/O is acknowledged, this signals the processor to proceed.

For IOR, read data is returned on F31:00.

Memory reads are acknowledged with the read data
5 returned on F31:00.

Each word of a quadword memory read is sent, MOK, MER or MDC, and acknowledged separately. The memory module may retain the bus for all four words.

For memory writes, there is no response on the F-path,
10 i.e. the processor does not wait.

ELECTRICAL SPECIFICATIONS

The S-bus uses four types of signals. As has been
15 described above, the logic signals are driven, OSC-, CLK-, TGRn-, FGRn-, or received, TRQn-, FRQn-, by circuits on the backplane. Bus signals connect to all module boards on the S-bus. The backplane terminates the bus lines, but does not send or receive signals.

20

S-bus Logic Signals:

1. OSC- and CLK- clocks are radially distributed to each module;

25 2. Bus request signals TRQn- and FRQn- are generated on each module board, and, in the disclosed embodiment, are Fairchild 74F logic signals; and

3. Bus grant signals TGRn- and FGRn- are generated on the S-bus backplane and are distributed to each module
30 board, and, in the disclosed embodiment, are Fairchild 74F logic signals.

S-Bus Signals:

1. Tri-state bus signals
T31:00-, TFN4:0-, TID7:0-, TPAR5:0-
35 F31:00-, FFN4:0-, FID7:0-, FPAR5:0-

2. Open collector bus control signals

TKEEP-, TREQ-, TRREQ-, THREQ-, TAK1:0-

TKEEP-, TREQ-, TRREQ-, THREQ-, TAK1:0-

In the disclosed embodiment, the logic used conforms
5 to the Fairchild 74F logic levels and loading, i.e.:

Bus Line Receivers: Input-High > 2.0 volt

with I-IH < 0.04ma

-Low < 0.8 volt

with I-IL < 0.6 ma

10 Bus Line Drivers: Output-High > 2.7 volt @ 1 ma

-Low < 0.5 volt @ 20 ma

Tri-State Bus Signals:

15 As described above, the F-path and the T-path each
uses 50 tri-state bus signals. These bus signals are driven
by the module that is presently the master of the bus path
synchronously with SCLK at 100 nsec cycles. In the
disclosed embodiment, these tri-state bus signals are
20 preferably driven and received with edge-triggered D-type
registers. Each slave receives the bus signals during every
bus operation cycle and decodes its address or ID.

In the disclosed embodiment, the master must drive the
function signal lines, TFN for the T-path or FFN for the F-
25 path, during every bus operation cycle for which it is the
master of the bus path. If the module does not use the bus
path, it must drive the function code to zero, i.e. high.
The lines for the other fields are allowed to float high to
save power since the bus termination dissipates significant
30 power only for low signals. Since tri-stated lines rise
slowly, they float high too slowly to guarantee an adequate
logic high; so the other fields are ignored.

The tri-state signals, as shown in FIG. 3, are
terminated through 150 ohms to 3 volts on the S-bus
35 terminator card. A series-pass voltage regulator (not
shown) is used to produce 3 volts on the terminator card.
This termination has an a.c. response similar to a split

resistor terminator, but dissipates little power for inactive (high) signals.

Terminator power for T-path or F-path for the resistors shown in FIG. 3 is given by:

5	50 tri-state bus lines * 20 ma = 1000 max,	500 average
	2 fast Open Collector * 35 ma = 70	35
	i.e. RREQ- and HREQ-	
	4 slow Open Collector * 22 ma = 90	45
	i.e. KEEP-, REQ-, and AK1:0	
10	TOTAL	1160 max, 580 average

In addition, a diode clamp (not shown) on terminator card 50 to 0.6 volts reduces negative excursions of the open collector signals.

The bus is terminated at one end only because double-ended termination requires too much drive current and dissipates substantially more power. Signal edges are absorbed by the terminator, but reflect at the far end. Thus, bus timing allows 20 nsec (> 2nsec/ft * 1.5 ft * 4) for settling. Thus, the timing for the bus is determined from the following:

20	delay SCLK to drivers disabled	15 nsec typical
	delay SCLK to drivers disabled	25 nsec maximum
	delay SCLK to drivers enabled	30 nsec typical
	delay SCLK to drivers enabled	45 nsec maximum
25	bus settling time	20 nsec maximum
	bus setup time	25 nsec maximum
	clock skew	10 nsec maximum
	total	100 nsec

Tri-state drivers provide cleaner, faster switching speeds than open-collector. However, severe noise problems can result whenever two opposing drivers are enabled simultaneously. A slight overlap of a few nsec is acceptable if the drivers are properly decoupled. Every driver should have an adjacent 0.1 uf decoupling capacitor. Any overlap increases system noise, but comparable current spikes occur when switching a high capacitance bus. However, consistent overlaps greater than 5 nsec must be

avoided. Each module board must reduce tri-state conflict by delaying bus enable versus bus disable by 10 nsec.

Positional Priority Signals:

5

As discussed hereinabove, each positional priority encoder such as encoder 316 in FIGs. 2 and 3, has one input, TRQn- or FRQn-, and one output, TGRn- or FGRn-, for each path for each S-bus module slot. In the disclosed embodiment, as previously described, each encoder circuit 316 comprises three Fairchild 74F148 priority encoders and three Fairchild 74F138 decoders. The maximum delay from any RQn- to any GRn- is 20.5 nsec. Each RQn- signal provides an input load of one Fairchild 74F unit, e.g. NOR 315 in FIG. 2, plus a 1K ohm pullup resistor, e.g. resistor 371 in FIG. 2. The 1K ohm pullup resistors hold unused inputs inactive high. Thus no jumper changes are needed when boards are inserted or removed.

10

•••
•••
•••
•••
•••
•••
•••

15

Bus path acquisition request and bus path grant signal timing is given by the following:

•••
•••
•••

20

delay SCLK to RREQ-, HREQ-	15 nsec max
settling time	30 nsec max
delay RREQ-, HREQ- to RQn-	10 nsec max
delay RQn- to GRn-	20 nsec max
settling time GRn- to LCLK	15 nsec max
clock skew	10 nsec max
total	100 nsec

•••

25

•••
•••

Bus Capacitance:

30

Bus connections have high distributed capacitance. This affects signal transmission by lowering the characteristic impedance and slows the rise time on the open collector signals.

35

per board: multiwire 3pf/inch x3" = 10pf, x 1.5" = 5pf			
input	74F	5	74F 5
output	74F	5	74F38 10
connector, pins, etc		5	5
5 TOTAL per board		25	25
x 22 boards =		550pf	550pf
backpanel: multilayer 2 pf/inch x 17" = 35pf			
TOTAL per line		585pf	585pf

10 Hold Times:

Data is transferred from a source register to a destination register during one bus operation cycle. The worst case delays and necessary setup times determine the cycle time (100 nsec nominal). In effect, clock skew shortens the cycle to 100-10 = 90 nsec.

Hold times are more difficult since the clock skew is comparable to the minimum delay path. Extra timing margin is desirable. Input registers are clocked on every cycle.

20 Output clocks are gated and may have an extra gate delay.

Clock Skew:

OSC- is radially distributed to each board in the system. It has a nominal 50% duty cycle, +- 10% or 5nsec. The high and low periods can nominally vary between 20 and 30 nsec. However, pulse width becomes less certain with each buffer. Only the falling edge of OSC- may be used for timing.

30

MEMORY OPERATIONS OVERVIEW

S-bus memory comprises one or more memory modules, each containing a controller and an array of semiconductor memory chips. Modules are selected by memory address using decoders on each board. Different size and speed modules can be mixed on the same bus.

35

There are three types of memory operations which can be performed in a memory module: read, write and special. The special operations combine a read-modify-write operation into a primitive command.

5

Interleaving:

Burst transfers from or to memory are divided into quadword block transfers. Each memory module interleaves
10 four rows of dynamic RAM on quadword boundaries to match bus bandwidth for a quadword transfer. Dynamic RAMs require a precharge time before they can be accessed again. The cycle time for a quadword read for any row, including a precharge time of 200 nanoseconds (ns), is 800 ns for the
15 disclosed embodiment. The memory module activates another row while a row that had just been accessed precharges. This means that successive quadword accesses are overlapped by 200 ns, thereby reducing the effective cycle time for these operations to 600 ns. However, one module cannot
20 match the bandwidth of both buses. Also, byte, halfword, and special operations are slower than bus speed.

Improved bandwidth is possible when there are two or more memory modules to share the work. These memory modules share the bus most effectively when bus operations
25 alternate between modules. Thus, two or four modules can be interleaved by quadwords. Burst transfers access each module cyclically, with successive quadwords being sent to different modules in turn. Module interleaving is determined by address decoders in each module.

Each module contains an input buffer which stores
30 addresses and data received while the module is busy performing previous operations. The buffer improves performance by allowing more effective sharing of the bus. However, buffer size is limited to provide adequate access
35 times for high-priority read operations --in the disclosed embodiment, operations already in the buffer are completed first.

The buffer preserves the sequence of operations to any given memory location. Otherwise, memory values could be uncertain. Operations to different addresses proceed in any order; the queue on any module is independent of the other modules.

Address Space:

32-bit memory addresses are transferred on T-path.
This provides a 4-gigabyte real address space.

Memory Read Operations:

The four memory read commands MR1, MR2, MR3, and MR4 read 1 to 4 words from memory without altering the contents of memory. MR1 reads one fullword, but it can be used for byte and halfword reads as well. Bytes and halfwords have the same alignment on the bus as in memory, i.e. the byte select bits of the data address, A01:00, are ignored by the memory modules.

MR4 reads an entire quadword, whereas, MR2 and MR3 read two or three words from the same quadword. Words are selected by memory address bits 03:02. The addressed word is transferred first; the remaining words are transferred in cyclic order: 0, 1, 2, 3, 0, 1, 2, etc. The 2-bit word address is returned with each word as part of the F-path function code.

The sequence of a read operation is:

```

bus-op
30 cycle [ 1 ][ 2 ][ 3 ][ 4 ] ... [ 5 ][ 6 ][ 7 ][ 8 ]
      [   ][   ]<---memory-read----->
T-pth[ rq ][ A  ][   ][ ak ] [   ][   ][   ][   ]
F-pth[   ][   ][   ][   ] [ rq ][ D  ][   ][ ak ]
cycle 1. source module, e. g. a processor, acquires T-path
35      (rq)
  
```

cycle 2. address cycle (A) - the source module,
e.g. processor:

(1) puts read function on TFN4:0
(MR1, MR2, MR3, MR4

5 (2) puts source unit ID onto TID7:0
(3) puts memory address onto T31:00

All memory modules latch T-path data every cycle. The source module then releases the bus after only the address (A) has been sent.

10 cycle 3. each memory module decodes the memory address to determine if it contains the addressed word. If not, it takes no further action. Parity is checked. The selected memory module begins the memory operation.

15 cycle 4. the selected memory module acknowledges (ak) the read message.

The memory module reads the requested data from the memory array and prepares to send it to the source module which requested it.

20 cycle 5. The memory module acquires the F-path (rq)

cycle 6. data transmission cycle (D) - the memory module:

(1) puts response function on FFN4:0

"MOKn" or "MDCn" indicates the data is valid

"MERn" indicates an uncorrectable memory
error

25 "n" = 0, 1, 2 or 3 indicates which word this is, within a quadword block. "n" is bits A03:02 of the address for the word on the bus.

30 (2) puts the source unit ID latched in cycle 3 onto FID7:0

(3) puts a data word on F31:00

The memory module releases F-path during the last data word transferred. Usually, all requested words are

35 transferred on consecutive cycles. However, memory data errors can slow or interrupt the transfer. The memory controller in the disclosed embodiment is allowed to

request the F-path before it has checked the data for validity. If there is an error, the next cycle is wasted and the corrected data word is sent on a later cycle.

- Each module, such as processors and so forth, latch data on the F-path during every cycle.
- 5 cycle 7. each module, such as processors and so forth, match FID to its own ID. If unequal, the module takes no further action. The source module checks the data for proper parity.
- 10 cycle 8. the source module acknowledges the item with either "here" or "fault." In the disclosed embodiment, the source modules are not allowed to respond "busy" or "not present" to memory operations they request. Any response other than
- 15 "here" is a malfunction.

Cycles 6 to 8 are repeated for each additional word accessed. For example, an MR4 transfers four words on F-path:

cycle ... [5][6][7][8][9][10][11]
 read---->]

20 F-path [rq][D0][D1][D2][D3][][]
 ack. [][][][d0][d1][d2][d3]

Memory Write Operations:

- 25 Memory write operations write byte, word, or multi-word data into memory. The data word(s) immediately follows the address on the T-path. F-path is not used.

- 30 The sequence of a write operation is:
- cycle [1][2][3][4][5][6][7][8]
 [][]<---memory-write-----> quadword---
- T-path [rq][A][D1][D2][D3][D4][][]
 ack. [][][][a][d1][d2][d3][d4]
- 35 D2, D3 and D4 are used only in quadword operations
- The sequence of a write operation is:

- cycle 1. source module, e. g. a processor, acquires T-path
(rq)
- 5 cycle 2. address cycle - this is the same as in a read
operation except that the function code specifies
a write: MW1, MW2, MW3, or MW4. The source module
retains the bus by asserting TKEEP.
- cycle 3. data cycle - the source module:
(1) puts write data onto T31:00
(2) puts its unit ID onto TID7:0
10 (3) puts a data code on TFN4:0, i.e. DW0, DW1,
DW2, or DW3

During the last data cycle of the normal memory
writes, function DWn specifies that writing ends on byte
"n." This is used for byte, halfword or other partial word
15 writes.

Function code DW3 must be used for other write data
cycles and for all special memory data on T-path.

Meanwhile, each memory module decodes the memory
address to determine if it contains the addressed word. If
20 not, it takes no further action. Parity is checked. The
selected memory module begins the memory operation.

cycle 4. The selected memory module acknowledges the
address cycle.

25 cycle 5. The selected memory modules acknowledges the data
cycle.

For MW2, MW3, and MW4, additional data is sent in
cycles 4, 5, and 6. Each item is checked and decoded on the
cycle after it is sent; then acknowledged on the following
cycle: 6, 7, and 8.

30

Multiple-word Write Operations:

The multiple-word operations (MW1-MW4) write a string
of data into a memory quadword. The string begins at the
35 addressed byte. Lower bytes, i.e. to the left, in this word
are not written. During the last word, the function code
TFN4:0 selects the last byte to be written.

Words are cyclic within a quadword. Writing begins at the addressed word and continues in cyclic order: 0, 1, 2, 3, 0, 1, 2, and so forth. The entire quadword can be written starting with any word.

- 5 Partial quadword writes are useful for string operations and for burst transfers which begin or end on non-aligned addresses. Data bytes are aligned on the bus as in memory. Unused bytes are ignored except for parity checking. When only one word is transferred, the address and TFN both limit which bytes are written.
- 10

<u>First word control</u>			<u>Last word control</u>		
<u>address</u>	<u>bytes written</u>		<u>TFN4:0</u>	<u>function</u>	<u>bytes written</u>
T01:00=00	0,1,2,3		DW0	00100	0
15 01	1,2,3		DW1	00101	0,1
10 10	2,3		DW2	00110	0,1,2
11	3		DW3	00111	0,1,2,3

Bytes written when only one word is transferred (MW1):

<u>Function code with data word</u>					
<u>address</u>	<u>DW0</u>	<u>DW1</u>	<u>DW2</u>	<u>DW3</u>	
...00	0	01	012	0123	
...01	none	1	12	123	
...10	none	none	2	23	
25 ...11	none	none	none	3	

Byte and Halfword Operations:

- 30 Byte and halfword operations are performed as partial word writes with one data word, MW1; they are special cases of partial word writes, as described above. The memory address specifies the first byte written; DWn specifies the last byte. The other bytes are not altered. Bus timing is the same as a fullword write; memory timing uses longer
- 35 read-modify-write cycles because the error correction codes are computed on fullwords. Each byte to be written must be properly aligned within the data field; the memory

controller does not shift data bytes. Unused bytes are ignored.

Byte Operations				Halfword Operations			
Byte	T01:00	TFN4:0		Halfword (bytes)	T01:00	TFN4:0	
5	0	00	00100 DW0	0	0,1	00	00101 DW1
	1	01	00101 DW1				
	2	10	00110 DW2	1	2,3	10	00111 DW3
	3	11	00111 DW3				

10 Special Memory Operations:

Special memory operations facilitate multiprocessor interaction. These operations are indivisible primitives which operate as read/modify/write cycles. They combine write and read data transfers on the S-bus. The original unmodified memory value is read and sent on F-path. Several types of operations are supported in the disclosed embodiment:

1. bit operations MRS and MRR set or reset any one bit within a data word. The number of this bit is selected by the write data value T04:00.
2. Exchange words MRW swaps a word with memory. The addressed word is read and returned on F-path; then the T-path data word is written.
3. Semaphore operations MRI and MRD increment or decrement the addressed word. The T-path data word is not used. By convention in the disclosed embodiment, all write messages are at least two words long.

The sequence of any special operation is:

```

30 eyc[ 1 ][ 2 ][ 3 ][ 4 ][ 5 ]...[ 6 ][ 7 ][ 8 ][ 9 ]
    [    ][    ]<-----read/modify/write cycle----->
    Tph[ rq ][ A  ][ DW ][ak-A][ak-D]
    Fph[    ][    ][    ][    ][    ]...[ rq ][ DR ][    ][ ak]

```

35 T-path operates as a memory write and F-path operates as memory read.

Test-and-Set Operations:

Test-and-Set (T&S) memory operations are used to coordinate activity within multi-tasking systems. T&S provide race-proof interlocks for allocating access to devices, tables, or other resources. The memory system performs an indivisible operation which: reads the interlock word, tests the specified bit, then sets it, and writes the word. This function is performed by the MRS command.

DIRECT TRANSFERS ON F-PATH OVERVIEW

In addition to memory read data described above, the F-path is used for inter-processor messages, broadcast interrupts, direct I/O and compatible I/O.

Interprocessor Messages:

Inter-processor "messages" transfer interrupts and three data words from any processor or channel to another designated processor or channel. Such messages are used by system designers to coordinate processors in multiprocessor systems, to initiate channel operations, and to reschedule tasks when I/O operations are completed.

In the disclosed embodiment, channels perform input/output tasks according to "channel command blocks" (CCB) stored in memory. CCB's contain information for selecting devices, functions, memory buffers and so forth connected with a particular I/O operation. The processor sends a message to an I/O channel to initiate the operation of a CCB. The message includes the CCB's address in memory. The channel then accesses the CCB using memory operations described hereinabove and performs the required functions. Data is transferred directly to or from memory. The channel can chain data buffers and commands under the direction of

the CCB. The channel reports its status to the processor by sending messages thereto.

In the disclosed embodiment, the interprocessor messages are: (a) "start I/O" command which is sent from a processor to a channel to initiate a CCB, (b) "halt I/O" command which is sent from a processor to a channel to halt execution of a CCB, (c) "end I/O" interrupt which is sent from a channel to a processor to interrupt the processor after a CCB is completed, or (d) "processor" interrupt which is sent from a first processor to a second processor. Although the source and destination modules are either processors or channels, these designations are flexible. For example, an I/O processor could accept messages as a channel or a channel could request I/O operations from another channel.

The interprocessor message is three words long:

word	FFN	FID	F31 - F00
1	18	unit-d	level, unit-s, parameter #1
2	13	unit-d	parameter #2
3	13	unit-d	parameter #3

where:

FID, i.e. FID7:0, is the ID of the destination module, unit-d; unit-s is the ID of the source module; level sets the "priority" of the CCB operation or interrupt, for example in the disclosed embodiment 0 is the highest priority; and parameter #1 is a halfword, parameters #2 and #3 are fullwords whose meaning is defined by software convention.

The sequence of a message is:

cycle	[1]	[2]	[3]	[4]	[5]
F-path	[rq]	[MSG]	[D1]	[D2]	[]
ackn	[]	[]	[]	[a-MS]	[a-D1]
cycle 1.	source module requests and acquires F-path (rq)				

cycle 2. source unit puts word 1 on the F-path (MSG):
 (1) puts function code MSGn on FFN4:0
 (2) puts the destination unit ID on FID7:0
 (3) puts level, source unit ID and parameter #1
 5 on F31:00

All processors and channels latch F-path data every cycle

cycle 3. source unit puts word 2 on the F-path (D1)
 (1) puts function code FDAT on FFN4:0, "data
 10 word"
 (2) puts destination unit ID on FID7:0
 (3) puts parameter #2 on F31:0

cycle 4. source unit puts word 3 on the F-path (D2)
 (1) puts function code FDAT on FFN4:0, "data
 15 word"
 (2) puts destination unit ID on FID7:0
 (3) puts parameter #3 on F31:0
 (4) destination module acknowledges word 1.

cycle 5. destination module acknowledges word 2.

cycle 6. destination module acknowledges word 3.

Broadcast Interrupts:

Broadcasts are interrupts which are sent
 25 simultaneously to all processors within an S-bus system.
 Processors receive all broadcasts regardless of their ID.
 Two broadcast interrupts used in the disclosed embodiment
 are "Pre-empt" (PRE) and "Start Real Time Clock" (SYNC).
 Broadcasts are not acknowledged on the F-path. Thus,
 30 processors must use the broadcast when it is received, i.e.
 they cannot respond with a "busy" acknowledge to request
 that the message be repeated.

The sequence of a broadcast interrupt is:

35	cycle	[1]	[2]	[3]	[4]
	F-path	[rq]	[FN]	[]	[]
	ackn	[]	[]	[]	[]

cycle 1. source unit requests and acquires F-path (rq)

cycle 2. BROADCAST: the module

(1) puts function code (PRE or another defined codes)

5 on FFN4:0

(2) puts zero in FID7:0

(3) puts the broadcast message on F31:0

All processors and channels latch F-path data every cycle.

10 cycle 3. All processors decode the broadcast message.

Broadcasts are not acknowledged.

"Start Real Time Clocks" is used to synchronize clocks within a multiprocessor system. In the disclosed embodiment, every processor includes a 64-bit "real time" clock which keeps precise time by counting system bus operation cycles. Once initialized, all clocks within the system will keep exactly the same time. SYNC does not send any data, i.e. bits F31:00 are ignored. However, the sender will place its ID into bits F23:16 for use by bus monitors.

20

The following procedure is used to initialize clocks:

step 1: One processor sends messages to all processors instructing them to stop their real-time clocks and to reload them with the 64-bit time contained in parameters #2 and #3 of words 2 and 3 of the interprocessor message.

25

step 2: the one processor waits for 100 microseconds to provide time for each other processor in the system to acknowledge its message interrupt and to reload its real-time clock.

30

step 3: the one processor issues SYNC on the F-path. This broadcast message is decoded by hardware within each processor and all processors start their clocks within two cycles to provide precise synchronization.

SYNC is a 1-word message whose format is:

35

FFN is x16; FID is sender's ID (used for a broadcast message to be decoded by all processors and channels on the S-bus system); and F23:16 is sender ID.

Direct Input/Output:

5 Direct I/O allows a processor to read or write a word from a selected device or channel. The complexities of channel I/O are avoided, but the processor must wait for a response. Direct I/O is useful for communicating with test equipment.

10 The processor can read (IOR) or write (IOW) data to a selected device. IOR sends one word and IOW sends data in a second word. The destination module responds with either an IOK, the operation was completed successfully, or an IER, error. Read data is returned with IOK.

15 The sequence of direct I/O is:

cyc[1][2][3][4][5] ... [6][7][8][9]

[] [] <---input/output----->

Fph[rq][IOR][][ak][]... [rq][IOK][][ak]
or

20 [rq][IOW][D][ak-W][ak-D]... [rq][IOK][][ak]

cycle 1. processor acquires F-path (rq)

cycle 2. select cycle - the processor:

(1) puts IOR or IOW on FFN4:0

(2) puts destination module unit ID on FID7:0

25 (3) puts subfunction/device on F31:00 as follows:

F31:28 (4-bits) subfunction

F27:24 (4-bits) not used

F23:16 (8-bits) sender unit ID

F15:00 (16-bits) channel/device number

30 cycle 3. data write cycle - IOW only - the processor

(1) puts FDAT on FFN4:0 "data word"

(2) puts destination unit ID on FID7:0

(3) puts write data on F31:00

cycle 4. destination module acknowledges select cycle (ak)

35 cycle 5. destination module acknowledges data write cycle, IOW only (ak-W)

[I/O unit does a data transfer]

cycle 6. destination module acquires the F-path (rq)
cycle 7. IOK or IER cycle - the destination module:
 (1) puts IOK or IER on FFN4:0
 (2) puts processor unit ID on FID7:0
 (3) puts data on F31:00 - response to IOR only
5 cycle 8. processor decodes and checks item received
cycle 9. processor acknowledges (ak)

Compatible I/O:

10 Compatible I/O operations are provided in the
disclosed embodiment by a "Direct Memory Interface" (DMI)
board to provide compatible 3200-series MUX and EDMA I/O
buses. A processor uses direct I/O operations to access the
DMI and a subfunction determines which specific MUX-bus
15 operation is performed.

Multiple S-bus Systems:

20 Two or more S-bus systems can be interconnected to
operate together. Processors can access memory in other
systems by using real addresses or by sending messages to
other modules using unit IDs.

25 Two S-buses are connected by a system cable which
symmetrically joins "system-bus expansion" board (SBX)
interfaces in each S-bus. Each S-bus can have a separate
power supply, diagnostic system and clock distribution
means.

30 An SBX monitors both the T-path and the F-path for
operations addressed to the remote system. On the T-path,
an SBX responds to selected memory addresses which are
stored, in the disclosed embodiment, as a bit map in a RAM.
For example, using a 64K x 4 bit RAM, one stores four bits
for each 64K byte block in the S-bus 4G byte real address
35 space. On the F-path, an SBX responds to selected unit IDs.
These are stored, in the disclosed embodiment, as a bit map
in a 256 x 4 bit RAM.

Messages received by a local SBX are acknowledged, buffered, and forwarded to a remote SBX. The remote SBX acquires the corresponding remote path and echos the message. The two SBX's reverse roles for operations from the remote system.

SBX operations on T-path and F-path are independent of each other, except for the fact that in the disclosed embodiment they share the same cables. F-path transfers have priority for the cable.

An SBX uses an entire bus operation cycle to decode an address. Thus, it can use fast RAM memories to store a bit map of each unit ID and each 1M-byte memory block to be accessed in the remote system. If desired, the SBX can only allow access to a subset of the remote system. Unit IDs and addresses are not translated by the SBX and messages are echoed without modification.

An SBX uses four unit ID's so that it can overlap memory reads. It substitutes one of these for the ID of each remote memory operation it receives. The original ID is not put on T-path, but is saved and returned with the read data. Thus, the IDs used to access memory do not need to be unique system-wide. The unit ID coming from a memory request is not matched against the ID RAM.

System Cable:

FIG. 1 shows an SBX 110 being connected to a system cable which comprises data paths 117 and 118. Data paths 117 and 118 are similar to T-path and F-path. Messages from S-bus 10 or 20 are sent by SBXs 110 and 210 onto a data path from a connection labelled CT and messages for S-bus 10 and 20 are received by SBXs 110 and 210 from a data path at a connection labelled CF. System cable 107 connects each CT connection of a local SBX, e.g., SBX 110, to the CF connection of a remote SBX, e.g., SBX 210.

The signal definitions are similar to that on the S-bus. An extra signal indicates which path each message is

from: CTT=1 selects T-path; CTT=0 selects F-path. Note that CTT is received by the remote SBX as CFT.

S-cable Signal definitions:

5	CT	CF	
	<u>transmitted</u>	<u>received as</u>	<u>Signal definition</u>
	CT31:00-	CF31:00-	32-bit data field
	CTFN4:0-	CFFN4:0-	5-bit function select
	CTID7:0-	CFID7:0-	8-bit unit ident. no.
10	CTPAR5:0-	CFPAR5:0-	parity bits
	<u>control signals:</u>		
	CTT-	CFT-	T-path (1) or F-path (0) mess.
	CTAK1:0-	CFAK1:0-	Acknowledge
15	CTRDY-	CFRDY-	Ready for more data

Each SBX transmits a continuous clock CTCLK from the CT connection; all items are synchronous with this clock. However, a remote SBX must be able to receive this clock, as CFCLK, asynchronously to its own clocks, because the remote system may operate on a different oscillator. However, an SBX should also be able to operate synchronously when S-buses operate synchronously.

DESCRIPTION OF COMPOSITE MEMORY MODULE

25 Composite Memory Module (CMM) 1000 is a combination storage control module and memory controller on one board for the Model 3280MPS computer system manufactured by Concurrent Computer Corporation. Several CMMs in a computer system may be interleaved on quadword boundaries in either 2-way or 4-way mode.

30 As shown in FIG. 5, CMM 1000 contains 1, 2, or 4 arrays of dynamic memory 601-604, each consisting of 156 dynamic RAMSs (64K x 1 or 256K x 1) organized into 4 rows
35 by 39 bits. This allows for 32-bit data storage with a 7-bit error-check-and-correct (ECC) code that provides single bit error detection and correction, double bit error

detection and some multiple bit error detection. Data storage capacity for CMM 1000 is 1, 2, or 4 megabytes using 64K x 1 dynamic RAMs and 4, 8, or 16 megabytes using 256K x 1 dynamic RAMs.

5 Packaging is accomplished using quad-single-in-line packages, QSIPs, which are ceramic substrates with 22 pins on 0.1 inch centers. Four RAMs are mounted on each and the QSIPs are mounted vertically on CMM 1000. This approach allows for four arrays of memory. It is possible to
10 depopulate this board to either 1 or 2 arrays.

As shown in FIG. 5, memory arrays 601-604 are each 64K x 1 bit dynamic RAMs with a nibble mode. In one embodiment, nibble mode provides high speed serial access of 2, 3, or 4 bits of data. Whenever a RAM is activated, it internally
15 accesses 4 bits of data as selected by the 8 row address bits and the most significant 6 bits of the column address. The 2 least significant bits of the column address, designated A3 and A6, select 1 of the 4 nibble bits for the initial access. The remaining nibble bits can then be
20 accessed by toggling the CAS control signal on lines 605 high and then low while the RAS control signals on lines 605 stay low. Toggling CAS on lines 605 causes A3 and A6 to be incremented internally while all other address bits remain unchanged. If more than 4 bits are accessed during a
25 nibble mode access, the address sequence repeats. If any bit is written during a nibble access, the new value will be read on any subsequent nibble access. 256K x 1 RAM chips produced by some manufacturers provide a slightly different nibble mode than that described above. In one instance, the
30 4 bits of data are selected by the 8 least significant bits of both the row and column addresses, and the most significant bit of both the row and column address are used to select 1 of the 4 nibble bits for the initial access.

Data to be written into memory is supplied to memory
35 arrays 601-604 through write data register (WDR) 610. Data read from memory is taken from memory arrays 601-604 and is buffered through the read data buffer (RDB) 612 and loaded

therefrom into read data register (RDR) 613. The address for the data in memory arrays 601-604 is held in address latch (AL) 614. The memory address is supplied to memory arrays 601-604 by means of address multiplexer (AMX) 616.

5 The 4 rows in each memory array are interleaved on quadword boundaries under the control of signals RAS0:3- on lines 605. Any time a row is activated, up to 4 words can be accessed by taking advantage of nibble mode. This allows high speed quadword access while only having 1 row of
10 memory active.

All commands for memory operations are initiated on CMM 1000 by supplying a master ID, function code, and 32-bit address on the T-path. As described above, transfers on the S-bus are performed at a 10MHz rate which is derived
15 from OSC- and kept in sync by CLK-. OSC- and CLK- are supplied to CMM 1000 over lines 619 to clock drivers 620. CMM 1000 accepts commands by constantly monitoring the 50 leads comprising T-path on line 621. This is done by sampling the T-path at every bus operation cycle with input
20 latch (IL) 622. IL 622 comprises seven octal latches which receive data from the T-path. The output of IL 622, i.e. all 50 input bits, are transmitted to parity check circuit (PCC) 671. PCC 671 computes the parity of the 50 bits and compares it with the transmitted value, TPAR4:0. If an
25 error occurs, a parity error signal is sent to T-path FIFO 624 and to T-path acknowledge (TPA) 673. In addition, 19 bits --T31:20, T05:04, and TFN4:0-- are sent from IL 622 to board select logic (BSL) 672 over line 627. BSL 672 uses bits TFN4:0 to determine whether the command is a memory
30 operation and uses bits T31:20 and T05:04 to determine whether the address of the memory operation is to be found within the memory arrays 601-604 which comprise CMM 1000. If both conditions are true, BSL 672 sends a signal over line 628 to TPA 673. In response, TPA 673 applies the
35 appropriate acknowledge response TAK1:00 to line 625.

In addition, when the function code of the received bus data item signifies that a memory operation is required

and an address match occurs for CMM 1000, 45 bits from SL 622 --T31:00, TFN4:0, TID7:0-- are sent over line 623 to FIFO 624. As described above, any write data, up to 4 words, accompanying a write command will follow on consecutive cycles on the T-path. This subsequently sent write data is also stored in FIFO 624. In the disclosed embodiment, FIFO 624 is 15 words deep and will accept commands until 11 locations are filled. CMM 1000 does not accept commands after FIFO 624 has 11 locations filled because 5 locations in FIFO 624 are required for a quadword write command. When FIFO 624 can no longer accept commands, a signal is sent to TPA 623 and TPA will transmit a "busy" acknowledge response as TAK1:0- on line 625. In the disclosed embodiment, T-path FIFO 624 comprises thirteen 16K x 4 RAMs. The data signals entering FIFO 624 is negative-true and positive-true leaving, i.e. the data signals are inverted.

Command signals in FIFO 624 are fed over line 630 to command latch (CL) 631 to start a memory operation: master unit ID TID7:0, memory operation function code TFN4:0 and status information. At the same time, the address of the memory operation in FIFO 624 is fed over lines 701 to address latch (AL) 614.

25

READ OPERATIONS

After CL 631 and AL 614 are loaded with a read command, a row of RAMs in one of memory arrays 601-604 is activated, as selected by CL 631 and AL 614, by signalling array drivers 47 to assert the proper RAS_n- signal on lines 605 low. AMX 616 first receives a row address for the data from AL 614 over line 705. Then, AMX 616 is switched to provide a column address to the arrays, which column address is received from AL 614 over line 706. CAS- is then asserted over line 605. Data is then read out of the selected row in memory arrays 601-604 over lines 635. The data is buffered in RDB 612 and stored in RDR 613. The

contents of RDR 613 are then transmitted to error-check-and-correct circuit (ECC) 640 for checking. If ECC 640 does not detect an error, the data is transmitted for storage to data output register (DOR) 645. F-path control circuit 57 acquires F-path and a bus data item is output over the 50 leads of the F-path by (1) placing the read data in DOR 645 on line 646, (2) placing the ID of the master which requested the operation in DOR 656 on line 660, the ID having been previously forwarded from CL 631 over line 665 to DOR 656, (3) placing the response code from DOR 658 on line 662, and (4) placing parity from DOR 657 onto line 661, the parity having been generated by the parity generation circuit (PGC) within special function gate array 59.

If additional words, up to three more, are to be accessed, signal CAS-, applied over line 605, is toggled by array control line driver 47.

WRITE OPERATIONS

A write operation starts exactly like a read operation through the time signal CAS- is first asserted over line 605. Write data is then sent from FIFO 624 over line 666 for storage in data input register (DIR) 667. The contents of DIR 667 are transmitted to ECC 640. Check bits are generated by ECC 640. The check bits and write data are then stored in WDR 610. Array control line driver 47 then asserts signal WE- to cause the write data in WDR 610 to be stored in memory arrays 601-604. If additional words are to be written into memory, up to three more, these words are taken from FIFO 624, loaded into DIR 667, stored into WDR 610, and written into memory by keeping signal WE- low and toggling signal CAS-.

Byte, halfword, and string write operations are performed using the above-described write operation and by supplying appropriate proper function codes and addresses to CMM 1000 in the incoming bus data item. For the first word

of any write operation, the 2 least significant bits of the address of the command are used to determine how many bytes of the first word are to be written into memory: 00 means write bytes 0-3, 01 means write bytes 1-3, 10 means write
5 bytes 2 and 3, and 11 means write byte 3. Bytes are ordered with byte 0 in data bits 31-24, byte 1 in data bits 23-16, byte 2 in data bits 15-8, and byte 3 in data bits 7-0. For the last word of any write operation, the accompanying function code determines how many bytes of that word are
10 written into memory: 00100 means write byte 0, 00101 means write bytes 0-1, 00110 means write bytes 0-2, and 00111 means write bytes 0-3. The combination of the 2 least significant address bits of the MW1 command along with the function code in the write data portion of the MW1 command,
15 many different partial writes can be performed.

Whenever a part of a word has to be written into memory, CMM 1000 first reads the word at the desired location out of memory into RDR 613, checks for errors, replaces part of the data in RDR 613 with write data from
20 DIR 667, generates new check bits, stores the modified word and the new check bits into WDR 610, and writes the contents of WDR 610 into memory.

SPECIAL OPERATIONS

25 In the disclosed embodiment, CMM 1000 performs 6 special operations. The first 5 of these read data out of memory, place the data on the F-path, modify the data, and write it back into memory.

30 The memory read and set, MRS, and the memory read and reset, MRR, start out exactly like read operations up through the time data is placed on the F-path. At that time, the data is latched into gate array 675. A 5-bit select code is then read from FIFO 624 and sent to gate
35 array 59. This code is derived from T04:00- on the cycle immediately following the MRS/MRR command on T-path. Gate array 59 then either sets or resets 1 of 32 bits in the

latched word. The modified word is then read out of the gate array and into ECC 640, new check bits are generated, the data bits and the check bits are stored in WDR 610. Then, the contents of WDR 610 are written into memory.

5 Memory read and increment, MRI, and memory read and decrement, MRD, are performed exactly like MRS/MRR except for the fact that no 5-bit code is taken from FIFO 624, and gate array 59 increments or decrements the word read out of memory by 1.

10 The memory exchange operation, MEX, is performed exactly like an MRS/MRR except for the fact that instead of using gate array 59 to modify the word read out of memory, a new 32-bit word is placed in DIR 667 from FIFO 624 and this new word is written into memory.

15 The sixth special operation, memory write diagnostic, MWD, does not place any data on the F-path. MWD reads a word out of memory, discards the 32-bit data, supplies a new 32-bit data through DIR 667 from FIFO 624, and writes the new word with the old check bits into memory through
20 WDR 610. This makes it possible to force errors on a subsequent read to the same location.

ECC

25 ECC 640 is implemented in the disclosed embodiment by use of the Texas Instrument TI 74ALS632 32-bit ECC chip. ECC 640 has the ability to detect and correct all single-bit errors in memory, detect all double-bit errors, and detect some multiple-bit errors.

30 If a single-bit error is detected during any memory operation, ECC 640 will latch the bad data off internal data bus 640 from RDR 613, correct the data, store the corrected data in WDR 610, and move data from WDR 610 to RDR 613. In the case of a read operation, new check bits
35 are generated for the new data in RDR 613, the data and the check bits are stored in WDR 610, and the contents of WDR 610 are written into memory. Write and special operations

continue normally after the corrected data has been placed in RDR 613.

When double or multiple-bit errors are detected, CMM 1000 sends the bad data out on the F-path with the proper response code during a read or special operation, except for MWD. In the case of a write operation, when a partial word is being written, CMM 1000 overwrites the addressed location with an all ones pattern. This will force a multiple-bit error condition back to the requesting master while performing a write operation. The data in memory is not modified if a multiple-bit error occurs during a read or special operation. Memory errors cannot occur during a MWD operation because the data read out of memory is not checked.

Clearly, those skilled in the art recognize that further embodiments of the present invention may be made without departing from its teachings.

20

25

30

35

THE CLAIMS DEFINING THE INVENTION ARE AS FOLLOWS:

1. A bus for transmitting data, function codes, module codes and parity information between modules connected thereto which comprises:

at least two paths which comprises:

- (a) 32 path lines for transmitting path data,
- (b) 5 path lines for transmitting path function codes,
- (c) 8 path lines for transmitting path identification codes,
- (d) 5 path lines for transmitting path parity information,
- (e) 4 path acquisition lines for transmitting path acquisition signals, and
- (f) 2 path acknowledge lines for transmitting path acknowledge codes;

means for interfacing modules to the at least one

path;

path acquisition means comprising:

- (a) means, responsive to module signals generated on the at least one module, for generating the path acquisition signals and
- (b) means, responsive to a path module acquisition request signal generated on the at least one module in response to the path acquisition signals, for generating path module acquisition grant signals;

path module acquisition means, disposed on the at least one module, comprising:

- (a) means for generating the path module signals,
- (b) means for generating the path module acquisition request signals in response to the path acquisition signals, and
- (c) means for determining whether the at least one module has acquired the path in response to the path module acquisition grant signals.



2. The bus of claim 1 wherein the path system acquisition means further comprises means for assigning positional priority to the modules and for giving preferential access to the path to modules having higher positional priority.

3. The bus of claim 2 wherein the path module acquisition means for determining whether at least one module has acquired access to the path in response to the path acquisition grant signal comprises means for generating a MINE and a MINE- signal having a first and a second logical value, respectively, if the at least one module has access to the path and having the second and first logical values, respectively, if the at least one module does not have access to the path.

4. The bus of claim 3 wherein the path module acquisition means comprises means, in response to a signal from the at least one module, for generating a bus request signal RG having the first logical value if the module wants to acquire access to the path and the second logical value otherwise.

5. The bus of claim 4 wherein the path module acquisition means comprises means, in response to the MINE- and RQ signals, for generating the "simple bus request" signal which has the second logical value when the at least one module wants to acquire access to the path and the path is not already accessed by the module, i.e., when RQ and MINE- have the first logical value, and which "simple bus request" signal has the second logical value otherwise.

6. The bus of claim 5 wherein the path system acquisition means further comprises means for ORing the "simple bus request" signals from the modules to generate the "composite simple request" signal, REQ-, on one of the first path acquisition lines.



7. The bus of claim 6 wherein the path module acquisition means further comprises a flip-flop for generating the RQ signal in response to a signal from the at least one module.

8. The bus of claim 2 wherein:

- (a) the path module acquisition means further comprises means for generating a "keep access to the path" signal as one of the path module signals; and
- (b) the path acquisition means further comprises means, in response to the "keep access to the path" signals from the modules, for generating a "composite keep access to the path" signal, KEEP-, as one of the path acquisition signals, whereby modules having access to the path and requiring the path to complete a task may get to retain access to the path for the next bus operation cycle.

9. The bus of claim 6 wherein the path module acquisition means further comprises means, responsive to the MINE signal and to a "more data" signal from the at least one module having the first logical value when the module requests a path access for more than a single bus operation and the second logical value otherwise, for generating a "keep access to the path" signal having the second logical value when the module seeks to retain access to the path and the first logical value otherwise.

10. The bus of claim 9 wherein the path system acquisition means further comprises means for ORing the "keep access to the path" signals from the modules to generate a "composite keep access to the path" signal, KEEP-, on one of the path bus acquisition lines.

11. The bus of claim 10 wherein the path module acquisition means further comprises means, responsive to the



KEEP- and REQ- signals, for generating a PASS- signal having the second logical value when any module does not seek to retain access the path and the first logical value otherwise.

12. The bus of claim 11 wherein the path module acquisition means further comprises means, responsive to RQ, for generating the path module acquisition request signal.

13. The bus of claim 12 wherein the path module acquisition means further comprises means, responsive to the PASS- and one of the path module acquisition grant signals, for generating GETBUS and LOSEBUS signals, having the first and second logical values, respectively, when the at least one module has obtained access to the path and the second and first logical values, respectively, otherwise.

14. The bus of claim 13 wherein the path module acquisition means further comprises means, in response to signals GETBUS and LOSEBUS, for generating signals MINE and MINE-.

15. The bus of claim 12 which further comprises means for inverting RQ to provide the path module acquisition request signal.

16. The bus of claim 14 wherein the path module acquisition means further comprises a flip-flop, in response to signals GETBUS and LOSEBUS, for generating signals MINE and MINE-.

17. The bus of claim 1 wherein the path acknowledge codes are open-collector signals.

18. The bus of claim 17 wherein the path data are tri-level signals.



19. A bus path acquisition circuit for modules connected to at least one bus path which transmits data, function codes, identification codes and parity information between modules connected thereto, which bus path acquisition circuit interfaces with a path acquisition circuit which generates path acquisition signals and path module acquisition grant signals, the bus path acquisition circuit comprising:

- (a) means for generating path module signals, in response to which path module signals the path system acquisition circuit generates the path acquisition signals;
- (b) means for generating path module acquisition request signals, in response to which path acquisition signals the path system acquisition circuit generates the path module acquisition grant signals; and
- (c) means for determining whether the module has acquired the path in response to the path module acquisition grant signals.

20. The bus path acquisition circuit of claim 19 wherein the means for determining whether the module has acquired access to the path in response to the path acquisition grant signal comprises means for generating a MINE and a MINE- signal having a first and a second logical value, respectively, if the module has access to the path and having the second and first logical values, respectively, if the module does not have access to the path.

21. The bus path acquisition circuit of claim 20 comprises means, in response to a signal from the module, for generating a bus request signal RQ having the first logical value if the module wants to acquire access to the path and the second logical value otherwise.



22. The bus path acquisition circuit of claim 21 comprises means, in response to the MINE- and RQ signals, for generating the "simple bus request" signal, which signal has the second logical value when the module wants to acquire access to the path and the path is not already accessed by the module, i.e., when RQ and MINE- have the first logical value, and which "simple bus request" signal has the second logical value otherwise.

23. The bus path acquisition circuit of claim 22 wherein the path system acquisition circuit further includes a circuit for ORing "simple bus request" signals from the modules to generate the "composite simple request" signal, REQ-, the bus path acquisition circuit comprising a flip-flop for generating the RQ signal in response to a signal from the module.

24. The bus path acquisition circuit of claim 22 which further comprises means, responsive to the MINE signal and to a "more data" signal from the module having the first logical value when the module requests a path access for more than a single bus operation and the second logical value otherwise, for generating a "keep access to the path" signal having the second logical value when the module seeks to retain access to the path and the first logical value otherwise.

25. The bus path acquisition circuit of claim 24 wherein the path system acquisition circuit further includes a circuit for ORing "keep access to the path" signals from the modules to generate a "composite keep access to the path" signal, KEEP-, the bus path acquisition circuit comprising means, responsive to the KEEP- and REQ- signals, for generating a PASS- signal having the second logical value when any module does not seek to retain access to the path and the first logical value otherwise.



26. The bus acquisition circuit of claim 25 which further comprises means, responsive to RQ, for generating the path module acquisition request signal.

27. The bus acquisition circuit of claim 26 further comprises means, responsive to the PASS- and one of the path module acquisition signals, for generating GETBUS and LOSEBUS signals, having the first and second logical values, respectively, when the module has obtained access to the bus and the first and second logical values, respectively, otherwise.

28. The bus acquisition circuit of claim 27 further comprises means, in response to signals GETBUS and LOSEBUS, for generating signals MINE and MINE-.

29. The bus of claim 1 wherein at least one module further comprises means for generating the acknowledge signals two bus operation cycles after a bus data item was sent on the at least one path.

30. A module for interfacing with a bus path for transmitting data, function, codes, identification codes and parity information between modules connected thereto which bus path includes:

at least one path which comprises:

- (a) 32 path lines for transmitting path data,
- (b) 5 path lines for transmitting path function codes,
- (c) 8 path lines for transmitting path identification codes,
- (d) 5 path lines for transmitting path parity information,
- (e) 4 path acquisition lines for transmitting path acquisition signals, and
- (f) 2 path acknowledge lines for transmitting path acknowledge codes,



- 63 -

the module comprising means for generating the
acknowledge signals two bus operation cycles after a bus
data item was sent on the path.

DATED this 7th day of May, 1991.

CONCURRENT COMPUTER CORPORATION

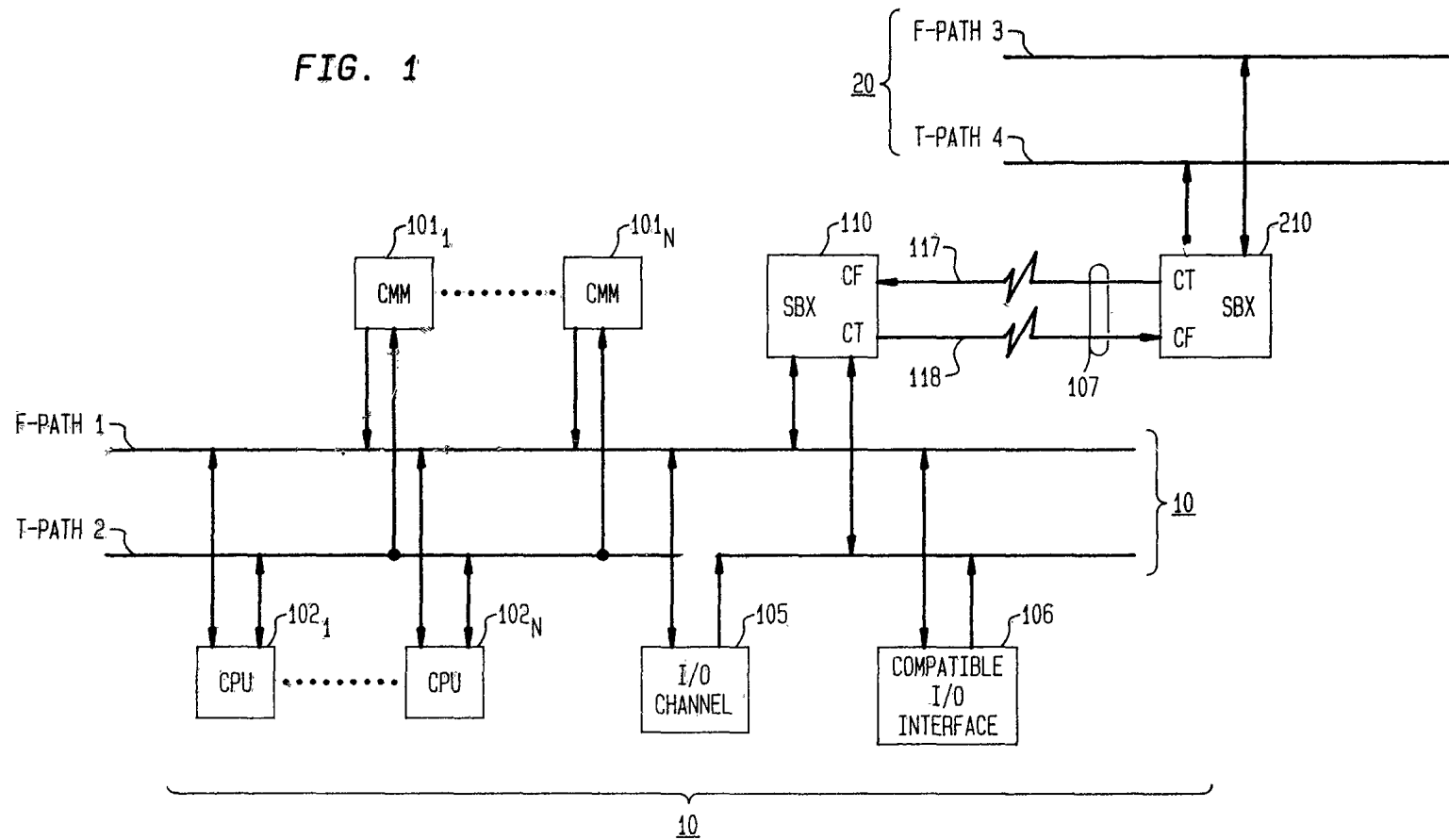
WATERMARK PATENT &
TRADEMARK ATTORNEYS
290 BURWOOD ROAD,
HAWTHORN, VICTORIA,
AUSTRALIA.

IAS:BB(11.21)



7 0 7 7 5 5 7

FIG. 1



1/5

75 257/87

FIG. 2

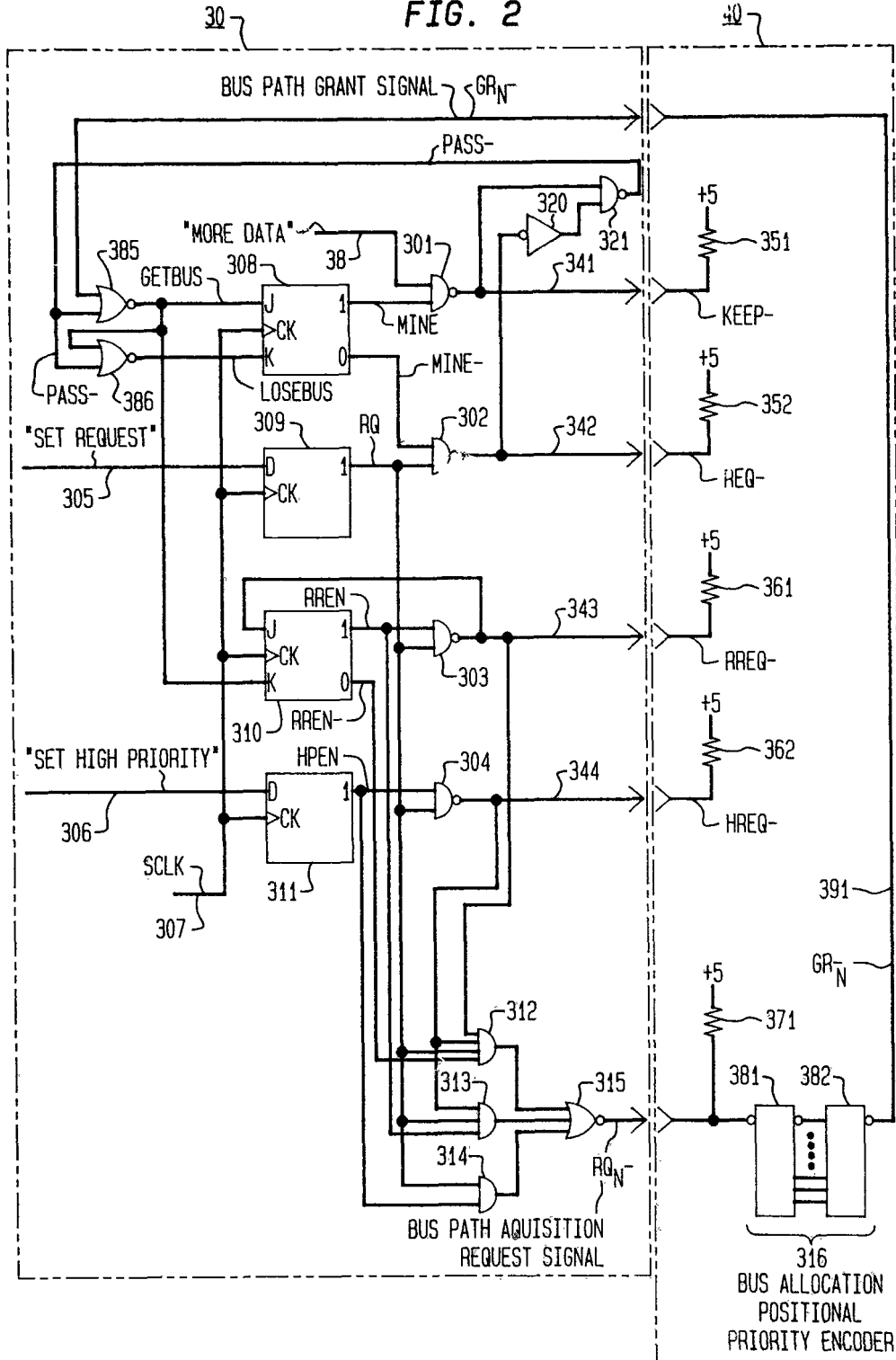
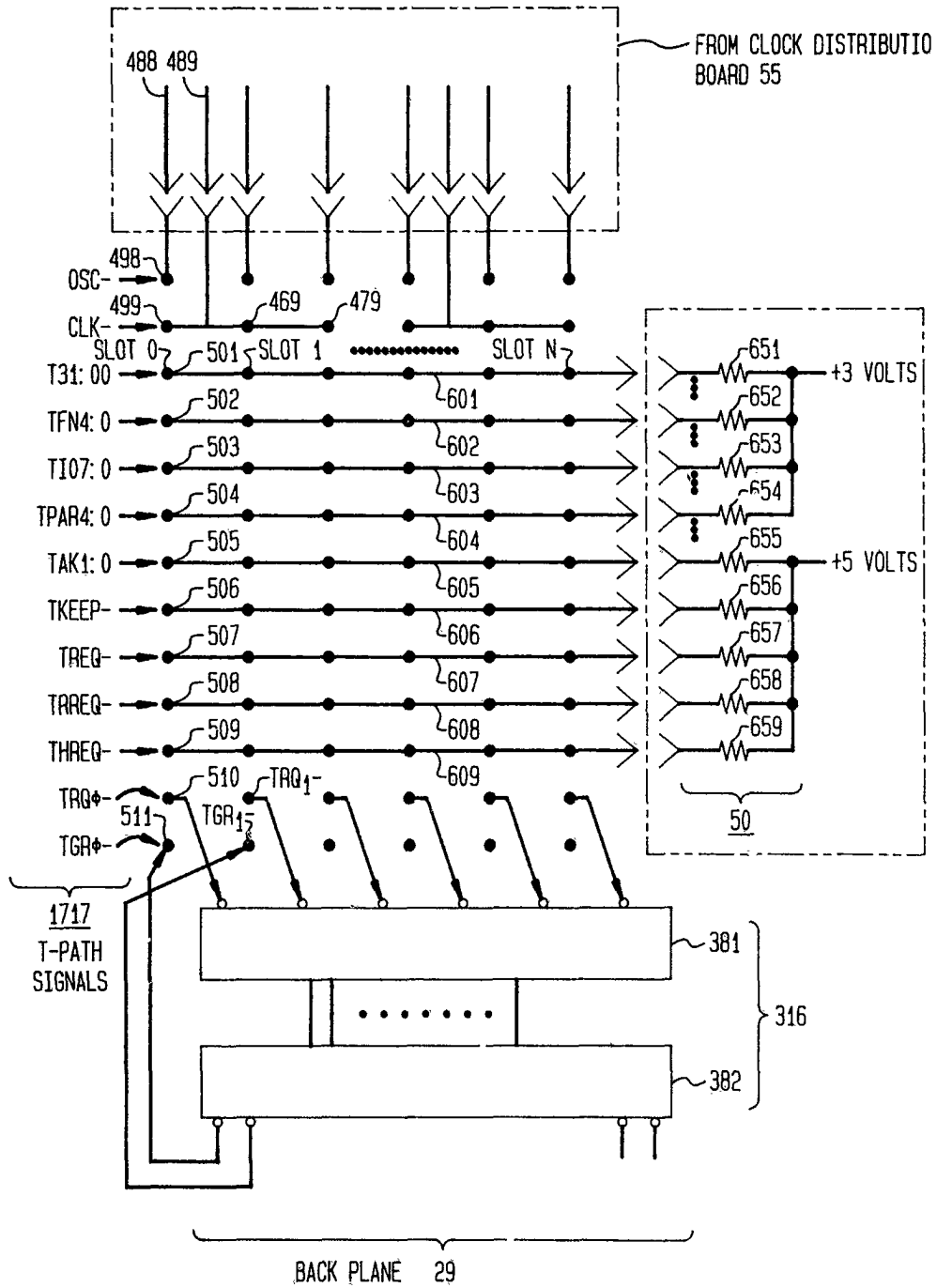
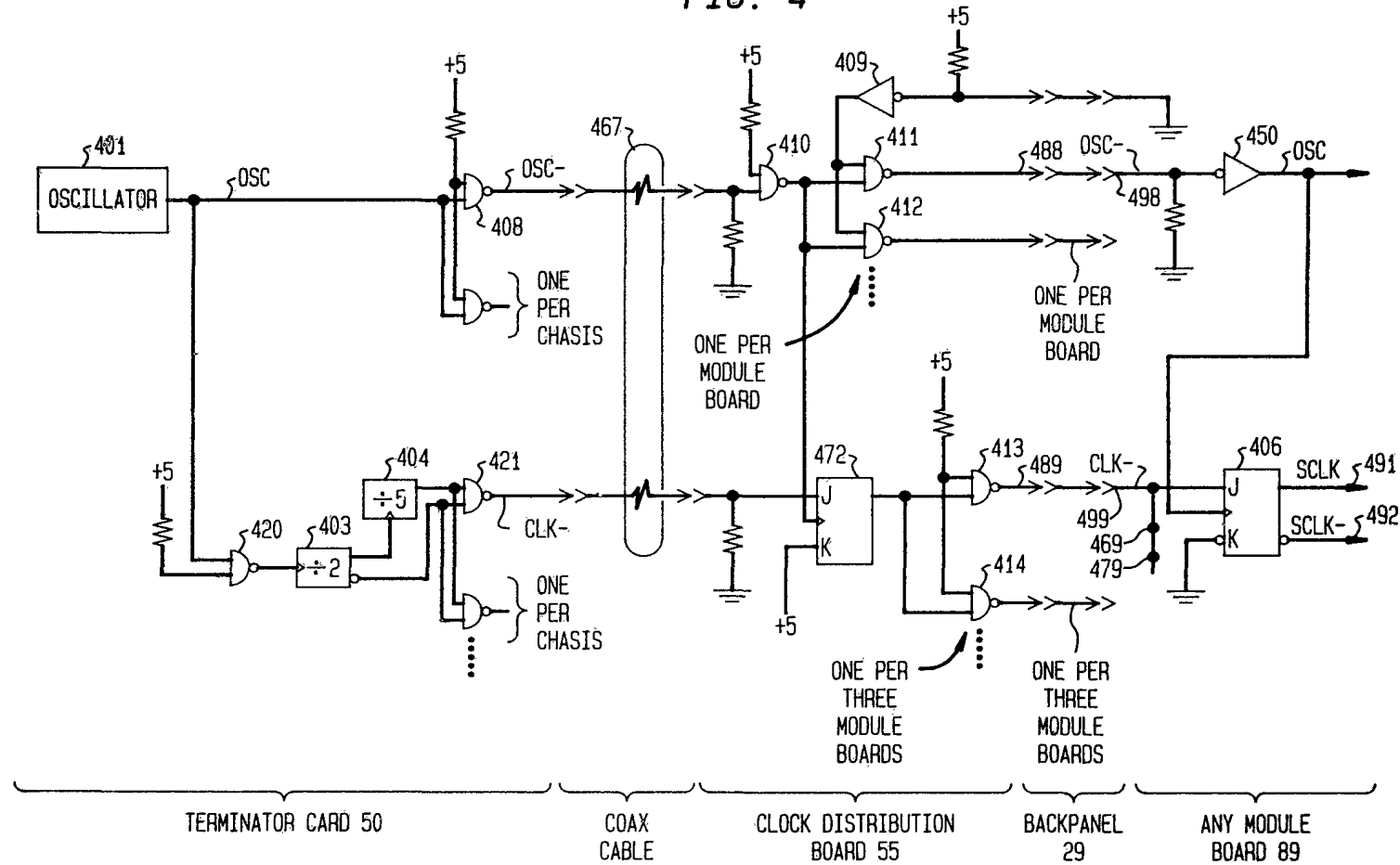


FIG. 3

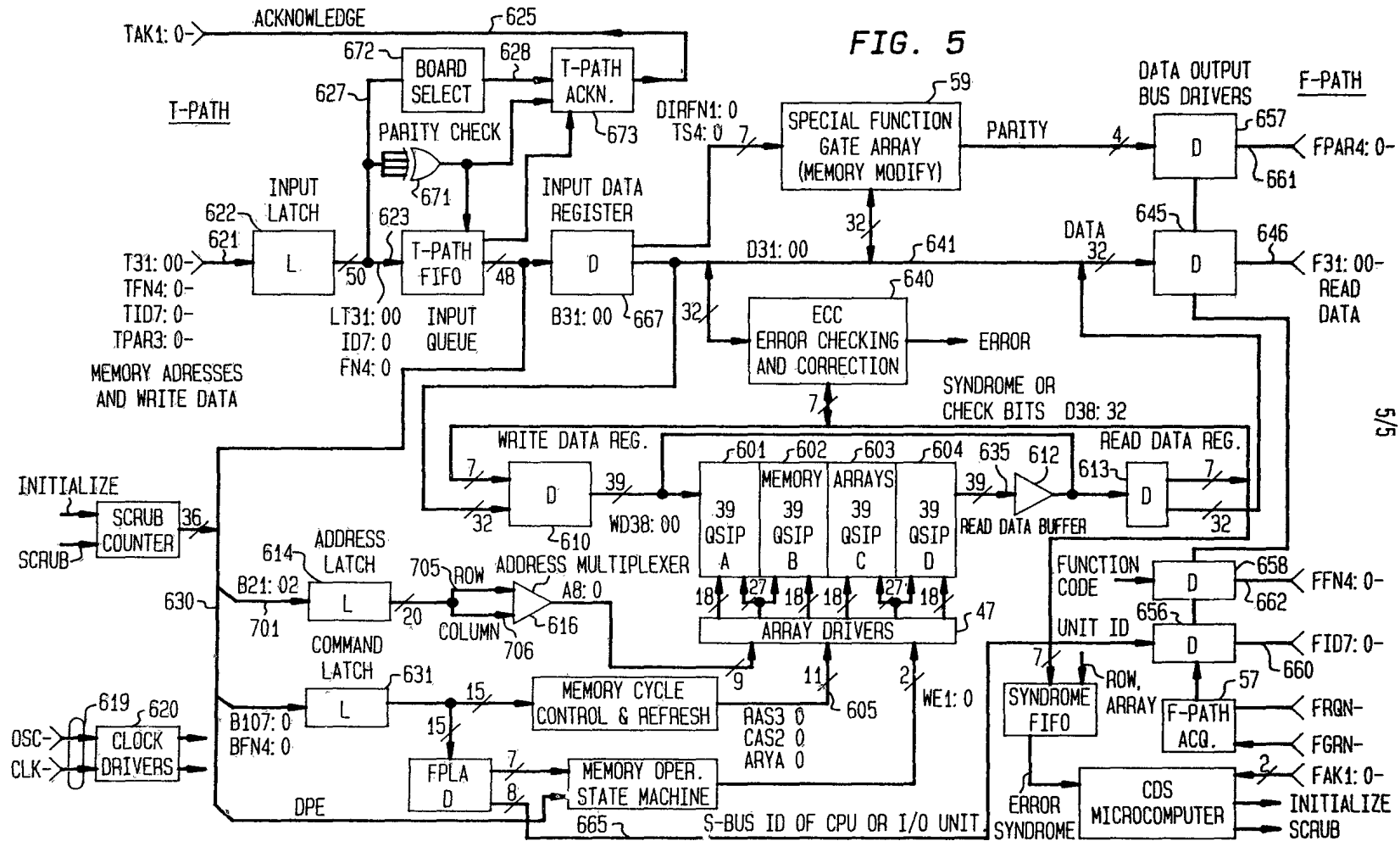


6 7 8 7 7 5 5 7

FIG. 4



0 7 0 7 7 0 0 7



5/5