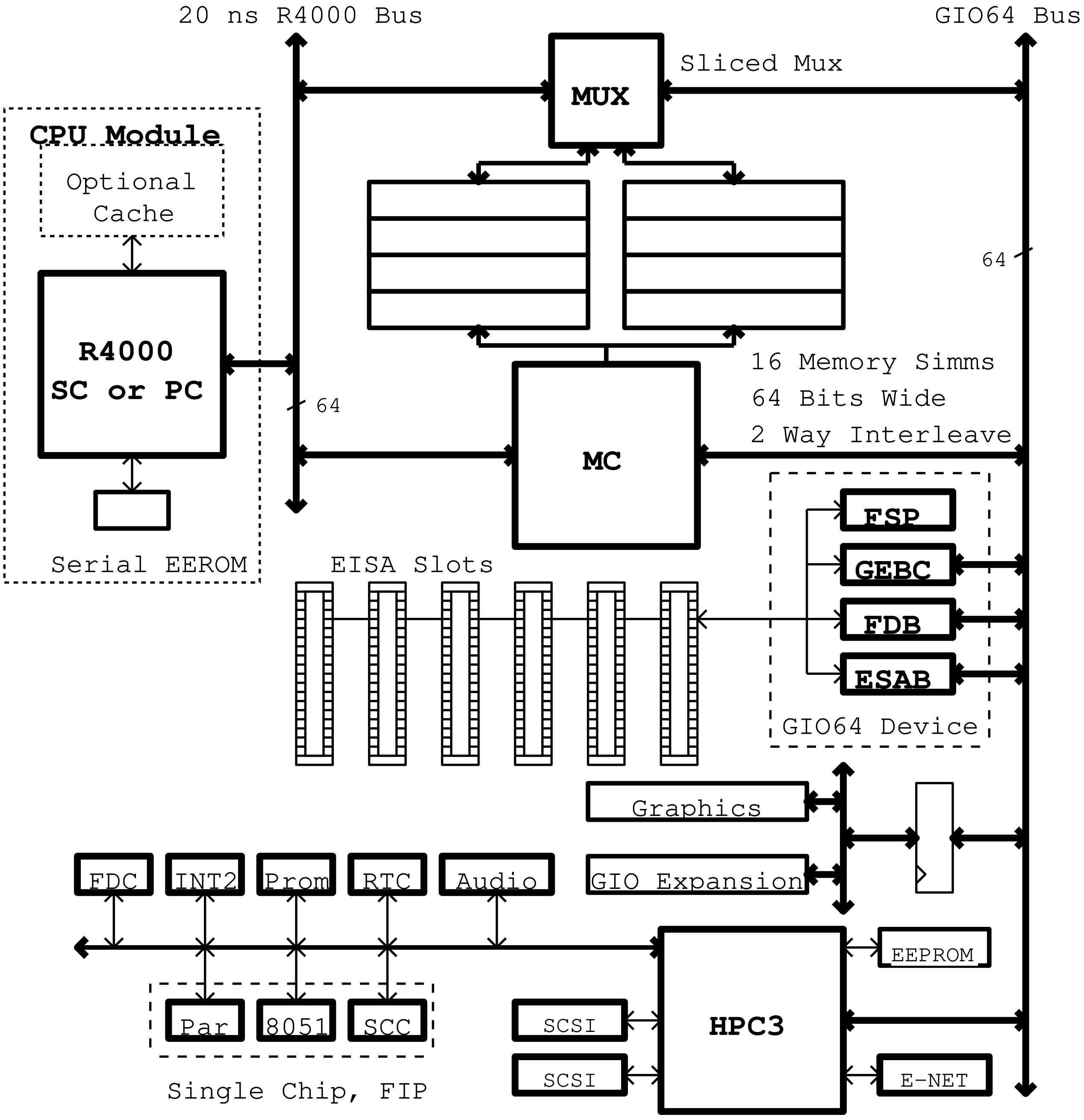
1. Introduction

This document specifies the architecture of the MUX gate array for the Fast Forward project. This array connects the R4000 processor to the the memory system and the GIO64 Bus. Each of the two MUX parts handle 36 bits including parity. A block diagram of the full machine is shown below:



1.1 MUX Block Diagram

The MUX chip is a data path part that is used to move data between the processor, memory, and GIO64 bus. Since the CPU, and GIO64 run at different clock rates the MUX will help solve flow control problems over the asynchronous boundaries. This will be accomplished using a fifo on writes from the CPU. The GIO64 bus will initially run at 33 MHz so that it can be compatible with GIO, but will be able to run at speeds up to 40 MHz. The interface between GIO64 and memory will be completely synchronous to the GIO64 clock while the interface between the CPU and memory will also be synchronous to the CPU clock (50 MHz). The fifo in the MUX will be used as a write buffer on CPU writes to memory and GIO64. A block diagram of the chip is shown on the next page.

There are six different basic operations that are performed by the MUX chip. These are CPU reads and writes to memory, CPU reads and writes to a GIO64 device, and GIO64 reads and writes to memory. Each of there operations will be explained in detail in the following sections.

The MUX chips has two different clocks, the CPU clock and the GIO clock. Some operations are synchronous to the GIO clock and others are synchronous to the CPU clock. The MUX control signals from MC are used for both types of operations so there is a pair of select signals that indicate to the MUX chip which set of operations are being selected and the clock to flop the MUX control signals. All of the MUX control signals are flopped before they are used. The two select signals are cpu_sel and gio_sel. The cpu_sel signals is active for CPU reads and and writes to main memory. The gio_sel is active for the other four MUX operations. The two select signals will never both be asserted at the same time.

1.2 Operation Select

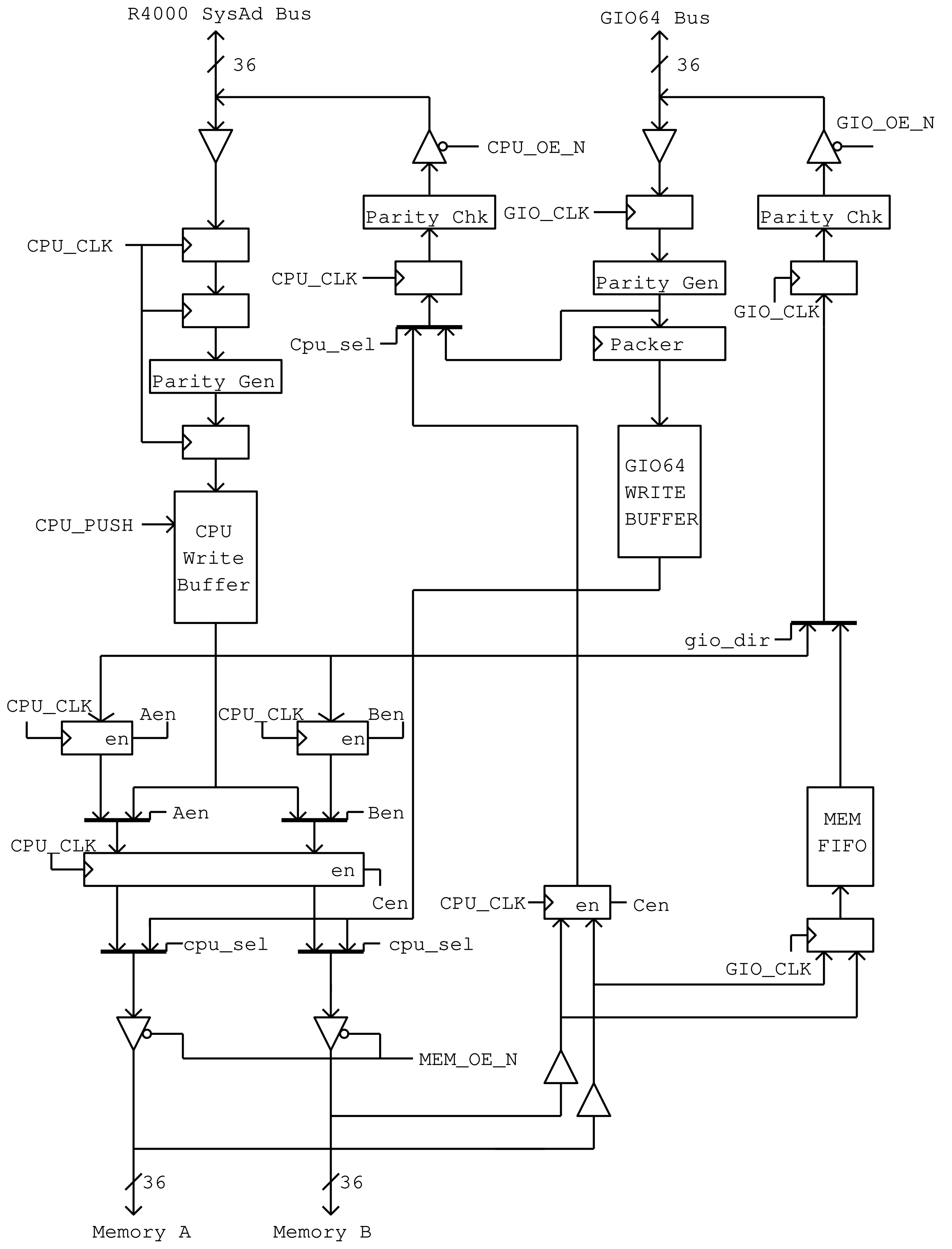
Four control signals: gio_sel, cpu_sel, mux_dir, and aen_mem determine the current MUX operation. Either gio_sel or cpu_sel will be asserted during each of the operations. Cpu_sel is synchronous to the cpu_clk and gio_sel is synchronous to the gio_clk. Below is a chart of the different operations and the source of the bus parity signals:

cpu_sel	mux_dir	aen_mem	Operation
0	0	0	CPU GIO64 device read. Parity is generated.
0	0	1	GIO64 write to memory. Parity is generated.
0	1	0	CPU GIO64 device write. Parity is generated.
0	1	1	GIO64 read from memory. Parity from memory.
1	0	d	CPU memory write. Parity is generated.
1	1	d	CPU memory read. Parity from memory.

During cpu memory reads and writes the aen_mem signals is used for another function.

Note that this document references the input csize64, which is no longer an input to MUX. The csize64 input was ncluded to support R3000 processors, but was removed as an input when R3000 support was no longer needed for Fast Forward machines. The internal core of MUX still contains csize64 references, but this signal is pulled high internally

MUX Block Diagram



1.3 Memory Reads and Writes By the CPU

During memory reads and memory writes by the CPU the cpu_sel signal is asserted and the MUX control signals are clocked with the CPU clock.

1.3.1 Processor Writes to Memory

Processor writes to memory are a two part operation. In the first part the data is buffered in the MUX chip until it can be written into memory. The data comes over the sysad bus and is flopped three times so that the MC chip has time to determine if the data is valid and send a CPU_PUSH signal to the MUX chip. If the data is not valid it will not be put into the fifo. The CPU memory write fifo is 32 entries deep. It is important to only put valid data into the fifo since there is no way to purge data from the fifo except to pop the fifo. The MC chip will keep track of how many data entries are in the fifo. CPU write data can be added to the MUX chip fifo at any time, even when the MUX chip is being used for a different operation. The cpu_sel signals does not have to be asserted to add data to the mux fifo, only the cpu_push signal needs to be asserted. The cpu_push signal is a dedicated signal so that write data can be added to the fifo at any time. This fifo needs to be 32 entries deep so that it can hold two complete 32 word block writes from the CPU, which is the largest block write the CPU can issue.

The csize64 control signal is a static signal that indicates that the processor has a full 64 bit interface. This signal should be connected to the r4k signal from the MC chip. When this signal is not asserted bottom 32 bits of the sysad bus, (sysad(31:0)), are copied to the top bits of the bus, (63:32), between the first and second set of data input flops. This signal is deasserted for an R3000 processor and will duplicate the write data so that it is ready to write into memory. This signal is pulled high internally since R3000 support was not required for Fast Forward machines.

The second part of a CPU write is actually writing the data into memory. When the fifo is full the MC chip will deassert the processor cpu_wrrdy_n signal so that the processor will not issue another write until the data in the write fifo has been written. The MC chip arbitrates for the memory system and then writes the data into memory by popping the data off the fifo and flopping it into the memory data write flops while it is written to memory. During this part of the operation the cpu_sel signal will be asserted. There are two sets of flops between the output of the CPU write buffer and the memory data outputs. The first set are controlled with the Aen and the Ben control signals. When either of these is active during a write to main memory by the CPU the CPU write buffer is popped. The second set of flops is controlled by the cen signal.

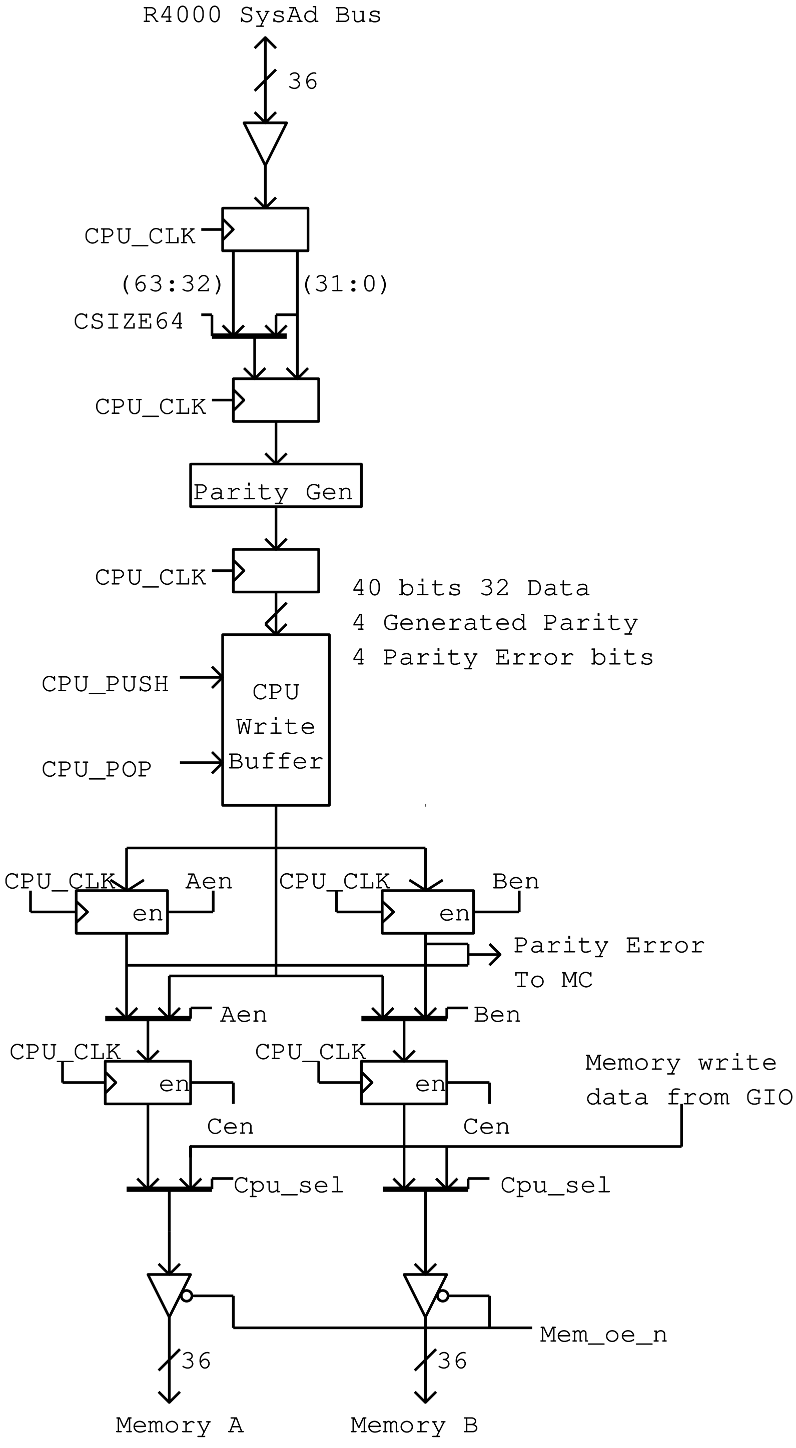
During the second part of the CPU write to memory the MUX control signals have the following meaning:

gio_sel	0	The gio_sel signal will be 0 during CPU writes to memory.
cpu_sel	1	The cpu_sel signal will always be 1 during the second part of a memory write.
cpu_push	d	The CPU can still be writing data to the CPU memory buffer during a memory write.

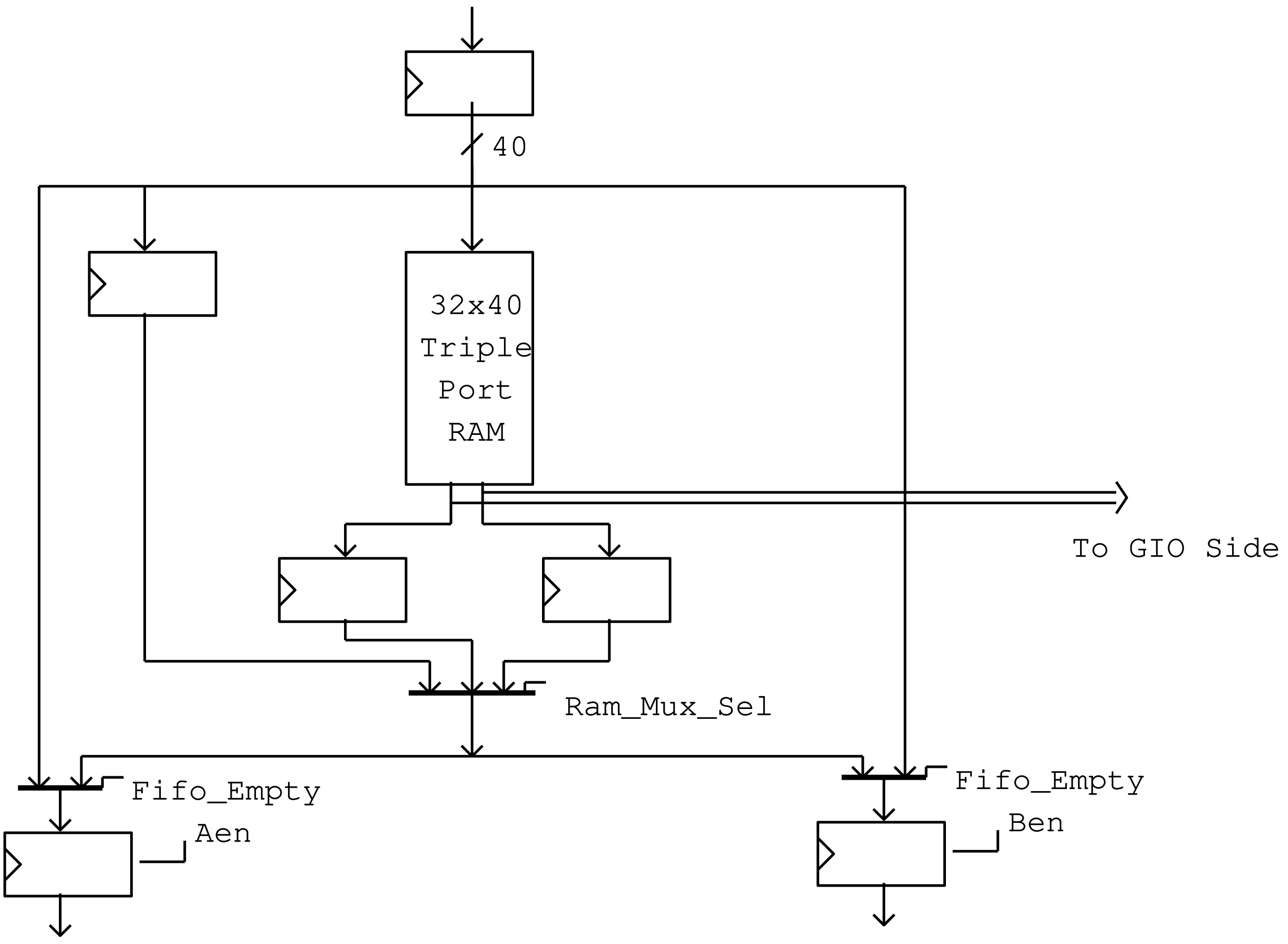
cpu_mem_oe		The memory output buffers will be turned on when this signal is asserted.
data_sel(2:0)	d	These signals are not used.
mux_dir	0	This signal should be 0 for memory writes.
graphics(1:0)	d	These signals are not used.
aen_mem		When this signal is asserted the A register will be enabled and the CPU memory buffer will be popped.
ben_ctrl		When this signal is asserted the B register will be enabled and the CPU memory buffer will be popped.
cen_fifo		When this signal is asserted the C register will be enabled. If both cen_fifo and aen_mem or ben_ctrl are both asserted in the same cycle the data from the CPU memory buffer instead of the A or B register will be loaded into the C register.
giostb	d	This signal is not used.
par_flush		When this signal is asserted bad memory parity will be written into main memory. This is used for diagnostics.

A block diagram of the CPU memory write path is shown on the following page.

Processor Writes To Memory



CPU Write Buffer Expanded



The CPU Write buffer is implemented such that the data being written does not have to be available to be read the same cycle. This is accomplished by using the by-pass muxes with the Fifo_Empty control signal. Also, if a read address changes value, the data associated with the new address does not have to be valid until two cycles later. This is accomplished by using a triple-port RAM and "ping-ponging" the read data from one port to another. The Ram_Mux_Sel control signal is used for this purpose. All additional control signals and muxes are included so that 65 MHz worse case timing can be met using RAMs currently available from LSI logic in their 100k series.

1.3.2 Processor Reads From Memory

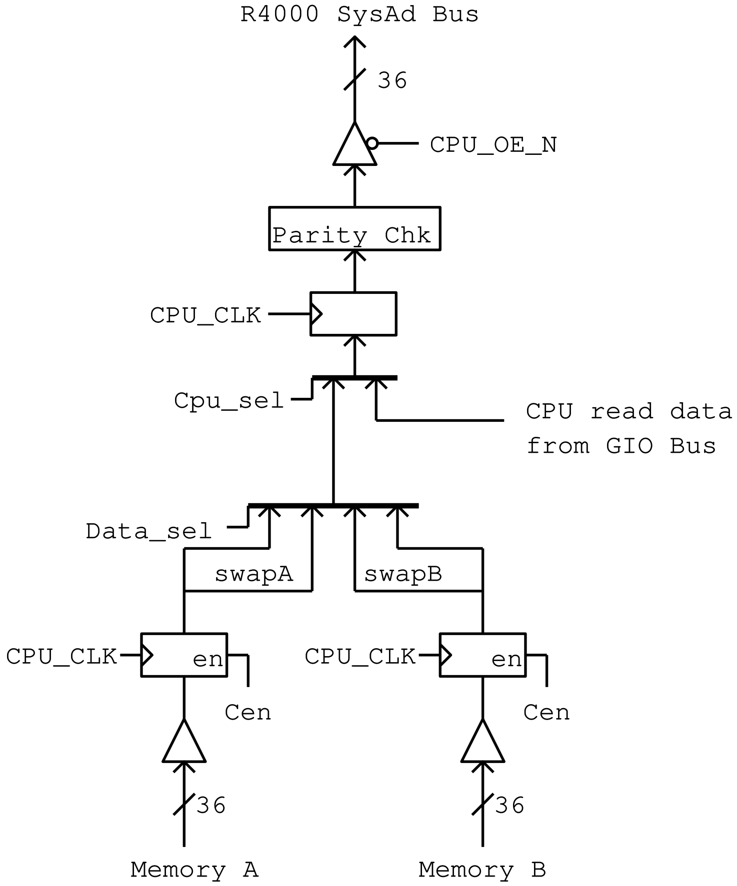
The MUX chip acts as an interleave multiplexer during CPU reads from memory. The data from memory comes in the two memory data ports on MUX and is flopped with the cpu_clk. This flop is enabled with the cen_fifo control signal. The data that is to be sent back to the processor is selected with the data_sel control signals. Parity is then generated and checked. This data is then flopped and sent over the sysad bus.

The MUX control signals during processor reads from memory have the following meaning:

gio_sel	0	The gio_sel signal will be 0 during CPU reads from memory.
cpu_sel	1	The cpu_sel signal will be 1 during CPU reads from memory.
cpu_push	d	The CPU can still be writing data to the CPU memory buffer during a memory read, although when the read is returning data to the processor the sysad bus is busy and so therefore can not be used by the CPU to send write data.
csize64	d	This signal is not used.
cpu_mem_oe		The sysad output buffers will be turned on when this signal is asserted.
data_sel(2)	d	This signals is not used.
data_sel(1:0)		These signal determine the source of the memory read data. 0 - mem_a(63:32) -> sysad(31:0), d -> sysad(63:32) 1 - mem_b(63:32) -> sysad(31:0), d -> sysad(63:32) 2 - mem_a -> sysad 3 - mem_b -> sysad
mux_dir	1	This signal should be 1 for memory reads.
graphics(1:0)	d	These signals are not used.
aen_mem	d	This signal is not used.
ben_ctrl	d	This signal is not used.
cen_fifo		When this signal is asserted the memory data input flop is enabled.
giostb	d	This signal is not used.
par_flush	d	This signal is not used.

A block diagram of the CPU memory read path is shown on the following page.

Processor Reads From Memory



1.4 GIO Operations

1.4.1 Processor Reads From GIO64/EISA

The R4000 processor can issue a read to any device on the GIO64 bus. The address is sent to the MC chip which decodes it and then arbitrates for the GIO64 bus. Once it has been granted the bus it sends out the address on the GIO64 bus. The read data from the GIO64 device is flopped by the MUX chip and then sent through a mux where a word swap operation may be performed. The data is then flopped again and sent over the sysad bus back to the processor. There are two different word swap operations that are needed. The first is directly sending gio_ad(63:0) to sysad(63:0). The second operation is to duplicate the lower word on the GIO64 bus onto both words of the sysad bus. This is necessary when the R4000 is reading from a 32 bit GIO64 device. The third operation is moving the high word on the GIO64 bus to the low word on the sysad bus. This is needed when the R3000 reads data from a 64 bit GIO64 device and the data is returned on the high bits of the bus.

The data that is returned to the processor is not synchronized in the MUX chip. Instead the data is held on the GIO64 bus for a few clocks before it is sent to the processor. As long as the cpu_mem_oe signal is asserted while gio_sel is asserted, mux_dir is deasserted, and aen_mem is deasserted the data from the GIO64 bus will be flopped onto the sysad bus.

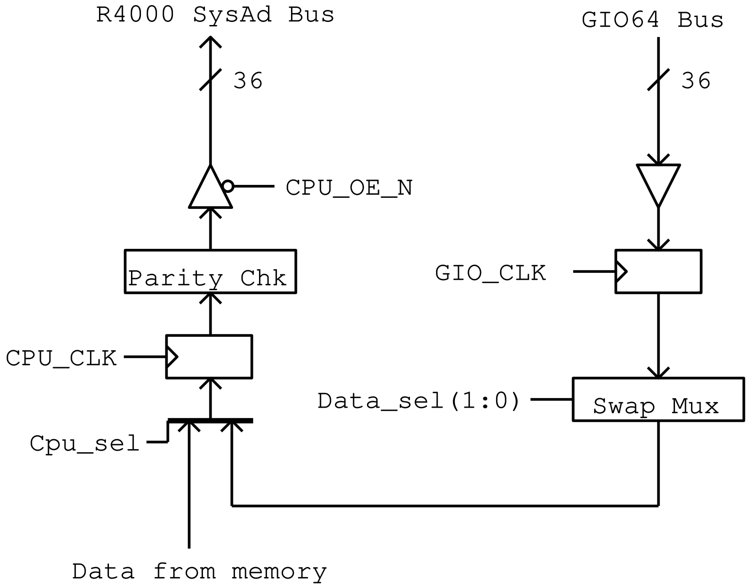
EISA reads look just like GIO64 device reads as far as the MUX is concerned.

The MUX control signals during processor reads from the GIO64 bus have the following meaning:

gio_sel	1	This is a GIO operation.
cpu_sel	0	This signal must be 0.
cpu_push	d	This signal will be 0 since the CPU is waiting for a read response.
csize64	d	This signal is not used.
cpu_mem_oe		The sysad output buffers will be turned on when this signal is asserted.
data_sel(2)	d	This signals is not used.
data_sel(1:0)		These signal determine the word swap operation. 0 - gio_ad(63:0) -> sysad(63:0) 1 - gio_ad(31:0) -> sysad(31:0) gio_ad(31:0) -> sysad(63:32) 2 - gio_ad(63:32) -> sysad(31:0), d -> sysad(63:32) 3 - reserved
mux_dir	0	This signal should be 0 for GIO64 reads.
graphics(1:0)	d	This signal is not used.
aen_mem	0	This signal should be 0 for GIO64 reads.
ben_ctrl	d	This signal is not used.
cen_fifo	d	This signal is not used.
giostb	d	This signal is not used.
par_flush	d	This signal is not used.

A block diagram of the CPU GIO64 read path is shown on the following page.

Processor Reads From GIO64/EISA



1.4.2 Processor Writes to GIO64

The R4000 processor can issue a write to any device on the GIO64 bus. Data is pushed into the cpu write fifo synchronous to the CPU_CLK and popped out synchronous to the GIO_CLK. The MC writes the data to the CPU write buffer using the cpu_push signal. The MC chip must make sure the data is in the write buffer before it tries to pop it off the fifo. Cache writes are supported to devices that are 64 bits wide when the processor is a R4000. For each piece of data to be written the MC chip sends the MUX chip a GIO command. The GIO command is put into a twelve entry command fifo in the MUX chip and then as each piece of data is transferred the command fifo is popped. When the fifo is empty the MUX should turn off it gio_ad output buffer. Normal GIO writes will only have one GIO command, but cache block writes to EISA can result in many commands. Part of the GIO command is which GIO delay signal to use to determine when the data has been transferred.

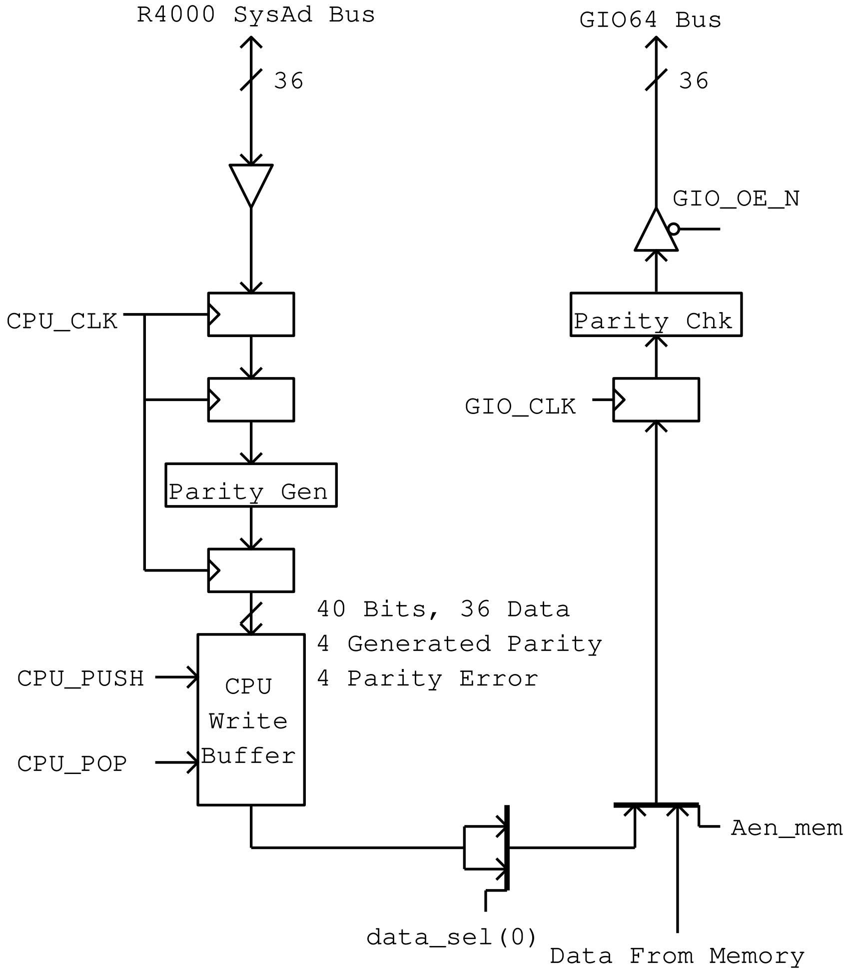
As each GIO command has been satisfied the MUX chip needs to wait a dead clock before transferring the data for the next GIO command. During cache block writes the CPU will assert masdly during these dead cycles.

The MUX control signals during processor writes to the GIO64 bus have the following meaning:

gio_sel	1	This is a GIO operation.
cpu_sel	0	This signal must be 0.
cpu_push	d	This signal is not used.
csize64	d	This signal is not used.
cpu_mem_oe	0	This signal should be 0.
data_sel(2:1)	d	These signals are not used.
data_sel(0)		This signal determines the word swap operation. 0 - sysad(63:0) -> gio_ad(63:0) 1 - sysad(31:0) -> gio_ad(63:32) sysad(63:32) -> gio_ad(31:0)
mux_dir	1	This signal should be 1 for CPU GIO64 writes.
graphics(1:0)		These signals determine which delay signal is from the addressed slave. 0 - slvdly 1 - grxdly0 2 - grxdly2 3 - grxdly3
aen_mem	0	This signal should be 0 for CPU GIO64 writes.
ben_ctrl	d	This signal is not used.
cen_fifo	d	This signal is not used.
giostb		This signal is asserted each time a new GIO command is sent to the MUX.
par_flush	d	This signal is not used.

A block diagram of the CPU GIO64 write path is shown on the following page.

Processor Writes To GIO64/EISA



1.4.3 GIO Writes To Memory

The MUX chip is used to buffer and pack data that is being written into main memory. The MUX chips packs data from 32 and 64 bit bus masters into 128 bit quad words to be written into main memory. There is also a six entry write buffer between the GIO64 bus and the memory data outputs. This isolates GIO bus stalls from the memory write.

GIO memory writes use GIO commands from the MC chip it indicate the packing operations that are to be performed. There is one GIO command for each piece of data that is transferred on the GIO64 bus. A GIO command is indicated by the MC chip asserting the giostb signal. The ben_ctrl signal is used to pop data off the write fifo. This signal is not part of the GIO command. When the fifo is empty the fifo output should be the contents of the GIO packing registers. The graphics signal is used to indicate to the MUX chip which delay signal is from the bus master. If the write fifo in the MUX chip gets filled up the MC chip will assert slvdly to throttle the transfer and the MUX chip will not have any new GIO commands. Once a piece of data has been transferred on the bus, which can be determined by looking at the bus masters delay signal the GIO command is popped and the next GIO command is executed.

If the transfer gets preempted the par_flush signal will be asserted to indicate that the GIO command fifo should be flushed of any valid entries and that the write fifo counters should be cleared.

The MUX chip does not have to worry about the write buffer being over filled. The MC chip has to keep track of how many entries are in the buffer. The MC chip controls how full the write buffer is by only sending out enough GIO commands to keep the write buffer full. When the MUX chip has run out of new GIO commands because the write buffer is full the MC chip will assert slvdly to stall the bus master.

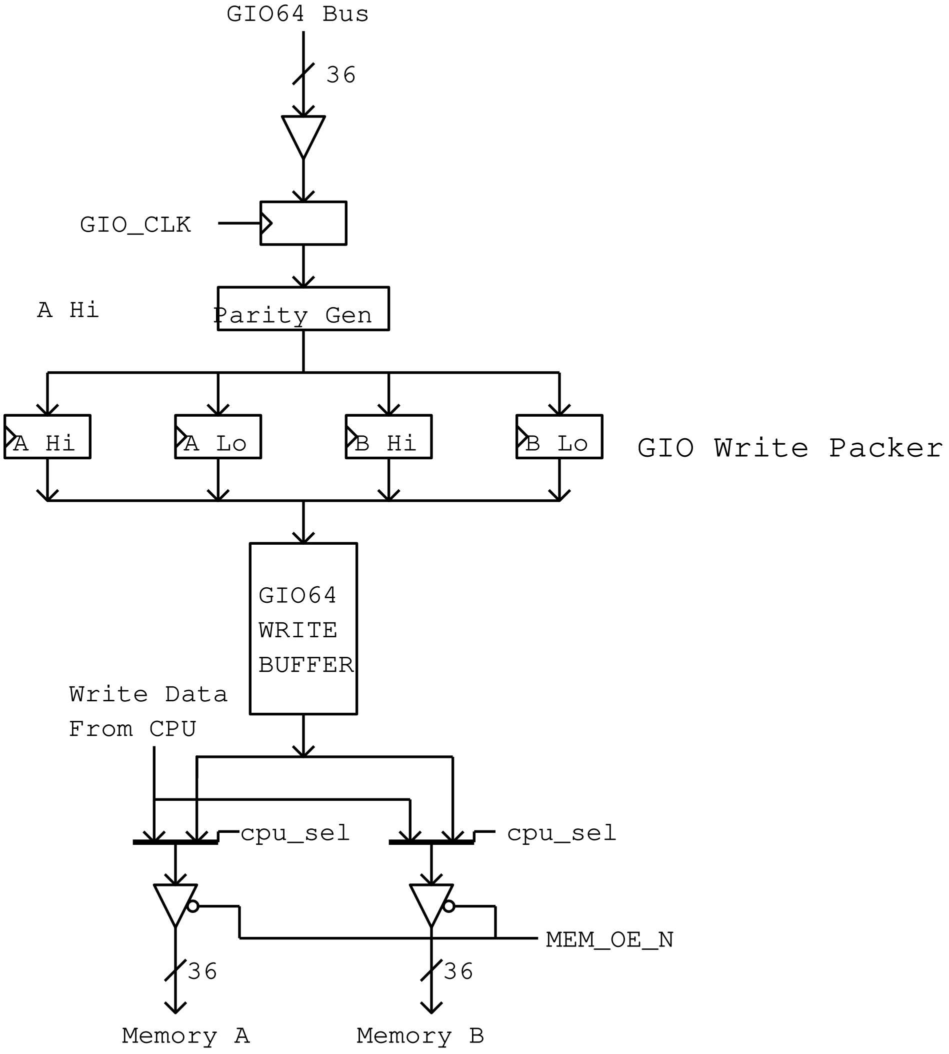
The MUX control signals during GIO64 writes to main memory have the following meaning:

```
gio_sel        1  This is a GIO operation.
cpu_sel        0  This signal must be 0.
cpu_push       d  This signal is not used.
csize64        d  This signal is not used.
cpu_mem_oe     0  This signal should be 0.
data_sel(2:0)     This signal determines which data in the packer
                  register is written.
                    0 - Write A Lo, gio_ad(31:0) -> A lo
                    1 - Write B Lo, gio_ad(31:0) -> B lo
                    2 - Write A Hi, gio_ad(31:0) -> A hi
                    3 - Write B Hi, gio_ad(31:0) -> B hi
                    4 - Write A Hi and Lo, gio_ad(31:0) -> A lo
                                 gio_ad(63:32) -> A hi
                    5 - Write B Hi and Lo, gio_ad(31:0) -> B lo
                                 gio_ad(63:32) -> B hi
                    6 - Reserved
                    7 - Reserved
mux_dir        0  This signal should be 0 for GIO64 writes to memory.
graphics(1:0)     These signals determine which delay signal is from the
                  addressed slave.
```

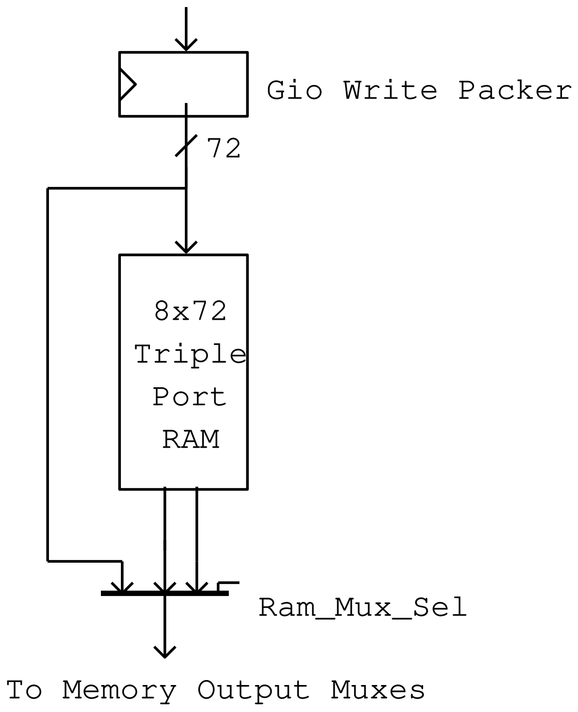
		0 - masdly
		1 - grxdly0
		2 - grxdly2
		3 - grxdly3
aen_mem	1	This signal should be 1 for GIO64 writes to memory.
ben_ctrl		This signal is used to pop data off the write fifo.
cen_fifo		This signal is used to push data onto the write fifo.
giostb		This signal is asserted each time a new GIO command is sent to the MUX.
par_flush		When this signal is asserted the GIO command fifo and the GIO write fifo should be flushed.

A block diagram of the GIO64 memory write path is shown on the following page.

GIO64 Writes To Memory



GIO Write Buffer Expanded



The GIO Write buffer is implemented such that the data being written does not have to be available to be read the same cycle. This is accomplished by using the by-pass muxes with the Ram_MUX_Sel control signal. Also, if a read address changes value, the data associated with the new address does not have to be valid until two cycles later. This is accomplished by using a triple-port RAM and "ping-ponging" the read data from one port to another. The Ram_Mux_Sel control signal is also used for this purpose. The additional control signal and mux are included so that 40 MHz worse case timing can be met using RAMs currently available from LSI logic in their 100k series.

1.4.4 GIO Reads From Memory

The Mux chip is a data buffer and unpacker during GIO64 reads from main memory. Like GIO memory reads the memory writes work off of a GIO command. The GIO command determines when the GIO read data fifo is popped and which data is sent over the GIO64 bus. The graphics signal indicates which master delay signal should be used in determining when a piece of data has been transferred. When a piece of data has been transferred the GIO command fifo is popped and the next command should be executed in the following cycle. The cen_fifo signal is part of the GIO command and indicates when the GIO read fifo should be popped. The ben_ctrl signal is not part of the GIO command and is used to push data onto the GIO read fifo. The GIO memory read fifo is three entries deep.

When the par_flush signal is asserted the GIO command fifo and the GIO read fifo should be flushed.

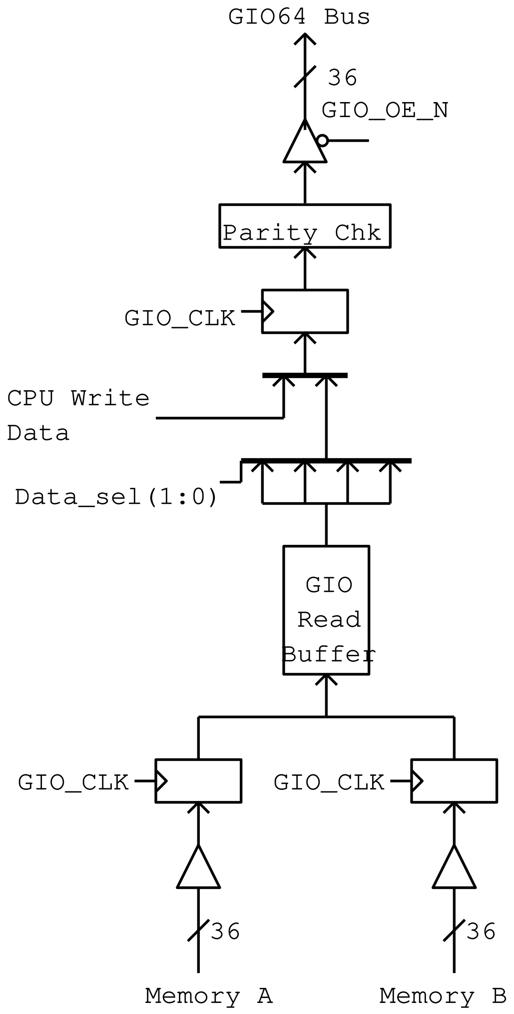
When the GIO command fifo is empty the MUX chip should stop driving the GIO bus. When the memory system can not keep up with a GIO64 device the MUX chip will run out of GIO commands. During this time the MC chip will assert slvdly to stall the GIO64 master.

The MUX control signals during GIO64 reads to main memory have the following meaning:

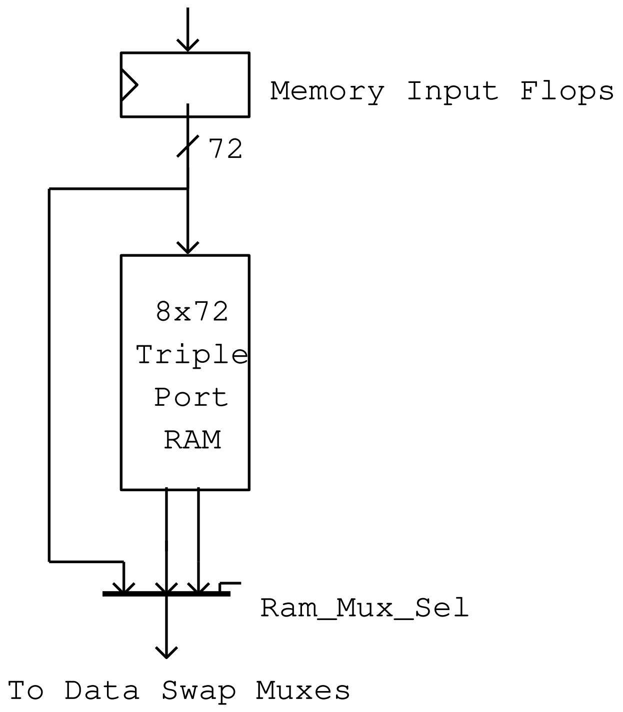
gio_sel	1	This is a GIO operation.
cpu_sel	0	This signal must be 0.
cpu_push	d	This signal is not used.
csize64	d	This signal is not used.
cpu_mem_oe	0	This signal should be 0.
data_sel(2)	d	This signal is used to determine whether MUX should drive the gio bus. This is included so that for back to back reads, MUX does not turn off its drivers, and then quickly turn them back on again..
data_sel(1:0)		This signal determines which data is sent over the GIO64 bus. 0 - Mem_A(63:0) -> GIO_AD(63:0) 1 - Mem_B(63:0) -> GIO_AD(63:0) 2 - Mem_A(63:32) -> GIO_AD(31:0), d -> GIO_AD(63:32) 3 - Mem_B(63:32) -> GIO_AD(31:0), d -> GIO_AD(63:32)
mux_dir	1	This signal should be 1 for GIO64 reads from memory.
graphics(1:0)		These signals determine which delay signal is from the addressed slave. 0 - masdly 1 - grxdly0 2 - grxdly1 3 - grxdly2
aen_mem	1	This signal should be 1 for GIO64 reads from memory.
ben_ctrl		This signal is used to push data on the read fifo.
cen_fifo		This signal is used to pop data off the read fifo.
giostb		This signal is asserted each time a new GIO command is sent to the MUX.
par_flush		When this signal is asserted the GIO command fifo and the GIO read fifo should be flushed.

A block diagram of the GIO64 memory read path is shown on the following page.

GIO64 Reads From Memory



GIO Read Buffer Expanded



The GIO Read buffer is implemented such that the data being written does not have to be available to be read the same cycle. This is accomplished by using the by-pass muxes with the Ram_MUX_Sel control signal. Also, if a read address changes value, the data associated with the new address does not have to be valid until two cycles later. This is accomplished by using a triple-port RAM and "ping-ponging" the read data from one port to another. The Ram_Mux_Sel control signal is also used for this purpose. The additional control signal and mux are included so that 40 MHz worse case timing can be met using RAMs currently available from LSI logic in their 100k series.

2. MUX Pins

There are 189 signal pins on the mux chip including five signals for test which at this time are not defined. This part will be packaged in the 240 pin MQFP.

2.1 Data Buses

sysad(31:0)	i/o	32 bits of R4000 processor bus, made up of 4 bytes. SYSAD R4000 R4000 (31:24) is (63:56) or (55:48) (23:16) is (39:32) or (47:40) (15: 8) is (31:24) or (23:16) (7: 0) is (7:0) or (15: 8)
sysad_par(3:0)	i/o	Parity over sysad.
gio_data(31:0)	i/o	32 bits of GIO64 data bus, made up of 4 bytes. GIODATA GIO64 GIO64 (31:24) is (63:56) or (55:48) (23:16) is (39:32) or (47:40) (15: 8) is (31:24) or (23:16) (7: 0) is (7:0) or (15: 8)
gio_par(3:0)	i/o	Parity over the gio_data.
mem_a(31:0)	i/o	Memory interleave A data.
mem_a_par(3:0)	i/o	Parity over mem_a.
mem_b(31:0)	i/o	Memory interleave B data.
mem_b_par(3:0)	i/o	Parity over mem_b.

2.2 MUX Control Signals

Some of the control signals serve more than one function. The function is dependent on whether or not the GIO bus is involved in the operation. The giosel and cpusel signals are used to determine which function is the current function.

giosel	in	Active during gio<->memory and cpu<->gio transfers. Mutually exclusive with cpusel
cpusel	in	Active during cpu<->memory transfers.Mutually exclusive with giosel
cpu_push	in	Push data onto the CPU write fifo.
csize64	in	This signal is used to indicate a full 64 bit transfer over the SYSAD bus. 0 - Processor is a R3000 1 - Processor is a R4000
cpu_mem_oe	in	Control buffer driving memory bus A and B or sysad depending on dir.
data_sel(2:0)	in	Source of MUX data. Exact meaning depends on the operation taking place.
mux_dir	in	Indicated source and destination of data, used with aen_mem, gio_sel and cpu_sel.
graphics(1:0)	in	determines whether to use grfxdly(2:0) or slvdly.

		0 - slvdly 1 - grxdly(0) 2 - grxdly(1) 3 - grxdly(2)
aen_mem	in	When cpu_sel = 1 enables flop used to catch data popped from the cpu write buffer during write operations to memory port A. when cpu_sel = 0 this is used to help indicate the source and destination of data.
ben_ctrl	in	When cpu_sel = 1 enables flop used to catch data popped from the cpu write buffer during write operations to memory port B. When cpu_sel = 0, is used to control GIO memory fifo.
cen_fifo	in	When cpu_sel = 1 enables data into the memory read and write flops. When cpu_sel = 0 is used to control GIO memory fifo.
giostb	in	Indicates that the GIO command which is made up of some of the above signals is valid and should be pushed onto the GIO command fifo.
par_flush	in	When cpu_sel = 1, generate bad parity on memory write data. When cpu_sel = 0, clean up state in MUX to complete the current memory operations and set up for the next memory operation.

2.3 GIO64 Signals.

masdly	in	Master delay signal.
slvdly	in	Slave delay signal.
grxdly(2:0)	in	Graphics delay signal. There are 3 of these signals.
mc_dly	in	For GIO slave reads from memory an early version of MC’s delay signal. For GIO slave writes to memory a copy of MC’s delay signal.

2.3 Clock Signals.

gio_clk	in	GIO64 clock, 33 or 40 MHZ
gio_pll_lp1	out	PLL phase detector output
gio_pll_lp2	in	PLL loop filter from board
gio_pll_vdd	in	PLL power
gio_pll_gnd	in	PLL gnd
cpu_clk	in	Processor clock, 50-65 MHZ
cpu_pll_lp1	out	PLL phase detector output
cpu_pll_lp2	in	PLL loop filter from board
cpu_pll_vdd	in	PLL power
cpu_pll_gnd	in	PLL gnd
pll_reset_n	in	PLL reset

2.4 Misc. Signals.

par_err(3:0)	out	Parity error detected on operation.

2.5 JTAG/ATPG Signals.

jtdi	in	JTAG data in
jtdo	out	JTAG data out
jtms	in	JTAG mode select
jtck	in	JTAG clock
tp0	in	JTAG tp0 mode select
tp1	in	JTAG tp1 mode select
entei	in	ATPG tristate enable inhibit

3.0 MUX Pin Cross-Reference Listing (299 Pin Package)

Pin Number	Name	Type	In	Out	Enable	Active
Pin 1 - C3	MC_DLY	INPUT	255	0	0	
Pin 2 - E5	GIOSTB	INPUT	248	0	0	
Pin 3 - B3	GIOSEL	INPUT	247	0	0	
Pin 4 - E6		NC	0	0	0	
Pin 5 - C4	GIO_DATA_31	BIDIR	293	323	259	LOW
Pin 6 - D6		NC	0	0	0	
Pin 7 - D5	GIO_DATA_30	BIDIR	292	322	259	LOW
Pin 8 - E7	GIO_DATA_29	BIDIR	290	321	259	LOW
Pin 9 - B4	VSS	POWER	0	0	0	
Pin 10 - C5	VDD	POWER	0	0	0	
Pin 11 - B5	GIO_DATA_28	BIDIR	289	320	259	LOW
Pin 12 - A5	GIO_DATA_27	BIDIR	288	319	259	LOW
Pin 13 - C6	GIO_DATA_26	BIDIR	287	318	259	LOW
Pin 14 - B6	GIO_DATA_25	BIDIR	286	316	259	LOW
Pin 15 - D7	GIO_DATA_24	BIDIR	285	315	259	LOW
Pin 16 - A6	VSS	POWER	0	0	0	
Pin 17 - C7	VDD	POWER	0	0	0	
Pin 18 - E8		NC	0	0	0	
Pin 19 - B7	PAR_FLUSH	INPUT	246	0	0	
Pin 20 - D8	CPUSEL	INPUT	162	0	0	
Pin 21 - A7	GIO_DATA_23	BIDIR	284	314	260	LOW
Pin 22 - E9	GIO_DATA_22	BIDIR	283	313	260	LOW
Pin 23 - C8	GIO_DATA_21	BIDIR	282	312	260	LOW
Pin 24 - D9	GIO_DATA_20	BIDIR	281	311	260	LOW
Pin 25 - B8	GIO_DATA_19	BIDIR	279	310	260	LOW
Pin 26 - C9	GIO_DATA_18	BIDIR	278	309	260	LOW
Pin 27 - A8	GIO_DATA_17	BIDIR	277	308	260	LOW
Pin 28 - C10	GIO_DATA_16	BIDIR	276	307	260	LOW
Pin 29 - B9	GIO_DATA_15	BIDIR	275	305	261	LOW
Pin 30 - E10	GIO_DATA_14	BIDIR	274	304	261	LOW
Pin 31 - B10	VSS	POWER	0	0	0	
Pin 32 - D10	VDD	POWER	0	0	0	
Pin 33 - B11	VSS	POWER	0	0	0	
Pin 34 - D11	GIO_DATA_13	BIDIR	273	303	261	LOW
Pin 35 - B12	GIO_DATA_12	BIDIR	272	302	261	LOW
Pin 36 - E11	GIO_DATA_11	BIDIR	271	301	261	LOW
Pin 37 - B13	GIO_DATA_10	BIDIR	270	300	261	LOW
Pin 38 - C11	GIO_DATA_9	BIDIR	268	299	261	LOW
Pin 39 - A14	GIO_DATA_8	BIDIR	267	298	261	LOW
Pin 40 - C12	GIO_DATA_7	BIDIR	266	297	262	LOW
Pin 41 - C13	GIO_DATA_6	BIDIR	265	296	262	LOW
Pin 42 - D12	GIO_DATA_5	BIDIR	264	325	262	LOW
Pin 43 - B14	GIO_DATA_4	BIDIR	294	324	262	LOW
Pin 44 - E12	VDD	POWER	0	0	0	
Pin 45 - A15	VSS	POWER	0	0	0	
Pin 46 - D13	GIO_DATA_3	BIDIR	291	317	262	LOW
Pin 47 - C14	GIO_DATA_2	BIDIR	280	306	262	LOW
Pin 48 - E13	GIO_DATA_1	BIDIR	269	326	262	LOW
Pin 49 - B15		NC	0	0	0	
Pin 50 - A16	VDD	POWER	0	0	0	
Pin 51 - C15	GIO_DATA_0	BIDIR	263	295	262	LOW
Pin 52 - B16	GIO_PAR_3	BIDIR	330	334	258	LOW
Pin 53 - A17	GIO_PAR_2	BIDIR	329	333	258	LOW
Pin 54 - C16	GIO_PAR_1	BIDIR	328	332	258	LOW
Pin 55 - B17	GIO_PAR_0	BIDIR	327	331	258	LOW
Pin 56 - D16	GRAPHICS_1	INPUT	250	0	0	
Pin 57 - D14	GRAPHICS_0	INPUT	249	0	0	
Pin 58 - C17	GRXDLY_2	INPUT	334	0	0	
Pin 59 - E14	GRXDLY_1	INPUT	253	0	0	
Pin 60 - B18	GRXDLY_0	INPUT	251	0	0	

MUX Pin Cross-Reference Listing (299 Pin Package cont)

Pin Number	Name	Type	In	Out	Enable	Active
Pin 61 - D15		NC	0	0	0	
Pin 62 - B19	GIO_PLL_LP1	PLL	0	0	0	
Pin 63 - E15	VDD	POWER	0	0	0	
Pin 64 - D17	VSS	POWER	0	0	0	
Pin 65 - C18	VSS	POWER	0	0	0	
Pin 66 - E16	GIO_PLL_LP2	PLL	0	0	0	
Pin 67 - C19	GIO_CLK	CLOCK	0	0	0	
Pin 68 - F16		NC	0	0	0	
Pin 69 - D18		NC	0	0	0	
Pin 70 - F17	GIO_PLL_AGND	PLL	0	0	0	
Pin 71 - E17	GIO_PLL_VSS	POWER	0	0	0	
Pin 72 - G16	GIO_PLL_VDD	POWER	0	0	0	
Pin 73 - D19	MEM_A_31	BIDIR	199	87	83	LOW
Pin 74 - E18	MEM_A_30	BIDIR	198	88	83	LOW
Pin 75 - D20	MEM_A_29	BIDIR	197	89	83	LOW
Pin 76 - E19	MEM_A_28	BIDIR	196	90	83	LOW
Pin 77 - F18	MEM_A_27	BIDIR	195	91	83	LOW
Pin 78 - E20	MEM_A_26	BIDIR	194	92	83	LOW
Pin 79 - G17		NC	0	0	0	
Pin 80 - F19		NC	0	0	0	
Pin 81 - H16	MEM_A_25	BIDIR	193	93	83	LOW
Pin 82 - G18	MEM_A_24	BIDIR	192	94	83	LOW
Pin 83 - H17	MEM_A_23	BIDIR	191	95	84	LOW
Pin 84 - F20	MEM_A_22	BIDIR	190	96	84	LOW
Pin 85 - J16	MEM_A_21	BIDIR	189	97	84	LOW
Pin 86 - G19	MEM_A_20	BIDIR	188	98	84	LOW
Pin 87 - J17	MEM_A_19	BIDIR	187	99	84	LOW
Pin 88 - H18	MEM_A_18	BIDIR	186	100	84	LOW
Pin 89 - J18	MEM_A_17	BIDIR	185	101	84	LOW
Pin 90 - G20	MEM_A_16	BIDIR	184	102	84	LOW
Pin 91 - K18		NC	0	0	0	
Pin 92 - H19		NC	0	0	0	
Pin 93 - K16		NC	0	0	0	
Pin 94 - J19		NC	0	0	0	
Pin 95 - K17		NC	0	0	0	
Pin 96 - K19	MEM_A_15	BIDIR	183	103	85	LOW
Pin 97 - L17	MEM_A_14	BIDIR	182	104	85	LOW
Pin 98 - L19	MEM_A_13	BIDIR	181	105	85	LOW
Pin 99 - L16	MEM_A_12	BIDIR	180	106	85	LOW
Pin 100 - M19	MEM_A_11	BIDIR	179	107	85	LOW
Pin 101 - L18	MEM_A_10	BIDIR	178	108	85	LOW
Pin 102 - N20	MEM_A_9	BIDIR	177	109	85	LOW
Pin 103 - M18	MEM_A_8	BIDIR	176	110	85	LOW
Pin 104 - N19	MEM_A_7	BIDIR	175	111	86	LOW
Pin 105 - M17	MEM_A_6	BIDIR	174	112	86	LOW
Pin 106 - P20	PAR_ERR_3	OUTPUT	0	240	0	
Pin 107 - M16	PAR_ERR_2	OUTPUT	0	239	0	
Pin 108 - N18	PAR_ERR_1	OUTPUT	0	241	0	
Pin 109 - N17	PAR_ERR_0	OUTPUT	0	238	0	
Pin 110 - P19	VDD	POWER	0	0	0	
Pin 111 - N16	VSS	POWER	0	0	0	
Pin 112 - R20		NC	0	0	0	
Pin 113 - P18	MEM_A_5	BIDIR	173	113	86	LOW
Pin 114 - R19	MEM_A_4	BIDIR	172	114	86	LOW
Pin 115 - T20	MEM_A_3	BIDIR	171	115	86	LOW
Pin 116 - R18	MEM_A_2	BIDIR	170	116	86	LOW
Pin 117 - T19	MEM_A_1	BIDIR	169	117	86	LOW
Pin 118 - U20	MEM_A_0	BIDIR	166	118	86	LOW
Pin 119 - T18	MEM_A_PAR_3	BIDIR	168	119	82	LOW
Pin 120 - U19	MEM_A_PAR_2	BIDIR	167	120	82	LOW

MUX Pin Cross-Reference Listing (299 Pin Package cont)

Pin Number	Name	Type	In	Out	Enable	Active
Pin 121 - P17	MEM_A_PAR_1	BIDIR	201	121	82	LOW
Pin 122 - T17	MEM_A_PAR_0	BIDIR	200	122	82	LOW
Pin 123 - P16	TP1	TEST	0	0	0	
Pin 124 - U18	TP0	TEST	0	0	0	
Pin 125 - R17		NC	0	0	0	
Pin 126 - V19	JTDO	JTAG	0	0	0	
Pin 127 - R16	JTMS	JTAG	0	0	0	
Pin 128 - V18	VSS	POWER	0	0	0	
Pin 129 - U17	PLL_RESET_N	PLL	0	0	0	
Pin 130 - T16	ENTEI	TEST	0	0	0	
Pin 131 - W19	JTDI	JTAG	0	0	0	
Pin 132 - T15		NC	0	0	0	
Pin 133 - W18	JTCK	JTAG	0	0	0	
Pin 134 - U15	CPU_MEM_OE	INPUT	163	0	0	
Pin 135 - V17	MEM_B_31	BIDIR	235	123	78	LOW
Pin 136 - T14	MEM_B_30	BIDIR	234	124	78	LOW
Pin 137 - U16	MEM_B_29	BIDIR	233	125	78	LOW
Pin 138 - W17	MEM_B_28	BIDIR	232	126	78	LOW
Pin 139 - V16	MEM_B_27	BIDIR	231	127	78	LOW
Pin 140 - W16	MEM_B_26	BIDIR	230	128	78	LOW
Pin 141 - X16		NC	0	0	0	
Pin 142 - V15	VDD	POWER	0	0	0	
Pin 143 - W15	MEM_B_25	BIDIR	229	129	78	LOW
Pin 144 - U14	MEM_B_24	BIDIR	228	130	78	LOW
Pin 145 - T13	MEM_B_23	BIDIR	227	131	79	LOW
Pin 146 - X15	MEM_B_22	BIDIR	226	132	79	LOW
Pin 147 - U13	MEM_B_21	BIDIR	225	133	79	LOW
Pin 148 - V14		NC	0	0	0	
Pin 149 - T12	MEM_B_20	BIDIR	224	134	79	LOW
Pin 150 - W14	MEM_B_18	BIDIR	222	136	79	LOW
Pin 151 - U12	MEM_B_19	BIDIR	223	135	79	LOW
Pin 152 - X14	MEM_B_17	BIDIR	221	137	79	LOW
Pin 153 - V12	MEM_B_16	BIDIR	220	138	79	LOW
Pin 154 - V13		NC	0	0	0	
Pin 155 - V11	CEN_FIFO	INPUT	161	0	0	
Pin 156 - W13	BEN_CTRL	INPUT	160	0	0	
Pin 157 - T11		NC	0	0	0	
Pin 158 - X13		NC	0	0	0	
Pin 159 - U11	AEN_MEM	INPUT	159	0	0	
Pin 160 - W12	MEM_B_15	BIDIR	219	139	80	LOW
Pin 161 - U10	MEM_B_14	BIDIR	218	140	80	LOW
Pin 162 - W11	MEM_B_13	BIDIR	217	141	80	LOW
Pin 163 - T10	MEM_B_12	BIDIR	216	142	80	LOW
Pin 164 - W10	MEM_B_11	BIDIR	215	143	80	LOW
Pin 165 - V10	MEM_B_10	BIDIR	214	162	80	LOW
Pin 166 - W9	MEM_B_9	BIDIR	213	145	80	LOW
Pin 167 - V9	MEM_B_8	BIDIR	212	146	80	LOW
Pin 168 - W8	MEM_B_7	BIDIR	211	147	81	LOW
Pin 169 - U9	MEM_B_6	BIDIR	210	148	81	LOW
Pin 170 - X7	DATA_SEL_2	INPUT	244	0	0	
Pin 171 - T9	DATA_SEL_1	INPUT	245	0	0	
Pin 172 - V8	DATA_SEL_0	INPUT	243	0	0	
Pin 173 - U8	READ	INPUT	256	0	0	
Pin 174 - W7	VDD	POWER	0	0	0	
Pin 175 - T8	VSS	POWER	0	0	0	
Pin 176 - X6	MEM_B_5	BIDIR	209	149	81	LOW
Pin 177 - V7	MEM_B_4	BIDIR	208	150	81	LOW
Pin 178 - W6	MEM_B_3	BIDIR	207	151	81	LOW
Pin 179 - X5	MEM_B_2	BIDIR	206	152	81	LOW
Pin 180 - V6	MEM_B_1	BIDIR	205	153	81	LOW

MUX Pin Cross-Reference Listing (299 Pin Package cont)

Pin Number	Name	Type	In	Out	Enable	Active
----------	----	----	--	---	------	------
Pin 181 - W5		NC	0	0	0	
Pin 182 - X4	MEM_B_0	BIDIR	202	154	81	LOW
Pin 183 - V5	VDD	POWER	0	0	0	
Pin 184 - W4	VSS	POWER	0	0	0	
Pin 185 - U7	MEM_B_PAR_3	BIDIR	204	155	82	LOW
Pin 186 - U5	MEM_B_PAR_2	BIDIR	203	156	82	LOW
Pin 187 - T7	MEM_B_PAR_1	BIDIR	237	157	82	LOW
Pin 188 - V4	MEM_B_PAR_0	BIDIR	236	158	82	LOW
Pin 189 - U6		NC	0	0	0	
Pin 190 - W3	RESET_N	INPUT	242	0	0	
Pin 191 - T6	VDD	POWER	0	0	0	
Pin 192 - V3	VSS	POWER	0	0	0	
Pin 193 - U4	CPU_PLL_VDD	POWER	0	0	0	
Pin 194 - T5	CPU_PLL_VSS	POWER	0	0	0	
Pin 195 - W2	CPU_CLK	CLOCK	0	0	0	
Pin 196 - R5		NC	0	0	0	
Pin 197 - V2		NC	0	0	0	
Pin 198 - R4	CPU_PLL_LP2	PLL	0	0	0	
Pin 199 - U3	CPU_PLL_LP1	PLL	0	0	0	
Pin 200 - P5	CPU_PLL_AGND	PLL	0	0	0	
Pin 201 - T4	MUX_DIR	INPUT	165	0	0	
Pin 202 - U2	CPU_PUSH	INPUT	164	0	0	
Pin 203 - T3	VSS	POWER	0	0	0	
Pin 204 - U1	VDD4	POWER	0	0	0	
Pin 205 - T2	SYSAD_31	BIDIR	29	61	76	LOW
Pin 206 - R3	SYSAD_30	BIDIR	28	60	76	LOW
Pin 207 - T1	SYSAD_29	BIDIR	26	58	76	LOW
Pin 208 - P4	SYSAD_28	BIDIR	25	57	76	LOW
Pin 209 - R2	SYSAD_27	BIDIR	24	56	76	LOW
Pin 210 - N5		NC	0	0	0	
Pin 211 - P3	SYSAD_26	BIDIR	23	55	76	LOW
Pin 212 - N4		NC	0	0	0	
Pin 213 - R1	VSS	POWER	0	0	0	
Pin 214 - M5	VSS	POWER	0	0	0	
Pin 215 - P2	VDD4	POWER	0	0	0	
Pin 216 - M4	SYSAD_24	BIDIR	21	53	76	LOW
Pin 217 - N3	SYSAD_25	BIDIR	22	54	76	LOW
Pin 218 - M3	SYSAD_23	BIDIR	20	52	75	LOW
Pin 219 - P1	SYSAD_22	BIDIR	19	51	75	LOW
Pin 220 - L3	SYSAD_21	BIDIR	18	50	75	LOW
Pin 221 - N2	SYSAD_20	BIDIR	17	49	75	LOW
Pin 222 - L5	SYSAD_19	BIDIR	15	47	75	LOW
Pin 223 - M2	SYSAD_18	BIDIR	14	46	75	LOW
Pin 224 - L4	SYSAD_17	BIDIR	13	45	75	LOW
Pin 225 - L2	SYSAD_16	BIDIR	12	44	75	LOW
Pin 226 - K4		NC	0	0	0	
Pin 227 - K2	VDD4	POWER	0	0	0	
Pin 228 - K5	SYSAD_15	BIDIR	11	43	74	LOW
Pin 229 - J2	SYSAD_14	BIDIR	10	42	74	LOW
Pin 230 - K3	SYSAD_13	BIDIR	9	41	74	LOW
Pin 231 - H1	SYSAD_12	BIDIR	8	40	74	LOW
Pin 232 - J3	SYSAD_11	BIDIR	7	39	74	LOW
Pin 233 - H2		NC	0	0	0	
Pin 234 - J4	SYSAD_10	BIDIR	6	38	74	LOW
Pin 235 - H3	SYSAD_9	BIDIR	35	67	74	LOW
Pin 236 - J5	SYSAD_8	BIDIR	34	66	74	LOW
Pin 237 - G1	SYSAD_7	BIDIR	33	65	73	LOW
Pin 238 - H4	SYSAD_6	BIDIR	32	64	73	LOW
Pin 239 - G2	VDD4	POWER	0	0	0	
Pin 240 - H5	VSS2_4	POWER	0	0	0	

MUX Pin Cross-Reference Listing (299 Pin Package cont)

Pin Number	Name	Type	In	Out	Enable	Active
Pin 241 - G3	SYSAD_5	BIDIR	31	63	73	LOW
Pin 242 - F1	SYSAD_4	BIDIR	30	62	73	LOW
Pin 243 - F2	SYSAD_3	BIDIR	27	59	73	LOW
Pin 244 - F3	SYSAD_2	BIDIR	16	48	73	LOW
Pin 245 - E1	SYSAD_1	BIDIR	36	68	73	LOW
Pin 246 - E2	SYSAD_0	BIDIR	5	37	73	LOW
Pin 247 - E3	SYSAD_PAR_3	BIDIR	3	71	77	LOW
Pin 248 - G4		NC	0	0	0	
Pin 249 - D2	SYSAD_PAR_2	BIDIR	2	70	77	LOW
Pin 250 - G5	SYSAD_PAR_1	BIDIR	4	72	77	LOW
Pin 251 - D3	SYSAD_PAR_0	BIDIR	1	69	77	LOW
Pin 252 - F4	VDD4	POWER	0	0	0	
Pin 253 - C2	VSS	POWER	0	0	0	
Pin 254 - E4	MASDLY	INPUT	254	0	0	
Pin 255 - B2	SLVDLY	INPUT	257	0	0	
Pin 256 - F5	VDD3	POWER	0	0	0	
Pin 257 - D4		NC	0	0	0	

The pin number column shows the pin number/grid array location of each signal pin.
The type column shows the direction of each signal pin.
The in/out columns show the position of each signal pin within the top level boundary scan chain.
The Enable column shows the position of each bidirectional signal pin’s output enable control signal within the top level boundary scan chain.
The active columns the polarity for each output enable control signal.

For example, the input flop of SYSAD_5 is the thirty-first element in the top level boundary scan chain, and the output flop of SYSAD_5 is the sixty-third element in the scan chain. The enable signal for SYSAD_5 is active low and is the seventy-third element in the top level scan chain.

4.0 MUX Pin Cross-Reference Listing (240 Pin Package)

Pin Number	Name	Type	In	Out	Enable	Active
Pin 1 - 1	VDD	POWER	0	0	0	
Pin 2 - 2	MC_DLY	INPUT	255	0	0	
Pin 3 - 3	GIOSTB	INPUT	248	0	0	
Pin 4 - 4	GIOSEL	INPUT	247	0	0	
Pin 5 - 5	GIO_DATA_31	BIDIR	293	323	259	LOW
Pin 6 - 6	GIO_DATA_30	BIDIR	292	322	259	LOW
Pin 7 - 7	GIO_DATA_29	BIDIR	290	321	259	LOW
Pin 8 - 8	VSS	POWER	0	0	0	
Pin 9 - 9	VDD	POWER	0	0	0	
Pin 10 - 10	GIO_DATA_28	BIDIR	289	320	259	LOW
Pin 11 - 11	GIO_DATA_27	BIDIR	288	319	259	LOW
Pin 12 - 12	GIO_DATA_26	BIDIR	287	318	259	LOW
Pin 13 - 13	GIO_DATA_25	BIDIR	286	316	259	LOW
Pin 14 - 14	GIO_DATA_24	BIDIR	285	315	259	LOW
Pin 15 - 15	VSS	POWER	0	0	0	
Pin 16 - 16	VDD	POWER	0	0	0	
Pin 17 - 17	PAR_FLUSH	INPUT	246	0	0	
Pin 18 - 18	CPUSEL	INPUT	162	0	0	
Pin 19 - 19	GIO_DATA_23	BIDIR	284	314	260	LOW
Pin 20 - 20	GIO_DATA_22	BIDIR	283	313	260	LOW
Pin 21 - 21	GIO_DATA_21	BIDIR	282	312	260	LOW
Pin 22 - 22	GIO_DATA_20	BIDIR	281	311	260	LOW
Pin 23 - 23	GIO_DATA_19	BIDIR	279	310	260	LOW
Pin 24 - 24	GIO_DATA_18	BIDIR	278	309	260	LOW
Pin 25 - 25	GIO_DATA_17	BIDIR	277	308	260	LOW
Pin 26 - 26	GIO_DATA_16	BIDIR	276	307	260	LOW
Pin 27 - 27	GIO_DATA_15	BIDIR	275	305	261	LOW
Pin 28 - 28	GIO_DATA_14	BIDIR	274	304	261	LOW
Pin 29 - 29	VDD	POWER	0	0	0	
Pin 30 - 30	VSS	POWER	0	0	0	
Pin 31 - 31	VSS3	POWER	0	0	0	
Pin 32 - 32	GIO_DATA_13	BIDIR	273	303	261	LOW
Pin 33 - 33	GIO_DATA_12	BIDIR	272	302	261	LOW
Pin 34 - 34	GIO_DATA_11	BIDIR	271	301	261	LOW
Pin 35 - 35	GIO_DATA_10	BIDIR	270	300	261	LOW
Pin 36 - 36	GIO_DATA_9	BIDIR	268	299	261	LOW
Pin 37 - 37	GIO_DATA_8	BIDIR	267	298	261	LOW
Pin 38 - 38	GIO_DATA_7	BIDIR	266	297	262	LOW
Pin 39 - 39	GIO_DATA_6	BIDIR	265	296	262	LOW
Pin 40 - 40	GIO_DATA_5	BIDIR	264	325	262	LOW
Pin 41 - 41	GIO_DATA_4	BIDIR	294	324	262	LOW
Pin 42 - 42	VDD	POWER	0	0	0	
Pin 43 - 43	VSS	POWER	0	0	0	
Pin 44 - 44	GIO_DATA_3	BIDIR	291	317	262	LOW
Pin 45 - 45	GIO_DATA_2	BIDIR	280	306	262	LOW
Pin 46 - 46	GIO_DATA_1	BIDIR	269	326	262	LOW
Pin 47 - 47	GIO_DATA_0	BIDIR	263	295	262	LOW
Pin 48 - 48	GIO_PAR_3	BIDIR	330	334	258	LOW
Pin 49 - 49	GIO_PAR_2	BIDIR	329	333	258	LOW
Pin 50 - 50	GIO_PAR_1	BIDIR	328	332	258	LOW
Pin 51 - 51	GIO_PAR_0	BIDIR	327	331	258	LOW
Pin 52 - 52	GRAPHICS_1	INPUT	250	0	0	
Pin 53 - 53	GRAPHICS_0	INPUT	249	0	0	
Pin 54 - 54	GRXDLY_2	INPUT	334	0	0	
Pin 55 - 55	GRXDLY_1	INPUT	253	0	0	
Pin 56 - 56	GRXDLY_0	INPUT	251	0	0	
Pin 57 - 57	GIO_PLL_LP1	PLL	0	0	0	
Pin 58 - 58	VDD	POWER	0	0	0	
Pin 59 - 59	VSS	POWER	0	0	0	
Pin 60 - 60	VDD3	POWER	0	0	0	

MUX Pin Cross-Reference Listing (240 Pin Package cont)

Pin Number	Name	Type	In	Out	Enable	Active
Pin 61 - 61	VDD	POWER	0	0	0	
Pin 62 - 62	VSS	POWER	0	0	0	
Pin 63 - 63	GIO_PLL_LP2	PLL	0	0	0	
Pin 64 - 64	GIO_CLK	CLOCK	0	0	0	
Pin 65 - 65	GIO_PLL_AGND	PLL	0	0	0	
Pin 66 - 66	GIO_PLL_VSS	POWER	0	0	0	
Pin 67 - 67	GIO_PLL_VDD	POWER	0	0	0	
Pin 68 - 68	MEM_A_31	BIDIR	199	87	83	LOW
Pin 69 - 69	MEM_A_30	BIDIR	198	88	83	LOW
Pin 70 - 70	MEM_A_29	BIDIR	197	89	83	LOW
Pin 71 - 71	MEM_A_28	BIDIR	196	90	83	LOW
Pin 72 - 72	MEM_A_27	BIDIR	195	91	83	LOW
Pin 73 - 73	MEM_A_26	BIDIR	194	92	83	LOW
Pin 74 - 74	VSS2	POWER	0	0	0	
Pin 75 - 75	VDD	POWER	0	0	0	
Pin 76 - 76	MEM_A_25	BIDIR	193	93	83	LOW
Pin 77 - 77	MEM_A_24	BIDIR	192	94	83	LOW
Pin 78 - 78	MEM_A_23	BIDIR	191	95	84	LOW
Pin 79 - 79	MEM_A_22	BIDIR	190	96	84	LOW
Pin 80 - 80	MEM_A_21	BIDIR	189	97	84	LOW
Pin 81 - 81	MEM_A_20	BIDIR	188	98	84	LOW
Pin 82 - 82	MEM_A_19	BIDIR	187	99	84	LOW
Pin 83 - 83	MEM_A_18	BIDIR	186	100	84	LOW
Pin 84 - 84	MEM_A_17	BIDIR	185	101	84	LOW
Pin 85 - 85	MEM_A_16	BIDIR	184	102	84	LOW
Pin 86 - 86	VSS3	POWER	0	0	0	
Pin 87 - 87	VSS	POWER	0	0	0	
Pin 88 - 88	VDD	POWER	0	0	0	
Pin 89 - 89	MEM_A_15	BIDIR	183	103	85	LOW
Pin 90 - 90	MEM_A_14	BIDIR	182	104	85	LOW
Pin 91 - 91	MEM_A_13	BIDIR	181	105	85	LOW
Pin 92 - 92	MEM_A_12	BIDIR	180	106	85	LOW
Pin 93 - 93	MEM_A_11	BIDIR	179	107	85	LOW
Pin 94 - 94	MEM_A_10	BIDIR	178	108	85	LOW
Pin 95 - 95	MEM_A_9	BIDIR	177	109	85	LOW
Pin 96 - 96	MEM_A_8	BIDIR	176	110	85	LOW
Pin 97 - 97	MEM_A_7	BIDIR	175	111	86	LOW
Pin 98 - 98	MEM_A_6	BIDIR	174	112	86	LOW
Pin 99 - 99	PAR_ERR_3	OUTPUT	0	240	0	
Pin 100 - 100	PAR_ERR_2	OUTPUT	0	239	0	
Pin 101 - 101	PAR_ERR_1	OUTPUT	0	241	0	
Pin 102 - 102	PAR_ERR_0	OUTPUT	0	238	0	
Pin 103 - 103	VDD	POWER	0	0	0	
Pin 104 - 104	VSS	POWER	0	0	0	
Pin 105 - 105	MEM_A_5	BIDIR	173	113	86	LOW
Pin 106 - 106	MEM_A_4	BIDIR	172	114	86	LOW
Pin 107 - 107	MEM_A_3	BIDIR	171	115	86	LOW
Pin 108 - 108	MEM_A_2	BIDIR	170	116	86	LOW
Pin 109 - 109	MEM_A_1	BIDIR	169	117	86	LOW
Pin 110 - 110	MEM_A_0	BIDIR	166	118	86	LOW
Pin 111 - 111	MEM_A_PAR_3	BIDIR	168	119	82	LOW
Pin 112 - 112	MEM_A_PAR_2	BIDIR	167	120	82	LOW
Pin 113 - 113	MEM_A_PAR_1	BIDIR	201	121	82	LOW
Pin 114 - 114	MEM_A_PAR_0	BIDIR	200	122	82	LOW
Pin 115 - 115	TP1	TEST	0	0	0	
Pin 116 - 116	TP0	TEST	0	0	0	
Pin 117 - 117	JTDO	JTAG	0	0	0	
Pin 118 - 118	JTMS	JTAG	0	0	0	
Pin 119 - 119	VSS	POWER	0	0	0	
Pin 120 - 120	VDD	POWER	0	0	0	

MUX Pin Cross-Reference Listing (240 Pin Package cont)

Pin Number	Name	Type	In	Out	Enable	Active
Pin 121 - 121	VDD3	POWER	0	0	0	
Pin 122 - 122	PLL_RESET_N	PLL	0	0	0	
Pin 123 - 123	ENTEI	OTHER	0	0	0	
Pin 124 - 124	JTDI	JTAG	0	0	0	
Pin 125 - 125	JTCK	JTAG	0	0	0	
Pin 126 - 126	CPU_MEM_OE	INPUT	163	0	0	
Pin 127 - 127	MEM_B_31	BIDIR	235	123	78	LOW
Pin 128 - 128	MEM_B_30	BIDIR	234	124	78	LOW
Pin 129 - 129	MEM_B_29	BIDIR	233	125	78	LOW
Pin 130 - 130	MEM_B_28	BIDIR	232	126	78	LOW
Pin 131 - 131	MEM_B_27	BIDIR	231	127	78	LOW
Pin 132 - 132	MEM_B_26	BIDIR	230	128	78	LOW
Pin 133 - 133	VSS	POWER	0	0	0	
Pin 134 - 134	VDD	POWER	0	0	0	
Pin 135 - 135	MEM_B_25	BIDIR	229	129	78	LOW
Pin 136 - 136	MEM_B_24	BIDIR	228	130	78	LOW
Pin 137 - 137	MEM_B_23	BIDIR	227	131	79	LOW
Pin 138 - 138	MEM_B_22	BIDIR	226	132	79	LOW
Pin 139 - 139	MEM_B_21	BIDIR	225	133	79	LOW
Pin 140 - 140	MEM_B_20	BIDIR	224	134	79	LOW
Pin 141 - 141	MEM_B_19	BIDIR	223	135	79	LOW
Pin 142 - 142	MEM_B_18	BIDIR	222	136	79	LOW
Pin 143 - 143	MEM_B_17	BIDIR	221	137	79	LOW
Pin 144 - 144	MEM_B_16	BIDIR	220	138	79	LOW
Pin 145 - 145	CEN_FIFO	INPUT	161	0	0	
Pin 146 - 146	BEN_CTRL	INPUT	160	0	0	
Pin 147 - 147	AEN_MEM	INPUT	159	0	0	
Pin 148 - 148	VSS2	POWER	0	0	0	
Pin 149 - 149	VSS3	POWER	0	0	0	
Pin 150 - 150	VDD	POWER	0	0	0	
Pin 151 - 151	MEM_B_15	BIDIR	219	139	80	LOW
Pin 152 - 152	MEM_B_14	BIDIR	218	140	80	LOW
Pin 153 - 153	MEM_B_13	BIDIR	217	141	80	LOW
Pin 154 - 154	MEM_B_12	BIDIR	216	142	80	LOW
Pin 155 - 155	MEM_B_11	BIDIR	215	143	80	LOW
Pin 156 - 156	MEM_B_10	BIDIR	214	162	80	LOW
Pin 157 - 157	MEM_B_9	BIDIR	213	145	80	LOW
Pin 158 - 158	MEM_B_8	BIDIR	212	146	80	LOW
Pin 159 - 159	MEM_B_7	BIDIR	211	147	81	LOW
Pin 160 - 160	MEM_B_6	BIDIR	210	148	81	LOW
Pin 161 - 161	DATA_SEL_2	INPUT	244	0	0	
Pin 162 - 162	DATA_SEL_1	INPUT	245	0	0	
Pin 163 - 163	DATA_SEL_0	INPUT	243	0	0	
Pin 164 - 164	READ	INPUT	256	0	0	
Pin 165 - 165	VDD	POWER	0	0	0	
Pin 166 - 166	VSS	POWER	0	0	0	
Pin 167 - 167	MEM_B_5	BIDIR	209	149	81	LOW
Pin 168 - 168	MEM_B_4	BIDIR	208	150	81	LOW
Pin 169 - 169	MEM_B_3	BIDIR	207	151	81	LOW
Pin 170 - 170	MEM_B_2	BIDIR	206	152	81	LOW
Pin 171 - 171	MEM_B_1	BIDIR	205	153	81	LOW
Pin 172 - 172	MEM_B_0	BIDIR	202	154	81	LOW
Pin 173 - 173	MEM_B_PAR_3	BIDIR	204	155	82	LOW
Pin 174 - 174	MEM_B_PAR_2	BIDIR	203	156	82	LOW
Pin 175 - 175	MEM_B_PAR_1	BIDIR	237	157	82	LOW
Pin 176 - 176	MEM_B_PAR_0	BIDIR	236	158	82	LOW
Pin 177 - 177	RESET_N	INPUT	242	0	0	
Pin 178 - 178	VDD	POWER	0	0	0	
Pin 179 - 179	VSS	POWER	0	0	0	
Pin 180 - 180	VDD3	POWER	0	0	0	

MUX Pin Cross-Reference Listing (240 Pin Package cont)

Pin Number	Name	Type	In	Out	Enable	Active
Pin 181 - 181	VDD	POWER	0	0	0	
Pin 182 - 182	CPU_PLL_VDD	POWER	0	0	0	
Pin 183 - 183	CPU_PLL_VSS	POWER	0	0	0	
Pin 184 - 184	CPU_CLK	CLOCK	0	0	0	
Pin 185 - 185	CPU_PLL_LP2	PLL	0	0	0	
Pin 186 - 186	CPU_PLL_LP1	PLL	0	0	0	
Pin 187 - 187	CPU_PLL_AGND	PLL	0	0	0	
Pin 188 - 188	MUX_DIR	INPUT	165	0	0	
Pin 189 - 189	CPU_PUSH	INPUT	164	0	0	
Pin 190 - 190	VSS	POWER	0	0	0	
Pin 191 - 191	VDD4	POWER	0	0	0	
Pin 192 - 192	SYSAD_31	BIDIR	29	61	76	LOW
Pin 193 - 193	SYSAD_30	BIDIR	28	60	76	LOW
Pin 194 - 194	SYSAD_29	BIDIR	26	58	76	LOW
Pin 195 - 195	SYSAD_28	BIDIR	25	57	76	LOW
Pin 196 - 196	SYSAD_27	BIDIR	24	56	76	LOW
Pin 197 - 197	SYSAD_26	BIDIR	23	55	76	LOW
Pin 198 - 198	VSS	POWER	0	0	0	
Pin 199 - 199	VDD4	POWER	0	0	0	
Pin 200 - 200	SYSAD_25	BIDIR	22	54	76	LOW
Pin 201 - 201	SYSAD_24	BIDIR	21	53	76	LOW
Pin 202 - 202	SYSAD_23	BIDIR	20	52	75	LOW
Pin 203 - 203	SYSAD_22	BIDIR	19	51	75	LOW
Pin 204 - 204	SYSAD_21	BIDIR	18	50	75	LOW
Pin 205 - 205	SYSAD_20	BIDIR	17	49	75	LOW
Pin 206 - 206	SYSAD_19	BIDIR	15	47	75	LOW
Pin 207 - 207	SYSAD_18	BIDIR	14	46	75	LOW
Pin 208 - 208	SYSAD_17	BIDIR	13	45	75	LOW
Pin 209 - 209	SYSAD_16	BIDIR	12	44	75	LOW
Pin 210 - 210	VSS	POWER	0	0	0	
Pin 211 - 211	VSS3_4	POWER	0	0	0	
Pin 212 - 212	VDD4	POWER	0	0	0	
Pin 213 - 213	SYSAD_15	BIDIR	11	43	74	LOW
Pin 214 - 214	SYSAD_14	BIDIR	10	42	74	LOW
Pin 215 - 215	SYSAD_13	BIDIR	9	41	74	LOW
Pin 216 - 216	SYSAD_12	BIDIR	8	40	74	LOW
Pin 217 - 217	SYSAD_11	BIDIR	7	39	74	LOW
Pin 218 - 218	SYSAD_10	BIDIR	6	38	74	LOW
Pin 219 - 219	SYSAD_9	BIDIR	35	67	74	LOW
Pin 220 - 220	SYSAD_8	BIDIR	34	66	74	LOW
Pin 221 - 221	SYSAD_7	BIDIR	33	65	73	LOW
Pin 222 - 222	SYSAD_6	BIDIR	32	64	73	LOW
Pin 223 - 223	VDD4	POWER	0	0	0	
Pin 224 - 224	VSS2_4	POWER	0	0	0	
Pin 225 - 225	SYSAD_5	BIDIR	31	63	73	LOW
Pin 226 - 226	SYSAD_4	BIDIR	30	62	73	LOW
Pin 227 - 227	SYSAD_3	BIDIR	27	59	73	LOW
Pin 228 - 228	SYSAD_2	BIDIR	16	48	73	LOW
Pin 229 - 229	SYSAD_1	BIDIR	36	68	73	LOW
Pin 230 - 230	SYSAD_0	BIDIR	5	37	73	LOW
Pin 231 - 231	SYSAD_PAR_3	BIDIR	3	71	77	LOW
Pin 232 - 232	SYSAD_PAR_2	BIDIR	2	70	77	LOW
Pin 233 - 233	SYSAD_PAR_1	BIDIR	4	72	77	LOW
Pin 234 - 234	SYSAD_PAR_0	BIDIR	1	69	77	LOW
Pin 235 - 235	VDD4	POWER	0	0	0	
Pin 236 - 236	VSS	POWER	0	0	0	
Pin 237 - 237	MASDLY	INPUT	254	0	0	
Pin 238 - 238	SLVDLY	INPUT	257	0	0	
Pin 239 - 239	VDD3	POWER	0	0	0	
Pin 240 - 240		NC	0	0	0	

5.0 MUX Output Drive Strengths

The Following table shows the drive strengths and associated LSI100K IO components for the MUX outputs:

Bus	LSI IO Component	Drive Strength	Voltage
SYSAD	BD4TL	4 mA TTL	3 Volt
GIO(31..16)	BD8TRPU	8 mA TTL w/ Pullup (moderate Slew Rate Control)	5 Volt
GIO(15..0)	BD8TRP	8 mA TTL (moderate Slew Rate Control)	5 Volt
MEM_A	BD4T	4 mA TTL	5 Volt
MEM_B	BD4T	4 mA TTL	5 Volt
PAR_ERR	BT4M	4 mA TTL w/ Tri-State enable	5 Volt
JTDO	B4	4 mA	5 Volt

6.0 MUX Input Buffers

The Following table shows the input buffers used for each MUX input signal:

Signal	LSI IO Component	Notes
SYSAD	BD4TL	3 Volt
GIO(31..16)	BD8TRPU	Pulled-up
GIO(31..0)	BD8TRP	
MEM_A	BD4T	
MEM_B	BD4T	
GIOSEL	TLCHT	
CPUSEL	TLCHT	
CPU_PUSH	TLCHT	
CPU_MEM_OE	TLCHT	
DATA_SEL	TLCHT	
MUX_DIR	TLCHT	
GRAPHICS	TLCHT	
AEN_MEM	TLCHT	
BEN_CTRL	TLCHT	
CEN_FIFO	TLCHT	
GIOSTB	TLCHT	
PAR_FLUSH	TLCHT	
MASDLY	TLCHT	
SLVDLY	TLCHT	
MC_DLY	TLCHT	
READ	TLCHT	
GRXDLY	TLCHT	
GIO_CLK	DDRV	
CPU_CLK	DDRV	
RESET_N	TLCHT	
JTDI	TLCHTU	Pulled-up
JTMS	TLCHTU	Pulled-up
PLL_RESET_N	TLCHTU	Pulled-up
TP0	TLCHTU	Pulled-up
TP1	TLCHTU	Pulled-up
ENTEI	TLCHN	Inverted
PLL_VSS	RDDRV	
PLL_VDD	RDDRV	
PLL_LP2	RDDRVPD	
PLL_AGND	RDDRVO	

7.0 MUX Set-up Times

The Following tables shows the setup times for the MUX inputs:

Signal	Set-up Time(ns)
SYSAD	5.18 (to cpu_clk)
GIO	3.81 (to gio_clk)
MEM_A	4.71 (to cpu_clk)
MEM_A	4.10 (to gio_clk)
MEM_B	4.71 (to cpu_clk)
MEM_B	4.10 (to gio_clk)
CPU_PUSH	3.65 (to cpu_clk)
CPU_MEM_OE	3.76 (to cpu_clk)
DATA_SEL	3.50 (to gio_clk)
DATA_SEL	3.67 (to cpu_clk)
MUX_DIR	3.65 (to gio_clk)
MUX_DIR	3.82 (to cpu_clk)
GRAPHICS	3.99 (to gio_clk)
AEN_MEM	3.53 (to gio_clk)
AEN_MEM	3.70 (to cpu_clk)
BEN_CTRL	3.51 (to gio_clk)
BEN_CTRL	3.68 (to cpu_clk)
CEN_FIFO	3.56 (to gio_clk)
CEN_FIFO	3.73 (to cpu_clk)
GIOSTB	3.51 (to gio_clk)
PAR_FLUSH	3.78 (to gio_clk)
PAR_FLUSH	3.95 (to cpu_clk)
MASDLY	3.58 (to gio_clk)
SLVDLY	3.67 (to gio_clk)
MC_DLY	3.60 (to gio_clk)
READ	3.45 (to gio_clk)
GRXDLY	13.88 (to gio_clk)

8.0 MUX Hold Times

The Following tables shows the hold times for the MUX inputs:

Signal	Hold Time(ns)
SYSAD	-2.78 (to cpu_clk)
GIO	-1.61 (to gio_clk)
MEM_A	-2.58 (to cpu_clk)
MEM_A	-1.87 (to gio_clk)
MEM_B	-2.59 (to cpu_clk)
MEM_B	-1.87 (to gio_clk)
CPU_PUSH	-1.80 (to cpu_clk)
CPU_MEM_OE	-1.73 (to cpu_clk)
DATA_SEL	-1.46 (to gio_clk)
DATA_SEL	-1.58 (to cpu_clk)
MUX_DIR	-1.80 (to gio_clk)
MUX_DIR	-1.80 (to cpu_clk)
GRAPHICS	-2.21 (to gio_clk)
AEN_MEM	-1.65 (to gio_clk)
AEN_MEM	-1.65 (to cpu_clk)
BEN_CTRL	-1.64 (to gio_clk)
BEN_CTRL	-1.64 (to cpu_clk)
CEN_FIFO	-1.69 (to gio_clk)
CEN_FIFO	-1.69 (to cpu_clk)
GIOSTB	-1.83 (to gio_clk)
PAR_FLUSH	-1.97 (to gio_clk)
PAR_FLUSH	-1.97 (to cpu_clk)
MASDLY	-1.72 (to gio_clk)
SLVDLY	-1.83 (to gio_clk)
MC_DLY	-1.74 (to gio_clk)
READ	-1.55 (to gio_clk)
GRXDLY	-2.65 (to gio_clk)

9.0 MUX Output Delays

The Following table shows the MUX output delays for 50 pF loading:

Signal	Worse Case Output Delay (ns)
SYSAD	14.26 (from cpu_clk)
MEM_A	21.42 (from cpu_clk)
MEM_A	21.43 (from gio_clk)
MEM_B	22.07 (from cpu_clk)
MEM_B	21.63 (from gio_clk)
GIO_DATA	13.85 (from gio_clk)

10.0 MUX Rev 1.2 Changes

The original spin of MUX contained a bug where bad parity was inadvertantly written to memory. The source of this bug lies with the definition of the par_flush signal, which is a multiplexed control line. When cpu_sel is asserted, an active par_flush signal will generate bad parity storage for CPU writes to memory. When gio_sel is asserted, an active par_flush signal will flush the gio command fifo, gio write buffer and gio read buffer. In the original version of MUX, when par_flush was asserted to flush the gio side of the chip and a CPU write occurred while par_flush was still asserted, bad parity would be generated for the CPU data to memory because par_flush was not qualified with cpu_sel. An AND gate was added in the second spin of MUX to qualify par_flush with cpu_sel in determining when to genereate intentional bad parity.

11.0 MUX Problems

There is a know problem that is believed to be with DMUX but the root cause of the problem is not completely known. The problem is when bad parity is generated by CPU writes to memory when the par_flush signal is asserted it appears to mess up the GIO memory fifo. This was found in IDE, when the parity test was run followed by the SCSI test, the SCSI test would sometime fail. If a VDMA was done between the parity and SCSI test that gets preempted the SCSI test always passes.

12.0 MUX Power

The DMUX power was measured in a Blackjack system.

3.3V	10 mA	33 mW
5.0V	250 mA	1250 mW
Total		1.28 W

13.0 Corrections and Additions

June 25, 1992:

a. csize64 is no longer an input pin.
b. depth of cpu write buffer is 32 entries deep
c. added expanded cpu write buffer drawing
d. added expanded gio write buffer drawing
e. added expanded gio read buffer drawing
f. data_sel(2) is used during gio reads to determine whether to turn off gio bus drivers
g. added specification of jtag and pll pins
h. added MUX pin cross-reference listing for both packages
i. added MUX rev 1.2 changes
j. added MUX output drive strengths
k. added MUX input buffers
l. added MUX setup/hold times
m. added MUX output delays

Silicon Graphics
Computer Systems

DMUX Chip Specification
R4000 Project

Draft 2.5
May 13, 2000

James Tornes
Prasad Paranjpe
Scott Sellers

Silicon Graphics Inc.

Table of Contents