METHEUS
OMEGA-GPU
GRAPHICS PROCESSING UNIT

Features

- 16 MHZ Clock
- Microcode in external RAM
- 16 Bit Processor Optimised for Graphic Data Processing
- 64 Bit Parallel Instruction Opcode
- 16 - 16 Bit Internal Registers
- Complete Graphics System when combined with companion Memory Control Unit.
- One 64 Bit Instruction executed per machine cycle.
- 16 Bit Loop Counter
- 6 Level Microprogram Stack
- Four Way Branches
- Writable Control Store Boot from a single PROM

Product Description

The OMEGA-GPU integrates the data processing portion of a two chip graphics drawing processor. This single data processing component can drive up to eight Memory Control Unit components (OMEGA-MCU) for up to 32 bits-per-pixel operation.

The OMEGA-GPU contains a complete 16-bit microcodable processor, microcode address sequencer, host interface and Graphics System Bus (Z-bus). The data processor consists of 16 general purpose registers, a 16-bit ALU, and a 16-bit accumulator. The external Writable Control Store (WCS) is addressed by a 15-bit microcode address generator. The 16-bit ALU implements all common arithmetic and logical operations plus a byte swap operation. The 8-bit Host Interface allows the host both a data communication capability and microcode debug operations. The Z bus is used to communicate between the GPU master and the other sections of a complete graphics system.

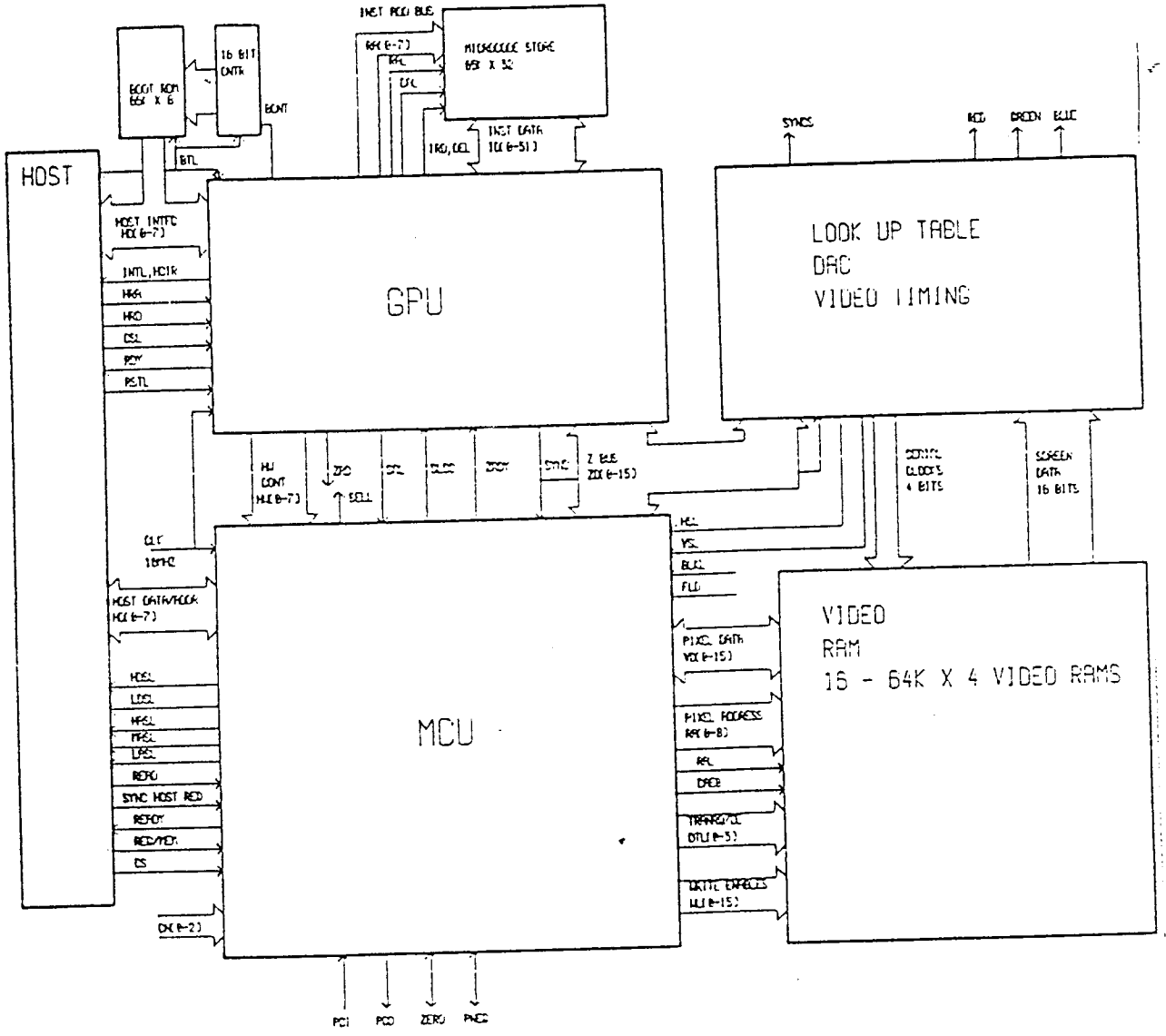
The OMEGA-GPU is implemented in low power CMOS technology and operates at input clock speeds up to 16 MHz.

The OMEGA-GPU is packaged in a low cost, 100 pin plastic flat-pack.

Functional Description

The OMEGA-GPU executes instructions from the 32-bit instruction memory (also called Writable Control Store) to interpret host graphic commands into commands for one to eight OMEGA-MCU components connected to the GPU throught the Graphic System Bus (Z Bus). Programs are available for the OMEGA-GPU to implement drawing primitives (lines, circles, etc.), fills, raster-ops, 2D transformation, and clipping.

GPU Based Graphics System Block Diagram
Four Bits per Pixel



Functional Description (continued)

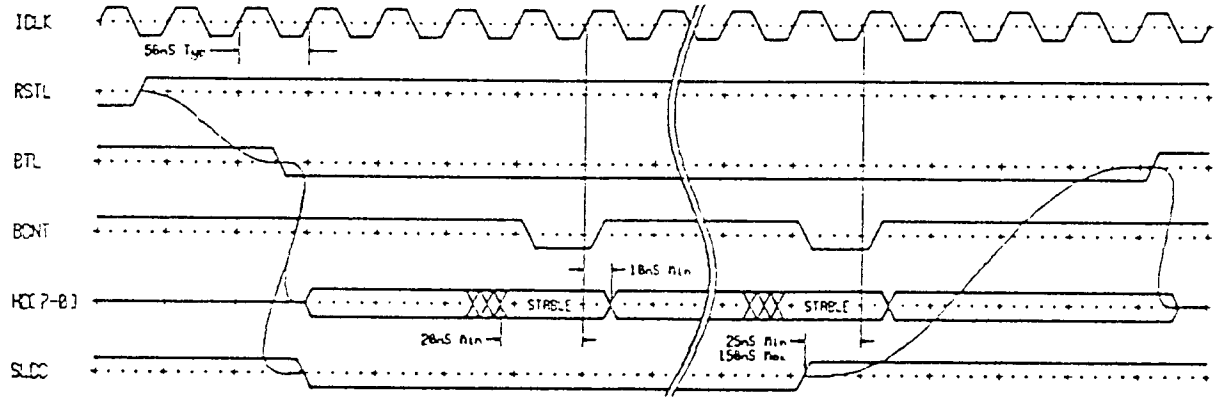
Booting the Instruction Memory

The instruction memory is loaded in one of two ways:

1) Auto Boot, or
2) Host Boot.

The auto boot process is initiated by reseting the GPU (RSTL signal) with the BTL signal high and SLCC low. This will cause the GPU to begin reading from an external ROM (Read Only Memory) attached to the Host data bus. After each read cycle, the GPU will increment an external counter (with BCNT) used to access the Boot ROM and then read the next byte. After 4 bytes are read a 32-bit write operation is performed into the instruction memory. This process continues until the GPU reads a one on the SLCC pin (driven by the external counter carry). The GPU then begins execution of the program at address 0. Note that the external ROM address counter must always be reset to zero by the RSTL signal.

The Host Boot process is initiated by reseting the GPU with the BTL signal low. After detecting this condition the internal GPU boot logic will wait for the host to load the instruction memory through internal registers in the GPU provided for that purpose (see the Host Accesible Registers section). The host then commands the GPU to begin execution at address 0 again using internal GPU registers.



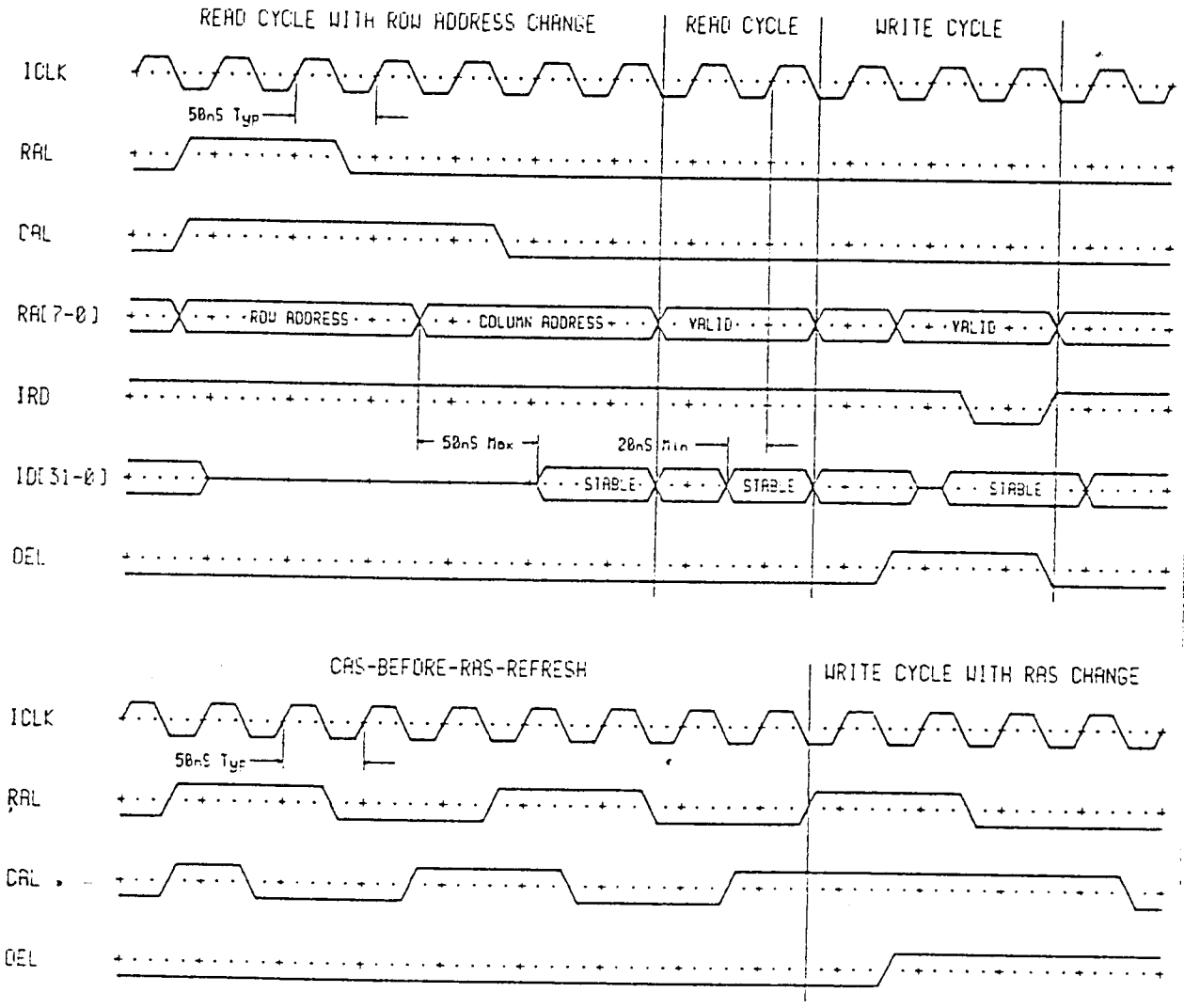
AUTO-BOOT TIMING

Functional Description (continued)

Instruction Memory Fetch

The instruction memory interface is optimised for low-cost Static Column dynamic Random Access Memories (SCRAM). The 16-bit address is multiplexed over 8 address lines using two control signals: Row Address strobe (RAL) and Column Address strobe (CAL). A third control signal, Instruction Read (IRD), is used in conjunction with CAL to determine if the instruction memory is to be read or written. The memory can also be implemented with Static RAM (SRAM) if an external latch is used to hold the last row address.

The internal GPU instruction memory control logic optimises the amount of address multiplexing that must be used by only asserting a row address when the most significant 8 bits of the address require changing. This feature requires that only "page mode" SCRAM or SRAM be used for instruction memory. Further, the refresh cycles required for use with SCRAM may be disabled when using SRAM. The GPU uses CAS-before-RAS refresh cycles.



INSTRUCTION MEMORY TIMING

Functional Description (continued)

Host Interface

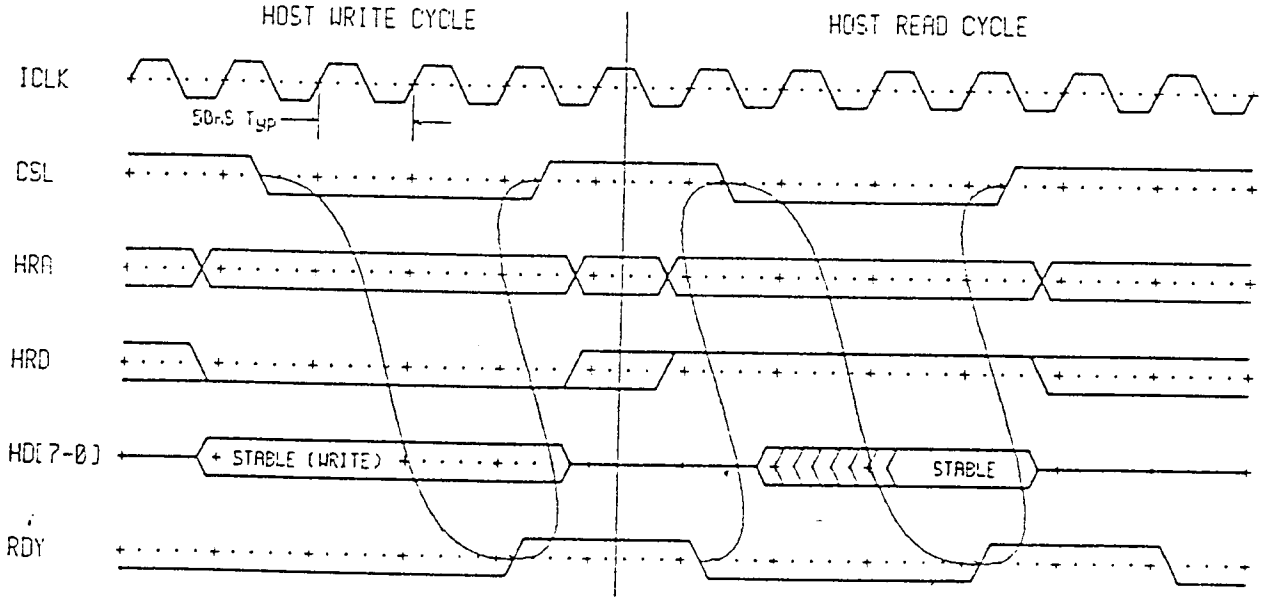
The Host Interface may be used for three types of operations.

1) Booting instruction memory as described earlier.
2) Diagnostics and debug of microcode.
3) Communication of Graphic commands to the GPU.

In any case, the host addresses one of a number of internal 8-bit registers of the GPU. In order to minimize the number of pins on the GPU and the address space consumed on the host, the GPU utilizes a indirect-register addressing scheme. The host uses only two addresses, the first contains a register pointer register and the second accesses one of sixteen 8-bit registers. The host loads the register pointer with the address of the register that is accessed with the second host address.

The host accesses the GPU with two mode signals and one control signal. The HRD mode signal selects a read operation if high and a write if low. The HRA mode signal selects the Register-Number-Register (register pointer) when high, and the selected host-accessible register when low. The main control signal is CSL. When CSL goes low, both mode signals must remain stable until the desired operation is completed. The CSL signal must remain low until the RDY signal goes high, indicating that the GPU has completed the operation. The GPU will clear RDY one system clock after the host has changed CSL to a high. The host must wait until RDY is low before bringing CSL low again to request another cycle.

CSL, HRD and HRA don't need to synchronized with the system clock, but the RDY signal will be synchronous.



HOST INTERFACE TIMING

Functional Description (continued)

Z Bus Interface

The Z Bus allows the GPU to communicate synchronously at a high rate of speed with slave devices it must support. The slave of primary importance is the Memory Control Unit (MCU) which directly manipulates the raster memory. Many other slaves may be addressed such as may be required to manipulate color maps, sync, graphics input, or other bus interfaces.

The Z bus can accommodate parallel operations.

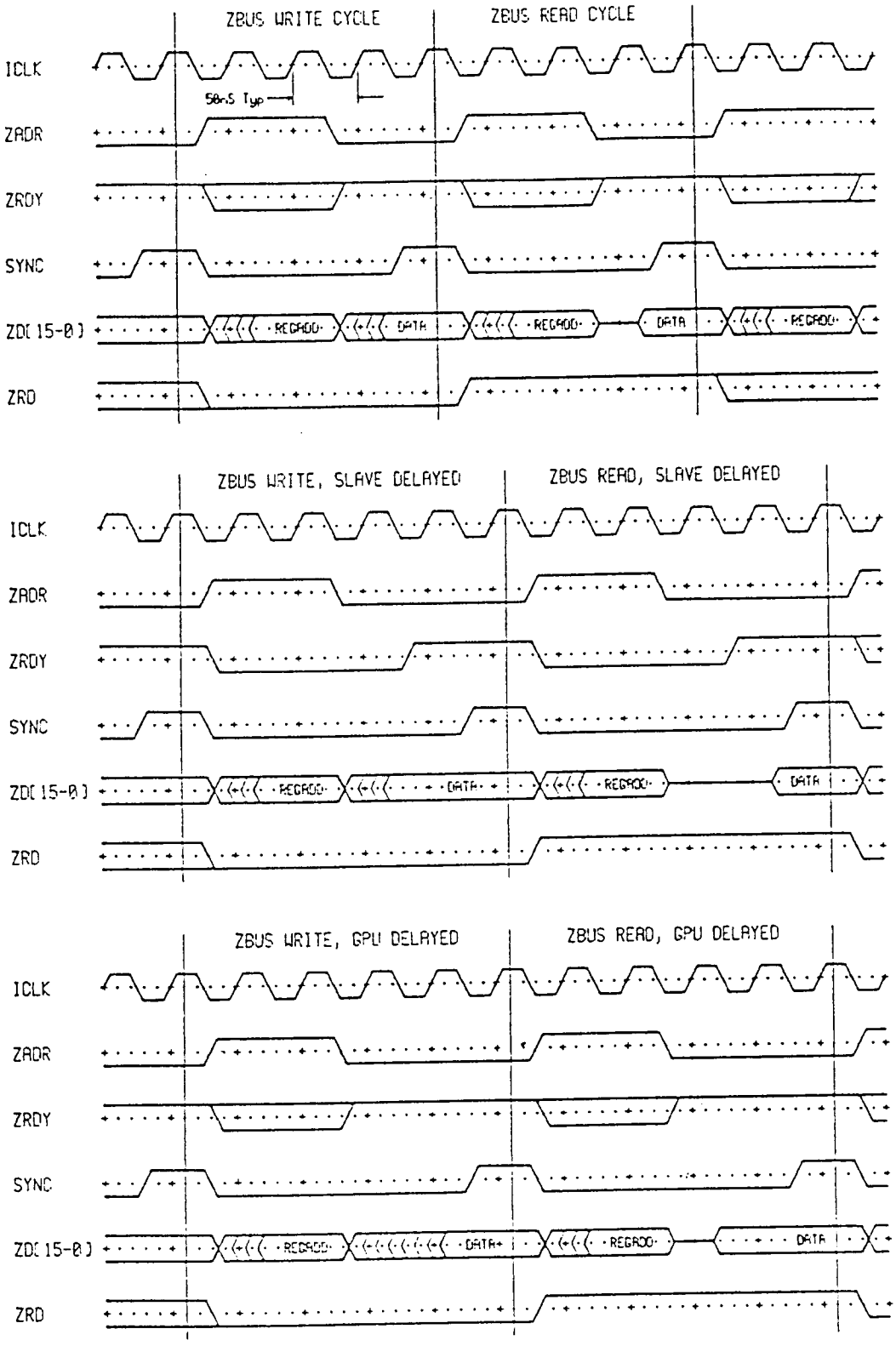
1) The main operation is register read and write. A Z bus cycle is split into two halves: a register select which takes 2 cycles, and the data operation, taking a minimum of two cycles. During the register select ZADR is high and a 6-bit register address is placed on the ZD[5-0] lines. ZD[6] also indicates a read or write, it is set if the the following data operation is a read. The selected slave decodes that address and produces data (on a read) as it also generates a ZRDY signal. The GPU then produces a SYNC signal, indicating the end of the Z bus cycle. When SYNC is high the data on ZD[15-0] is latched by the bus reader. The GPU selects a slave by writing to the reserved SELECT address (0x01) followed by a 16-bit data pattern in which each bit can turn a slave on or off. In this way up to 16 slaves may be used and any of the 16 may be selected at the same time for broadcast commands. The first 8 bits, ZD[7-0] are reserved for MCU chips. An additional register address (0x00) is reserved for the NOP register when a Z bus cycle occurs with no register operation. Several more register addresses are reserved for GPU internal registers as detailed later.

2) In addition to a register operation, the Z bus can execute hardware commands with the 8 hardware control lines HW[7-0]. These lines can implement 8 separate hardware commands or using various encodings implement more commands at a loss of parallelism. In either case the exact meaning of a specific hardware command byte is determined entirely by the selected slave device. Hardware commands are executed by the slave on the positive clock edge during SYNC.

3) The selected Z bus slave can also return to the GPU a single condition code line which is defined by the slave and sensed by the GPU only with appropriate microcode. The SLCC line must be synchronous and meet the timing conditions specified in the A.C. characteristics. It is sampled by the GPU during SYNC.

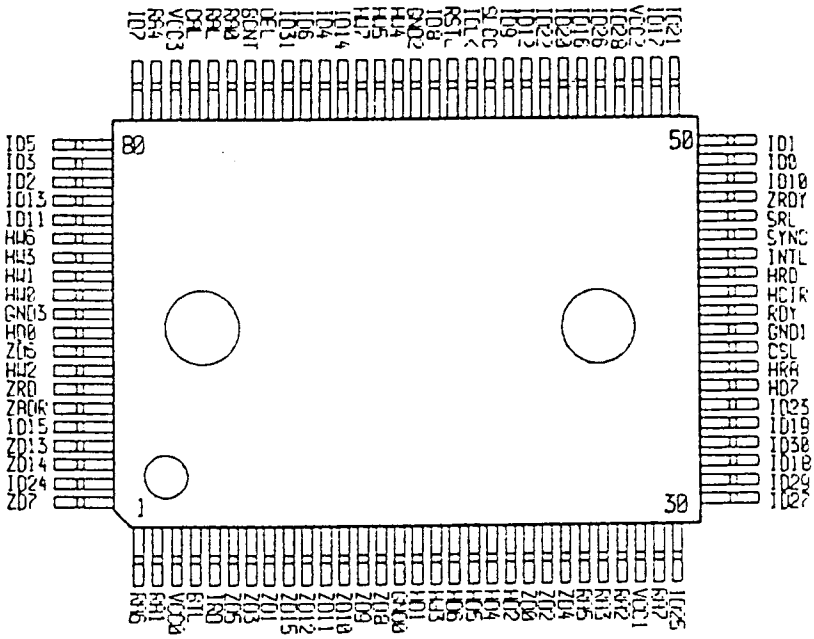
Functional Description (continued)

Typical Z Bus Cycles



Pin Descriptions

100 Pin Plastic Flat Pack



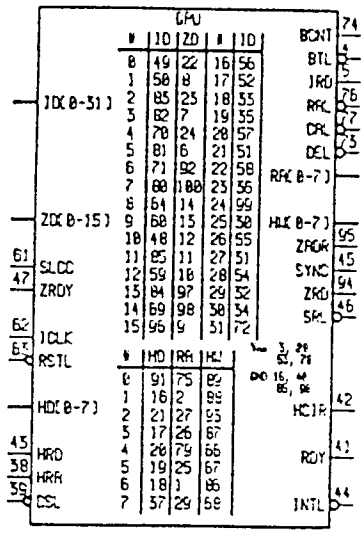
Pin Descriptions (continued)

Pin Names to Pin Numbers Alphabetically

BCNT	74	HWJ1	66	ID2	63	INTL	44	VCC3	78
BTL	4	HWJ2	93	ID20	57	IRD	5	ZADR	95
CAL	77	HWJ3	87	ID21	51	OEL	73	ZD0	22
CSL	39	HWJ4	66	ID22	58	RA0	75	ZD1	8
GND0	15	HWJ5	67	ID23	36	RA1	2	ZD10	12
GND1	40	HWJ6	86	ID24	99	RA2	27	ZD11	11
GND2	65	HWJ7	68	ID25	30	RA3	26	ZD12	10
GND3	90	ICLK	62	ID26	55	RA4	79	ZD13	97
HCIR	42	ID0	49	ID27	31	RA5	25	ZD14	98
HD0	91	ID1	50	ID28	54	RA6	1	ZD15	9
HD1	16	ID10	48	ID29	32	RA7	29	ZD2	23
HD2	21	ID11	85	ID3	62	RAL	76	ZD3	7
HD3	17	ID12	59	ID30	34	RDY	41	ZD4	24
HD4	20	ID13	84	ID31	72	RSTL	83	ZD5	6
HD5	19	ID14	69	ID4	70	SLCC	61	ZD6	92
HD6	18	ID15	96	ID5	81	SRL	46	ZD7	100
HD7	37	ID16	56	ID6	71	SYNC	45	ZD8	14
HRA	38	ID17	52	ID7	60	VCC0	3	ZD9	13
HRD	43	ID18	33	ID8	64	VCC1	28	ZRD	94
HWJ0	69	ID19	35	ID9	60	VCC2	53	ZRDY	47

Pin Descriptions (continued)

Schematic Symbol



Pin Descriptions (continued)

OMEGA-GPU Signals for Instruction Memory		
Signal	Pin	Direction
BTL	I/O, Open-Drain	The boot-enable pin is used to control the microcode boot process. Immediately after reset this pin is monitored as an input. If BTL is pulled high, the GPU initiates the auto-boot process, otherise the chip waits for the host to load microcode. During the auto-boot process, this pin is driven low and at the end of the boot it will float high.
BCNT	Output	The boot count pin is used to increment the boot ROM counter during the auto-boot process. The boot ROM counter must be incremented by the rising edge.
RA[7-0]	Output	The instruction RAM address pins communicate the address to be read or written to the instruction RAM. The address is a 16-bit address that is multiplexed out using the CAL and RAL strobes as required.
RAL	Output	The row address strobe indicates a valid row address is present on the RA bus. This is a low true signal.
CAL	Output	The column address strobe indicates that a valid column address is present on the RA bus. The instruction RAM must produce instruction data within the access time calculated from the clock frequency. CAL also requests CAS-before-RAS refresh cycles from the RAM if it is asserted before RAL. This is a low true signal.
IRD	Output	The instruction read pin drives the read/write control for the instruction memory. When IRD is high, the instruction RAM operation is a read. When IRD goes low, it is used as a write strobe (the GPU uses WE controled write cycles).
OEL	Output	The output enable pin controls the data output drivers on the instruction RAM. This is a low true signal.
ID[31-0]	I/O	The instruction data pins read instructions from the instruction RAM to control the execution of the GPU. They also drive the instruction RAM when writing data during booting and scratch pad operations.

Pin Descriptions (continued)

OMEGA-GPU Signals for Z Bus		
Signal	Pin	Direction
SRL	Output	The synchronous reset pin is a low true signal used to reset the Z bus slaves. It follows the RSTL signal input to the GPU by 2 clock cycles and is synchronous with the system clock.
SYNC	Output	The sync pin is used to synchronize the Z bus slaves with the GPU. Zbus data is accepted by the GPU (read) or the slave (write) on the positive edge of the system clock while sync is high. The next Z bus register address is valid on the rising edge of the second system clock after sync goes low.
ZADR	Output	The Z bus address pin indicates that an address is active on the ZD[6-0] pins instead of data. The selected slave should latch that address on the rising edge of the clock after ZADR is asserted.
ZRD	Output	The Z bus read pin indicates that the current Z bus direction is a read into the GPU.
ZD[15-0]	I/O	The 16 Z bus data pins transfer the register address and then the register data over the Z bus to (or from) the Z bus slaves. When ZADR is high, ZD[5-0] have the register address and ZD[6] has a 1 if the next data transfer is a read, or 0 for a write.
ZRDY	Input	The Z bus ready pin is driven by the slave when it is ready to accept data on a write or has driven the data on a read. The next clock cycle after ZRDY the GPU will assert SYNC to terminate the Z bus cycle if all of the GPU internal processing is complete.
SLCC	Input	The slave condition code pin is driven by the currently selected slave. The condition of this signal is testable by the sequencer branch logic in the GPU. During auto-boot, the SLCC signal should be reset, and then set to a 1 as soon as the last instruction in the boot ROM has been read.
HW[7-0]	Output	The 8 Z bus hardware control pins are decoded by the selected Z bus slave to implement specific hardware commands. The hardware control pins are active at the positive clock edge while SYNC is high.

Pin Descriptions (continued)

OMEGA-GPU Signals for the Host Interface		
Signal	Pin	Direction
CSL	Input	The chip select pin is the primary strobe that signals the Host's intention to read or write internal GPU registers. The signal is low true.
HRA	Input	The host register address pin selects whether the host is accessing the GPU internal register select register (when high) or the register currently selected by the last write to the register select register.
HRD	Input	The host read pin is driven by the host and selects either a read (high) or write operation to the selected internal register.
HD[7-0]	I/O	The 8 host data pins are used to transfer the register read and write data.
RDY	Output	The ready pin is set when the current host operation is complete, telling the host that it may take away its request (CSL). The host must wait for RDY to be cleared before making another request.
INTL	Output, Open-Drain	The interrupt pin returns a low-true signal to the host when any of the internal interrupt conditions become true and are enabled.
HCIR	Output	The host communication input ready pin is set when the Communication Input Register (CIR) is ready for data. The pin is cleared when the host writes to the CIR. HCIR has the same sense as bit 2 in the host status register (SREG, see Host Accessible Registers). HCIR is intended for use with FIFO-buffered host interfaces.

OMEGA-GPU Miscellaneous Signals		
Signal	Pin	Direction
ICLK	Input	The Clock pin is used to input the basic system clock. This clock is four times the GPU instruction execution rate.
RSTL	Input	The Reset pin forces the GPU to reinitialize at a known state. This signal is low true.

Host Accessible Registers

The host port on the GPU represents two addresses in the host address space. The highest address is a read/write register called the Register Number Register which contains the four-bit address of the selected internal GPU register. The selected register can be read or written by the host at the other host address (low address). This section of the data sheet documents the contents of these registers accessible by the host.

Register Number Register

The Register Number Register is a R/W register accessed by the host, using the protocol described in the host interface section, with the HRA pin high.

RNR	REGISTER NUMBER REGISTER							
HRA Pin	Bit7	6	5	4	3	2	1	Bit0
1	0	0	0	0	REGN[3-0]			

The host must set this register with the number of the GPU internal register that the host wishes to read or write.

GPU/Host Registers

The following sixteen registers are accessed by the host with the HRA pin low and the correct register number in the Register Number Register as described above.

REGN[3-0]	Name	Description
0	SREG	Interrupt Status Register
1	IREG	Interrupt Enable Register
2	CREG	Control Register
3	REGN	Register Number Register (value always 3)
4	IDR0	Instruction Data Register bits 7-0
5	IDR1	Instruction Data Register bits 15-8
6	IDR2	Instruction Data Register bits 23-16
7	IDR3	Instruction Data Register bits 31-24
8	IRAR0	Instruction RAM Address Register bits 7-0
9	IRAR1	Instruction RAM Address Register bits 15-8
A	CMA0	Current Microcode Address bits 7-0
B	CMA1	Current Microcode Address bits 14-8
C	GDB0	GPU Data Bus bits 7-0
D	GDB1	GPU Data Bus bits 15-8
E	CIR	Communication Input Register
F	COR	Communication Output Register

Status Register

The Status Register is a read only register that returns the information the host requires to handle interrupts from the GPU. If the interrupt output is not used or enabled to the host, this register can be used to poll for GPU events.

SREG	STATUS REGISTER							
Reg. No.	Bit7	6	5	4	3	2	1	Bit0
0x0	INT	0	0	0	HALT	CBR	CIR	BKPT

BKPT — Breakpoint is set when a breakpoint is reached in the microcode and the processor is halted. Breakpoint is cleared after the processor is disabled and enabled with SSEN or if the processor is single stepped (see Control Register). BKPT can cause an interrupt if enabled with the interrupt-enable register.

CIR — Communication Input Ready bit is set when the GPU reads the byte from the Communication Input Register. CIR is cleared when a byte is written from the host that fills the CIRD. CIR can cause an interrupt if enabled.

CBR — The Command Byte Ready bit is set when a byte has been written to the host by the GPU. The CBR is cleared when the host reads the byte. CBR can cause an interrupt if enabled.

HALT — The Halt bit is set when the GPU processor is halted for any reason. The HALT bit is cleared when the GPU processor runs. HALT can cause an interrupt if enabled.

INT — The Interrupt bit is set when the interrupt-output pin is driven low. This bit is an OR of the enabled interrupt conditions (BKPT, CBR, CIR, and HALT).

Interrupt Enable Register

The Interrupt Enable register is a R/W register with four bits to enable each of the four conditions which might cause an interrupt. These interrupt enables are used by the host as required to control the host program interrupt handling. The IREG also contains the flag bits which the GPU can read in its status register.

IREG	INTERRUPT ENABLE REGISTER							
Reg. No.	Bit7	6	5	4	3	2	1	Bit0
0x1	HFLG[3-0]				ENHLT	ENCBR	ENCIR	ENBKP

ENBKP — Enables the Breakpoint interrupt when set.

ENCBR — Enables the Command Byte Ready interrupt when set.

ENCIR — Enables the Command Input READY interrupt when set.

ENHLT — Enables the Halt interrupt when set.

HFLG[0-3] — User-definable flag bits, used to indicate status to the GPU.

Control Register

The Control Register is a R/W register used by the host to control execution of the GPU and host access to Instruction Data Memory. The ZERO, STEP, BKEN, and SSEN bits are used in the microcode debugger. The TST bit is used for chip testing. The remaining bits are used to command Instruction Data Memory Operations.

CREG	CONTROL REGISTER							
Reg. No.	Bit7	6	5	4	3	2	1	Bit0
0x2	BKEN	INCID	IDOP	RDID	TST	ZERO	STEP	SSEN

SSEN — The Single Step Enable bit is set by the host to halt the GPU at the current instruction. The HALT bit will be set in the Status Register and a interrupt generated if the ENHLT bit is set.

STEP — The Step bit is set by the host to command the GPU (which must be halted with SSEN) to execute the current micro-instruction. Once STEP is set it will remain set until a single micro-instruction has been executed, therefore the host must read a zero at this bit before forcing a step.

ZERO — The Zero bit forces the GPU to execute the next instruction from address zero. The bit may be cleared by the host or by the GPU when the instruction at zero is executed. If the host sets the bit and clears it before the GPU is allowed to execute an instruction, the address will not go to zero.

TST — The Test bit can be read by the GPU from its status register. When the test bit is set it is intended that the GPU begin executing test microcode.

RDID — The Read Instruction Data bit makes the next IDOP a read from Instruction Data Memory into the Instruction Data Registers.

IDOP — The Instruction Data Operation bit causes the Instruction Data Memory to be read or written (see RDID) to or from the Instruction Data Registers. The bit will be reset when the operation is complete.

INCID — The Increment Instruction Data bit enables a increment of the Instruction RAM Address register after the next IDOP.

BKEN — The Break Enable bit when set halts the GPU and sets SSEN whenever a breakpoint is detected in the micro-instruction (see the BKPT bit in the GPU Instruction Field section).

Instruction Data Registers

The Instruction Data Registers are R/W registers that hold the four bytes of data to be read or written to the Instruction RAM. The address to be used for the write or read is stored in the two byte host Instruction RAM Address Register. The actual read or write is generated by a command to the Control Register. The Instruction Data Register bytes IDR[1-0] are the same register as the GPU ISPDL, and bytes IDR[3-2] are the same as the GPU ISPDH register, therefore the GPU must be halted to read or write the IDR.

IDR	INSTRUCTION DATA REGISTER							
Reg. No.	Bit7	6	5	4	3	2	1	Bit0
0x4	ID[7-0]							
0x5	ID[15-8]							
0x6	ID[23-16]							
0x7	ID[31-24]							

The Instruction Data Register and the associated address and control are primarily used for the host boot of the Instruction memory. The host zeroes the address, puts four bytes of the program code in the IDRs, causes a write and increments the address until the memory is initialized. These registers may also be used for host diagnostic of the instruction memory RAM. Since they are shared with the GPU processor for different functions, the host must save and restore these registers when debugging microcode.

Instruction RAM Address Register

The Instruction RAM Address Registers are R/W registers used as the address into the Instruction Memory. This address is used when data is read or written to the Instruction Data Registers (see above).

IRAR	INSTRUCTION RAM ADDRESS REGISTER							
Reg. No.	Bit7	6	5	4	3	2	1	Bit0
0x8	IA[7-0]							
0x9	IA[15-8]							

These registers may be incremented as a counter by a command in the Control Register. Since the IRAR is the same as the GPU ISPADR register, the GPU must be halted to read or write the IRAR. Please see the usage description for the Instruction Data Register.

Current Microcode Address

The Current Microcode address is a read-only register that contains the address of the instruction the GPU is executing. The contents of this register are unreliable if the GPU is not halted (see Control Register).

CMA	CURRENT MICROCODE ADDRESS							
Reg. No.	Bit7	6	5	4	3	2	1	Bit0
0xA	UA[7-0]							
0xB	0	UA[14-8]						

This register is used by diagnostics and the microcode debugger. Diagnostics use the registers to verify correct operation of the GPU. The debugger uses the registers to verify and debug microcode programs.

GPU Data Bus

The GPU Data Bus register is a read-only register which reflects the current state of the internal GPU data Bus. The contents of this register are unreliable if the GPU is not halted (see Control Register). The GPU Data Bus register is used for diagnostics and debugging similar to the Current Microcode Address register.

GDB	GPU DATA BUS							
Reg. No.	Bit7	6	5	4	3	2	1	Bit0
0xC	D[7-0]							
0xD	D[15-8]							

Communication Input Register

The Communication Input Register is a R/W register used for Host to GPU byte-wide communication. The Communication Input Register is written by the host according to the status bit CIR. The register is readable by the host as a diagnostic aid.

CIR	COMMUNICATION INPUT REGISTER							
Reg. No.	Bit7	6	5	4	3	2	1	Bit0
0xE	CIRD[7-0]							

The host can write this register when the CIR bit in the host status register (SREG) is set. Writing the Communication Input Register clears the CIR bit. The CIR bit will be set when the GPU processor reads the Ioport register.

Host Accessible Registers (continued)

Communication Output Register

The Communication Output Register is a read only register used for GPU to Host byte-wide communication. The Communication Output Register is read according to the status bit CBR by the host.

COR	COMMUNICATION OUTPUT REGISTER							
Reg. No.	Bit7	6	5	4	3	2	1	Bit0
0xF	CORD[7-0]							

The host can read this register when the CBR bit in the host status register (SREG) is set. Reading the Communication Output Register clears the CBR bit. The CBR bit will be set when the GPU processor writes the Ioport register.