Galileo
Technology, Inc.

System Controller For i960JX Processors

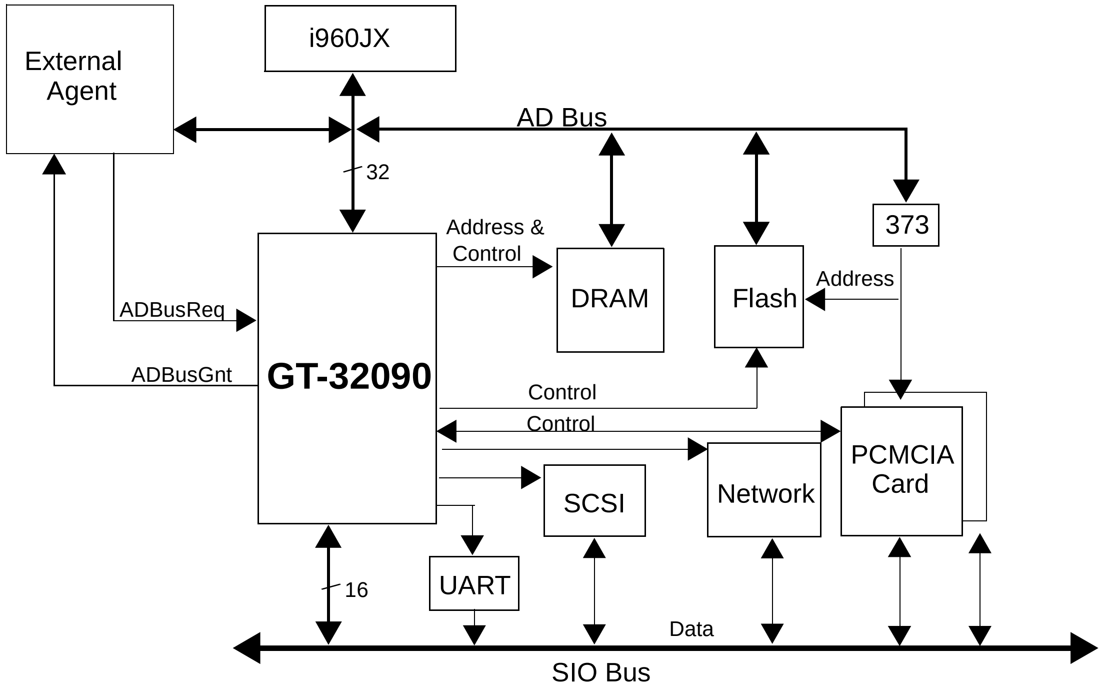
GT- 32090

Preliminary, Rev. 2.0
March 1996

NOTE: Always contact Galileo Technology for possible updates before starting a design.

FEATURES

- Integrated system controller for embedded applications
- Supports the i960JX family of CPUs
- 16-33MHz bus frequency
- Flexible DRAM controller
  - Page mode and EDO DRAMs
  - 128MByte address space
  - 256K-4M device depth
  - 1-4 banks supported directly
  - Up to 8 banks supported indirectly
  - 32-bit data width
  - Non-interleaved
  - Different size for each bank
  - Zero wait state to first data at 16 and 20MHz
  - One wait state to first data and no wait state to burst data at 25MHz
  - Two wait states to first data and no wait state to burst data at 33MHz
- Flexible AD bus device controller
  - 128MByte address space
  - 4 chip selects
  - Per bank programmable timing
  - Supports several types of standard memories (ROM / Flash / SRAM) and I/O controllers
  - External wait support
  - 8-, 16-, and 32-bit device (and boot) support
- High performance DMA
  - Three independent channels
  - Chaining via linked lists of records
  - Transfers through a 16-byte internal FIFO or fly-by
  - Moves data between SIO, memory, and devices
  - Packing and unpacking of 8-bit and 16-bit data to/from the SIO bus into 32-bit data on the CPU bus
  - Packing and unpacking from/to the SIO bus concurrent with AD bus activity
  - Fixed and round robin programmable priorities
- Simple I/O bus (SIO bus)
  - Simple Read/Write bus for glueless interface to low cost peripherals
  - 8/16-bit wide bus
  - Four chip selects
  - Programmable timing
  - External wait support
- PCMCIA
  - Supports two PCMCIA devices directly
- JTAG
- 5V
- 160 PQFP



OVERVIEW

The GT-32090 is a low cost, highly integrated single-chip System Controller for the i960Jx Family. It provides high system performance, while reducing cost, complexity, device count, and board space. The GT-32090 controls two separate and independent buses, the CPU's 16 to 33MHz 32-bit wide address/data bus, and a 16-bit I/O bus. The two buses can work concurrently at different frequencies.

The DRAM Controller supports up to 128MBytes of standard or EDO DRAM. It supports up to four 32-bit wide banks directly, or up to eight banks indirectly, with zero wait states to first data or burst data at 16 and 20MHz. At 25 and 33MHz, there is one wait state and two wait states respectively, to first data, and at both frequencies there are no wait states to burst data. Various refresh and addressing modes are supported.

The Device Controller supports up to 4 devices directly, and includes various programmable timing and wait state mechanisms that can be setup individually for each device. Typical devices supported include DRAM, ROM, Flash, and SRAM, as well as high-performance master peripherals.

The powerful three-channel DMA Controller has data alignment capabilities and sophisticated chaining support via link lists. The DMA can move data between devices on the CPU bus, or between devices on the CPU bus and devices on the I/O bus. DMA transfers can go through an on-board 16-byte FIFO, or directly if in fly-by mode. Packing and unpacking of 8-bit and 16-bit data from/to the I/O bus occurs concurrently with activity on the CPU bus, increasing overall system bandwidth.

The I/O bus is a simple 16-bit read/write bus that interfaces to a large variety of low cost support components like UARTs, SCSI controllers, Ethernet controllers, and other devices. The I/O bus supports 8- or 16-bit peripherals, as well as slave DMA and PCMCIA devices. Three 4-byte FIFOs provide efficient support for the gathering of 8-bit or 16-bit data to/from different peripherals, in the endianess chosen by the designer.

The GT-32090 includes a direct interface to two 8-bit or 16-bit PCMCIA slots.

REFERENCE DESIGN

Galileo makes available the Galileo-5 Evaluation and Development System, an ISA card which greatly facilitates the development of embedded control systems based on the i960JA/JD/JF Intel processors.

The centerpiece of the system is the GT-32090 System Controller, which integrates most of the core logic necessary in embedded applications.

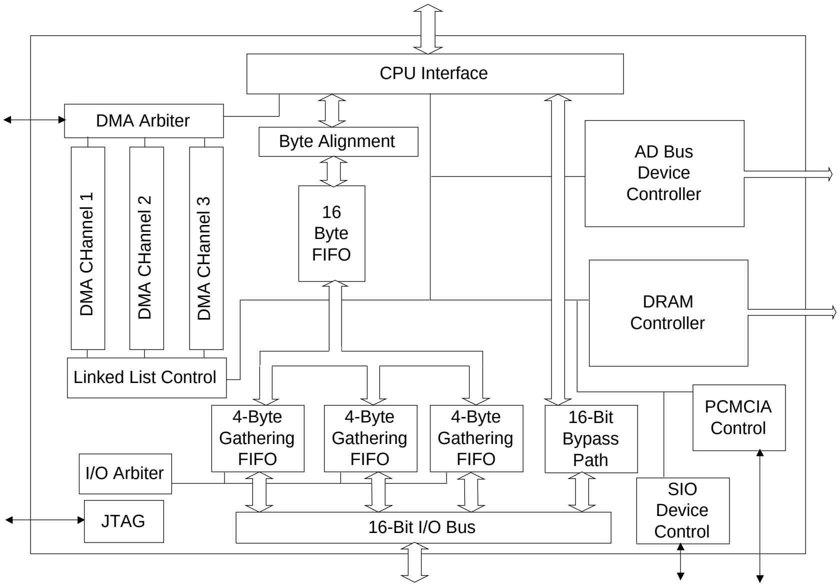
The Galileo-5 board serves two main objectives:

a) It allows customers to easily evaluate the performance of a GT-32090 based system, using their own software, as opposed to generic benchmarks.

b) It greatly facilitates and expedites the development of the final product, since hardware designers can use its design as a reference and software designers can use it to start porting software ahead of their own hardware platform.

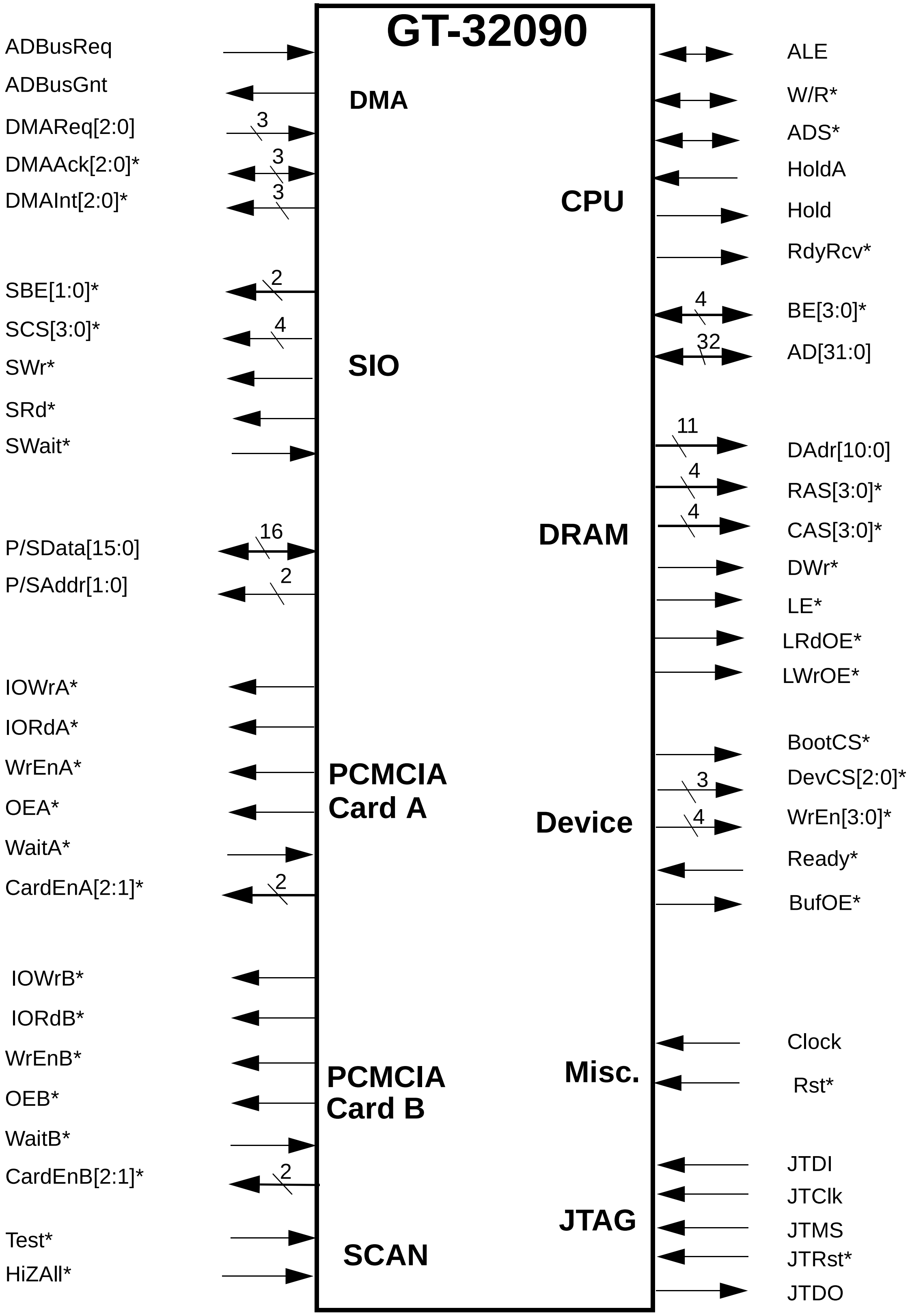
The shipping configuration of the Galileo-5 includes a 33MHz i960JA CPU, 4MBytes of DRAM, sockets for 2MBytes of Flash, a 512KByte EPROM, a DUART, Intel MON960 software, and 2 PCMCIA sockets. Jumpers allow customers to evaluate a large variety of system configurations, and expansion connectors allow customer-designed options to be easily interfaced to.

The Galileo-5 can either be used in stand-alone fashion, or it can be plugged directly into an ISA slot of standard personal computers.



1 PIN INFORMATION

1.1 Logic Symbol



1.2 Pin Assignment Table

Pin Name	Type	Description
CPU Interface		
AD[31:0]	I/O	**Address/Data Bus:** Multiplexed address and data bus for communication between the processor, devices, DRAM, any external agent, and the GT-32090.
ALE	I/O	**Address Latch Enable:** A strobe for latching the address into the GT-32090 and into external latches for devices, PCMCIA cards, and SIO bus peripherals. It is an input during a CPU or external agent access, and an output during a DMA access.
ADS*	I/O	**Address Strobe**: Indicates a valid address and the start of a new bus access. It is an input during a CPU or external agent access, and an output during a DMA access.
BE[3:0]*	I/O	**Byte Enable:** Selects which of the four bytes on the AD bus participate in the current bus access. It is an input during a CPU or external agent access, and an output during a DMA access.
Hold	O	**Hold**: A request from the GT-32090 to acquire the AD bus.
HoldA	I	**Hold Acknowledge:** An indication by the CPU that it has relinquished the AD bus to the GT-32090.
RdyRcv*	O	**Ready/Recover**: Indicates when the data on the AD bus can be sampled or removed. During a device turn-off time, it indicates to the CPU not to drive the address on the AD bus.
W/R*	I/O	**Write/Read:** Specifies if the access is a write or a read access. It is an input during a CPU or external agent access, and an output during a DMA access. Used also to control the direction of the bi-directional transceiver for the devices on the AD bus.
DRAM		
DAdr[10:0]	O	**DRAM Address/Device Burst Address**: Eleven multiplexed address bits to the DRAM. DAdr[1:0] provides the word burst address (same meaning as the CPU's A3, A2 pins) for all 32-bit accesses, be they to DRAM or to devices.
RAS[3:0]*	O	**Row Address Select**: Supports four banks of DRAM.
CAS[3:0]*	O	**Column Address Select:** Supports byte writes to DRAM.
DWr*	O	**DRAM Write:** Signals a write access to the DRAM.
LE*	O	**Latch Enable:** When active, latches the DRAM data into external latches.
LRdOE*	O	**Latch Read Output Enable:** When active, outputs the data from the DRAM's external latches onto the AD bus.
LWrOE*	O	**Latch Write Output Enable:** When active, outputs the data from the DRAM's external latches onto the DRAM's data pins.
AD Bus Devices		
WrEn[3:0]*	O	**Write Enable:** Byte write enable to devices on the AD bus.
DevCS[2:0]*	O	**Device Chip Select**: Programmable chip select signals to devices on the AD bus.
BootCS*	O	**Boot Chip Select**: Programmable chip select signal to the boot device on the AD bus.

Pin Name	Type	Description
Ready*	I	**Ready**: When not active, it extends the access to a device on the AD bus by adding wait cycles.
BufOE*	O	**Buffer Output Enable:** This signal has similar functionality to the DEN* signal of the i960JX, but is active during accesses to devices only. It is active during the data phase of accesses to devices on the AD bus. It is used with the W/R* to control external data transceivers.
PCMCIA & SIO Shared Signals		
P/SAddr[1:0]	O	**PCMCIA/SIO Address**: Byte and half-word addresses for PCMCIA and SIO accesses.
P/SData[15:0]	I/O	**Data Bus:** Shared data bus for SIO and PCMCIA devices.
SIO Interface		
SBE[1:0]*	O	**SIO Byte Enable:** Selects which of the two bytes on the SIO bus participates in the current data transfer.
SCS[3:0]*	O	**SIO Chip Select:** Chip Select for devices on the SIO bus.
SRd*	O	**SIO Read:** Active during a read from a device on the SIO bus.
SWr*	O	**SIO Write:** Active during a write to a device on the SIO bus.
SWait*	I	**SIO Wait:** Extends bus cycle, used to generate wait states by SIO devices.
PCMCIA Card A		
CardEnA[2:1]*	O	**Card Enable A:** CardEnA[1] enables the even address bytes, and CardEnA[2] enables the odd address bytes.
OEA*	O	**Output Enable A:** Controls the output of data from card.
WaitA*	I	**Wait A:** Extends bus cycle, used to generate wait states by the card.
WrEnA*	O	**Write Enable A:** Indicates write accesses by the GT-32090 to the card.
IORdA*	O	**I/O Read A:** Activated to read data from the card’s I/O space.
IOWrA*	O	**I/O Write A**: Activated to write data to the card’s I/O space.
PCMCIA Card B		
CardEnB[2:1]*	O	**Card Enable B:** CardEnB[1] enables the even address bytes, and CardEnB[2] enables the odd address bytes.
OEB*	O	**Output Enable B:** Controls the output of data from card.
WaitB*	I	**Wait B:** Extends bus cycle, used to generate wait states by the card.
WrEnB*	O	**Write Enable B:** Indicates write accesses by the GT-32090 to the card.
IORdB*	O	**I/O Read B:** Activated to read data from the card’s I/O space.
IOWrB*	O	**I/O Write B**: Activated to write data to the card’s I/O space.
DMA		
ADBusReq	I	**Bus Request:** Signals a request from the external agent to the GT-32090 for acquisition of the AD bus.

Pin Name	Type	Description
ADBusGnt	O	**Bus Grant:** Signals that the GT-32090 grants mastership on the AD bus to the external agent.
DMAReq[2:0]	I	**DMA Request:** A request for a DMA operation directed at the internal arbiter. When Rst* is asserted these bits are sampled into the Device Address Space register bits [6:4].
DMAAck[2:0]*	I/O	**DMA Acknowledgement:** Outputs - The internal arbiter grants a DMA operation to a requesting device. Inputs - When Rst* is asserted, these bits are sampled into the Internal Space Decode & Control register bits [5:3]. These pins are inputs when Rst* is active, until 2 clocks after Rst* is deasserted. They are outputs at all other times.
DMAInt[2:0]*	O	**Interrupt:** Each interrupt corresponds to one of the DMA channels.
Miscellaneous		
Rst*	I	**Reset:** Resets all internal logic and state machines to a known state.
Clock	I	**Clock:** The input clock to the GT-32090.
JTAG		
JTDI	I	**Test Data Input Pin:** JTAG’s serial input
JTMS	I	**Test Mode Select:** Used to issue serial instructions to the TAP controller’s state machine.
JTClk	I	**Test Clock:** JTAG’s input clock
JTRst*	I	**Test Reset:** Active low asynchronous reset for the TAP controller
JTDO	O	**Test Data Output:** JTAG’s serial output
SCAN		
Test*	I	**Test:** When active, indicates that the GT-32090 is in Scan mode. 0 - Scan mode 1 - Normal operation
HiZAll*	I	**Tri-State:** When active, the GT-32090 drives all outputs and I/O pins to High impedance. 0 - Tri-State 1 - Normal operation

2 FUNCTIONAL DESCRIPTION

2.1 CPU Interface

The GT-32090 has a glueless interface to the Intel i960JX family of processors with bus frequencies between 16MHz and 33MHz. External agents can take control of the AD bus and access all of the GT-32090 resources. The GT-32090 handles the priorities between the internal DMA resources, the CPU and the external agent. It will request the AD bus from the CPU and relinquish the bus to the external agent. For systems that need extension of the capabilities offered by the GT-32090, up to four GT-32090 devices can support one CPU. For example, with four GT-32090's, the system can have up to 512MBytes of DRAM and 12 DMA channels.

2.2 Address Space Decode

The GT-32090 decodes the address space in two stages. In the first stage, the decoding is done to eight groups that include the following: four spaces of 32 MBytes for the AD bus devices, 128 MBytes for the DRAM, 128 MBytes for the Simple I/O bus (SIO bus), 256 MBytes for the PCMCIA, and 64 MBytes for the internal address space. The decoding in the first stage is done by comparing the high order address bits with the Address Space registers' values. This way, four groups of devices are provided in order to enable mappings of 8-, 16- and 32-bit devices in the same system, using different PMCONs (PMCONs are the i960JX's Physical Memory Configuration registers - see i960JX User's Manual). In the second stage, there is additional decoding to the specific device bank, DRAM bank, internal register, SIO bank, or PCMCIA bank. The sub-decoding for the DRAM and the AD bus device banks is programmable while the sub-decoding for the internal registers, the PCMCIA cards, and the SIO devices, is fixed.

Each DRAM bank can have a programmable address space of 1MByte to 16MBytes that can be located anywhere in the DRAM address space. The address space size programmability enables a contiguous address space when accessing the different DRAM banks even when the banks have different sizes. The decoding is done by comparing address bits 25:20 to be between two values (High and Low) in the DRAM bank registers.

Each AD bus device bank can have a programmable address space of 2MBytes to 32MBytes. Similar to the DRAM banks, it enables a contiguous address space for different AD device bank sizes. The decoding is done by comparing address bits 24:21 to be between two values (High and Low) in the AD device bank registers.

The PCMCIA is allocated a 256MByte block that has a fixed sub-decoding to four 64MByte spaces. The first 128MBytes are allocated to card A and the second 128MBytes to card B. Each 128MBytes are divided so that the first 64MBytes are for memory space and the second 64MBytes are for I/O address space.

The SIO space is sub-decoded to four fixed 16 MByte spaces for the SIO Chip Selects, or eight 16MByte spaces with external sub-decoding to create a 128MByte space.

The internal address space is sub-decoded for the control and status registers that reside on the GT-32090.

The GT-32090 will not respond to addresses that are not allocated to it, since that address space might be used for other devices.

2.3 DRAM Controller

The DRAM controller supports 4 banks of page mode or EDO DRAMs. DRAM types supported are those with 0.5K, 1K, and 2K refresh, as well as asymmetric RAS/CAS addressing. The depth of the DRAM devices can vary for each bank separately from 256K to 4M, and the width of all banks is 32-bits. With these options, each DRAM bank size can vary from 1MByte to 16MBytes. The DRAM timing is optimized for the different frequencies and device types supported. There is optional support for an external bi-directional latch on the DRAM's data bus for improved DRAM performance.

At 33MHz, a CPU read access from an EDO DRAM will have the pattern xxDDDD, which means 2 wait-states to first data and zero wait-state for each additional word. At 33MHz, with standard DRAMs, the pattern will be xxDxDxDxDx, meaning 2 wait-states to the first data and 1 wait-state for each additional word. At 25MHz, standard DRAM with latch or EDO DRAM will have a performance of xDDDD (one wait-state to first data). At 25MHz, standard DRAM without latches will be xxDxDxDxDx (2 wait-states to first data, 1 wait-state to burst data. At 20MHz and 16MHz using standard DRAM with latches or EDO DRAM, the pattern will be DDDD, zero wait-states to the first data and zero wait-states for each additional word. At 20MHz and 16MHz, DRAM performance is xxDxDxDxDx. DMA burst accesses can be one per clock, using the same parameters as for the CPU access.

Refresh can be programmed to different periods by a 16-bit refresh counter. Staggered and non-staggered refresh modes are supported. In staggered mode, the four banks of DRAM will be refreshed with one cycle delay between each bank, while in non-staggered mode all four banks will be refreshed together.

For systems that require more than 64MBytes of DRAM, it is possible to add 4 more DRAM banks with a simple external multiplexer, to get to a total of 128MBytes of DRAM with 8 RAS control signals, using Address[26] to multiplex between two banks. For instance, the demultiplexing can be implemented with a Quality Semiconductor QS3257.

2.4 AD Bus Device Controller

The GT-32090 supports directly four devices on the AD bus, one boot device and three general purpose devices such as Flash, SRAM, ROM, FIFO, or any other read/write peripheral device. External logic can sub-decode the four chip selects to any number, using the Address for the sub-decoding. The device controller has several control signals to enable read and write accesses (including chip selects, reads, writes, and buffer control). Each Chip Select has a programmable address space of 2MBytes to 32MBytes to enable contiguous address space between the different banks. Byte writes are enabled through four Write Enable signals. The write signals can be shaped by specifying in the Device Bank Parameters registers the following: the number of cycles from the assertion of DevCS* to the first assertion of write (CsToWr); the number of cycles the write pulse is active (WrActive); and the number of cycles the write signal is non-active between consecutive writes(WrHigh). The timing parameters of the write signals determine the length of active DevCS* and DMAAck* (when allocated to an AD bus device).

In read cycles, the following parameters are programmable: the number of cycles from the assertion of Chip Select to the rising edge of the clock that samples the first data (DelayToFirst), the number of cycles from when data is sampled to the next time data is sampled (DelayToNext); and the number of cycles between the deassertion of DevCS* to a new AD bus cycle (TurnOff).

Each device can be configured as 8-, 16- or 32-bits wide, by programming the appropriate i960Jx's PMCON register to the desired bus width, and mapping the appropriate Device n Address Space decode register into this address space. Each PMCON configures the bus width of a memory region with a specific Addr[31:29] while the Device n Address Space decode register maps device #n into a region with a specific Addr[31:25].

The device controller supports read or write bursts of up to four data elements. The burst address is supported by a 2-bit wide address bus that is multiplexed with the two least significant bits of the DRAM address (DAdr[1:0]) when the device is 32-bits wide. For an 8-bit device the burst address is the CPU's BE[1:0]*, and for a 16-bit device the burst address is the CPU's {A2, BE[1]*}. For an 8-bit device the write signal is WrEn[0]*, and for a 16-bit device the write signals are WrEn[3]* (Write High Byte) and WrEn[0]* (Write Low Byte).

The Ready* pin enables an extension of a device cycle beyond the values that are programmed in the Device Bank Parameters register. All the control signals will continue to be in their state when the Ready* signal is sampled inactive, and until it becomes active. The internal state machine counters will continue to count to the programmed values, even when Ready* is HIGH. The control signals will change only when the Ready* is LOW and the counters are at terminal count. After insertion of wait states and re-assertion of Ready*, there are two clock cycles to data transfer. The use of the Ready* signal is individually optional for each bank through a programmable bit in the Device Bank Parameters registers.

In systems where the devices on the AD bus represent a large load, or where there are devices with long turn-off times, the GT-32090 supports an optional bi-directional transceiver (245 type) to isolate the device bus from the CPU's AD bus. The BufOE* signal controls the bi-directional transceiver's OE* and the W/R* signal controls its direction.

2.5 DMA

The DMA controller can move data between devices on the AD bus, or between devices on the AD bus and devices on the SIO bus. There are two DMA subsystems on the GT-32090, one handles the DMA activity on the SIO bus, and the other handles the activity on the AD bus. Each DMA subsystem has its own data storage resources and arbiters. The two DMA subsystems can work simultaneously or independently, except for the time that they transfer data between the SIO bus and the AD bus. Both subsystems can have at one time three DMA channels active. Each of the three channels can be allocated to service the SIO bus or the AD bus.

DMA accesses can be initiated by an external request by asserting one of the three DMAReq pins (Demand mode), or by setting an internal bit in a register (Block mode). Access can be non-aligned both at the source and at the destination, and up to 64KBytes of data can be transferred in each transaction.

The AD bus DMA can transfer data in two ways: through an internal 16-byte FIFO, or directly between the DRAM and an AD bus device ("fly-by"). In the internal mode, data is transferred from the source device/DRAM into the internal FIFO, and from the internal FIFO to the destination device/DRAM. In "fly-by" mode, the access is to

DRAM with assertion of DevCS*, DMAAck*, and WrEn*, if necessary. The WrEn* signals are not toggled. The length of each DMA access can be limited to 1, 2, or 4 32-bit words.

For a 32-bit device, the write signals are WrEn[3:0]*. For a 16-bit device, the write signals are WrEn[3]* for write high byte, and WrEn[0]* for write low byte. For an 8-bit device, the write signal is WrEn[0]*.

The SIO bus DMA can transfer data between an 8- or 16-bit wide device on the SIO bus, and a 32-bit wide device/DRAM on the AD bus. In read accesses (from an SIO device to an AD bus device), the SIO DMA uses one of three internal packing registers (one for each channel), that pack 8- or 16-bit data into 32-bit words. In write accesses (from an AD bus device to an SIO device), the SIO DMA uses one of three internal unpacking registers (one for each channel), that unpack 32-bit words into 8- or 16-bit data.

The SIO channel can be programmed as read (source) or write (destination) depending on bit 5 of the Channel Mode register.

During an SIO DMA, data is transferred in two stages that involve one of the SIO local packing and unpacking registers, and the FIFO in the AD DMA unit.

During an SIO DMA read access, the SIO device will arbitrate for the SIO bus in the local SIO arbiter, and will move data into its channel packing register. When the register is full, or when the DMA counter reaches terminal count, the packing register will arbitrate in the AD DMA arbiter for the AD DMA FIFO, and will transfer the data to it. When the data is in the FIFO, the FIFO will request the AD bus and will transfer data to the AD bus device.

During an SIO DMA write access, the AD DMA will move data from the AD bus to its internal FIFO. From the FIFO, it will move the data into the SIO unpacking register, and then the SIO DMA will unpack and transfer the data to the requesting SIO device.

The SIO packing/unpacking register can be flushed by writing to the Channel Flush/Reset register (see section 3.11).

The SIO channel can be programmed to two possibilities of arbitration:
1) Access Arbitration: Arbitration between SIO channels is done in every SIO DMA access (byte or 16-bit word).
2) Word Arbitration: Arbitration between SIO channels is done only when the DMA finishes packing/unpacking its register (four byte transfers or two 16-bit word transfers).

The DMA controller supports chained and non-chained modes of operation. In the non-chained mode, the CPU programs the DMA channel for each DMA transaction. In chained mode, the DMA controller programs itself for the next DMA operation by fetching the information from a linked list of records in memory.

The DMA controller can be programmed to assert an interrupt in chained mode, at the end of every DMA transaction or when the Next Pointer Register is NULL and Byte Count reaches terminal count. In non-chained mode, the DMA will assert an interrupt every time the Byte Count reaches terminal count.

There are two separate arbiters for DMA accesses. One arbiter prioritizes accesses between devices on the SIO bus for data transfers between SIO devices and their packing/unpacking registers. The second arbiter prioritizes accesses between devices on the AD bus and SIO devices. The two arbiters have programmable priorities and are identical in their functionality. The arbiter programmable options work as follows: Channels 0 and 1 are in one group, and channel 2 in the second group. Inside the two channel group, the priority can be fixed with a selected channel number having the higher priority, or both can have the same priority in round robin fashion. The same scheme applies between the two groups, they can have fixed or round robin priority.

In systems with EDO DRAM at 33MHz or Page Mode DRAM at 16 to 25MHz that require high bandwidth, the data phase should be extended. In order to extend the data phase, a bi-directional latch should be used. The GT-32090 controls the latch- read output enable (DRAM Data to CPU Bus), write output enable (CPU Data to DRAM Data), read latch enable (DRAM Data). The latching at the CPU bus is done with the use of the system clock. The correct timing of the controls is derived from the system’s parameters - ADFreq, Type, and Latch, which are programmed in the DRAM Parameters register.

2.6 SIO

The SIO interface is a simple Read/Write with Chip Select bus interface. The interface includes: a dedicated 16-bit wide data bus that is shared with the PCMCIA devices, an address bus that is shared with the AD bus devices, dedicated byte and 16-bit word address, byte enables (SBE[1:0]*), four chip selects (SCS[3:0]*), read (SRd*) and write (SWr*) control signals, and a flow control signal (SWait*), in addition to the DMA signals.

The SWait* pin is used to extend a data cycle. Two cycles prior to the end of an access, SWait* is sampled. If it is asserted, then the access is extended. When SWait* is again deasserted, two more cycles will be executed until the end of the data transfer.

Standard address space is 64MBytes. Each of the Chip Select signals has a 16 MByte address space, can be configured as 8-bit or 16-bit wide, and has programmable timing. Big and little endian conversion is supported for each device. The SIO supports up to three slave DMA devices as described in section 2.5. Further sub-decoding can result in four Chip Selects for a total of 128 MBytes of address space.

Timing for an SIO bus DMA or CPU access is programmable. The user can specify the width of active SRd* and SWr* signals, and the Turn-Off time of the device. The width of SRd* and SWr* dictate the shapes of SCS* and DMAAck*. They are asserted one cycle before SRd* or SWr* become active, and remain active for one cycle after SRd* or SWr* is deasserted. The GT-32090 will not start a new read cycle from a different SIO device and will not start a new write access on the SIO bus as long as the Turn-Off time is not satisfied. The Turn-Off Width will start counting when SCS* is deasserted to terminal count.

When the channel is programmed to Word arbitration, a device will be served until all the remaining bytes are packed/unpacked. Between accesses, DMAAck* will be deasserted for 1 cycle and SWr* /SRd* for 3 cycles. When the channel is programmed to Burst Mode with Word arbitration, DMAAck* will stay asserted through the entire burst. It will be asserted one cycle before SRd* (or SWr*) becomes active, and remain asserted until one cycle after the last SRd* (or SWr*) of the burst is deasserted. The SRd* or SWr* signals will be asserted for the number of clocks programmed in PulsWid and deasserted for one clock.

The partition between the devices is fixed and decoding in the SIO is done on address bits 25:24.
SCS[0]* : Address[25:24] = 00
SCS[1]* : Address[25:24] = 01
SCS[2]* : Address[25:24] = 10
SCS[3]* : Address[25:24] = 11

2.7 PCMCIA

Two PCMCIA cards can be supported directly by the GT-32090. Each card has a 128MByte address space dedicated to it, 64MBytes for I/O space and 64MBytes for memory space. There is support for big or little endian data formats and 8-bit or 16-bit accesses. The timing of the control signals to the cards is programmable. All the control signals between the cards and the GT-32090 can be connected without glue logic except for the static control and status signals. They are interfaced to the GT-32090 through an external latch and buffer, as shown in the application section.

The partition between PCMCIA devices is fixed and decoding is done on address bits 27:26.

PCMCIAa Memory	: Address[27:26] = 00
PCMCIAa I/O	: Address[27:26] = 01
PCMCIAb Memory	: Address[27:26] = 10
PCMCIAb I/O	: Address[27:26] = 11

2.8 JTAG (Boundary Scan)

The GT-32090 supports JTAG test features compatible with the IEEE Standard Test Access Port And Boundary Scan Architecture (IEEE 1149.1.A).

The JTAG features supported are:

Test Name	Binary Code
EXTEST	000
SAMPLE	010
IDCODE	001
STCTST	101
INTEST	100
BYPASS	111

The GT-32090 IDCode is 03290115h.

Bit[31:28] : Version: ‘0000’

Bit[27:12] : Part Number: ‘0011001010010000’

Bit[11:1] : Manufacturer ID: ‘00010001010’

Bit[0] : IDCode’s bit[0] is ‘1’ by definition.

Boundary Scan Pins Order:

All the control signals are active LOW, that is, the output is enabled when its control signal is "0".

pad signal	chain pos	pad type	scan type	control signal
-	[145]	ctrl	norm	s_io_oe_buf3_Z
P/SData[0]	[144]	bidir	norm	s_io_oe_buf3_Z
P/SData[1]	[143]	bidir	norm	s_io_oe_buf3_Z
P/SData[2]	[142]	bidir	norm	s_io_oe_buf3_Z
P/SData[3]	[141]	bidir	norm	s_io_oe_buf3_Z
P/SData[4]	[140]	bidir	norm	s_io_oe_buf3_Z

pad signal	chain pos	pad type	scan type	control signal
P/SData[5]	[139]	bidir	norm	s_io_oe_buf3_Z
P/SData[6]	[138]	bidir	norm	s_io_oe_buf3_Z
P/SData[7]	[137]	bidir	norm	s_io_oe_buf3_Z
P/SData[8]	[136]	bidir	norm	s_io_oe_buf3_Z
P/SData[9]	[135]	bidir	norm	s_io_oe_buf3_Z
P/SData[10]	[134]	bidir	norm	s_io_oe_buf3_Z
P/SData[11]	[133]	bidir	norm	s_io_oe_buf3_Z
P/SData[12]	[132]	bidir	norm	s_io_oe_buf3_Z
P/SData[13]	[131]	bidir	norm	s_io_oe_buf3_Z
P/SData[14]	[130]	bidir	norm	s_io_oe_buf3_Z
P/SData[15]	[129]	bidir	norm	s_io_oe_buf3_Z
Test*	[128]	input	obsrv	
Rst*	[127]	input	obsrv	
-	[126]	ctrl	norm	adbusgnt_pad_bst_oe_PO
ADBusGnt	[125]	3state	norm	adbusgnt_pad_bst_oe_PO
ADBusReq	[124]	input	obsrv	
-	[123]	ctrl	norm	p_test*
DMAInt[2,0:1]*	[122:120]	3state	norm	p_test*
Hold	[119]	3state	norm	p_test*
RdyRcv*	[118]	3state	norm	p_test*
-	[117]	ctrl	norm	ad_en_buf2_Z
AD[0]	[116]	bidir	norm	ad_en_buf2_Z
AD[1]	[115]	bidir	norm	ad_en_buf2_Z
AD[2]	[114]	bidir	norm	ad_en_buf2_Z
AD[3]	[113]	bidir	norm	ad_en_buf2_Z
AD[4]	[112]	bidir	norm	ad_en_buf2_Z
AD[5]	[111]	bidir	norm	ad_en_buf2_Z
AD[6]	[110]	bidir	norm	ad_en_buf2_Z
AD[7]	[109]	bidir	norm	ad_en_buf2_Z
AD[8]	[108]	bidir	norm	ad_en_buf2_Z
AD[9]	[107]	bidir	norm	ad_en_buf2_Z
AD[10]	[106]	bidir	norm	ad_en_buf2_Z
AD[11]	[105]	bidir	norm	ad_en_buf2_Z
AD[12]	[104]	bidir	norm	ad_en_buf2_Z
AD[13]	[103]	bidir	norm	ad_en_buf2_Z
AD[14]	[102]	bidir	norm	ad_en_buf2_Z
AD[15]	[101]	bidir	norm	ad_en_buf2_Z
AD[16]	[100]	bidir	norm	ad_en_buf2_Z
AD[17]	[99]	bidir	norm	ad_en_buf2_Z
AD[18]	[98]	bidir	norm	ad_en_buf2_Z
AD[19]	[97]	bidir	norm	ad_en_buf2_Z
AD[20]	[96]	bidir	norm	ad_en_buf2_Z
AD[21]	[95]	bidir	norm	ad_en_buf2_Z
AD[22]	[94]	bidir	norm	ad_en_buf2_Z
AD[23]	[93]	bidir	norm	ad_en_buf2_Z
AD[24]	[92]	bidir	norm	ad_en_buf2_Z
AD[25]	[91]	bidir	norm	ad_en_buf2_Z
AD[26]	[90]	bidir	norm	ad_en_buf2_Z
AD[27]	[89]	bidir	norm	ad_en_buf2_Z
AD[28]	[88]	bidir	norm	ad_en_buf2_Z
AD[29]	[87]	bidir	norm	ad_en_buf2_Z
AD[30]	[86]	bidir	norm	ad_en_buf2_Z
AD[31]	[85]	bidir	norm	ad_en_buf2_Z
Clock	[84]	input	obsrv	
-	[83]	ctrl	norm	p_io_oe1_buf_Z
BE[3:0]*	[82:79]	bidir	norm	p_io_oe1_buf_Z
-	[78]	input	obsrv	
-	[77]	ctrl	norm	p_io_oe0*
ALE	[76]	bidir	norm	p_io_oe0*
HoldA	[75]	input	obsrv	
W/R*	[74]	bidir	norm	p_io_oe0*
ADS*	[73]	bidir	norm	p_io_oe0*
-	[72]	ctrl	norm	d_test1*
LE*	[71]	3state	norm	d_test1*
LRdOE*	[70]	3state	norm	d_test1*
LWrOE*	[69]	3state	norm	d_test1*
DWr*	[68]	3state	norm	d_test1*
RAS[3:0]*	[67:64]	3state	norm	d_test1*
-	[63]	ctrl	norm	d_test0_buf_Z
CAS[3:0]*	[62:59]	3state	norm	d_test0_buf_Z
-	[58]	ctrl	norm	d_test0_i*
DAdr[0:10]	[57:47]	3state	norm	d_test0_i*
Ready*	[46]	input	obsrv	
-	[45]	ctrl	norm	p_test_buf_Z
BufOE*	[44]	3state	norm	p_test_buf_Z
-	[43]	ctrl	norm	d_test1*
WrEn[3:0]*	[42:39]	3state	norm	d_test1*
BootCS*	[38]	3state	norm	d_test1*
DevCS[2:0]*	[37:35]	3state	norm	d_test1*
-	[34]	ctrl	norm	s_test*
SCS[3:0]	[33:30]	3state	norm	s_test*
DMAReq[2:0]	[29:27]	input	obsrv	
-	[26]	ctrl	norm	s_test1*
DMAAck[2:0]*	[25:23]	bidir	norm	s_test1*
SBE[1:0]*	[22:21]	3state	norm	s_test*
SRd*	[20]	3state	norm	s_test*
SWr*	[19]	3state	norm	s_test*
SWait*	[18]	input	obsrv	

pad signal	chain pos	pad type	scan type	control signal
-	[17]	ctrl	norm	sp_test*
IOWrA*	[16]	3state	norm	sp_test*
IORdA*	[15]	3state	norm	sp_test*
WrEnA*	[14]	3state	norm	sp_test*
OEA*	[13]	3state	norm	sp_test*
CardEnA[2:1]*	[12:11]	3state	norm	sp_test*
WaitA*	[10]	input	obsrv	
IOWrB*	[9]	3state	norm	sp_test*
IORdB*	[8]	3state	norm	sp_test*
WrEnB*	[7]	3state	norm	sp_test*
OEB*	[6]	3state	norm	sp_test*
CardEnB[2:1]*	[5:4]	3state	norm	sp_test*
WaitB*	[3]	input	obsrv	
-	[2]	ctrl	norm	s_test_Z
P/SAddr[1:0]	[1:0]	3state	norm	s_test_Z

2.9 Reset Configuration

The GT-32090 must acquire some knowledge of the system before it is configured by the software. The following configuration pins are sampled from when Rst* is asserted, until 4 Clock cycles after it is deasserted:

DMAAck[2:0]* - Indicates the three MSB's of the GT-32090 Internal address space decode, bits 31:29. These bits are sampled into the Internal Address Space Decode & Control register bits [5:3].

DMAReq[2:0] - Indicates the three MSB's of the GT-32090's four devices address, bits 31:29. These bits are sampled into bits [6:4] of the Device n Address Space registers.

3 REGISTER TABLES

The GT-32090's internal registers are memory mapped and can be accessed by the CPU. The registers' address is comprised of the value in the Internal Address Space Decode and Control register and the register's Offset. For example, to access the "Channel 0 DMA Byte Count" register (Offset 0x800) immediately after reset, assuming that during reset DMAAck[2:0]*'s value was "110", the following occurs: the value in the "Internal Address Space Decode and Control" register bits [5:0] will be "0x33", which must match the AD bus bits [31:26] as "110011", and the offset being 0x800, will result in a 32-bit address of 0xcc000800. The location of the registers in the memory space can be changed by changing the value programmed into the Internal Address Space Decode and Control register. For example, after changing the value in this register by writing to 0xcc00001c a value of "0x28", an access to the "Channel 0 DMA Byte Count" register will be with 0xa0000800.

The GT-32090's internal registers are 32-bits wide and consequently the appropriate memory region must be configured as a 32-bits region, which is done by programming the CPU's appropriate PMCON register.

3.1 Register Map

Description	Offset
Group Address Space	
DRAM Address Space	0x000
Device 0 Address Space	0x004
Device 1 Address Space	0x008
Device 2 Address Space	0x00c
Boot Device Address Space	0x010
SIO Address Space	0x014
PCMCIA Address Space	0x018
Internal Address Space Decode and Control	0x01c
DRAM Address Space	
RAS[0] Decode Address	0x400
RAS[1] Decode Address	0x404
RAS[2] Decode Address	0x408
RAS[3] Decode Address	0x40c
Device Address Space	
CS[0] Decode Address	0x410
CS[1] Decode Address	0x414
CS[2] Decode Address	0x418
BootCS Decode Address	0x41c
DRAM Refresh Configuration	
Refresh Configuration	0x420
DRAM Parameters	
DRAM Parameters	0x424
Device Parameters	
Device Bank0 Parameters	0x428
Device Bank1 Parameters	0x42c
Device Bank2 Parameters	0x430
Device Boot Bank Parameters	0x434

DMA Record	
Channel 0 DMA Byte Count	0x800
Channel 1 DMA Byte Count	0x804
Channel 2 DMA Byte Count	0x808
Channel 0 DMA Source Address	0x810
Channel 1 DMA Source Address	0x814
Channel 2 DMA Source Address	0x818
Channel 0 DMA Destination Address	0x820
Channel 1 DMA Destination Address	0x824
Channel 2 DMA Destination Address	0x828
Channel 0 Next Record Pointer	0x830
Channel 1 Next Record Pointer	0x834
Channel 2 Next Record Pointer	0x838
DMA Channel Control	
Channel 0 Control	0x840
Channel 1 Control	0x844
Channel 2 Control	0x848
DMA Arbiter	
Arbiter Control	0x860
SIO Configuration	
Arbiter Control	0xc00
Channel Flush/Reset	0xc04
Channel 0 Mode	0xc08
Channel 1 Mode	0xc0c
Channel 2 Mode	0xc10
Channel 3 Mode	0xc20
PCMCIA Configuration	
PCMCIA A Mode	0xc18
PCMCIA B Mode	0xc1c

3.2 Group Address Space

There are eight main address space segments that the GT-32090 decodes. The segments are DRAM, four AD devices, PCMCIA, SIO devices, and Internal registers.

DRAM Address Space, Offset: 0x000

Bits	Field Name	Function	Initial Value
4:0	Decode	The DRAM banks will be accessed when AD bits 31:27 match the value programmed in bits 4:0.	0x00

Device 0 Address Space, Offset: 0x004

Bits	Field Name	Function	Initial Value
6:0	Decode	The AD device bank will be accessed when AD bits 31:25 match the value programmed in bits 6:0.	{DMAReq[2:0],1,1,1,1} * Bits [6:4] are sampled during reset from pins DMAReq[2:0].

Device 1 Address Space, Offset: 0x008

Bits	Field Name	Function	Initial Value
6:0	Decode	The AD device bank will be accessed when AD bits 31:25 match the value programmed in bits 6:0.	{DMAReq[2:0],1,1,1,1} * Bits [6:4] are sampled during reset from pins DMAReq[2:0].

Device 2 Address Space, Offset: 0x00c

Bits	Field Name	Function	Initial Value
6:0	Decode	The AD device bank will be accessed when AD bits 31:25 match the value programmed in bits 6:0.	{DMAReq[2:0],1,1,1,1} * Bits [6:4] are sampled during reset from pins DMAReq[2:0].

Boot Device Address Space, Offset: 0x010

Bits	Field Name	Function	Initial Value
6:0	Decode	The AD device bank will be accessed when AD bits 31:25 match the value programmed in bits 6:0.	{DMAReq[2:0],1,1,1,1} * Bits [6:4] are sampled during reset from pins DMAReq[2:0].

SIO Address space, Offset: 0x014

Bits	Field Name	Function	Initial Value
4:0	Decode	The SIO banks will be accessed when AD bits 31:27 match the value programmed in bits 4:0.	0x10

PCMCIA Address Space, Offset: 0x018

Bits	Field Name	Function	Initial Value
3:0	Decode	The PCMCIA cards will be accessed when AD bits 31:28 match the value programmed in bits 3:0.	0xa

Internal Address Space Decode & Control, Offset: 0x01c

Bits	Field Name	Function	Initial Value
5:0	Decode	Registers inside the GT-32090 will be accessed when AD bits 31:26 match the value programmed in bits 5:0.	{DMAAck[2:0]*,0,1,1} * Bits [5:3] are sampled during reset from DMAAck[2:0]*.
7:6	Reserved		
8	Endian	The endianess of the AD Bus. This bit is used for reads and writes to internal registers in order to interpret correctly the data on the CPU bus upon access to the GT-32090. 0 - Little endian 1 - Big endian	0x0
9	HoldMask	Masks all requests (i.e. internal DMA or external agent) from generating assertion of Hold when set. Should be set by the CPU only and can be used in critical routines of the CPU. Enables the processor to disable all bus and DMA requests. 0 - Enable requests 1 - Mask requests	0x1
10	ExtOwn	The External Agent can own the bus regardless of the state of its ADBusReq signal, as long as this bit is set to ‘1’. When set to ‘1’, the GT-32090 asserts Hold continuously. Set only by the External Agent and only when it is master of the bus. 0 - Release the bus 1 - Own the bus	0x0
11	Reserved	Must be ‘0’	0x0

3.3 DRAM Address Space

The values of the Address Decode registers determine which physical DRAM bank will be accessed when the CPU or DMA issues an address. The address decoding is done by comparing address bits 25:20 to be equal or higher than the value in the Low fields, and equal or lower than the value in the High fields. Note that address bit 26 is not a part of the internal decoding and can be used for external decoding to generate two DRAM banks' RASs using one GT-32090 RAS.

RAS[0] Decode Address, Offset: 0x400

Bits	Field Name	Function	Initial Value
5:0	Low	Lowest decoded address value for RAS[0] activation.	0x0
11:6	High	Highest decoded address value for RAS[0] activation.	0x0f

RAS[1] Decode Address, Offset: 0x404

Bits	Field Name	Function	Initial Value
5:0	Low	Lowest decoded address value for RAS[1] activation.	0x10
11:6	High	Highest decoded address value for RAS[1] activation.	0x1f

RAS[2] Decode Address, Offset: 0x408

Bits	Field Name	Function	Initial Value
5:0	Low	Lowest decoded address value for RAS[2] activation.	0x20
11:6	High	Highest decoded address value for RAS[2] activation.	0x2f

RAS[3] Decode Address, Offset: 0x40c

Bits	Field Name	Function	Initial Value
5:0	Low	Lowest decoded address value for RAS[3] activation.	0x30
11:6	High	Highest decoded address value for RAS[3] activation.	0x3f

3.4 Device Address Space

The values of the Address Decode registers determine which physical device bank will be accessed when the CPU or DMA issues an address. The address decoding is done by comparing address bits 24:21 to be equal or higher than the value in the Low fields, and equal or lower than the value in the High fields.

CS[0] Decode Address, Offset: 0x410

Bits	Field Name	Function	Initial Value
3:0	Low	Lowest decoded address value for CS[0] activation.	0x0
7:4	High	Highest decoded address value for CS[0] activation.	0x1

CS[1] Decode Address, Offset: 0x414

Bits	Field Name	Function	Initial Value
3:0	Low	Lowest decoded address value for CS[1] activation.	0x2
7:4	High	Highest decoded address value for CS[1] activation.	0x3

CS[2] Decode Address, Offset: 0x418

Bits	Field Name	Function	Initial Value
3:0	Low	Lowest decoded address value for CS[2] activation.	0x4
7:4	High	Highest decoded address value for CS[2] activation.	0x5

BootCS Decode Address, Offset: 0x41c

Bits	Field Name	Function	Initial Value
3:0	Low	Lowest decoded address value for BootCS activation.	0x7
7:4	High	Highest decoded address value for BootCS activation.	0x7

3.5 DRAM Refresh Configuration

This register specifies refresh parameters. The time between DRAM refresh cycles is programmable, with the option to refresh all the banks at the same time or in staggered fashion.

Refresh Configuration, Offset: 0x420

Bits	Field name	Function	Initial Value
15:0	RefIntCnt	Refresh interval count value. The number of clock cycles between refreshes. The GT-32090 will perform CAS before RAS refreshes every ‘RefIntCnt’ clock cycles.	0xf8
16	StagRef	Staggered refresh. 0 - All the RAS signals are asserted simultaneously at refresh. 1 - Staggered refresh.	0x0

3.6 DRAM Parameters

This register specifies the DRAM timing parameters and the DRAM's refresh type support. The parameter BanknRef which configures the refresh type support, can be set for each DRAM bank independently. The DRAM state machines are optimized to different bus frequencies. In order to control the number of cycles to first data, the ADFreq bits should be set according to the bus frequency.

DRAM Parameters, Offset: 0x424

Bits	Field name	Function	Initial Value
0	Type	DRAM type used. 0 - Standard page mode 1 - EDO	0x0
1	Latch	Defines whether the DRAM has an external latch on its data lines or not. 0 - No external latch 1 - With external latch	0x0
3:2	ADFreq	Processor bus frequency. 00 - 16MHz 01 - 20MHz 10 - 25MHz 11 - 33MHz	0x11
5:4	Bank0Ref	DRAM refresh type support. 00 - 1/2K Refresh (9 bits row, 9 to 11 bits column) 01 - 1K Refresh (10 bits row, 9 to 11 bits column) 10 - 2K Refresh (11 bits row, 9 to 11 bits column) 11 - Not used	0x0
7:6	Bank1Ref	DRAM refresh type support. 00 - 1/2K Refresh (9 bits row, 9 to 11 bits column) 01 - 1K Refresh (10 bits row, 9 to 11 bits column) 10 - 2K Refresh (11 bits row, 9 to 11 bits column) 11 - Not used	0x0
9:8	Bank2Ref	DRAM refresh type support. 00 - 1/2K Refresh (9 bits row, 9 to 11 bits column) 01 - 1K Refresh (10 bits row, 9 to 11 bits column) 10 - 2K Refresh (11 bits row, 9 to 11 bits column) 11 - Not used	0x0
11:10	Bank3Ref	DRAM refresh type support. 00 - 1/2K Refresh (9 bits row, 9 to 11 bits column) 01 - 1K Refresh (10 bits row, 9 to 11 bits column) 10 - 2K Refresh (11 bits row, 9 to 11 bits column) 11 - Not used	0x0

3.7 Device Parameters

Device parameters can be different for each bank. The shape of the different control signals that are active in a device access can be programmed. The access time (in number of cycles) of the device during read accesses should be programmed into the AccToFirst field, to set the time data from the device will be ready to be sampled by the CPU or by the GT-32090. AccToNext should be programmed with the time data from the device can be sampled in consecutive accesses during burst accesses. The DevCS* will be deasserted after the last data is latched and to prevent bus contention the TurnOff field specifies the number of cycles (from the deassertion of DevCS*) to the beginning of the next bus transaction. The write signals pulse can be shaped as well. The parameters specify the number of cycles from the beginning of the cycle to the assertion of the write signals (CSToWr), the number of cycles the write is active (WrActive), and the number of cycles the write signals are inactive (WrHigh) between consecutive writes in a burst access. The Ready* signal that can be used to insert wait states to each device data transfer can be enabled/disabled per bank via the Ready field.

Device Bank0 Parameters, Offset: 0x428

Bits	Field name	Function	Initial Value
1:0	TurnOff	The number of cycles between the deassertion of DevCS* and a new AD bus cycle.	0x3
4:2	AccToFirst	The number of cycles from DevCS* active to first data (determined by the access time of the device). Actual value is n+2 (‘00’ is not allowed).	0x7
7:5	AccToNext	The number of cycles between the sampling of data and the next sampling point of data (time difference between sampling points in a burst sequence). Actual value is n+1.	0x7
9:8	CSToWr	The number of cycles from DevCS* to first assertion of WrEn* Actual value is n+2 (‘00’ is not allowed).	0x3
11:10	WrActive	The number of cycles WrEn* is active. Actual value is n+1.	0x3
13:12	WrHigh	The number of cycles between deassertion and assertion of WrEn*. Actual value is n+1 (‘00’ is allowed for 32-bit devices only).	0x3
14	Ready	Controls bus cycle extension, can be disabled by DevCS*. 0 - Enabled, will extend the bus cycle 1 - Disabled	0x0
15	DevWidth	Determines the width configuration for the device. 0 - The device is configured as 8-bits wide. 1 - The device is configured as 16- or 32- bits wide.	0x1

Device Bank1 Parameters, Offset: 0x42c

Bits	Field Name	Function	Initial Value
15:0	Various	Fields function as in Device Bank0.	0xbfff

Device Bank2 Parameters, Offset: 0x430

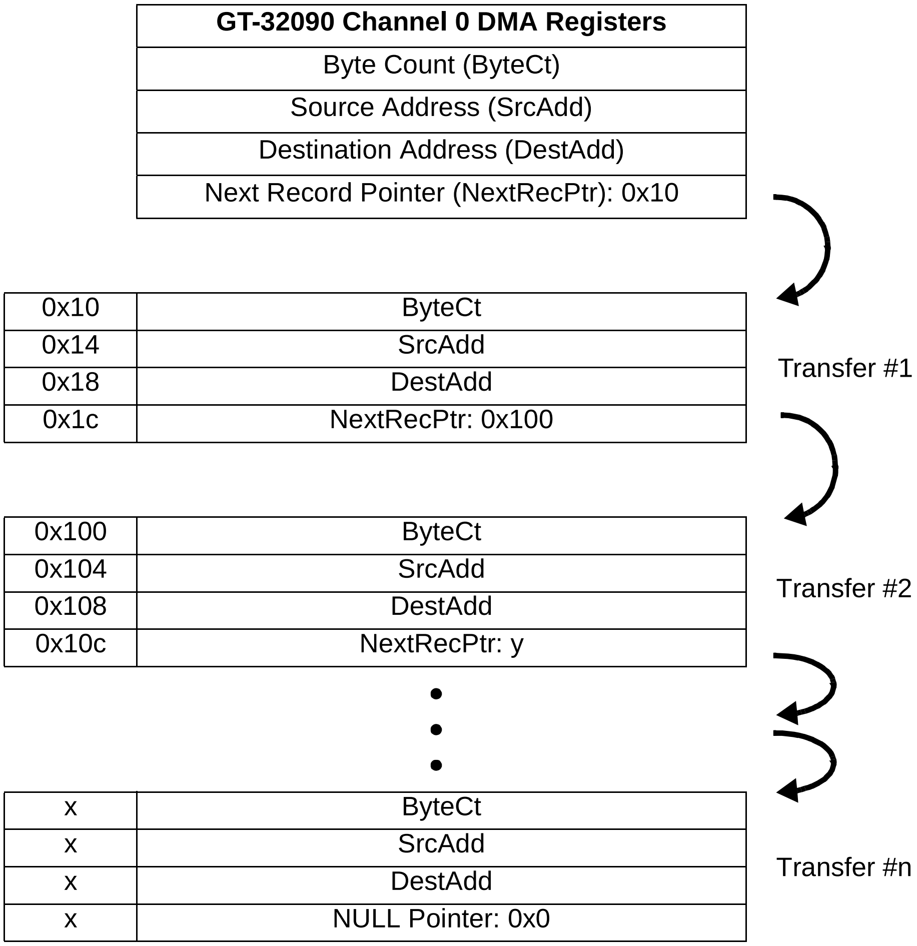
Bits	Field Name	Function	Initial Value
15:0	Various	Fields function as in Device Bank0.	0xbfff

Device Boot Bank Parameters, Offset: 0x434

Bits	Field Name	Function	Initial Value
15:0	Various	Fields function as in Device Bank0.	0xbfff

3.8 DMA Record

Each DMA channel record includes four registers: byte count, source address, destination address, and a pointer to the next record. The active record can be written by the CPU, or by the DMA controller in the process of fetching a new record from memory. The structure of the record is illustrated in the following example:



GT-32090 Channel 0 DMA Registers
Byte Count (ByteCt)
Source Address (SrcAdd)
Destination Address (DestAdd)
Next Record Pointer (NextRecPtr): 0x10

0x10	ByteCt
0x14	SrcAdd
0x18	DestAdd
0x1c	NextRecPtr: 0x100

0x100	ByteCt
0x104	SrcAdd
0x108	DestAdd
0x10c	NextRecPtr: y

x	ByteCt
x	SrcAdd
x	DestAdd
x	NULL Pointer: 0x0

Channel 0 DMA Byte Count, Offset: 0x800

Bits	Field Name	Function	Initial Value
15:0	ByteCt	The number of bytes that are left in DMA transfers.	0x0

Channel 1 DMA Byte Count, Offset: 0x804

Bits	Field Name	Function	Initial Value
15:0	ByteCt	The number of bytes that are left in DMA transfers.	0x0

Channel 2 DMA Byte Count, Offset: 0x808

Bits	Field Name	Function	Initial Value
15:0	ByteCt	The number of bytes that are left in DMA transfers.	0x0

Channel 0 DMA Source Address, Offset: 0x810

Bits	Field Name	Function	Initial Value
31:0	SrcAdd	The address that the DMA controller will read the data from.	0x0

Channel 1 DMA Source Address, Offset: 0x814

Bits	Field Name	Function	Initial Value
31:0	SrcAdd	The address that the DMA controller will read the data from.	0x0

Channel 2 DMA Source Address, Offset: 0x818

Bits	Field Name	Function	Initial Value
31:0	SrcAdd	The address that the DMA controller will read the data from.	0x0

Channel 0 DMA Destination Address, Offset: 0x820

Bits	Field Name	Function	Initial Value
31:0	DestAdd	The address that the DMA controller will write the data to.	0x0

Channel 1 DMA Destination Address, Offset: 0x824

Bits	Field Name	Function	Initial Value
31:0	DestAdd	The address that the DMA controller will write the data to.	0x0

Channel 2 DMA Destination Address, Offset: 0x828

Bits	Field Name	Function	Initial Value
31:0	DestAdd	The address that the DMA controller will write the data to.	0x0

Channel 0 Next Record Pointer, Offset: 0x830

Bits	Field Name	Function	Initial Value
31:0	NextRecPtr	The address for the next record of DMA. A value of 0 means a NULL pointer (end of the chained list).	0x0

Channel 1 Next Record Pointer, Offset: 0x834

Bits	Field Name	Function	Initial Value
31:0	NextRecPtr	The address for the next record of DMA. A value of 0 means a NULL pointer (end of the chained list).	0x0

Channel 2 Next Record Pointer, Offset: 0x838

Bits	Field Name	Function	Initial Value
31:0	NextRecPtr	The address for the next record of DMA. A value of 0 means a NULL pointer (end of the chained list).	0x0

3.9 DMA Channel Control

Each DMA channel has a control register to set its mode of operation independently of the other two channels. A channel can be programmed to transfer data through the GT-32090 (internal) or directly between DRAM and devices (fly-by). In the internal mode, the DMA reads data from the source address (SIO, devices or DRAM) into an internal 16-byte FIFO. From the internal FIFO, the data is written to a destination address (SIO, devices or DRAM) that can be independent from the source address. In fly-by mode, data can be transferred only between a 32-bit wide DRAM and a 32-bit device through the AD bus. The DMA controller will generate addresses for the DRAM and DMAAck* for the device. Transfer direction (DRAM to device or device to DRAM) has to be programmed in the Control register's FlyByDir field.

Source addresses and destination addresses can be programmed to increment, decrement, or hold the same value throughout the DMA transfer. For devices that can absorb a limited number of bytes at a time, the channel can be programmed to limit the number of bytes transferred in each DMA cycle. DMA accesses can be initiated by an external source (Demand mode) by asserting one of the three DMAReq[2:0] pins, or by an internal request (Block mode) until the byte count reaches zero.

All three channels have chaining support via linked lists of records. When the chaining mode is enabled, the DMA controller will fetch the information (the record) for a new DMA transfer directly out of memory without involving the CPU. The location of the next record is in the Next Record Pointer register (NextRecPtr) and the DMA controller will fetch records every DMA transfer end until it reaches the NULL pointer (NULL pointer is zero).

There are several mechanisms for status and control of the DMA operations. A status interrupt can be programmed to be asserted every time the DMA byte count reaches zero, or only when byte count reaches zero and the record is the last record in the chain (the NextRecPtr is NULL). In addition, there is a status bit that indicates whether a channel is active or not. A channel is active when it is enabled and its byte count is other than zero, or in chained mode when both its byte count is not equal to zero or its NextRecPtr is not equal to NULL, or when it is disabled and its internal FIFO is not empty. A channel can be controlled by disabling it temporarily, and a next record fetch can be forced in chained mode even if the current DMA has not ended.

Channel 0 Control, Offset: 0x840

Bits	Field name	Function	Initial Value
0	FlyBy	Selection of fly-by or normal operation. 0 - No fly-by mode, data is read into the DMA's FIFO and then written to the destination address. 1 - Fly-by mode, data is transferred to/from devices on the AD Bus to/from DRAM	0x0
1	FlyByDir	This bit is meaningful only in fly-by mode. 0 - Read from DRAM 1 - Write to DRAM	0x0
3:2	SrcDir	Source Direction. 00 - Increment source address 01 - Decrement source address 10 - Hold in the same value 11 - Not allowed	0x0
5:4	DestDir	Destination Direction. 00 - Increment destination address 01 - Decrement destination address 10 - Hold in the same value. 11 - Not allowed	0x0
8:6	DataTransLim	Data Transfer Limit in each DMA access. 000 - 4 bytes 001 - 8 bytes 011 - 16 bytes 111 - Reserved	0x0
9	ChainMod	Chained Mode. 0 - Chained mode; when a DMA access is terminated, the parameters of the next DMA access will come from a record in memory that a NextRecPtr register points to. 1 - Non-chained mode; only the values that are programmed by the CPU directly into the ByteCt, SrcAdd, and DestAdd registers are used.	0x0
10	IntMode	Interrupt Mode. 0 - Interrupt asserted every time the DMA byte count reaches terminal count. 1 - Interrupt every NULL pointer (in Chained mode).	0x0
11	TransMod	Transfer Mode. 0 - Demand 1 - Block	0x0
12	ChanEn	Channel Enable. 0 - Disable the channel for DMA accesses 1 - Enable the channel for DMA accesses	0x0

Bits	Field name	Function	Initial Value
13	FetNexRec	Fetch Next Record. 0 - Normal state (this bit is never written as ‘0’, this is the toggle state after fetch is completed). 1 - Forces a fetch of the next record (even if the current DMA has not ended). This bit is reset after fetch is completed (meaningful only in Chained mode).	0x0
14	DMAActSt	DMA Activity Status (read only). 0 - Channel is not active 1 - Channel is active	0x0

Channel 1 Control, Offset: 0x844

Bits	Field Name	Function	Initial Value
14:0	Various	Fields function as in Channel 0 Control	0x0

Channel 2 Control, Offset: 0x848

Bits	Field Name	Function	Initial Value
14:0	Various	Fields function as in Channel 0 Control	0x0

Some extra notes on DMA programming follow:

- Non-Chained mode:

In this mode, Source, Destination and Byte Count should be initialized prior to enabling the channel. ChainMod should be set to ‘1’.

- Chained mode:

All the channel record’s parameters for the current transaction (Source, Byte Count, Destination and Next Record Pointer) should be initialized in DRAM or Devices. The address of the first record should be initialized by writing it to the NextRecPtr of the channel. The channel should be enabled via ChanEn=1, the FetNexRec should be set to ‘1’ and the ChainMod should be set to ‘0’.

- Restarting a disabled channel

In Non-Chained mode, ChanEn should be set to ‘1’.

In Chained mode, the software should find out if the first fetch took place. If it did, only ChanEn should be set to ‘1’. If it did not, the FetNexRec should also be set to ‘1’.

- Reprogramming an active channel

The channel should first be disabled via ChanEn=0. Then it must be assured that the channel is no longer active (for example by polling the DMAActSt of the channel). New DMA parameters should be programmed prior to re-enabling the channel via ChanEn=1.

3.10 DMA Arbiter

The DMA controller has a programmable arbitration scheme between its three channels. The channels are grouped into two groups, one group includes channel 0 and 1, and the other group includes only channel 2. The channels in the first group can be programmed so that one of them will have the higher priority, or they can both have the same priority in round robin fashion. The priority between the two groups can be programmed in a similar way so that a selected group has a higher priority, or they can both have the same priority in round robin.

The priority scheme has additional flexibility with the programmable Priority Option. With the Priority Option the DMA bandwidth allocation can be divided in a fairer way. For example, if the PrioOpt bit is set to ‘0’ and the PrioGrps field is set as ‘10’, the requesting devices will get the DMA in the order 0,1,2,0,1,2,0,1,2,0,1,2,..... (assuming that PrioChan1/0 is set to round robin), while if the PrioOpt bit is set to ‘1’ the requesting devices will get the DMA in the order 0,1,0,1,0,1,2,2,2,..... The DMA arbiter control register can be reprogrammed any time regardless of the channels’ status (active or not active).

Some arbitration examples follow to facilitate the understanding of this register:

1. Assuming all 3 channels are requested all the time,
   with Arbiter Control register = 0x40, the order will be: 0,2,1,2,0,2,1,2,0,2,1,2,.....
   with Arbiter Control register = 0x0, the order will be: 0,1,2,0,1,2,0,1,2,.....
2. Assuming all 3 channels are requested all the time ,
   with Arbiter Control register = 0x51, the order will be: 2,2,2,2,..1,1,1,1,...,0,0,0,0,.....
   with Arbiter Control register = 0x11, the order will be: 2,1,2,0,2,1,2,0,.....
3. Assuming all 3 channels are requested all the time,
   with Arbiter Control register = 0x50, the order will be: 2,2,2,2,....,1,0,1,0,...
   with Arbiter Control register = 0x10, the order will be: 2,0,2,1,2,0,2,1,.....

Arbiter Control, Offset: 0x860

Bits	Field name	Function	Initial Value
1:0	PrioChan1/0	Priority between Channel 0 and Channel 1. 00 - Round robin 01 - Priority to channel 1 over channel 0 10 - Priority to channel 0 over channel 1 11 - Reserved	0x0
3:2	Reserved	Must be 0x0	0x0
5:4	PrioGrps	Priority between the group of channels 0&1, and channel 2. 00 - Round robin 01 - Priority to channel 2 over 0&1 10 - Priority to channels 0&1 over 2 11 - Reserved	0x0
6	PrioOpt	Defines the arbiter behavior for the high priority device. 0 - High priority device will relinquish the bus for a requesting device for one DMA transaction. 1 - High priority device will be granted as long as it requests the bus.	0x0

3.11 SIO Configuration

These registers configure and control the SIO channels. The registers include: the SIO local Arbiter Control register that enables several priority options between the DMA channels that are assigned to the SIO; the Channel Flush/Reset register that enables the CPU to clear the data from the DMA packing and unpacking registers; and the four Channel Mode registers that define the device parameters (bus width, endianess), the shape of the control signals (pulse width, burst support, turn-off width), and the DMA parameters (request polarity, arbitration boundaries, DMA direction, DMA channel assignment). **Note that channel 3 does not have DMA capabilities nor burst support.** Channel 3 is for CPU access only.

The Channel Flush/Reset register should only be written to the value of ‘1’. After completion of the Flush/Reset operation, the appropriate bit will be reset to ‘0’.

Arbiter Control, Offset: 0xc00

Bits	Field name	Function	Initial Value
1:0	PrioChan1/0	Priority between Channel 0 and Channel 1. 00 - Round robin 01 - Priority to channel 1 over channel 0 10 - Priority to channel 0 over channel 1 11 - Reserved	0x0
3:2	Reserved	Must be 0x0	0x0
5:4	PrioGrps	Priority between the group of channels 0&1, and channel 2. 00 - Round robin 01 - Priority to channel 2 over 0 & 1 10 - Priority to channels 0 & 1 over 2 11 - Reserved	0x0
6	PrioOpt	Defines the arbiter behavior for the high priority device 0 - High priority device will relinquish the bus for a requesting device for one DMA transaction. 1 - High priority device will be granted as long as it requests the bus.	0x0

Channel Flush/Reset, Offset: 0xc04

Bits	Field name	Function	Initial Value
0	FlushRstCh0	During an SIO DMA read access, writing ‘1’ will flush the contents of the packing register into the target device on the AD bus. During an SIO DMA write access, writing ‘1’ will clear the unpacking register.	0x0
1	FlushRstCh1	Field functions as in bit 0.	0x0
2	FlushRstCh2	Field functions as in bit 0.	0x0

Channel 0 Mode, Offset: 0xc08

Bits	Field name	Function	Initial Value
0	DMASel	Selects if the DMA is to an SIO device or to an AD bus device. 0 - AD bus DMA 1 - SIO bus DMA	0x0
1	BusWid	The data path width of the device on the SIO. 0 - 8-bits 1 - 16-bits	0x0
2	Endian	Defines the device byte ordering. 0 - Little endian 1 - Big endian	0x0
3	ArbBound	This bit defines whether the channel arbitrates for the SIO bus during a DMA operation every access (byte or 16-bit word) or only when the DMA is finished packing or unpacking its register. 0 - Every access 1 - Every word	0x0
4	BurstMode	Selects if the DMAAck* corresponding to this channel will stay LOW through a burst DMA or will be deasserted after every read or write. In Burst Mode, the strobe signal (i.e. SWr*, SRd*) is deasserted for one cycle during the access. 0 - DMAAck* does not stay LOW 1 - DMAAck* stays LOW	0x0
5	WrRd	Defines the DMA direction. 0 - Read from an SIO device 1 - Write to an SIO device	0x1
9:6	PulsWid	Pulse width, the number of cycles the SRd* or SWr* signals will be asserted. The actual number will be n+1.	0xf
11:10	TOWid	Turn-Off width, the number of cycles between the deassertion of the SCS* corresponding to this channel in a read access and a) the start of a read cycle to a PCMCIA card or to an SIO device, or b) a write access to a PCMCIA card or an SIO device.	0x3
12	DMAReqPol	Selects the polarity of the DMAReq signal corresponding to this channel. 0 - Active HIGH 1 - Active LOW	0x0
13	TotalMask	When this bit is ‘1’ the corresponding DMAReq is masked. 0 - Enable DMA request. 1 - Mask DMA request.	0x1
14:17	ByteEn	Byte enables of the packing/unpacking register. These bits are read only. This field is only relevant for unpacking.	oxf

Channel 1 Mode, Offset: 0xc0c

Bits	Field name	Function	Initial Value
17:0	Various	Fields function as in Channel 0.	0x3efe0

Channel 2 Mode, Offset: 0xc10

Bits	Field name	Function	Initial Value
17:0	Various	Fields function as in Channel 0.	0x3efe0

Channel 3 Mode, Offset: 0xc20

Bits	Field name	Function	Initial Value
0	Reserved	Must be ‘0’	0x0
1	BusWid	The data path width of the device on the SIO. 0 - 8-bit 1 - 16-bit	0x0
2	Endian	Defines the device byte ordering. 0 - Little endian 1 - Big endian	0x0
5:3	Reserved		0x0
9:6	PulsWid	Pulse width, the number of cycles the SRd* or SWr* signals will be asserted. The actual number will be n+1.	0xf
11:10	TOWid	Turn-off width, the number of cycles between the deassertion of SCS* in a read access and a) the start of a read cycle to a PCMCIA card or to an SIO device, or b) a write access to a PCMCIA card or an SIO device.	0x3

3.12 PCMCIA Configuration

The PCMCIA mode registers set the timing and device parameters for each PCMCIA card. The parameters are the device bus width, the endianess of the device, and the timing of the control signals to the cards.

PCMCIA A Mode, Offset: 0xc18

Bits	Field name	Function	Initial Value
0	MemBusWid	Selects 8- or 16-bit memory device data width. 0 - 8-bit 1 - 16-bit	0x0
1	I/OBusWid	Selects 8- or 16-bit I/O device data width. 0 - 8-bit 1 - 16-bit	0x0
2	Endian	Defines the PCMCIA card byte ordering. 0 - Little endian 1 - Big endian	0x0

Bits	Field name	Function	Initial Value
3	Reserved	Must be ‘0’.	0x0
7:4	MemPulsWid	The number of cycles OEA*, WrEnA* will be active (the actual number is n+1).	0xf
9:8	MemTOWid	The number of cycles between the deassertion of CardEnA[2:1]* in a read access and a) the start of a read cycle from the second PCMCIA card or from an SIO device, or b) a write access to a PCMCIA card or an SIO device	0x3
11:10	Reserved	Must be ‘00’	0x0
15:12	I/OPulsWid	The number of cycles IOWrA*, or IORdA* will be active (the actual number is n+1).	0xf
17:16	I/OTOWid	The number of cycles between the deassertion of CardEnA[2:1]* in a read access and a) the start of a read cycle to the second PCMCIA card or to an SIO device, or b) a write access to a PCMCIA card or an SIO device.	0x3

PCMCIA B Mode, Offset: 0xc1c

Bits	Field name	Function	Initial Value
17:0	Various	Fields function as in PCMCIA A.	0x3f3f0

4 RESTRICTIONS

4.1 CPU Interface

a) The DRAM space must be configured as a 32-bit bus in the i960Jx’s registers.
b) The SIO and the PCMCIA bus must be configured as a 16-bit bus in the i960Jx’s registers. Configuring 8-bit devices is possible by programming the GT-32090’s registers only.
c) The GT-32090’s internal registers space must be configured as a 32-bit bus in the i960Jx’s appropriate PMCON register.

4.2 AD Bus Device Controller

a) When signal shaping parameters like WrActive, AccToFirst, AccToNext are programmed to their minimum allowed values, the Ready* signal will be ignored during the relevant time.
b) The minimum allowed values of the parameters CSToWr and AccToFirst is 1 (i.e 3 clock cycles).
c) The minimum allowed value of the parameter WrHigh is 1 when the device is either 8- or 16-bits wide, and 0 when it is 32-bits wide.

4.3 DMA

a) DMA transactions which involve the SIO require the address bits [1:0] of both source and destination to have the same value.
b) For DMA on the SIO bus, when the SIO channel is configured to Burst Mode, source and destination addresses, as well as byte count, must be word-aligned.
c) There is DMA support only for an AD bus device which is 32-bits wide.
d) For each transfer, the minimum byte count is 4. For DMA to/from the SIO, the field DataTransLim in the DMA Channel Control register is restricted to the value ‘000’ - a maximum of 4 bytes.
e) In order to restart a channel after it has been enabled, it should first be checked that the DMAActSt bit is set to NOT ACTIVE (see section 3.9 for details).
f) When the source or destination address is decremented, both addresses should be word-aligned (that is, A1 and A0 should be both zero), and Byte Count should be a multiple of 4.
g) When using the address Hold option in the source direction (SrcDir in section 3.9), the source address should be word-aligned.
h) When using the address Hold option in the destination direction (DestDir in section 3.9), both Source and Destination addresses should be word-aligned.
i) Records’ addresses (NextRecPtr) should be a multiple of 16.
j) In fly-by mode, both source and destination addresses should be word-aligned (that is, A1 and A0 should be both zero), and byte count should be a multiple of 4.
k) When the DMA is programmed to Demand mode (with or without SIO channel), first the DMA registers should be programmed and then the corresponding SIO register (Channel Mode register).
l) For DMA on the SIO bus, the SIO register (Channel Mode register) should be programmed last in a Configuration of DMA process. In a Re-configuration of DMA with an SIO device this register should be written twice: once with a ‘0’ value to bit 0 (DMASel) and then with a ‘1’ value to that same bit, precisely in this sequence.

4.4 SIO & PCMCIA

a) When the parameter PulsWid is programmed to less then 2 (i.e. less then 3 clock cycles) the appropriate Wait* signal will be ignored during the assertion of the read or write signals.

4.5 Memory Mapping

a) When the same value is programmed to two or more of the Device n Address Space registers (which configure bits[31:25] of device #n’s address space), the regions corresponding to their CS[n] Decode Address registers (which configure High and Low values to bits[24:21] of device #n’s address space) must not be overlapped.

5 PINOUT TABLE

5.1 160 pin PQFP (sorted by number)

Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name
1	P/SData[0]	36	AD[3]	71	BE[3]*
2	P/SData[1]	37	AD[4]	72	BE[2]*
3	P/SData[2]	38	AD[5]	73	BE[1]*
4	P/SData[3]	39	AD[6]	74	BE[0]*
5	P/SData[4]	40	AD[7]	75	ALE
6	P/SData[5]	41	AD[8]	76	HoldA
7	P/SData[6]	42	AD[9]	77	W/R*
8	P/SData[7]	43	AD[10]	78	ADS*
9	VSS	44	AD[11]	79	LE*
10	VDD	45	VSS	80	LRdOE*
11	P/SData[8]	46	AD[12]	81	LWrOE*
12	P/SData[9]	47	AD[13]	82	DWr*
13	P/SData[10]	48	AD[14]	83	RAS[3]*
14	P/SData[11]	49	AD[15]	84	VSS
15	P/SData[12]	50	AD[16]	85	RAS[2]*
16	P/SData[13]	51	AD[17]	86	RAS[1]*
17	P/SData[14]	52	AD[18]	87	RAS[0]*
18	P/SData[15]	53	AD[19]	88	CAS[3]*
19	HiZAll*	54	AD[20]	89	CAS[2]*
20	Test*	55	AD[21]	90	CAS[1]*
21	Rst*	56	VSS	91	CAS[0]*
22	ADBusGnt	57	VDD	92	DAdr[0]
23	ADBusReq	58	AD[22]	93	DAdr[1]
24	VSS	59	AD[23]	94	DAdr[2]
25	VDD	60	AD[24]	95	DAdr[3]
26	DMAInt[2]*	61	AD[25]	96	VDD
27	DMAInt[0]*	62	VDD	97	VSS
28	DMAInt[1]*	63	VSS	98	DAdr[4]
29	Hold	64	AD[26]	99	DAdr[5]
30	RdyRcv*	65	AD[27]	100	DAdr[6]
31	AD[0]	66	AD[28]	101	VSS
32	AD[1]	67	AD[29]	102	VDD
33	VDD	68	AD[30]	103	DAdr[7]
34	VSS	69	AD[31]	104	DAdr[8]
35	AD[2]	70	VSS	105	DAdr[9]

Pin #	Signal Name	Pin #	Signal Name	Pin #	Signal Name
106	DAdr[10]	125	VSS	144	WaitA*
107	Ready*	126	VDD	145	IOWrB*
108	BufOE*	127	DMAAck[2]*	146	IORdB*
109	WrEn[3]*	128	DMAAck[1]*	147	WrEnB*
110	WrEn[2]*	129	DMAAck[0]*	148	VSS
111	VSS	130	SBE[1]*	149	VDD
112	WrEn[1]*	131	SBE[0]*	150	OEB*
113	WrEn[0]*	132	SRd*	151	CardEnB[2]*
114	BootCS*	133	SWr*	152	CardEnB[1]*
115	DevCS[2]*	134	SWait*	153	WaitB*
116	DevCS[1]*	135	VSS	154	P/SAddr[0]
117	DevCS[0]*	136	Clock	155	P/SAddr[1]
118	SCS[3]*	137	VDD	156	JTClk
119	SCS[2]*	138	IOWrA*	157	JTRst*
120	SCS[1]*	139	IORdA*	158	JTMS
121	SCS[0]*	140	WrEnA*	159	JTDI
122	DMAReq[2]	141	OEA*	160	JTDO
123	DMAReq[1]	142	CardEnA[2]*		
124	DMAReq[0]	143	CardEnA[1]*		

6 DC CHARACTERISTICS - *PRELIMINARY/SUBJECT TO CHANGE*

6.1 Absolute Maximum Ratings

Symbol	Parameter	Min.	Max.	Unit
Vdd	Supply Voltage	-0.3	6.5	V
Vi	Input Voltage	-0.3	Vdd+0.3	V
Vo	Output Voltage	-0.3	Vdd+0.3	V
Io	Output Current		24	mA
Iik	Input Protect Diode Current		+-20	mA
Iok	Output Protect Diode Current		+-20	mA
Top	Operating Temperature	0	85	oC
Tstg	Storage Temperature	-40	125	oC
ESD			2000	V
Pt	Power Dissipation		1.5	W

6.2 Recommended Operating Conditions

Symbol	Parameter	Min.	Typ.	Max.	Unit
Vdd	Supply Voltage	4.75		5.25	V
Vi	Input Voltage	0		Vdd	V
Vo	Output Voltage	0		Vdd	V
Top	Operating Temperature	0		70	oC
Cin	Input Capacitance		7.2		pF
Cout	Output Capacitance		7.2		pF

6.3 DC Electrical Characteristics Over Operating Range

(TC=0-70^{o}C; Vdd=+5V, +/-5%)

Symbol	Parameter	Test Condition	Min.	Typ.	Max.	Unit
Vih	Input HIGH level	Guaranteed Logic HIGH level	2.0		Vdd + 0.5V	V
Vil	Input LOW level	Guaranteed Logic LOW level	-0.5		0.8	V
Voh	Output HIGH Voltage	IoH = 2 mA IoH = 4 mA IoH = 8 mA IoH = 12 mA IoH = 16 mA IoH = 24 mA	2.4			V

Symbol	Parameter	Test Condition	Min.	Typ.	Max.	Unit
Vol	Output LOW Voltage	IoL = 2 mA IoL = 4 mA IoL = 8 mA IoL = 12 mA IoL = 16 mA IoL = 24 mA			0.4	V
Iih	Input HIGH Current				+-1	uA
Iil	Input LOW Current				+-1	uA
Iozh	High Impedance Output Current				+-1	uA
Iozl	High Impedance Output Current				+-1	uA
Vh	Input Hysteresis	Vdd = 4.5V Vdd = 5.0V Vdd = 5.5V	0.52 0.54 0.56		0.60 0.61 0.62	mV
Icc	Operating Current				220	mA

NOTE:

Pullup/Pulldown resistors are 45KOhm minimum, 65KOhm typical, 80KOhm maximum.

7 AC TIMING - *PRELIMINARY/SUBJECT TO CHANGE*

(TC= 0-70^{o}C; VDD= +5V, +/- 5%)

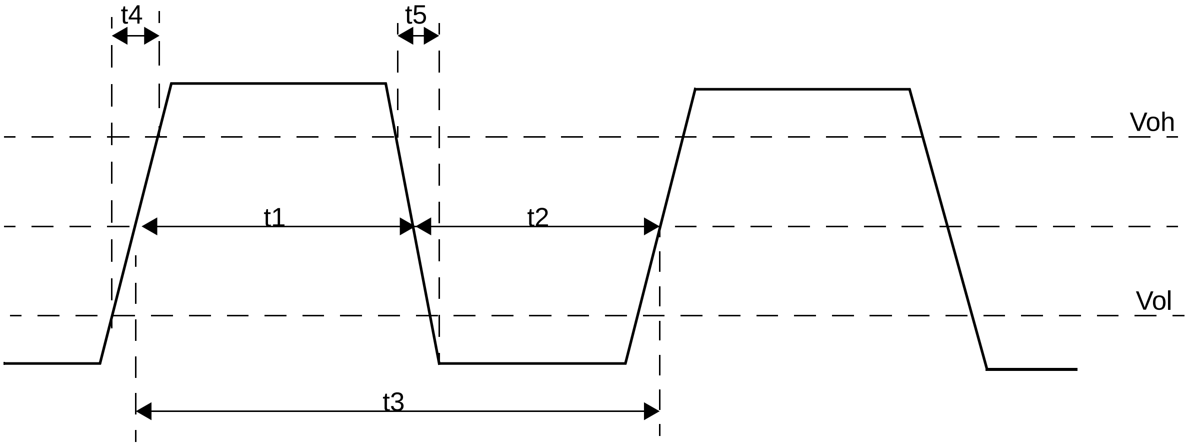
Symbol	Signals	Description	Min	Max	Unit
t1	Clock	Pulse Width High	13		nS
t2	Clock	Pulse Width Low	13		nS
t3	Clock	Clock Period	30		nS
t4	Clock	Rise Time		3	nS
t5	Clock	Fall Time		3	nS
t6	Rst*	Active	10		Clock
t7	DAdr[10:0],	Delay from Clock Rising or Falling Edge	2	20	nS
t7	ADBusGnt, DMAAck[2:0]*, DMAInt[2:0]*, SCS[3:0]*, SWr*, SRd*, P/SAddr[1:0], IOWrA*, IOWrB*, IORdA*, IORdB*, WrEnA*, WrEnB*, OEA*, OEB*, CardEnA[2:1]*, CardEnB[2:1]*, W/R*, ADS*, Hold, RdyRcv*, BE[3:0]*, AD[31:0], RAS[3:0]*, CAS[3:0]*, DWr*, LE* (asserted), LRdOE*, LWrOE*, BootCS*, DevCS[2:0]*, WrEn[3:0]*, BufOE*	Delay from Clock Rising Edge	2	15	nS
t8	DAdr[10:0]	Delay from Clock Falling Edge	2	20	nS
t8	P/SData[15:0]	Delay from Clock Falling Edge	2	17	nS
t8	SBE[1:0]*, CAS[3:0]*,	Delay from Clock Falling Edge	2	15	nS
t8	LE* (deasserted)	Delay from Clock Falling Edge	2	11	nS
t9	All Inputs	Setup	10		nS
t10	All Inputs	Hold	1		nS
t11	All Outputs	Float Delay	2	18	nS
t12	All Outputs	Drive Delay	2	12	nS

Notes:

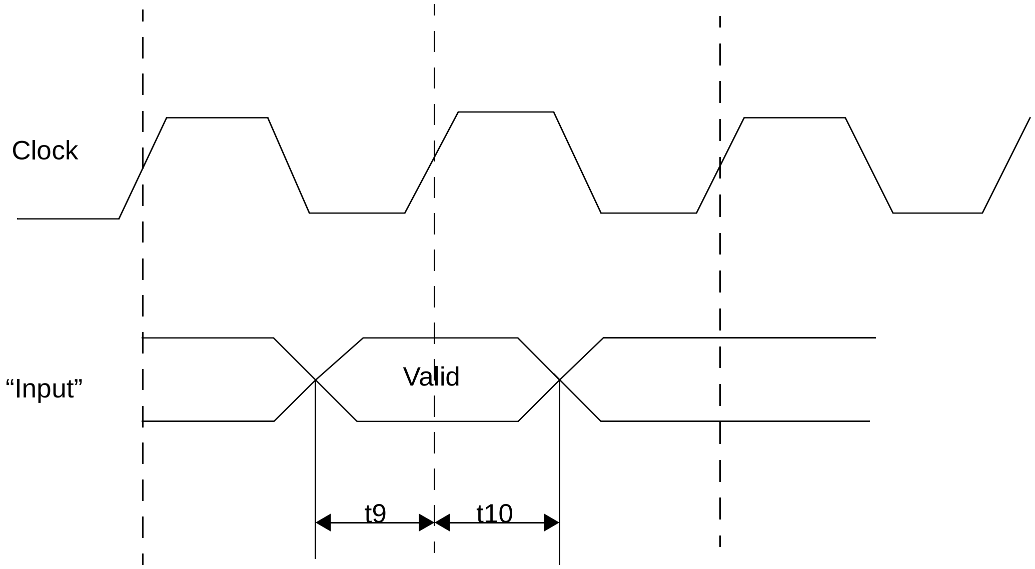
1. All Delays, Setup, and Hold times are referred to Clock rising edge, unless stated otherwise.

2. All outputs are specified for 50pf load.

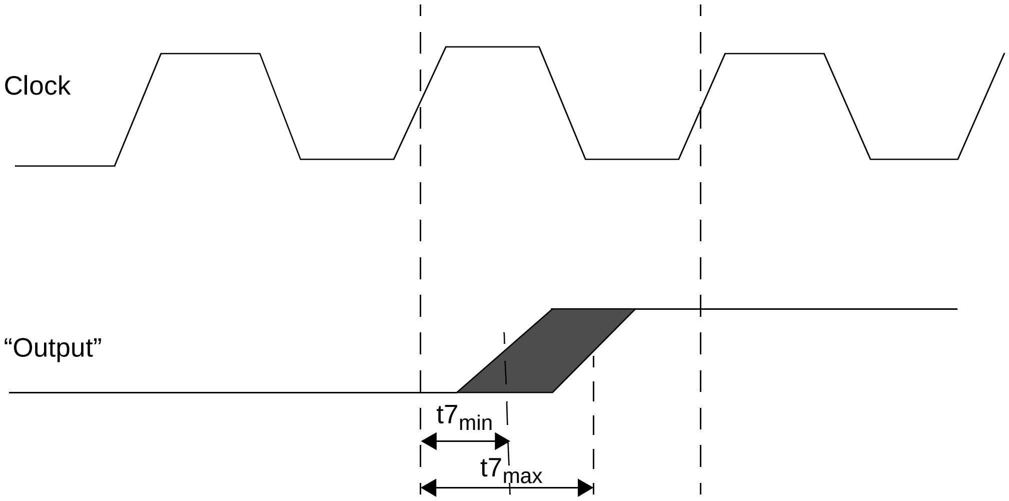
3. “All Inputs” and “All Outputs” also refer to I/O signals’ behavior.



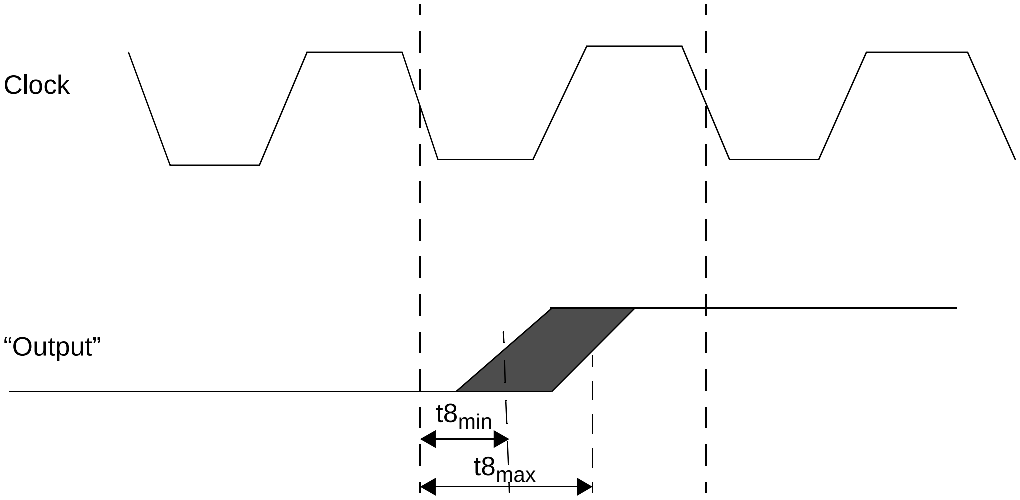
Clock



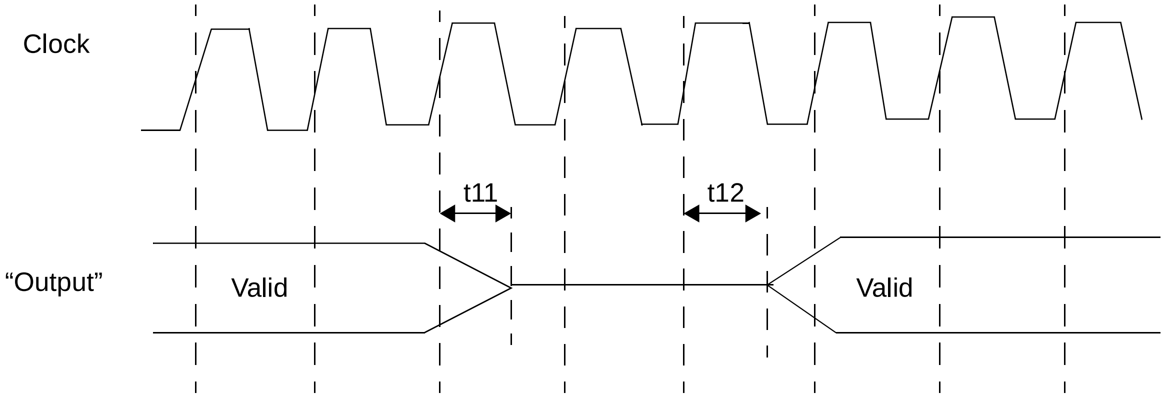
Inputs Setup and Hold



Output Delay from Clock Rising Edge

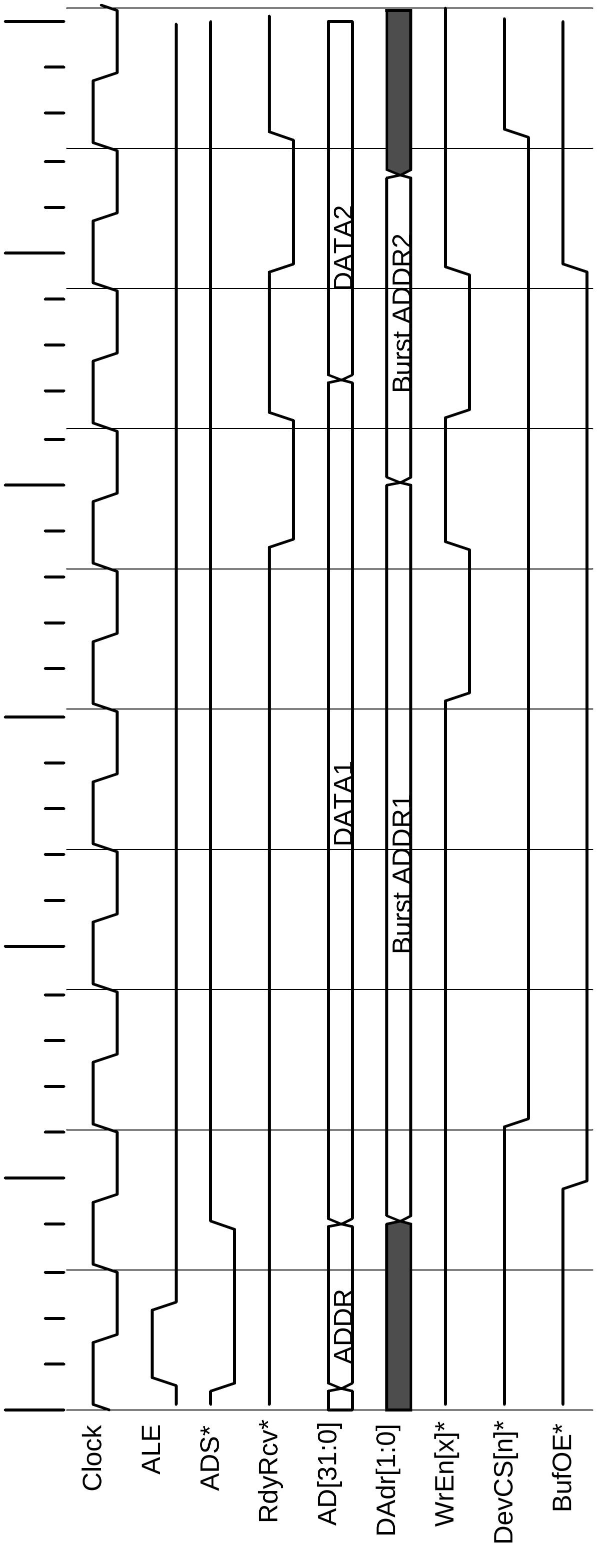


Output Delay from Clock Falling Edge

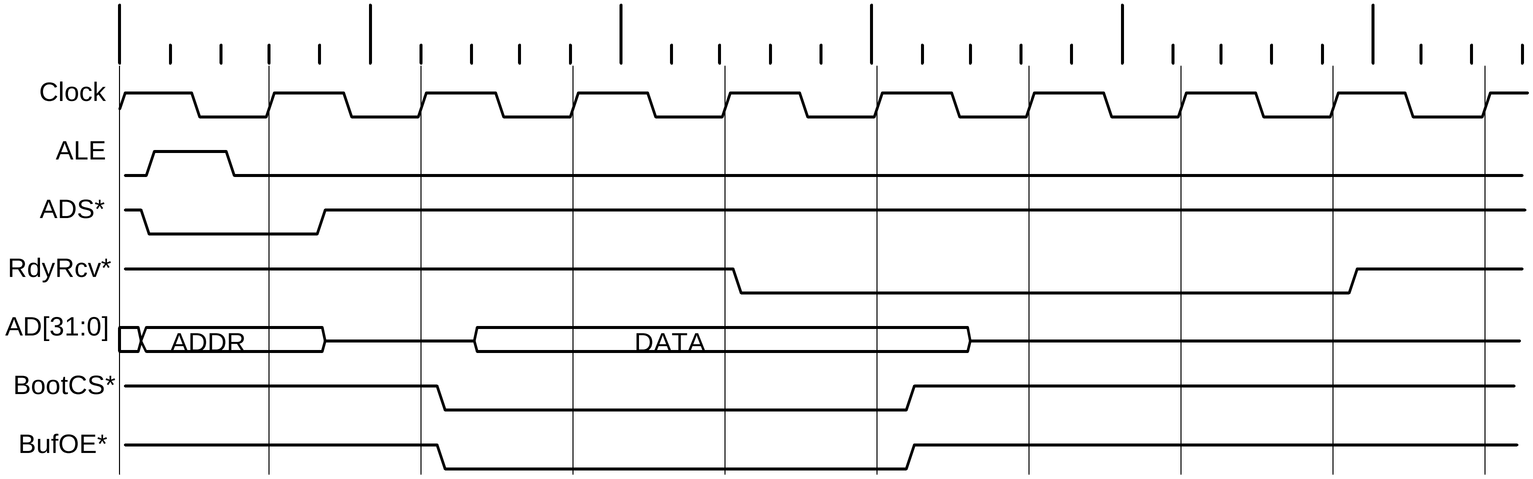


Output Float and Drive Delay

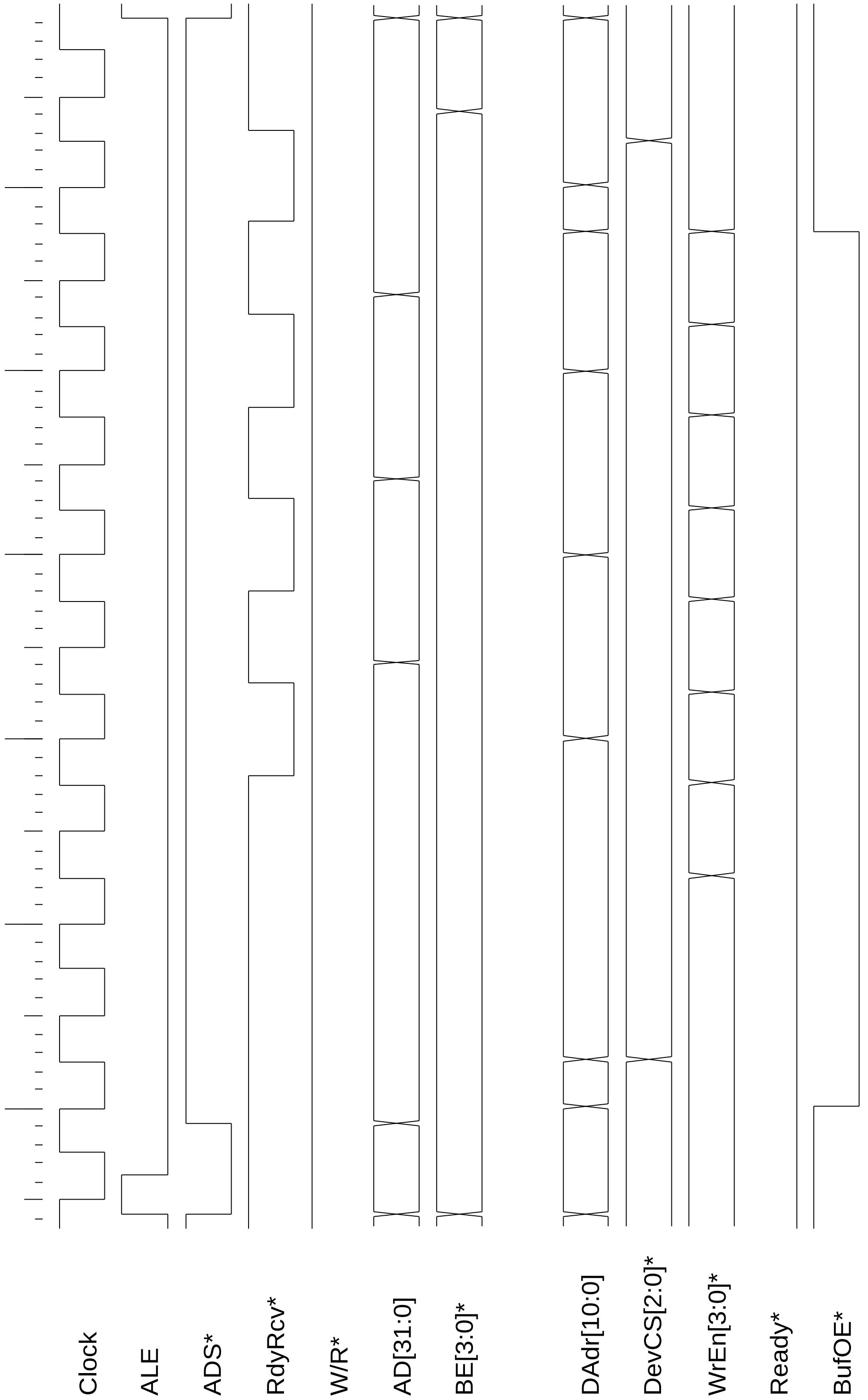
8 FUNCTIONAL WAVEFORMS



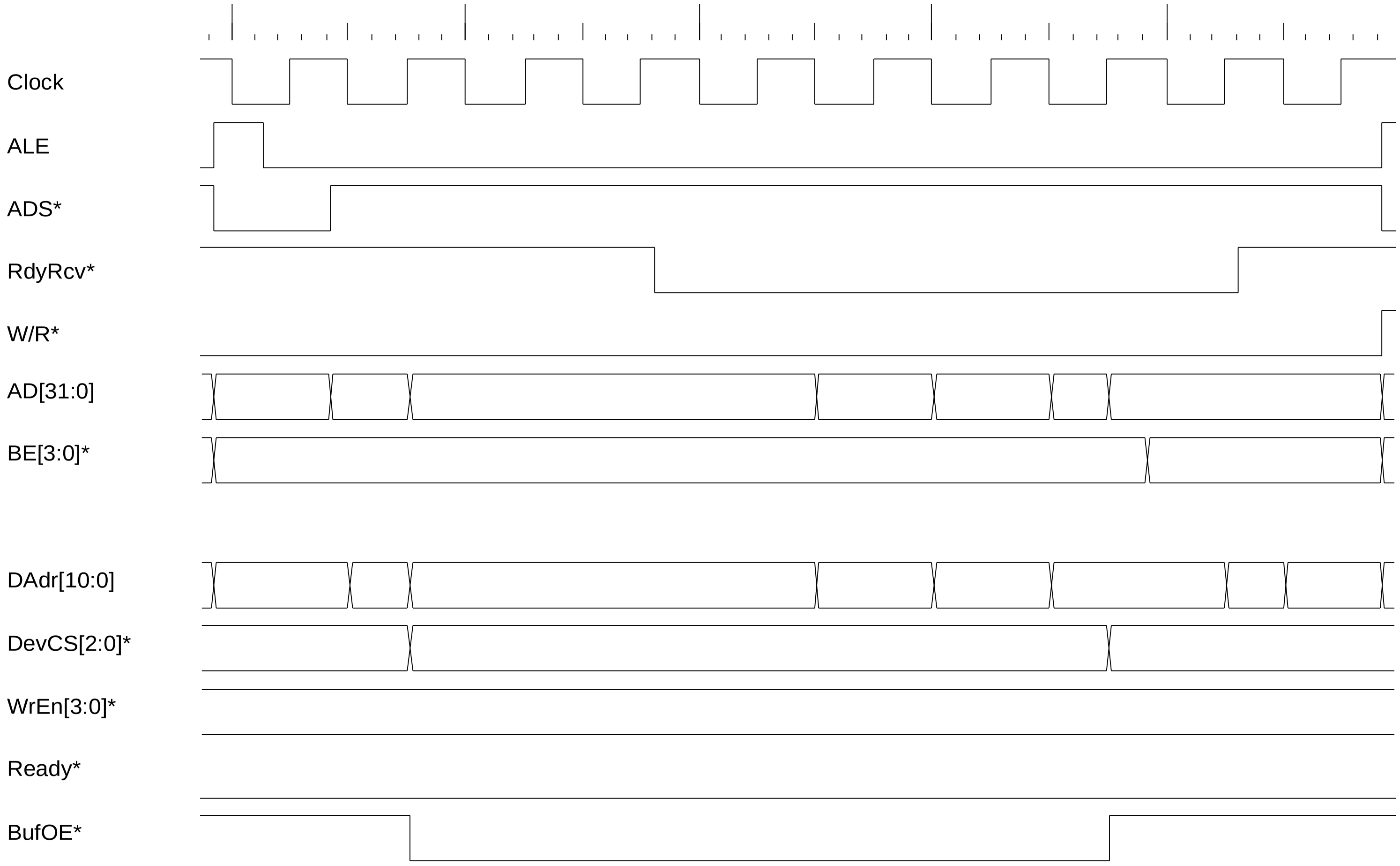
Double Word Write to 32-Bit Device



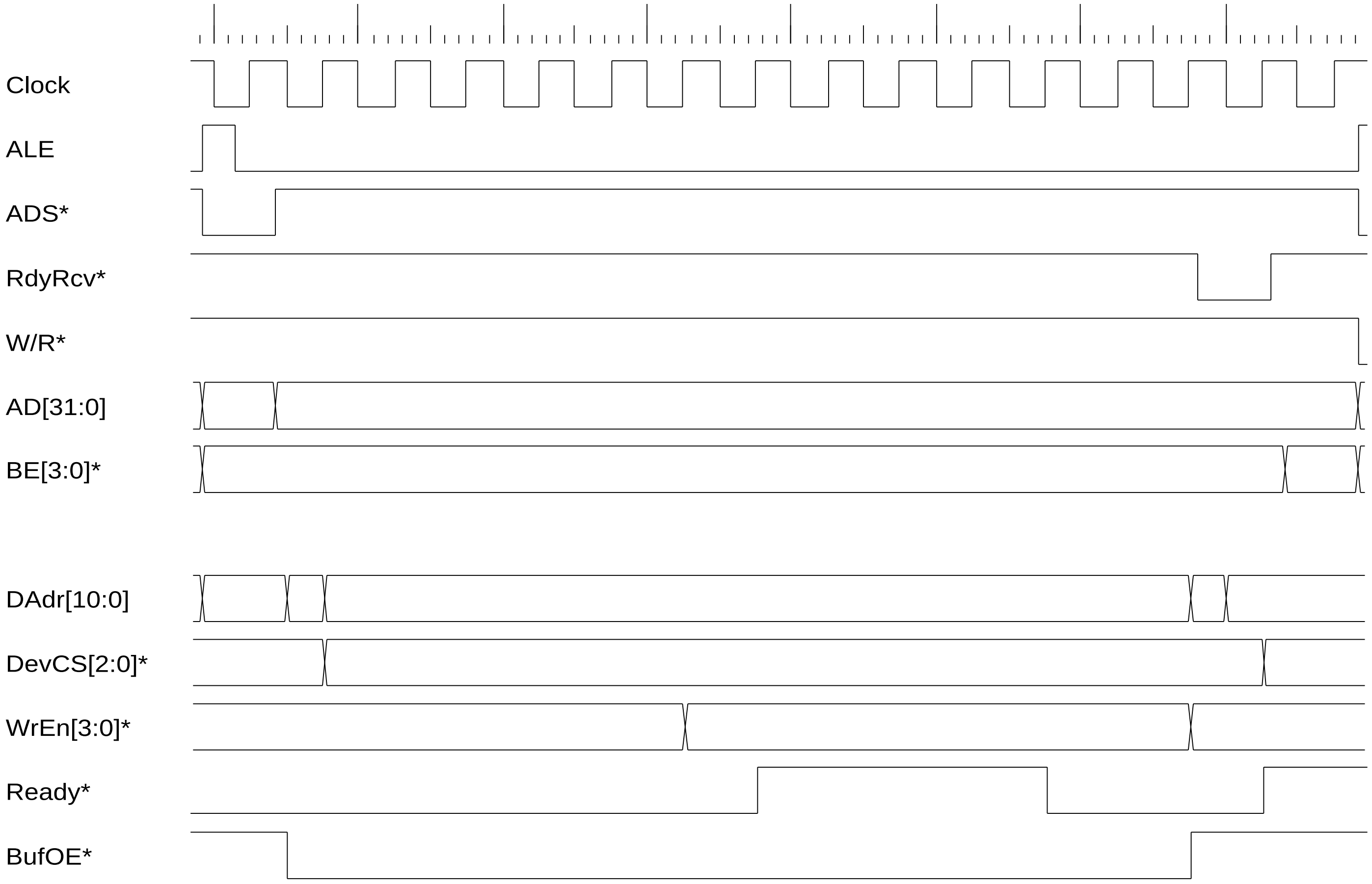
Byte Read from 8-Bit Boot Device



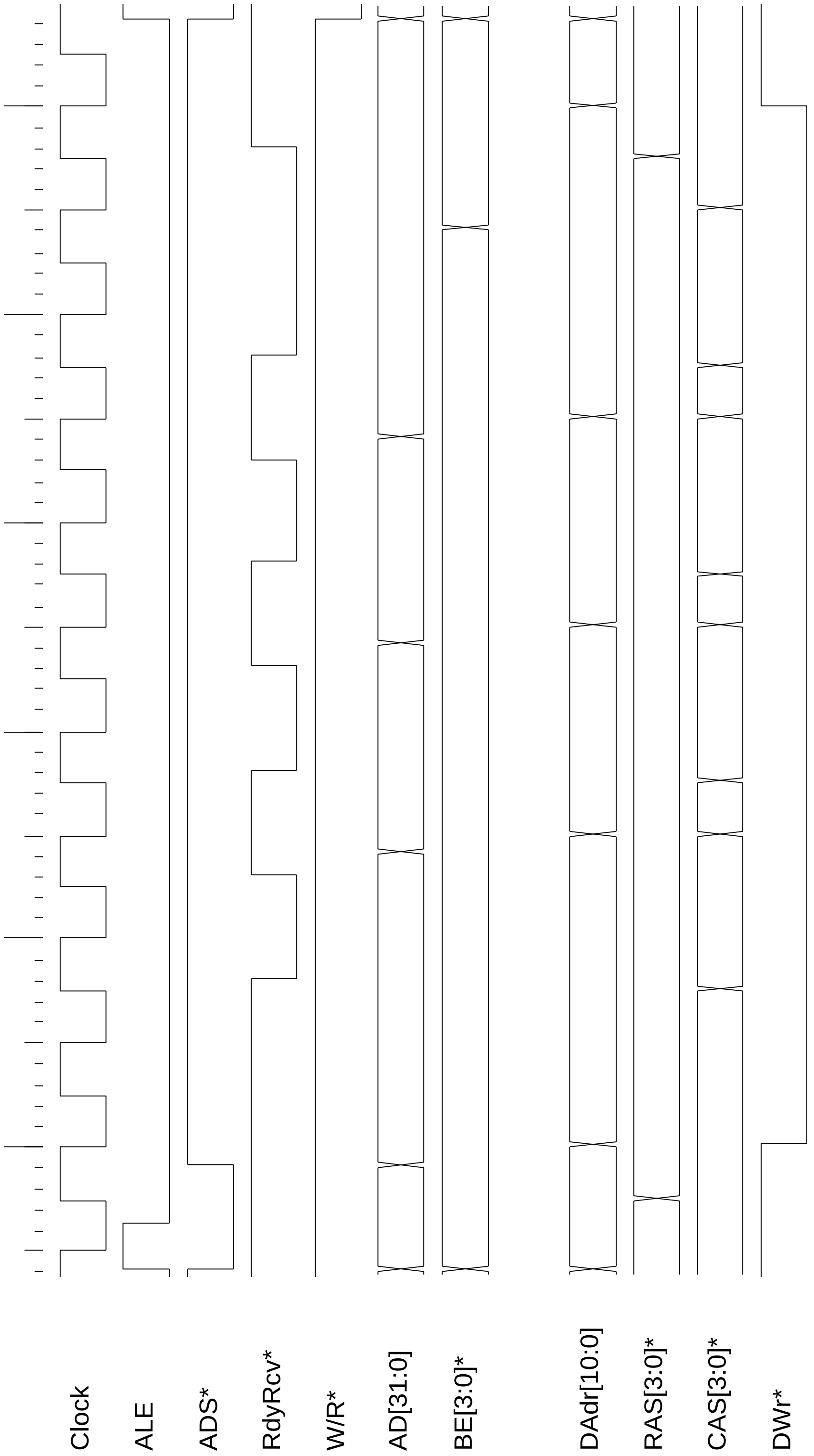
32-Bit Device Write, 2 Cycles CS2Write, 1 Cycle Write Active/Not Active



32-Bit Device Read, 2 Cycles to First, 1 Cycle Delay to Next and Turn-off

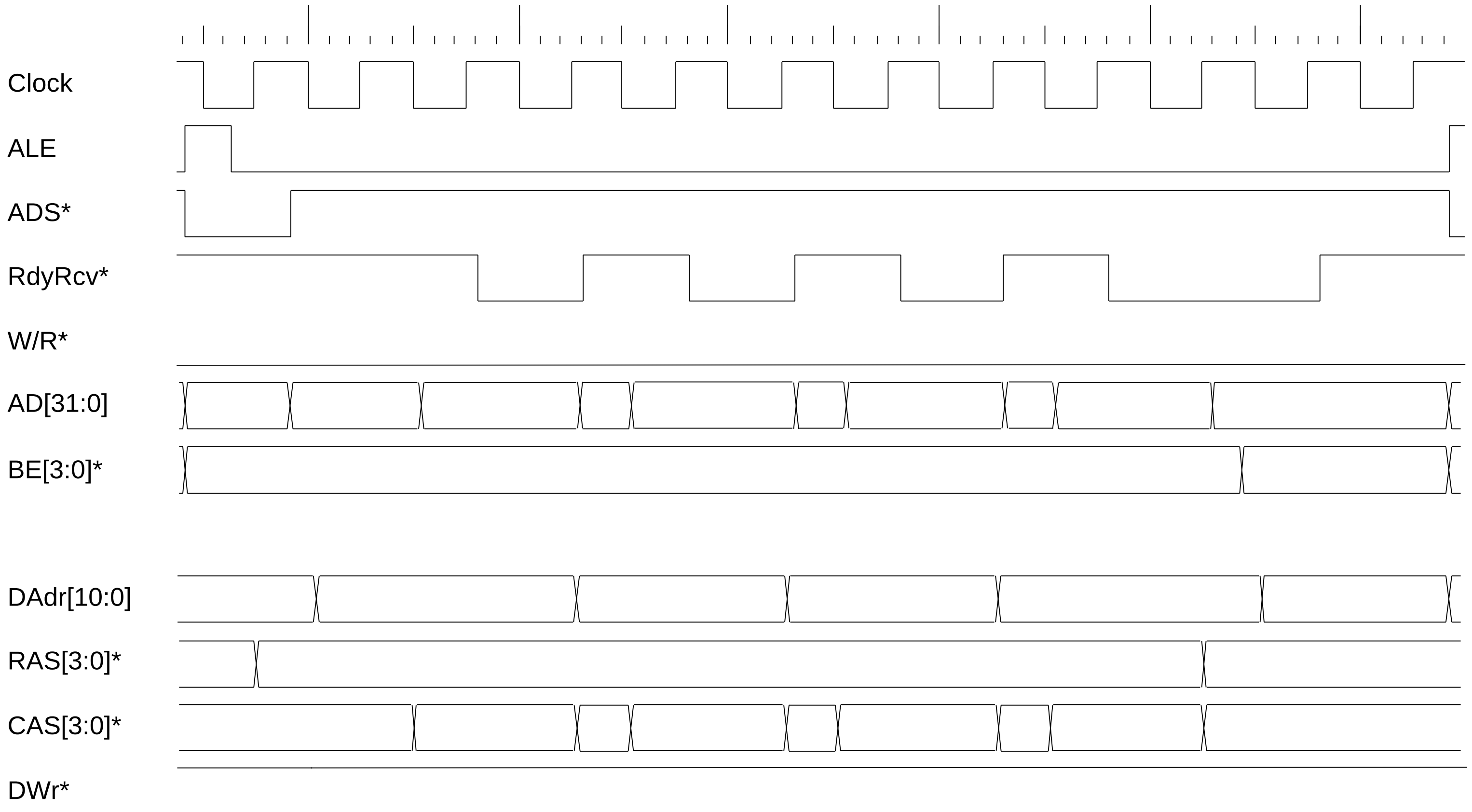


Device with Maximum Delays, 2 Bytes Write

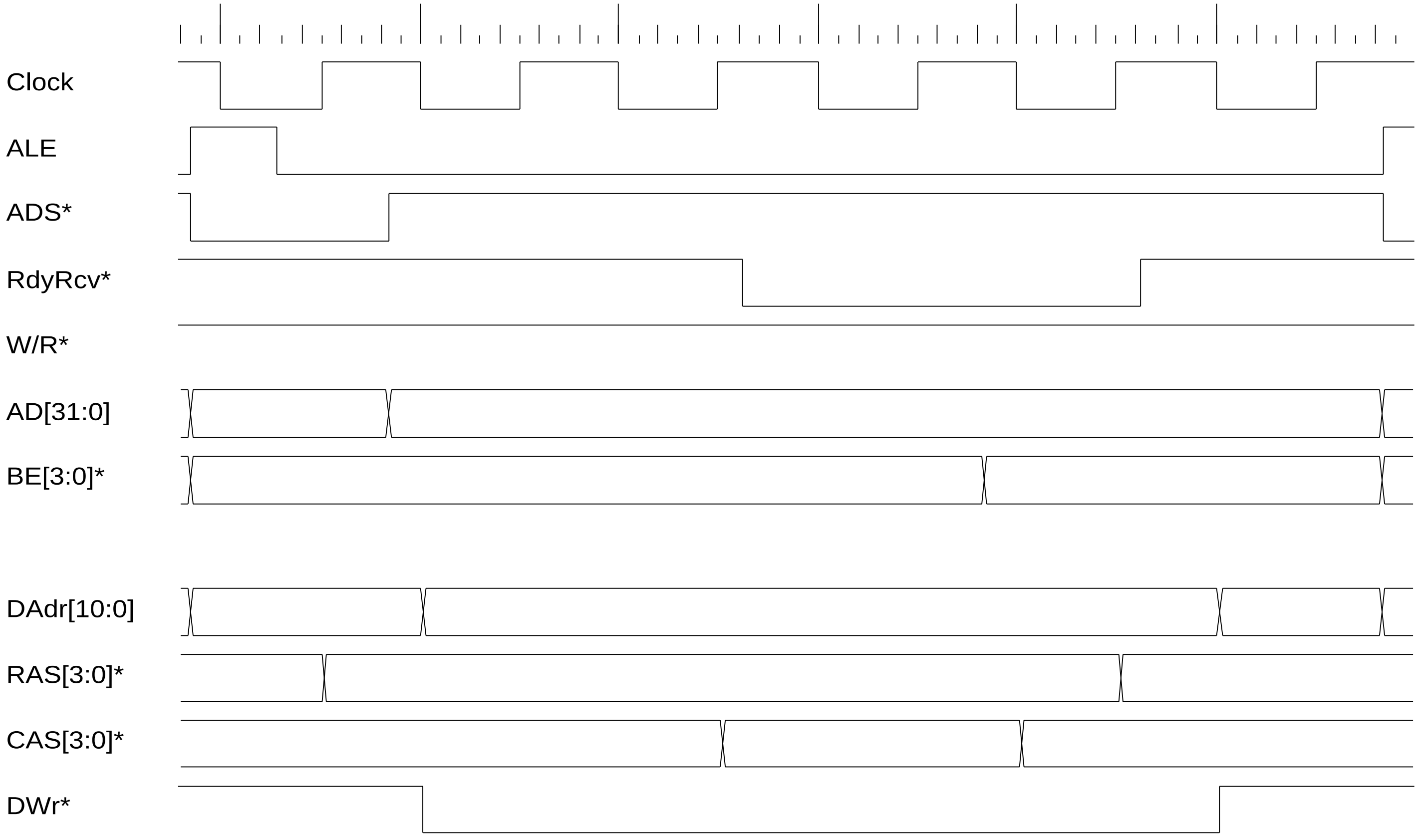


Standard DRAM 32-Bit Write 25 MHz

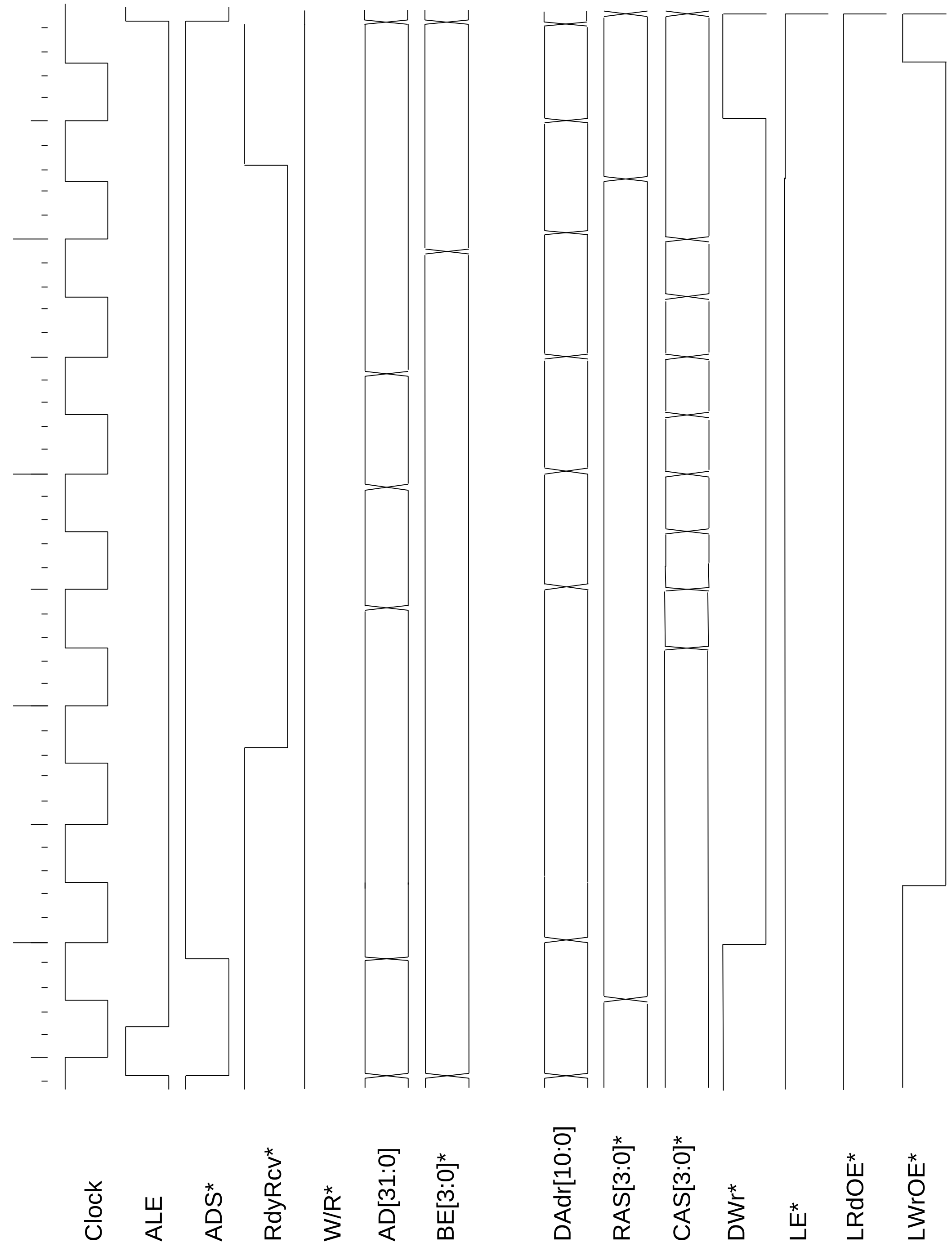
•



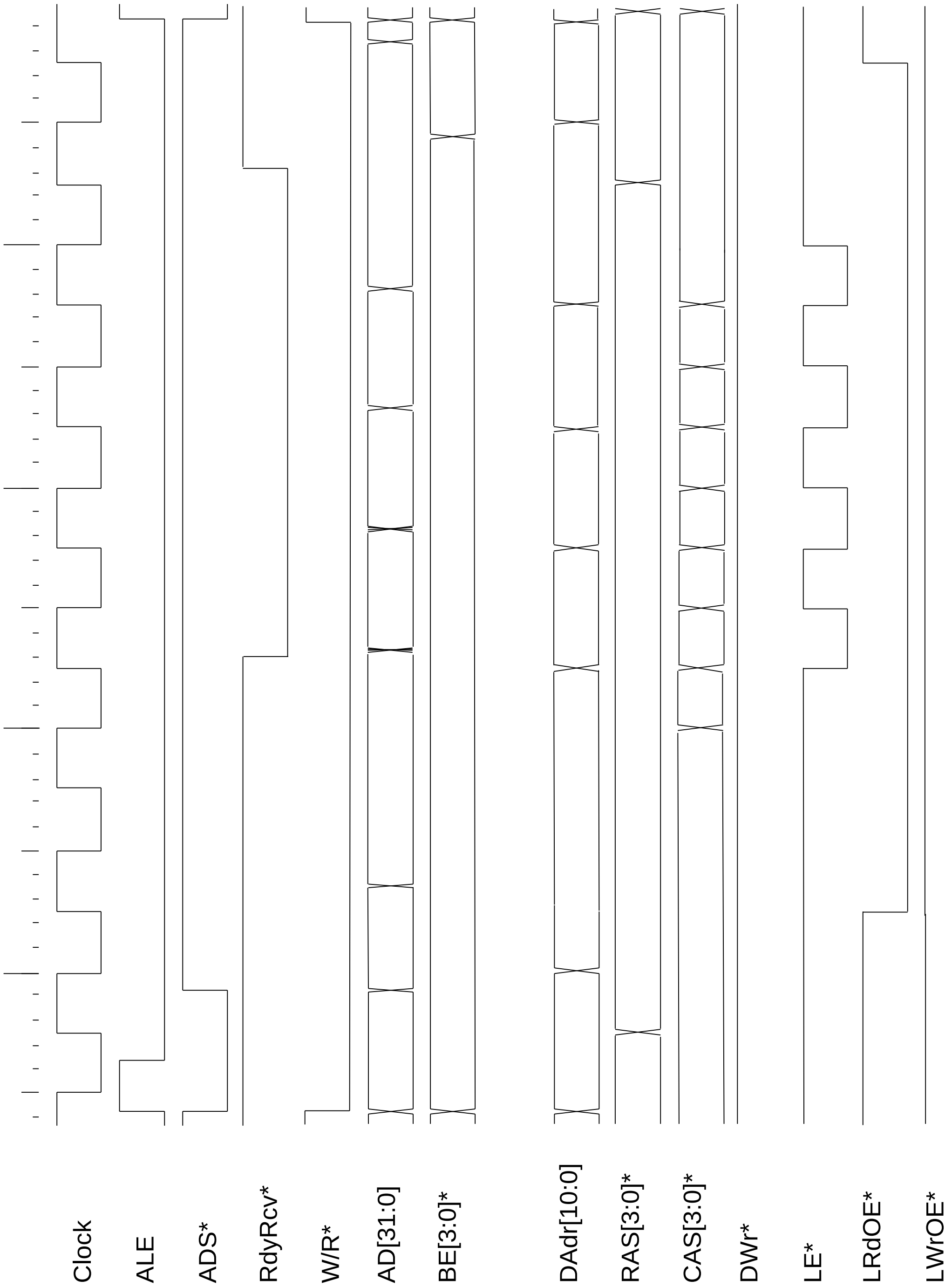
Standard DRAM 32-Bit Read 25MHz



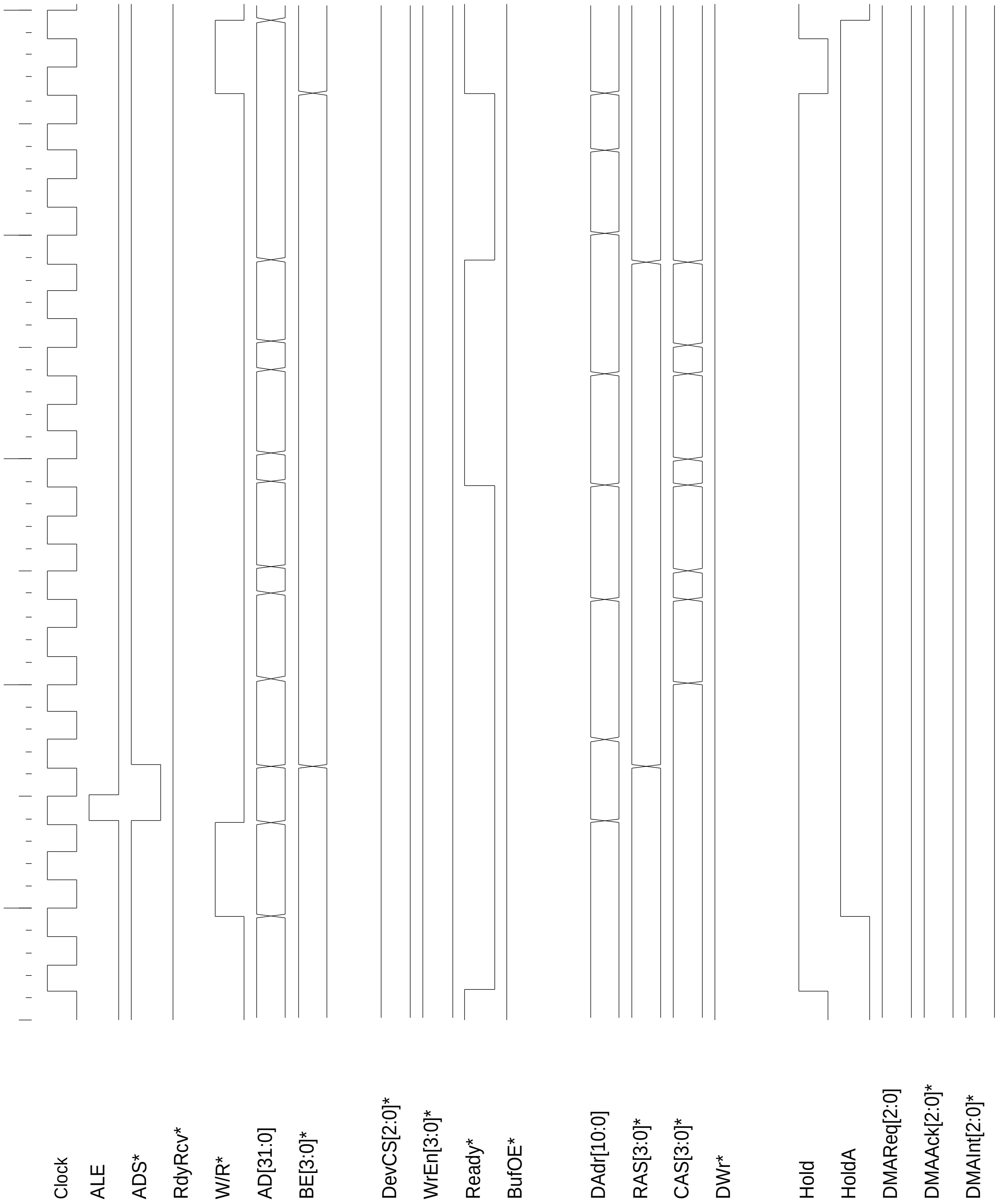
Standard DRAM One Byte Write 25MHz



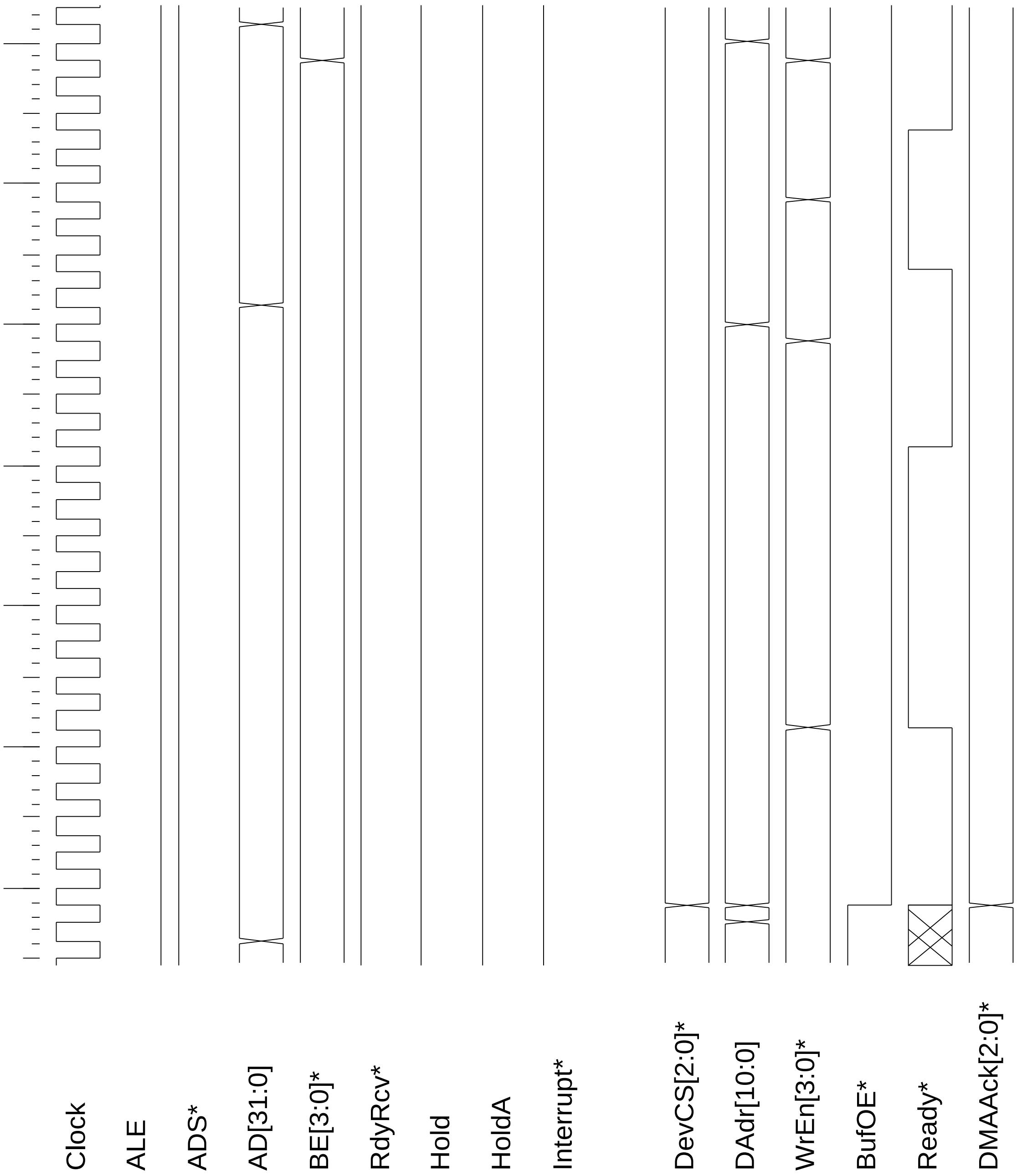
EDO with Latch 32-Bit Write 33MHz



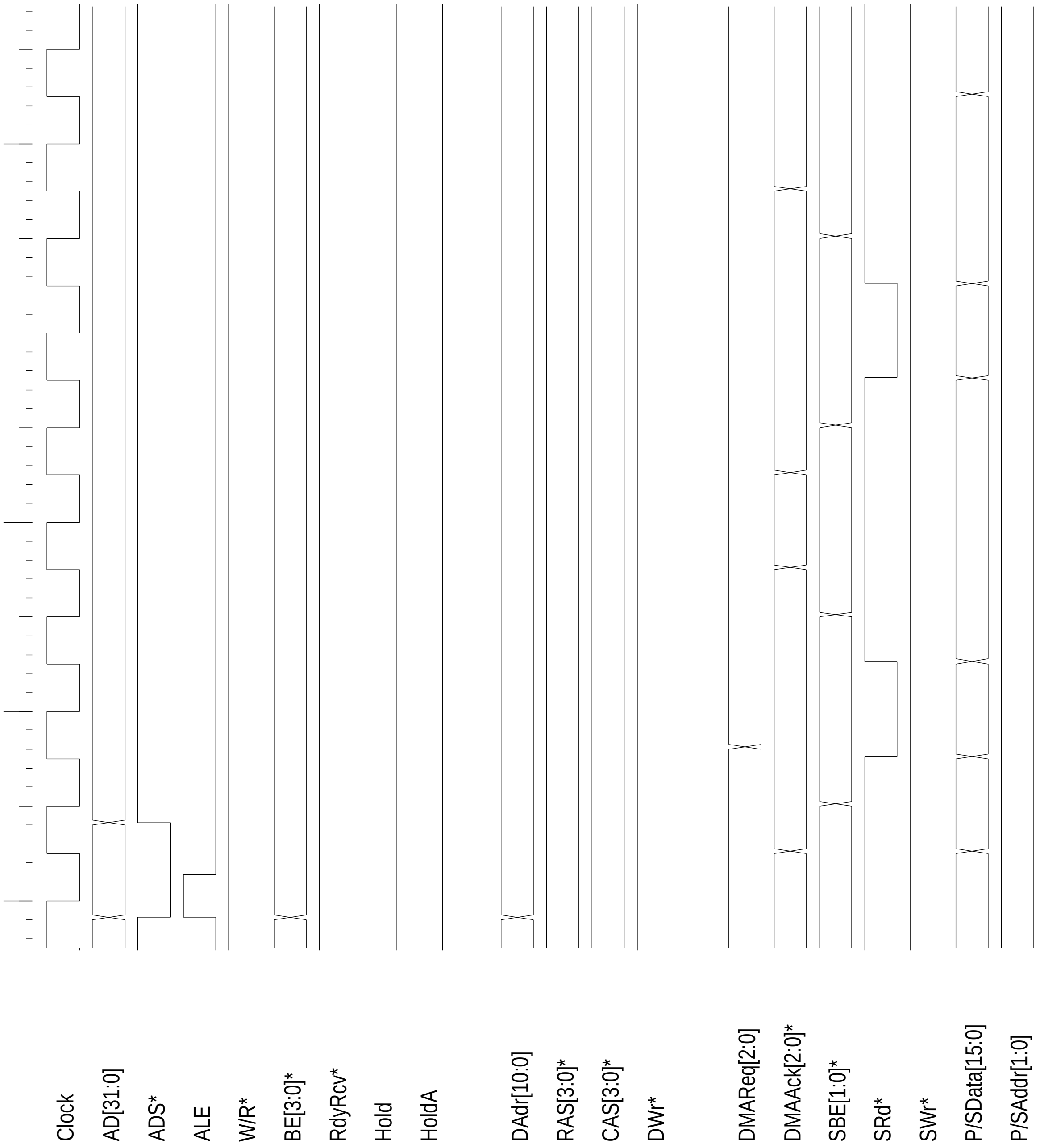
EDO with Latch 32-Bit Read 33MHz



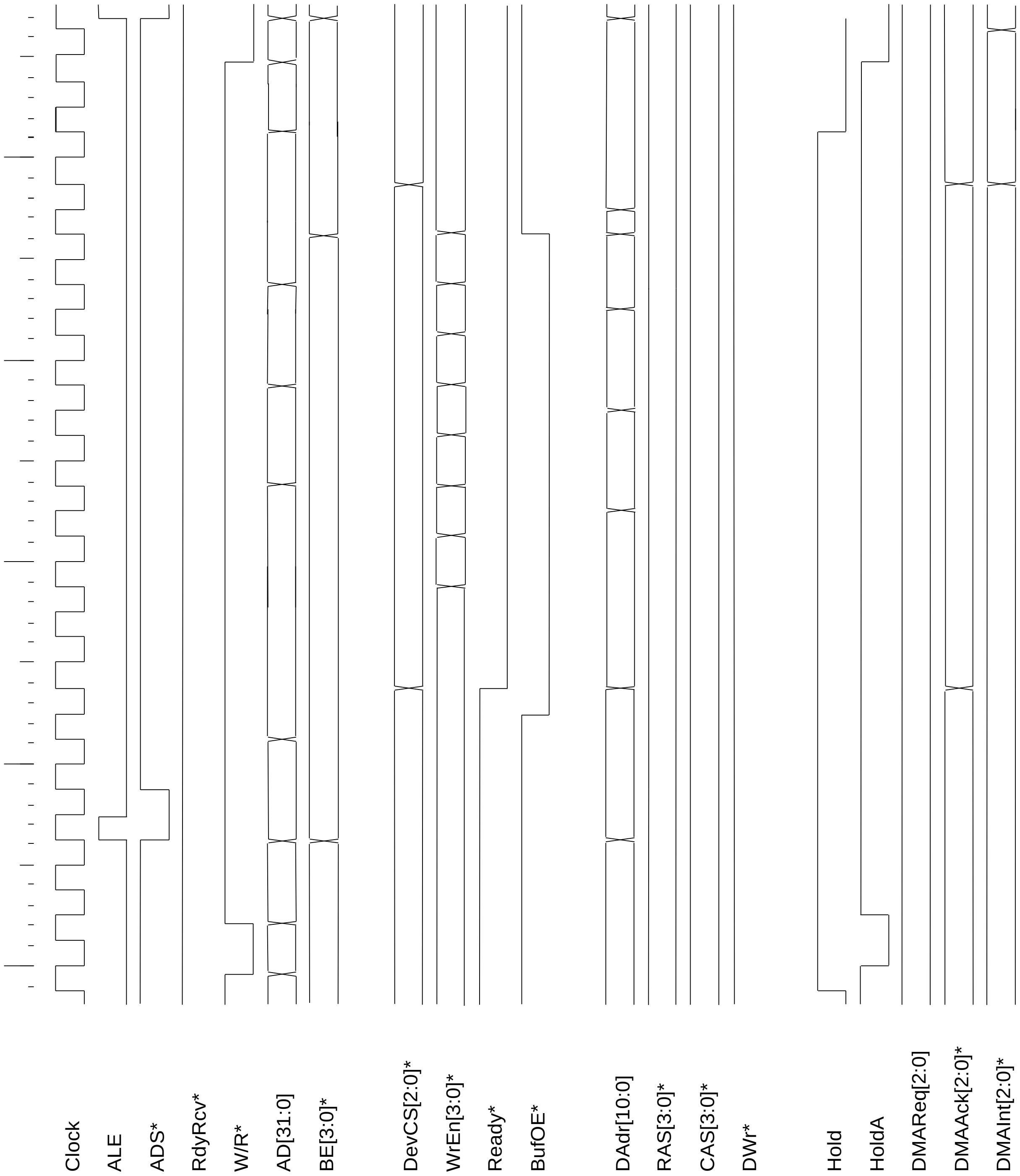
DMA Burst Read From DRAM



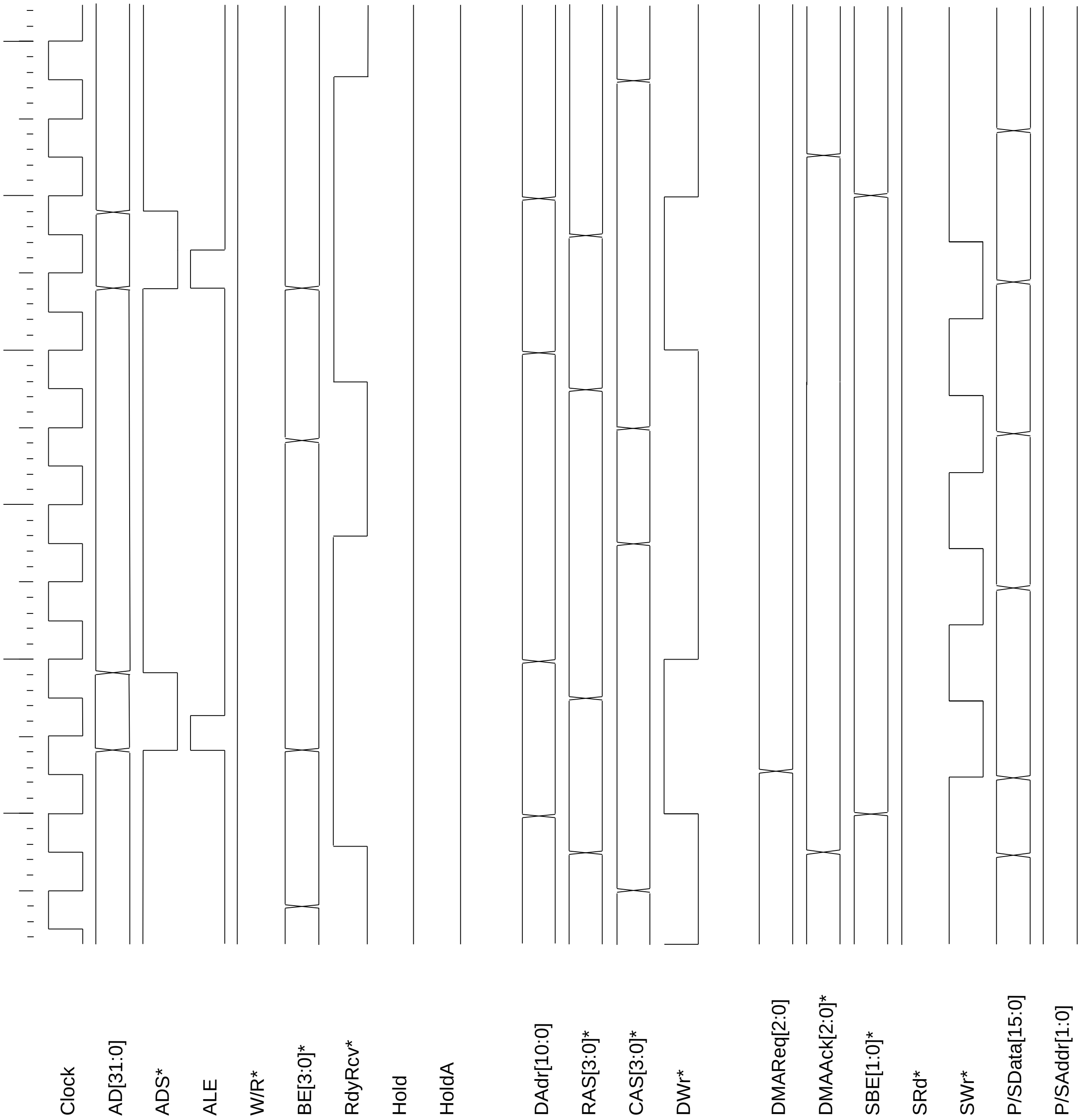
DMA Write with Ready* Extending a Cycle



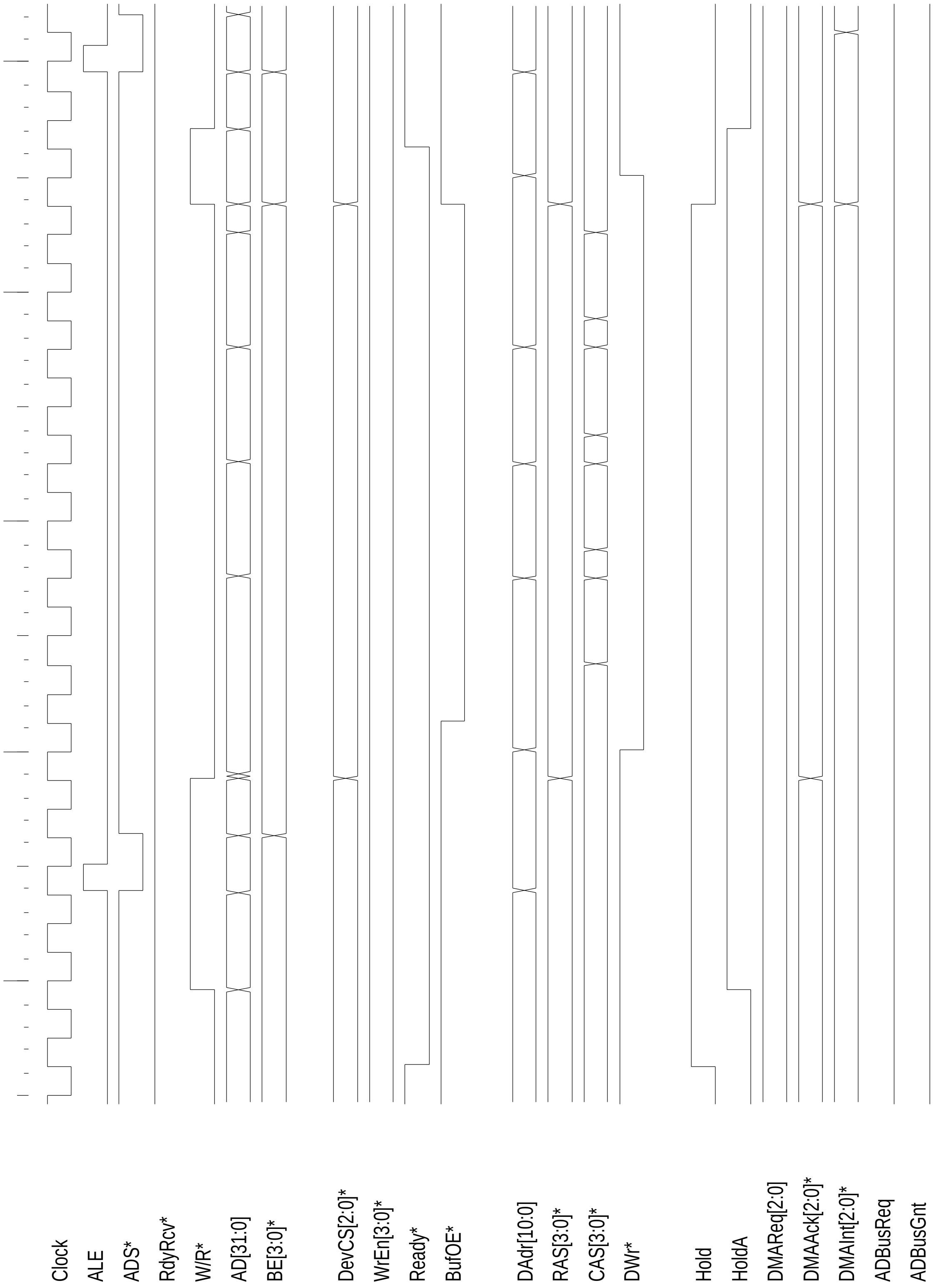
SIO DMA Double 16-Bit Read in Word Arbitration Mode



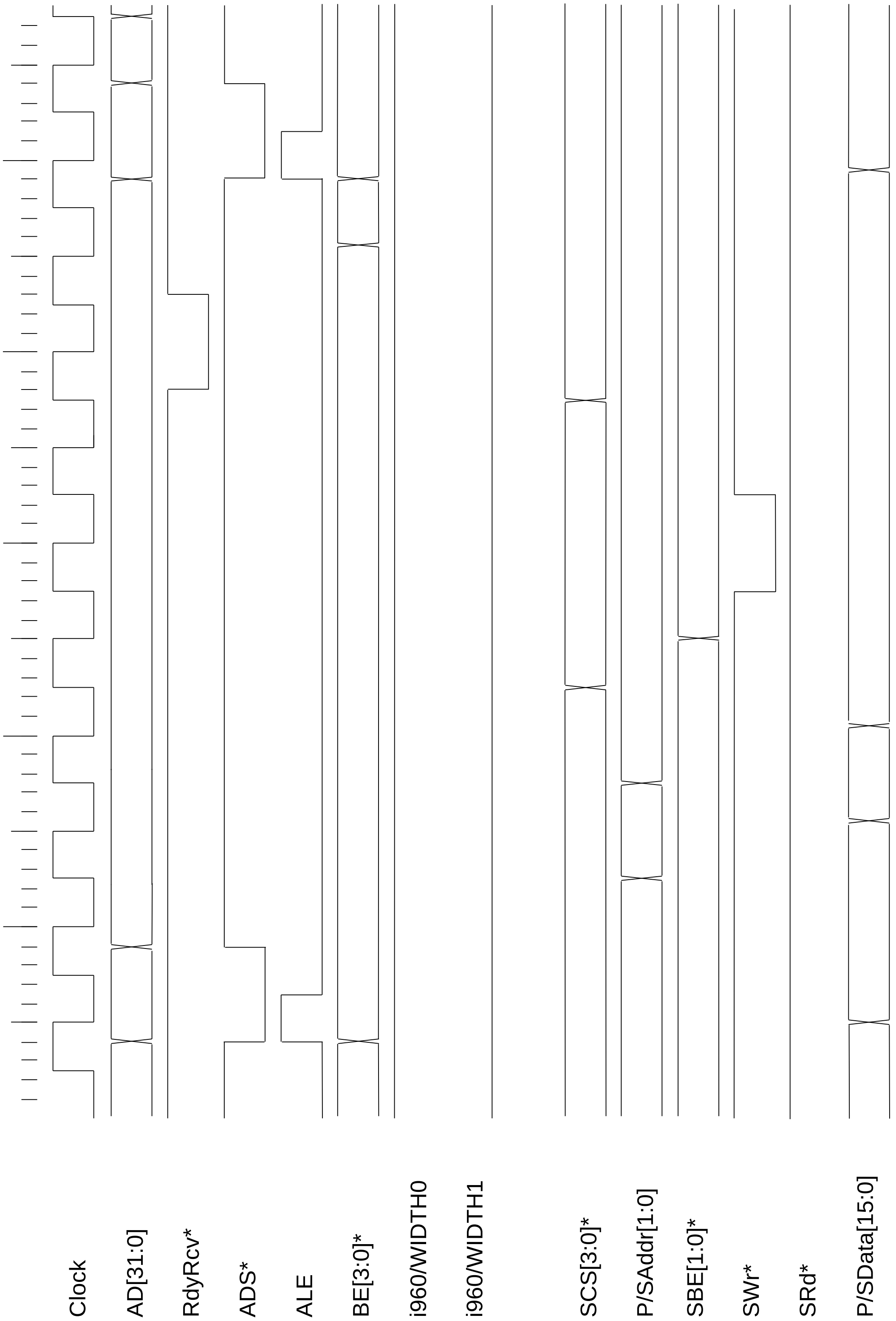
DMA Burst Write to an AD Bus Device



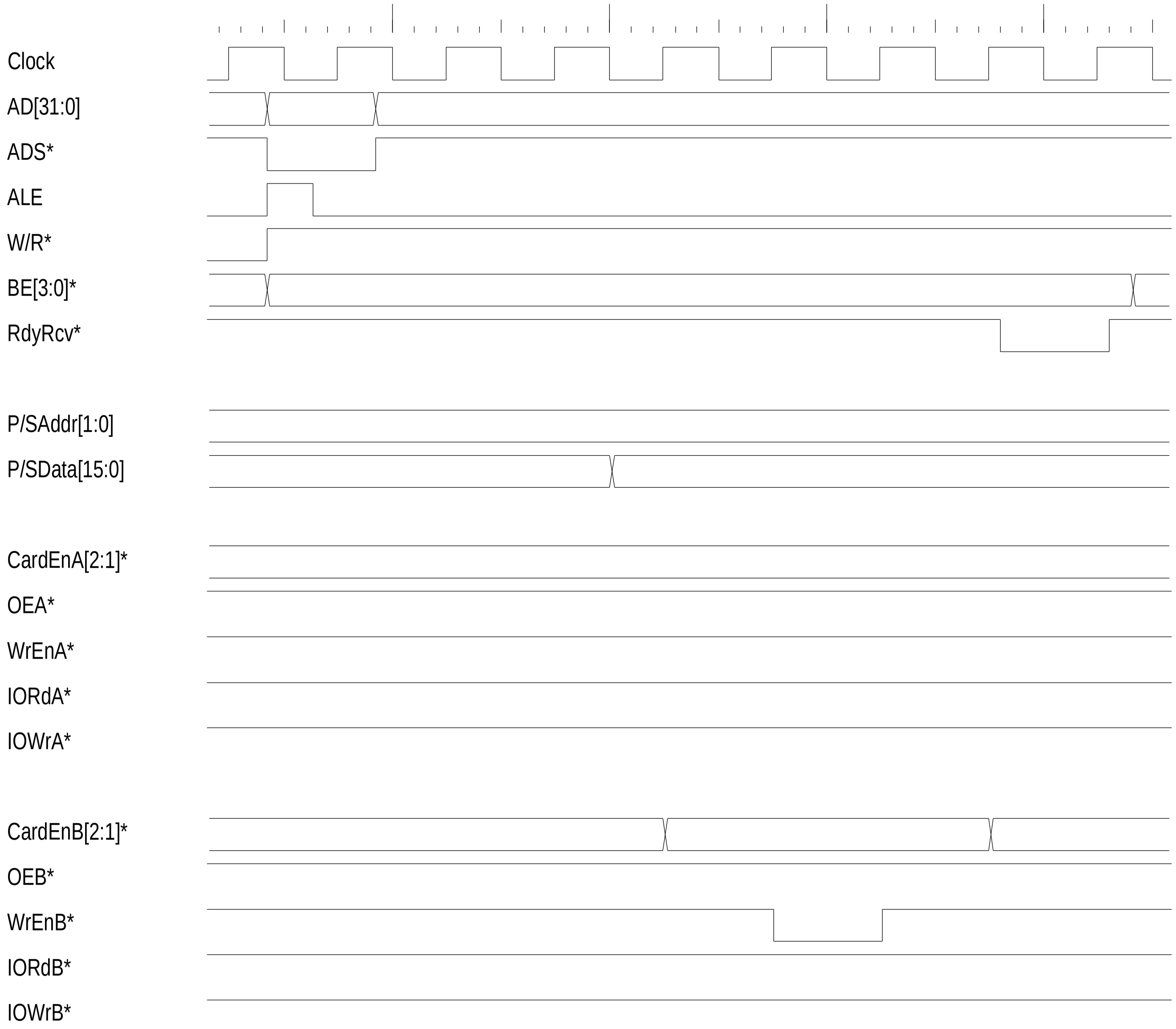
SIO DMA Four Byte Write in Burst Mode



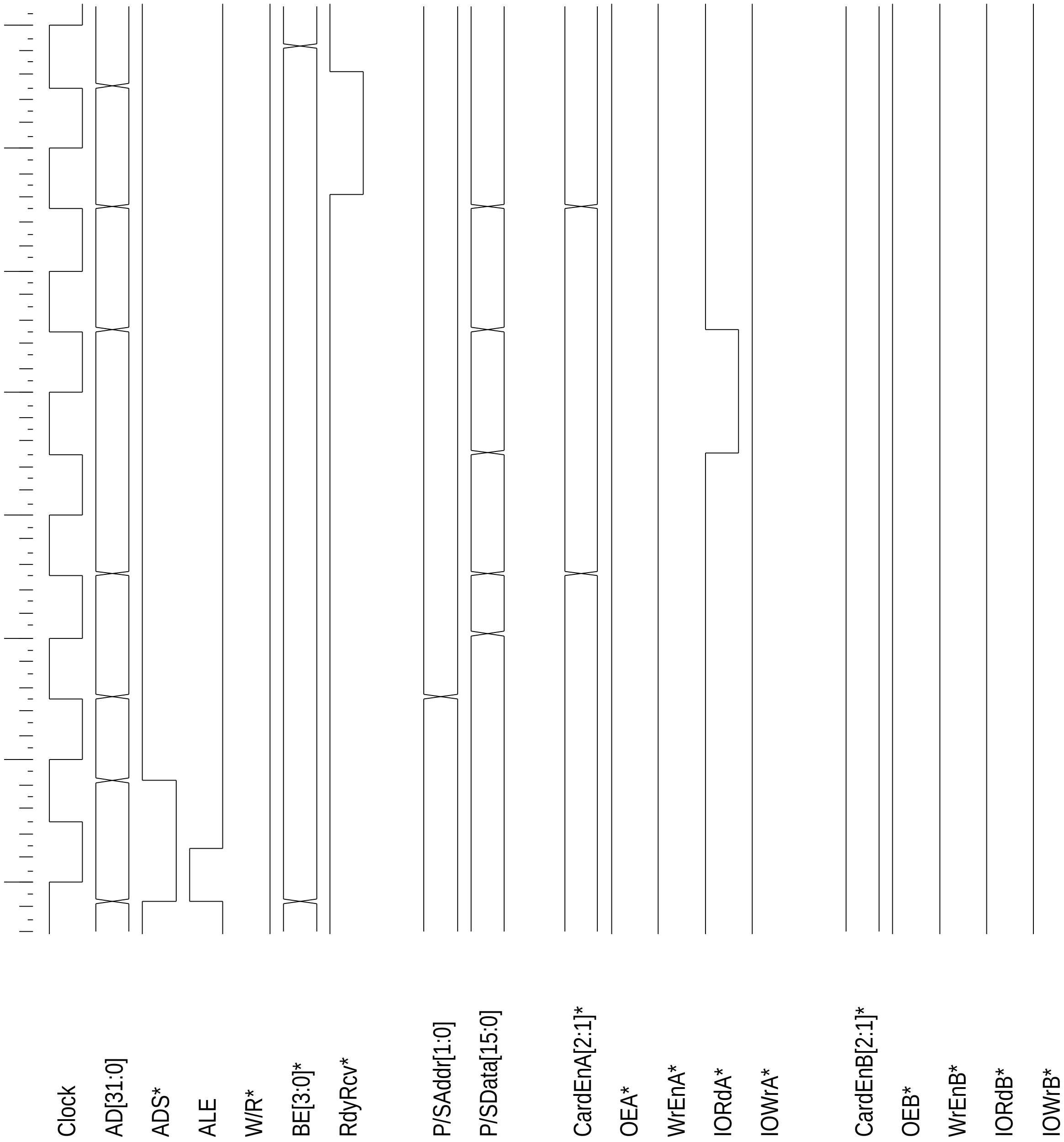
DMA Fly-By Mode Device to DRAM Burst Transfer



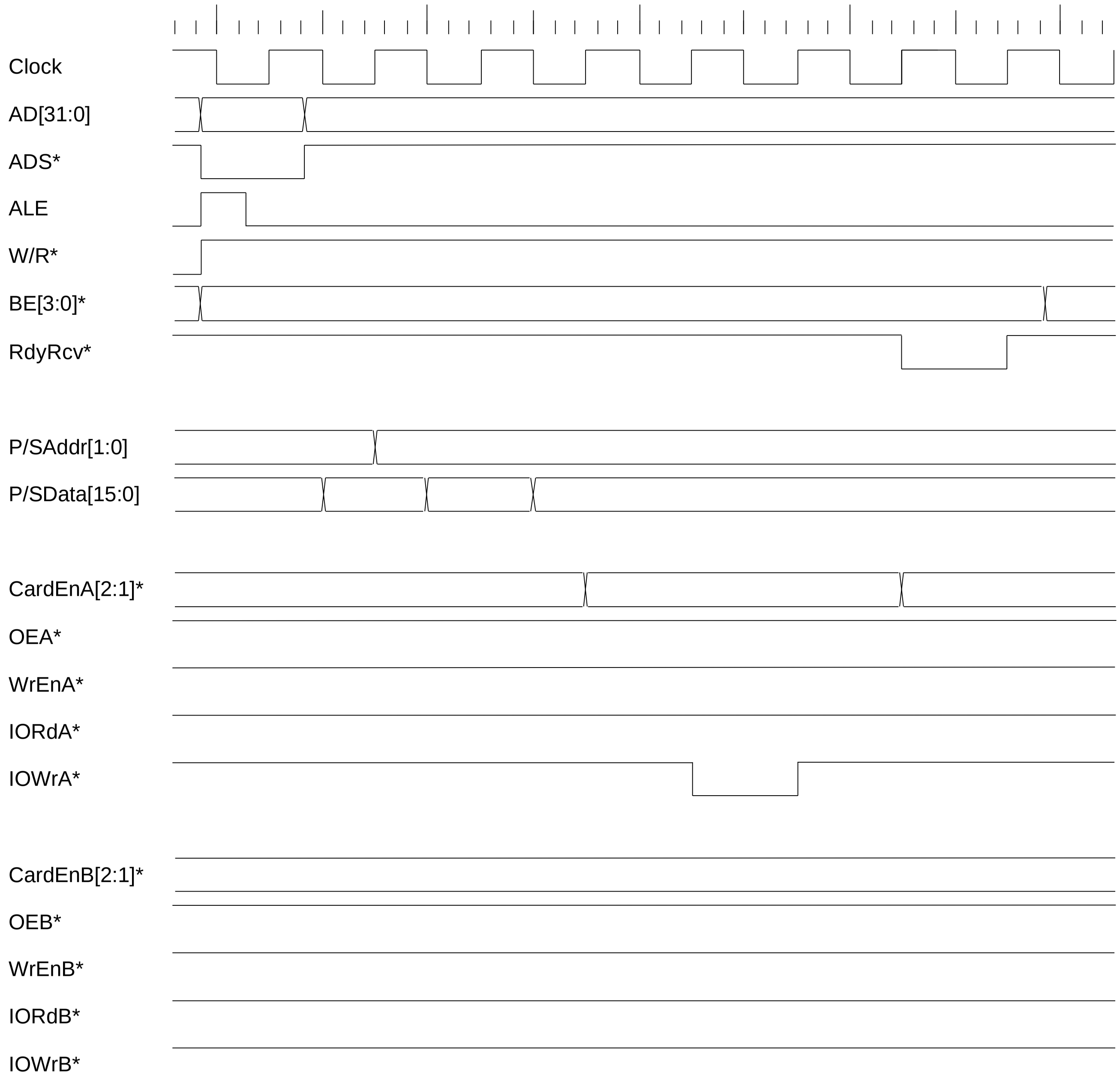
16-Bit Write to 16-Bit SIO Device



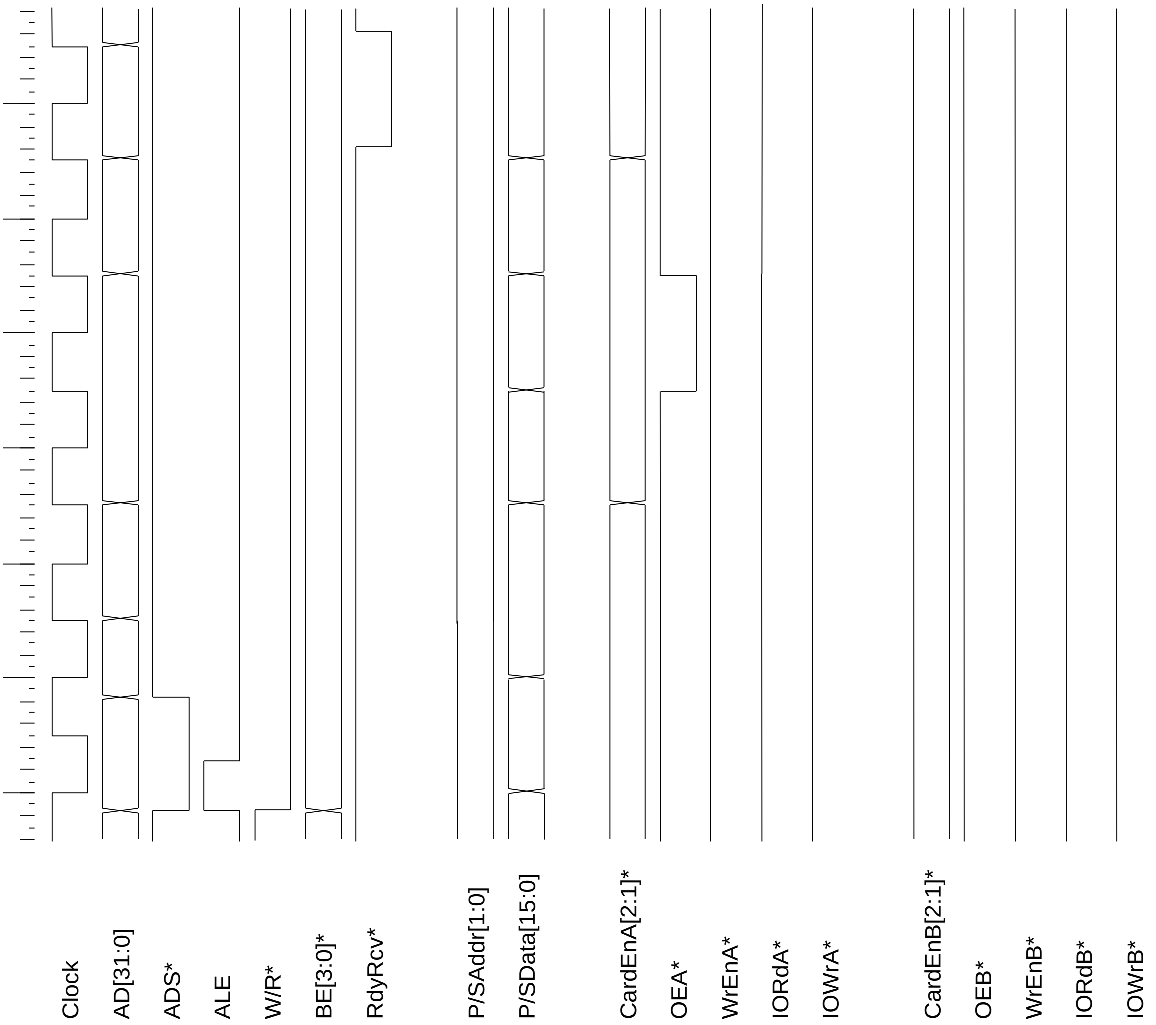
Byte Write to 8-Bit PCMCIA Memory



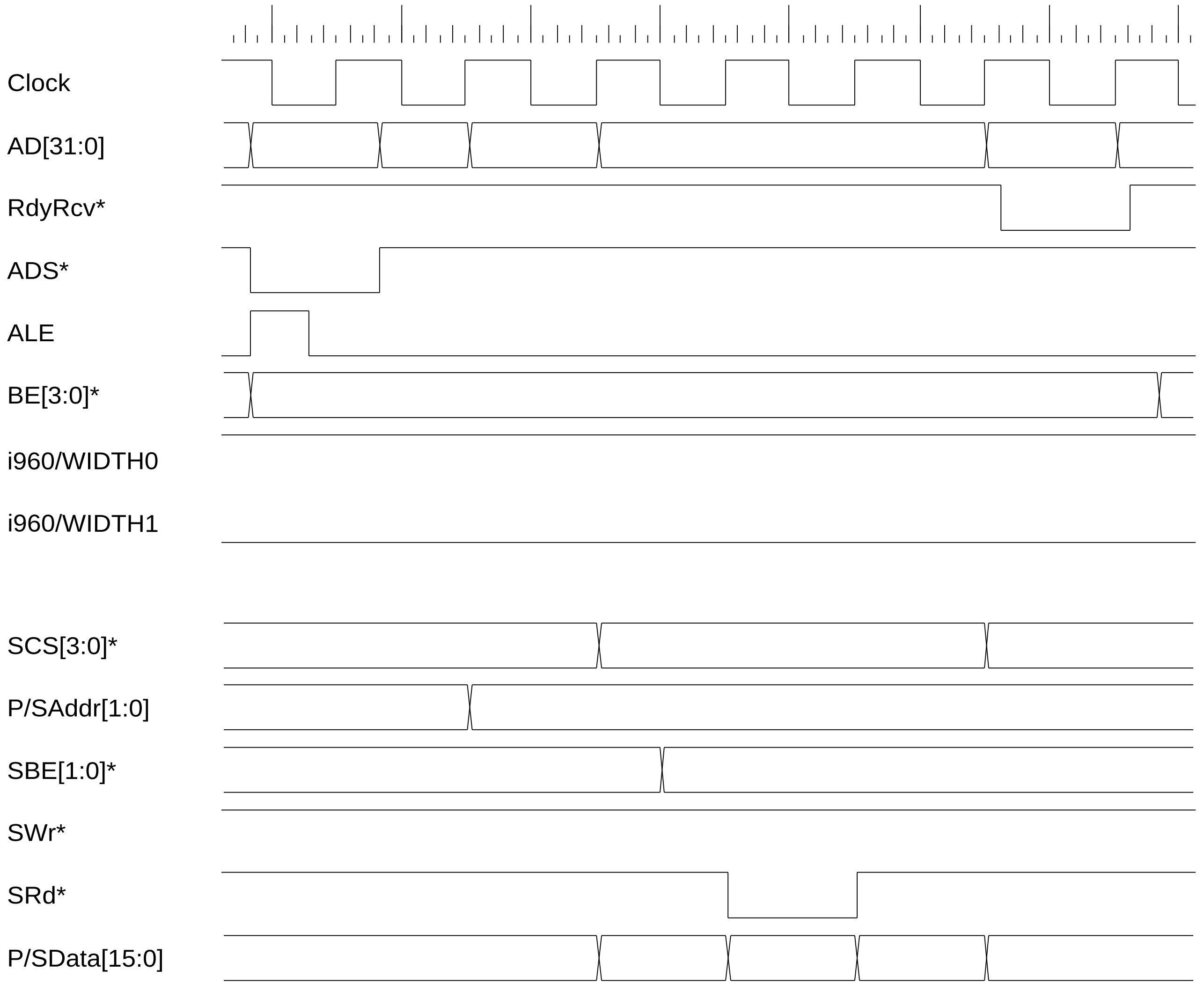
Read High Byte from 16-Bit PCMCIA I/O



16-Bit Write to 16-Bit PCMCIA I/O



16-Bit Read from PCMCIA Memory



MSB Read from 16-Bit SIO Device

9 APPLICATIONS: CONNECTING THE MEMORY BUSES

A GT-32090 system can be assembled in different configurations. The DRAM has to be 32-bits wide and the devices may be 8-, 16-, or 32-bits wide. The user may choose to work with external latches for the DRAM (depending on the operating frequency) and can choose to work with a bi-directional transceiver for the devices to reduce turn-off time and the need for drivers on the CPU's AD bus.

9.1 Choosing Standard DRAM or EDO DRAM.

The choice of which type of memory to use is a function of the performance goals for the system. EDO DRAM, being so common these days, offers an excellent alternative for best performance. Be aware of the following restrictions:
If working with EDO DRAM at 33 MHz you must use a latch for the data.
If working with EDO DRAM at less than 33 MHz you cannot use a latch for the data.
If working with standard DRAM at 33 MHz you cannot use a latch for the data.
If working with standard DRAM at less than 33 MHz you may use a latch for the data (giving you the performance of an EDO.)

This can be summarized as follows:

Memory Type	Frequency	Latch	Performance
EDO	33MHz	Required	xxDDDD
	25MHz	Not Allowed	xDDDD
	20MHz	Not Allowed	DDDD
	16MHz	Not Allowed	DDDD
DRAM	33MHz	Not Allowed	xxDxDxDxDx
	25MHz	Optional	xDDDD (latches), xxDxDxDxDx (no latches)
	20MHz	Optional	DDDD (latches), xxDxDxDxDx (no latches)
	16MHz	Optional	DDDD (latches), xxDxDxDxDx (no latches)

9.2 Working Without Data Latches/Transceivers

9.2.1 DRAM Connection

Connection	Memory Width	Connect...	To...
DRAM Address	32-bit	DAdr[10:0]	DRAM address pins
DRAM Data[1]	32-bit	AD[31:0]	DRAM data pins
DRAM Control	32-bit	RAS[3:0]*	DRAM RAS pins
		CAS[3:0]*	DRAM CAS pins
		DWr*	DRAM write pins

Notes:
1. AD[31:24] is always the MSB and AD[7:0] is always the LSB, regardless of CPU endianess.

9.2.2 Device Connection

Connection	Memory Width	Connect...	To...
Device Address	32-bit	{ LAdd[24:4][1], DAdr[1:0] }	Device address pins
	16-bit	{ LAdd[24:3][1] , A2[2] , BE[1]* }	Device address pins
	8-bit	{ LAdd[24:2][1] , BE[1:0]* }	Device address pins
Device Data	32-bit	AD[31:0][3]	Device data pins
	16-bit	AD[15:0][4]	Device data pins
	8-bit	AD[7:0]	Device data pins
Device Control	32-bit	WrEn[3:0]*	Device write pins
	16-bit	WrEn[3]*, WrEn[0]*[5]	Device write pins
	8-bit	WrEn[0]*	Device write pins

Notes:
1. LAdd[25:2] is the output of the 373 address latches sampled by the ALE signal of the CPU'S AD bus.
2. A2 is the A2 bit of the CPU.
3. AD[31:24] is always the MSB and AD[7:0] is always the LSB, regardless of endianess.
4. AD[15:8] is always the MSB and AD[7:0] is always the LSB, regardless of endianess.
5. WrEn[3]* is for AD[15:8], WrEn[0]* is for AD[7:0].

9.3 Working With Data Latches/Transceivers

9.3.1 DRAM Connection

Connection	Memory Width	Connect...	To...
DRAM Address	32-bit	DAdr[10:0]	DRAM address pins
DRAM Data[1,2]	32-bit	AD[31:0]	Latch I/O's B side
		Latch I/O's A side	DRAM data pins
DRAM Control	32-bit	RAS[3:0]*	DRAM RAS pins
		CAS[3:0]*	DRAM CAS pins
		DWr*	DRAM Write pins
		'0'	CEAB* of 543 data latches
		'0'	CEBA* of 543 data latches
		LWrOE*	OEBA* of 543 data latches
		LRdOE*	OEAB* of 543 data latches
		Clock	LEBA* of 543 data latches
		LE*	LEAB* of 543 data latches

Notes:
1. AD[31:24] is always the MSB and AD[7:0] is always the LSB, regardless of CPU endianess.
2. Must have a 32 bi-directional latch (such as 2 IDTFCT16543).

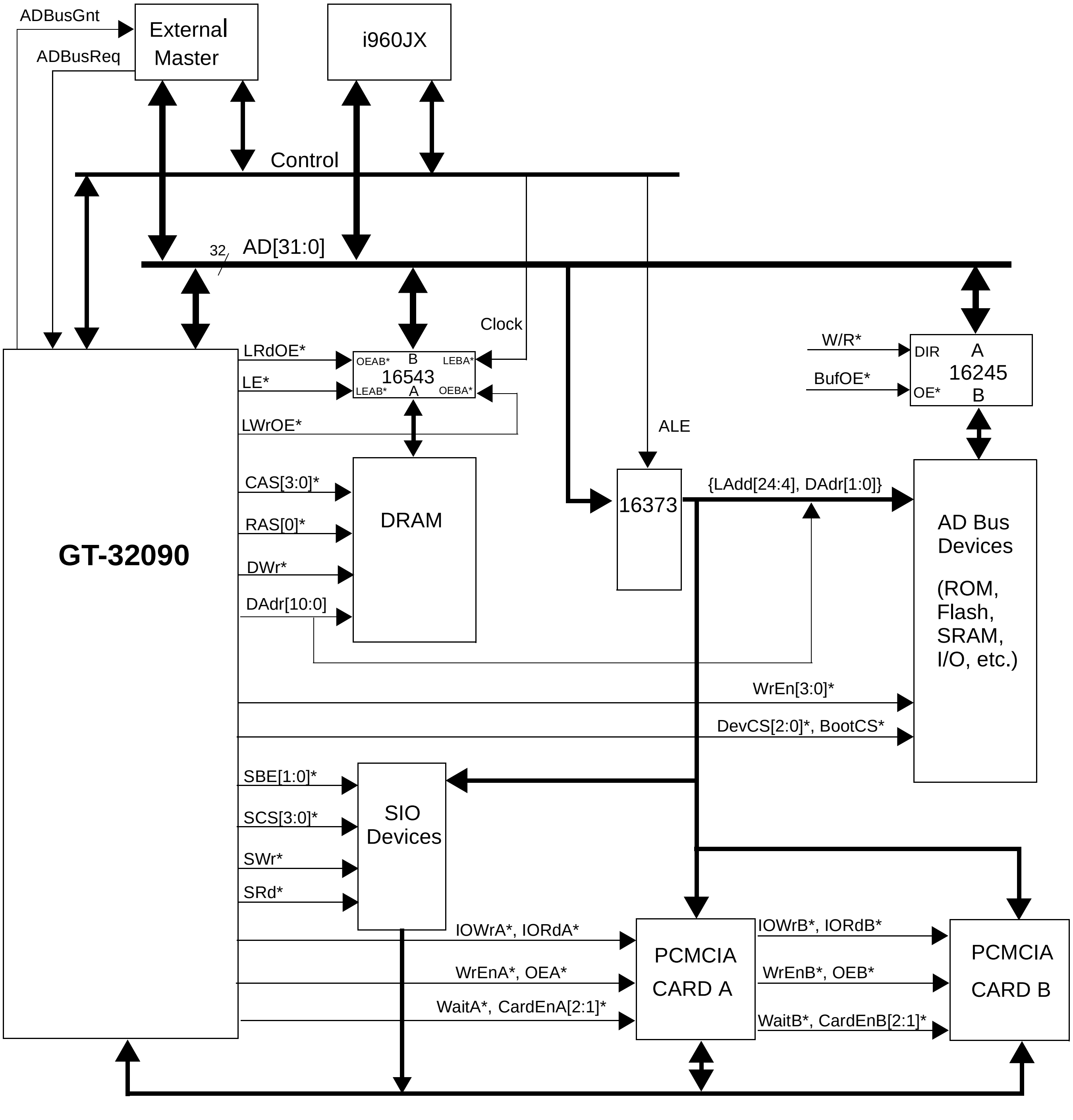
9.3.2 Device Connection

Connection	Memory Width	Connect...	To...
Device Address	32-bit	{ LAdd[24:4][1] , DAdr[1:0] }	Device address pins
	16-bit	{ LAdd[24:3][1] , A2[2] , BE[1]* }	Device address pins
	8-bit	{ LAdd[24:2][1] , BE[1:0]* }	Device address pins
Device Data	32-bit	AD[31:0][3]	Transceiver I/O's A side
		Transceiver I/O's B side	Device data pins
	16-bit	AD[15:0][4]	Transceiver I/O's A side
		Transceiver I/O's B side	Device data pins
	8-bit	AD[7:0]	Transceiver I/O's A side
		Transceiver I/O's B side	Device data pins
Device Control	32-bit	WrEn[3:0]*	Device write pins
		BufOE*	OE* of 16245 transceiver
		W/R*	DIR of 16245 transceiver
	16-bit	WrEn[3]* , WrEn[0]*[5]	Device write pins
		BufOE*	OE* of 16245 transceiver
		W/R*	DIR of 16245 transceiver
	8-bit	WrEn[0]*	Device write pins
		BufOE*	OE of 16245 transceiver
		W/R*	DIR of 16245 transceiver

Notes:
1. LAdd[25:2] is the output of the 373 address latches sampled by the ALE signal of the CPU's AD bus.
2. A2 is the A2 bit of the CPU.
3. AD[31:24] is always the MSB and AD[7:0] is always the LSB, regardless of endianess.
4. AD[15:8] is always the MSB and AD[7:0] is always the LSB, regardless of endianess.
5. WrEn[3]* is for AD[15:8], WrEn[0]* is for AD[7:0].

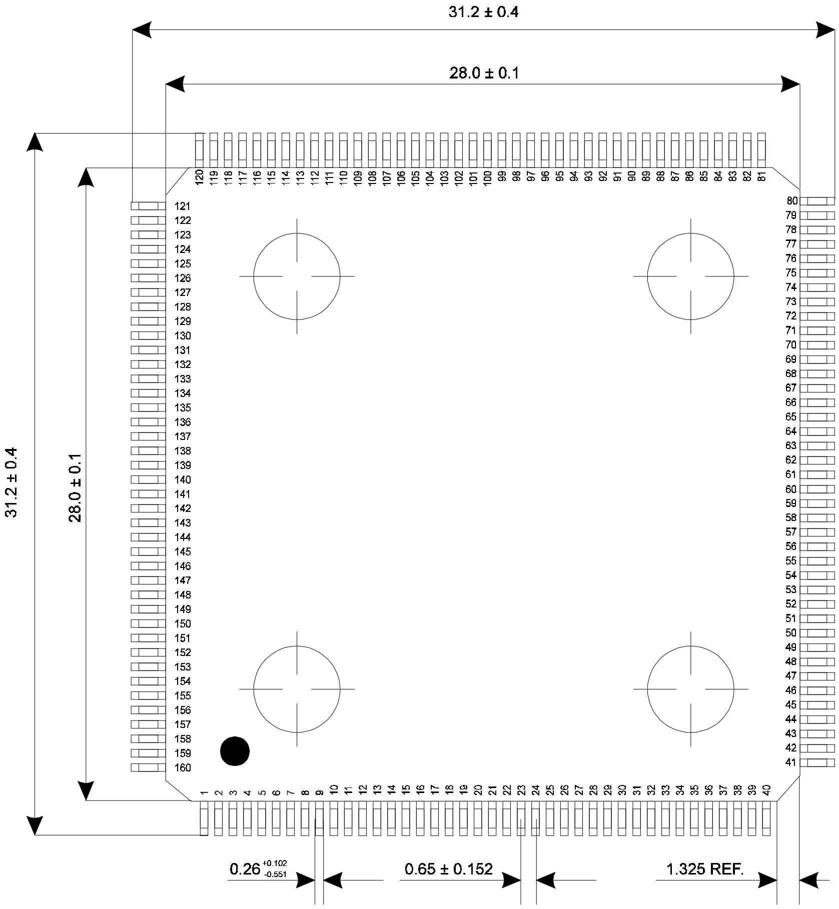
10 SYSTEM CONFIGURATIONS

The basic GT-32090 system shown below includes 4 banks of 32-bit wide DRAM, 32-bit devices, two PCMCIA slots, and 8- or 16-bit devices on the SIO bus. In this system, data can be moved via DMA between AD bus devices and DRAM, between SIO devices and DRAM, and between SIO devices and AD bus devices.



10 PACKAGE

10.1 160-Pin Quad Flat Package (QFP, EIAJ)



10.2 160 - Pin Quad Flat Package Expanded View (QFP, EIAJ)

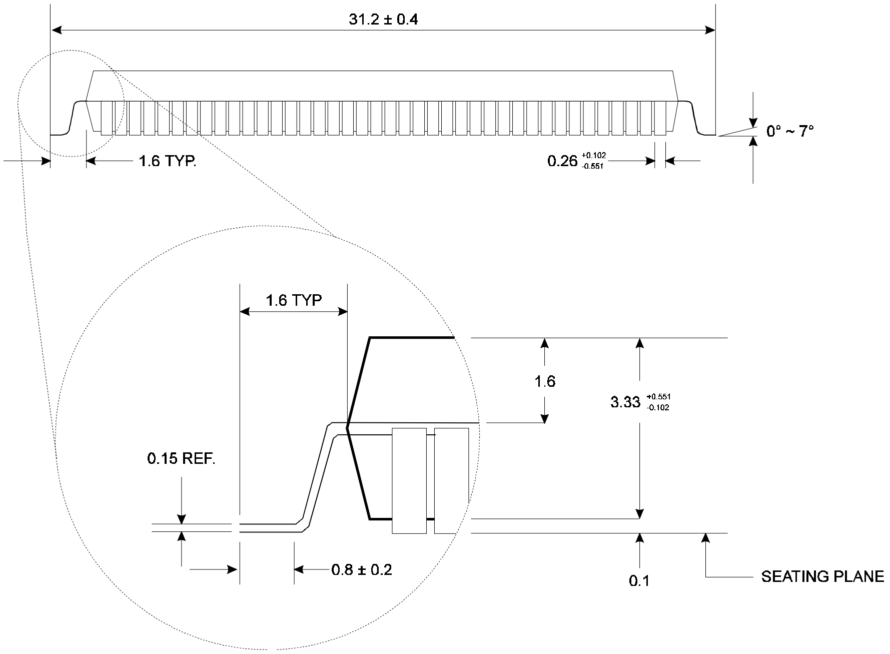


Fig. 10.2 : Expanded View