BROADCOM®

PRODUCT APPLICATION NOTE

BCM1250 Generic Bus Interface to ATA/ATAPI PIO Mode 3 (IDE) Hard Disk

10/08/01

16215 Alton Parkway • P.O. Box 57013 • Irvine, CA 92619-7013 • Phone: 949-450-8700 • Fax: 949-450-8710

REVISION HISTORY

Revision #	*Date*	*Change Description*
12500-AN200-R	04/05/01	Initial Release.
1250-AN201-R	10/08/01	Changed document number.

Broadcom Corporation
16215 Alton Parkway
P.O. Box 57013
Irvine, CA 92619-7013

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Section 1: Overview

This application note describes the interface between the BCM1250 Generic Bus and an ATA/ATAPI (IDE) hard disk drive, hereafter simply named IDE.

Please reference the following documents for additional information:

- BCM1250 User's manual.
- BCM1250 Datasheet.
- ATA/ATAPI-4 specification.

The BCM1250 generic bus offers a tremendous amount of flexibility that a system designer can take advantage of when interfacing to various I/O devices, such as, FLASH/ROM, PCMCIA, hard disk, etc.

By configuring the generic bus registers to meet the I/O device's timing specifications a system designer eliminates the need for any external logic.

It is this level of programmability, and rich set of features that makes the BCM1250 cost effective when implementing an I/O interface.

In this application note we will demonstrate the interface between the BCM1250 generic bus and an IDE disk drive. The considerations used in this example can be applied to implementing and configuring other types of interface on the generic bus.

The document begins with a block diagram, followed by a description of the interfaces and timing tables. At the end of the document there is an appendix that has the actual OrCAD schematic, and gives some IDE background.

Section 2: BCM1250 to IDE Electrical Interface

BLOCK DIAGRAM

This section will begin by providing a high-level block diagram and explaining the interfaces between the BCM1250 and the IDE hard disk drive interface.

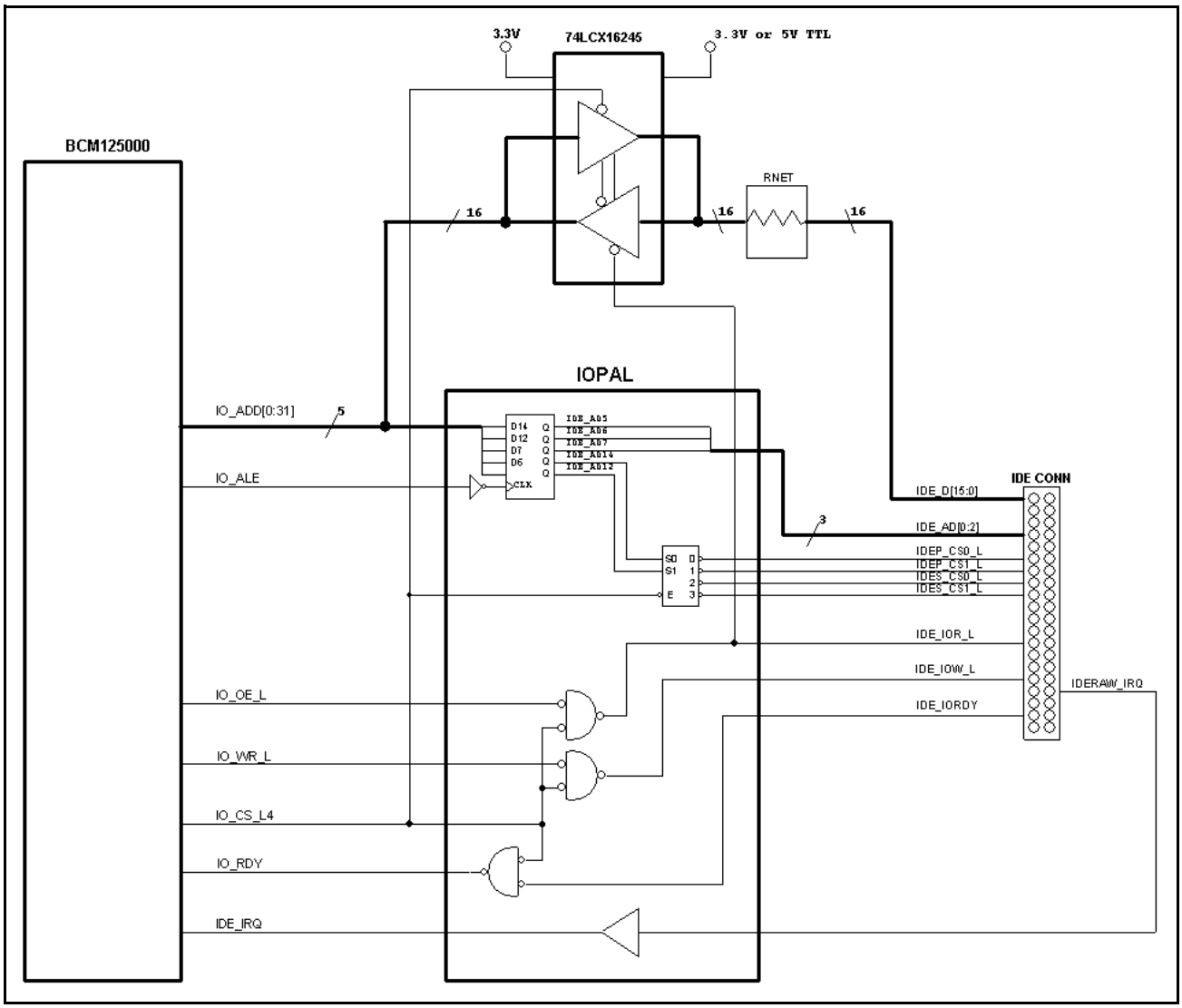

Figure 1: BCM1250 Block Diagram

As shown in Figure 1, the BCM1250 interfaces to the IDE hard disk with few external components. Only a level translator buffer, a PAL for address decode, termination resistors, and the standard 40-pin IDE connector are needed.

The IDE interface requires the address and chip selects to be valid for the entire IDE cycle, and a 16-bit data path for data transfers. Therefore, the generic bus must be configured for multiplexed mode, and 16-bit wide data bus. When the

BCM1250 outputs an address in multiplexed mode the address will only stay valid until the chip select is asserted, then the bus will turn around for a data transfer. The address signals out of the generic bus (IO_AD[14,12,7,6,5]) must be latched with IO_ALE, and used together with IO_CS_L4 to generate the two IDE chip selects, thereby holding them valid through the entire cycle.

In this application example a PAL (Xilinx XC9572XL-10) is used to provide both the address latch and chip select decode. The IOPAL was chosen because we needed to provide additional functions on our reference design. It provides voltage translation for the IDE control signals, and a PCMCIA interface to the generic bus. A simple latch (xx373) and address decoder (xx139) could also be used.

Data transfers between the generic bus and the IDE interface is via the muxed address/data bus of the BCM1250, and the IDE data bus.

The data bus connection between the IDE and generic bus reflect the little endian nature of the IDE bus and the big endian alignment on the generic bus. The lower IDE data byte, IDE_D[7:0], connects to IO_AD[31:24] and the upper data byte, IDE_D[15:8], connects to IO_AD[23:16]. This is done using a 74lcx16245 voltage level translating buffer (see Figure 4, ”BCM1250 to IDE Schematic,” on page 14). The BCM1250 generic bus is NOT 5V tolerant, so a voltage translator is required. A FET-switch (e.g. a QuickSwitch) could be used in place of the 74lcx16245. In this use it is better to use an active driver since the interface between the 40-pin connector and the IDE drive is up to 18 inches of flat ribbon cable that looks like a capacitor at lower frequencies, and a transmission line at higher frequencies.

Furthermore, series termination will normally be needed to prevent reflections and ringing on the interface. The system designer needs to provide a balance between termination resistors, and source driver strength in order to limit the ringing and reflections to acceptable levels. The correct choice of using a 74lcx16245, a QuickSwitch or some other buffering logic depends on the implementation details, for example the decision may change if the ribbon cable link were not used.

IDE Addressing Description

The IDE interface is defined by the ATA/ATAPI specifications, and originated as the AT bus interface ported to the hard disk via a 40-pin header.

The IDE hard disk drives use two chip selects (IDE_CS_L0, IDE_CS_L1), three address bits (IDE_AD[2:0]), and I/O read or I/O write (IDE_IOR_L, IDE_IOW_L) to access their I/O registers. All registers are 8-bits wide except for the data register, which is 16-bits wide. All 8-bit register transfers are on the lower data byte (IDE_D[7:0]) of the data bus regardless of whether it is an odd or even address.

In many implementations two interfaces are supported with each interface supporting two drives. For standard PC legacy I/O mappings, the primary interface is located at 1Fx/3Fx and the secondary interface is located at 17x/37x.

The simplest option considered was to map the interface at the normal primary and secondary offsets in one of the generic bus regions, software could then access the registers by adding the normal offsets to the base address of the region. However, the generic bus moves bytes on their natural byte lane and is always big endian. Therefore, for a 16-bit wide interface the BCM1250 would expect the bytes at odd addresses to be transferred on IDE_D[15:8], whereas the drive always uses IDE_D[7:0]. Thus this simple mapping cannot be used.

In order for both the drive and BCM1250 to transfer byte accesses on IDE_D[7:0] the drive registers must all be mapped to even addresses in the generic bus region (while being presented as odd addresses to the drive). Also, to keep the software porting effort low a simple mapping is preferred; an easy solution to both of these is to shift the address. The example implementation uses a 5 bit shift between the address used on the generic bus and the address presented to the drive (i.e. the IDE_AD[0] is connected to the generic bus IO_AD[5]).

The address only needs to be shifted by a single bit to satisfy the need for registers to be on even generic bus addresses. The choice to shift by five bits was made by considering the ways the drive will be accessed and the operation of the generic bus interface. Software will only ever be doing byte accesses to the control registers, so it makes no difference what shift is used. However, transfers to and from the 16 bit data register will be done in bursts and when the drive indicates it is ready for the data transfer stage of a command it is always able to move a full 512 byte block. The CPU could do the transfer using a series of 256 16 bit accesses, each of which would require a separate ZBbus transaction. If the CPU does a wider access the generic bus interface will automatically break it into a sequence of 16 bit accesses (four for a 64 bit access) at ascending generic bus addresses. If all of these addresses are aliases of the drive's data register then the CPU access will result in the larger number of bytes being transferred to or from the drive's data fifo. The number of generic bus transactions is not changed, but the number of ZBbus transactions and the number of in-flight transaction buffer entries used will be lower. The maximum size transfer that can be generated by software from the CPU to the uncacheable generic address space is 64 bits (using LD and SD instructions), but if the disk block is moved using the data mover as a DMA engine it will generate 32 byte accesses. The generic bus controller will break these up into sixteen 16 bit wide accesses, so there must be sixteen aliases of the drive's data register for the data mover access to work. To get sixteen aliases of the 16 bit register the shift of five bits is needed between the generic and drive address lines.

In addition to the address shift, the address decoder for the drive chip selects must be designed. Table 1 shows the standard PC IDE I/O address mapping and their respective chip selects.

Table 1: Standard PC IDE I/O Address Mapping

Interface Number	*PC CS0-Decode*	*PC CS1 - Decode*	*A9*	*A8*	*A7*	*A6*	*A5*	*A4*	*A3*	*A2*	*A1*	*A0*
Interface 1	01F0h-01F7h		0	1	1	1	1	1	0	X	X	X
		03F6h-03F7h	1	1	1	1	1	1	0	1	1	X
Interface 2	0170h-0177h		0	1	0	1	1	1	0	X	X	X
		0376h-03774	1	1	0	1	1	1	0	1	1	X

From the table it is apparent that the addresses differ in bit 9 for chip selects (CS0-, and CS1-), and bit 7 for the primary and secondary interfaces. With the shift of five bits these become generic address bits 14, and 12 respectively. In this example, the incomplete decode is acceptable because only the IDE interface(s) are on this generic bus region 4 chip select.

Putting the decode together with the offset to form the address for accessing the drive registers software will have to shift the standard register address left by five bits and add the region base address. For example, if the base address of the chip select 4 (IO_CS_L4) region is set to 00_1234_0000, a write to address 00_1234_3E20 (00_1234_0000 + (01F1 <<5)) will result in a write on the generic bus to chip select 4 address of 0x3E20.From the generic bus the PAL will decode address 0x3E20 and IDE_CS_L0 will be asserted and the address on IDE_AD[2:0] will be 1, accessing the register on the drive that would normally be found at I/O address 1F1.

IDE BUS TERMINATION

Termination resistors are necessary in order to attenuate any reflections on the data bus. The 40-pin cable is modeled as a transmission line with a typical characteristic impedance of 110 ohms when connected in a ground-signal-ground configuration. When there are fewer grounds the characteristic impedance is higher. For the data lines adjacent to ground the characteristic impedance is approximately 160 ohms. 110 ohms applies to data strobes.

So, for higher speed timing modes such as mode 3 you must terminate the lines properly by matching impedances. For example, from transmission line theory ringing occurs when the source impedance is lower than the cable impedance, and the termination impedance is higher than the cable impedance. Therefore, make sure that the driver output impedance and the series termination resistor together equal the cable impedance. This assumes that the 40-pin IDE connector is at one end of the cable, and not in the middle.

When adding a second IDE connector for a second pair of hard drives double series termination is suggested by the ATA/ATAPI specification. Please refer to ATA/ATAPI specification Annex C for additional details.

CONTROL SIGNALS

The IDE drive produces two status signals IDE_IORDY (indicating that the drive is ready to proceeded with an access) and IDE_IRQ (the drive interrupt line). Both are 5V logic signals and must therefore be buffered before being presented to the BCM1250 to avoid damage to the input pads. This is done in the IOPAL, as shown in Figure 1. It could equally well be done using any 3.3V buffer with 5V tolerant inputs. In the example implementation the IDE_IORDY signal is combined with the chip select for the drive, this is done because there are other peripherals that use the acknowledgement based access mode on the generic bus. If the IDE device was the only device using the IO_RDY signal to the BCM1250 there would be no need for the external logic because the signal is ignored for chip selects that use fixed timing.

The address and chip select signals to the drive are generated by the IOPAL as described in the previous section. The IDE_IOR_L and IDE_IOW_L read and write strobes are generated directly from the equivalent signals on the generic bus (IO_OE_L and IO_WR_L), but they are masked out if the IDE region chip select is inactive. This is done to prevent the drive interface AC timing requirements being violated if other peripherals on the generic bus have the strobe signals active for shorter than the minimum time permitted in the ATA/ATAPI specification. Arguably this is not needed because the chip select signals to the drive will be deasserted during the short strobe, but the standard only lists the minimum strobe width and does not indicate that this need only be met when an access is in progress.

CONFIGURING THE GENERIC BUS

The BCM1250 allocates 767 MBytes of memory for generic bus I/O, and 4KBytes for configuration and status. These addresses are physically mapped at 00_1006_0000 to 00_3FFF_FFFF.

The 4 KBytes of configuration and status are mapped in this manner:

- 2 KBytes, 00_1006_1000 to 00_1006_17FF is allocated for configuration.
- 2 KBytes, 00_1006_1800 to 00_1006_1FFF is allocated for status.

There are eight regions defined within this address space. These eight regions are selected by their respective chip selects, IO_CS_L[7:0]. Two of the eight regions are fixed, IO_CS_L0 is used for Flash/ROM boot device, and IO_CS_L6 for PCMCIA when it is used (otherwise IO_CS_L6 is free).

Each region is configured as a 32-bit multiplexed address/data bus, or as an 8-bit data bus, and 24-bit address bus.

The base address and size of memory required for the interface is configured in the following registers of the generic bus:

- **io_ext_start_addr**
- **io_ext_mult_size**

The base address, **io_ext_start_addr**, will have bits [39:30], and [15:0] set to zero, only bits [29:16] are set in the register by software to the address of at which the region should start. An access made at this address will become an access with address zero on the generic bus with the region's chip select asserted. The size of the region is set in the register **io_ext_mult_size**, in increments of 64 KBytes. For the IDE application the smallest size of 64 KBytes is sufficient.

Two timing modes are supported by the BCM1250; fixed and acknowledgement. For fixed mode timing the access is entirely controlled by the parameters set in the configuration registers of the generic bus. In acknowledgement-based mode a ready/busy signal from the device controls the length of the access. The IDE interface requires use of an acknowledgement based protocol to achieve its highest transfer speeds. There is also a timeout which sets the longest the BCM1250 will wait for a peripheral to signal it is ready. The IDE timing (see Table 5) parameter TB sets the maximum the drive can extend the cycle as 1250 ns, so the generic bus timeout is configured for 2 us. In each mode the timing parameters of an access are configured in the **io_ext_time0_cfg0** and **io_ext_time1_cfg1** registers as described in the next section.

The timing mode parameters are set in the **io_ext_cfg** register. This register is also used to select the 16 bit wide data path and therefore multiplexed AD mode.

The following tables provide the settings necessary to set up the generic bus for Region 4, Address/Data Multiplex mode, and 16-bit data width mode.

Table 2: io_ext_cfg

Generic Bus Region Configuration Register ➠ io_ext_cfg_4 = 00_1006_1020			
Bits	***Name***	***Value***	***Description***
0	rdy_active	1'b1	IO_RDY is set for active high assertion
1	io_ena_rdy	1'b1	Acknowledgement-based access. (Also see bit 0)
3:2	io_width_sel	2'h1	2'b01:2 byte Data width is 16 bits.
4	io_parity_ena	1'b0	Disable parity check.
5	Reserved	1'b0	Reserved, must be zero.
6	io_parity_type	1'b0	When low even parity is used, when high odd parity is used.
7	io_nonmux	1'b0	This bit is set to 0 for multiplexed mode.
15:8	io_timeout	8'h02	Time out value. This is set to "2" to give a 2µs timeout because the maximum time the drive can extend the cycle is given as 1250ns (in the IDE specification).

Table 3: io_ext_mult_size

Generic Bus Region Configuration Register ➠ io_ext_mult_size_4 = 00_1006_1120			
Bits	***Name***	***Value***	***Description***
11:0	io_mult_size	12'h000	Memory region size set for 64 KB.
15:12	Reserved	4'h0	Reserved.

Table 4: io_ext_start_addr

Generic Bus Region Configuration Register ➠ io_ext_start_addr_4 = 00_1006_1220			
Bits	***Name***	***Value***	***Description***
13:0	io_start_addr	14'hx	Bits [29:16] of start address segment. This register will be set by the initialization code to the address the IDE interface should be mapped into the memory map. If this address is outside the generic bus range the region is disabled.
15:14	Reserved	2'h0	Reserved.

TIMING CONFIGURATION

This section describes the derivation of the generic bus timing parameters to use the PIO MODE 3 timing described in the ATA/ATAPI specification. The IDE interface requires that an acknowledgement-based access be used for any of the higher performance modes. The table below describes the other timing parameters that must be met.

Table 5: Standard IDE Mode 3 Timing

	PIO Mode 3 Timing Parameters	***Min (ns)***	***Max (ns)***
t0	Cycle time.	180	-
t1	Address and chip selects valid to IOR, IOW asserted.	30	-
t2	IOR/IOW pulse width.	80	-
t2i	IOR/IOW recovery time.	70	-
t3	Data setup to IOW high.	30	-
t4	IOW high to data hold.	10	-
t5	Data setup to IOR high.	20	-
t6	IOR high to data hold.	5	-
t6z	IOR high to data tristated.	-	30
t9	IOR/IOW high to chip selects and Address valid hold.	10	-
trd	Read data valid to IORDY asserted.	0	-
ta	IOR/IOW low to IORDY low.	-	35
tb	IORDY pulse width.	-	1250
tc	IORDY assertion to release.	-	5

The ATA/ATAPI PIO mode 3 specification states that t0, t2, and t2i shall be met, and points out the minimum total cycle time requirement is greater than the sum of the minimum times for t2 and t2i. These dictate the basic timing parameters for the generic bus.

This paragraph gives an overview and the next two sections analyze the timing in more detail. The address latch in the PAL does not need the IO_ALE asserted for more than a cycle, so the io_ale_width can be set to its minimum value of 1. The drive requires the same timing for chip select and address lines, so the generic bus chip select (and hence the decoded chip selects to the drive) should be produced as early as possible by setting ale_to_cs to its minimum value of 1. The need to meet t1 fixes the earliest possible time for the strobe signals and leads to cs_to_oe being set to 3 cycles, and ale_to_wr to 4 cycles. The idle cycles parameter is set in order to meet recovery time of 70ns minimum before the assertion of IDE_IOR or IDE_IOW for the next cycle. The pulse width of both oe and wr asserted is set longer than the required 80ns min. to meet the total required cycle time (t0) of 180ns minimum.

READ CYCLE TIMING EXAMPLE

This example will describe how the generic bus timings were chosen based on the read cycle. It should be read in conjunction with the IDE specification, the Generic bus timing parameter table in the BCM1250 User's manual, and the timing diagram below.

- Set ale_width to 1 cycle. This is the minimum value and is sufficient for the address to be captured by the PAL. In this case the falling edge of the IO_ALE is used to capture the address (if a standard 373 latch were used the latch would be open while ALE is high and closed on the falling edge). The IDE interface does not use the byte enables that are output with the address when ale is asserted.
- Set ale_to_cs to 1 cycle. This is its minimum value. The IDE address and chip selects need to have 30ns of setup before IDE_IOR_L is asserted, and IDE_IOR_L is qualified with IO_OE_L. The sooner IO_CS_L4 is asserted the sooner IO_OE_L can be asserted.
- Set cs_to_oe to 3 cycles to meet the IDE address and chip select 30ns minimum setup time before asserting IDE_IOR_L. There is a potential here for the delay through the PAL to cause a violation of the IDE t1 timing specification of 30ns minimum. The maximum delay from the falling edge of IO_ALE through the decode logic to assert the IDE chip select is given as 17.9ns, and the delay from chip select assertion (which happens 10ns after IO_ALE deasserts) to the IDE chip select assertion is maximum 10ns. The chip select path dominates; so the maximum delay from the generic bus chip select assertion to the IDE chip select being valid is the PAL input to output delay of 10ns. The generic bus will assert IO_OE_L 30ns after chip select, and the IDE_IOR_L will be one delay through the PAL (10ns) later. In the case where the delay on the chip select is maximum and the delay on IDE_IOR_L is minimum the setup time for chip select to read will be violated. This could be fixed by using a faster PAL (or adding a cycle to the access), but since the paths are input to output on the same PAL the delays will be similar and this is not likely to be a problem.
- IDE_IOR_L is asserted a maximum of 10ns after assertion of IO_OE_L. See bullet above.
- Read pulse width generation: IDE_IOR_L is asserted a maximum of 10ns after the assertion of IO_OE_L, and negated a maximum of 10ns after negation of IO_OE_L, based on the worst-case delay through the PAL. The IDE specification says that the minimum pulse width for IDE_IOW or IDE_IOR is 80ns. Because the delay through the PAL can be less than 10ns, and probably will be, another 10ns should be added to the IDE_IOR_L width thereby making the pulse width somewhere between 80-to-90ns, depending on the PAL delay. This guarantees the 80ns pulse width.
- According to the IDE specification (tA) IDE_IORDY will be valid no later than 35ns after IDE_IOW_L or IDE_IOR_L is asserted. This is shorter than the minimum width required for the strobe, so the io_cs_width parameter (which sets the first point at which the acknowledgement will be tested) is set by the required strobe width rather than when the ready line becomes valid. In the fastest case (when the drive does not request a wait) the data will be valid 60ns (t2-t5) after IDE_IOR_L asserts, adding this to the cs_to_oe delay (3 cycles) and the additional 10 ns for the IO_OE_L to IO_IOR_L delay gives the data valid on the IDE bus 100ns after the IO_CS_L4 asserts. This sets the earliest point at which the acknowledgement should be examined to see if the cycle should end, and therefore cs_width should be set to 10. If the cycle is extended by the drive the latest the data becomes valid is the point at which the IDE_IORDY asserts. In either case the data has to pass through the databus buffer and meet the 8.5ns setup time into the BCM1250 before the IO_OE_L is deasserted, so the rdy_smpl parameter should be set to 2 to allow 20ns for the data to propagate and meet setup.
- The same 10ns PAL delay will be taken for the deassertion of IO_OE_L to propagate to IDE_IOR_L. From this point the IDE specification requires a 10ns hold on the chip select and address. The same argument could be used as above for cs_to_oe to allow the oe_to_cs parameter to be a single cycle, but there is no advantage in doing so. There needs to be at least 100ns of recovery time on the IDE strobes (to meet the 180ns cycle time using the minimum 80ns asserted) so chip select can be held asserted for an additional cycle (and one cycle less explicit idle time used). Thus oe_to_cs can be set to 2. As will be seen in the next section, the write cycle requires this setting.
- The cycle ends one cycle after IO_CS_L4 is deasserted.
- The number of idle cycles must be selected to meet both the IDE_IOR_L recovery time and the IDE cycle time. Normally two cases need to be considered: first when the subsequent generic bus cycle is also to the IDE, and secondly when it is to some other device. Since the chip select masks the strobe signals only the first need be considered for this interface. There were 9 cycles where the strobe was active, so to meet the 180ns cycle time it must be inactive for 9 more cycles. Three of these have already been taken from the oe_to_cs of 2 cycles and the 1 cycle from chip select deassertion to the access ending. At the start of the next access there will be 1 cycle where ALE is asserted, 1 cycle ale_to_cs and 3 cycles cs_to_oe. These give a total of 8 cycles, one less that the requirement. Therefore the idle_cycles parameter must be set to 1.

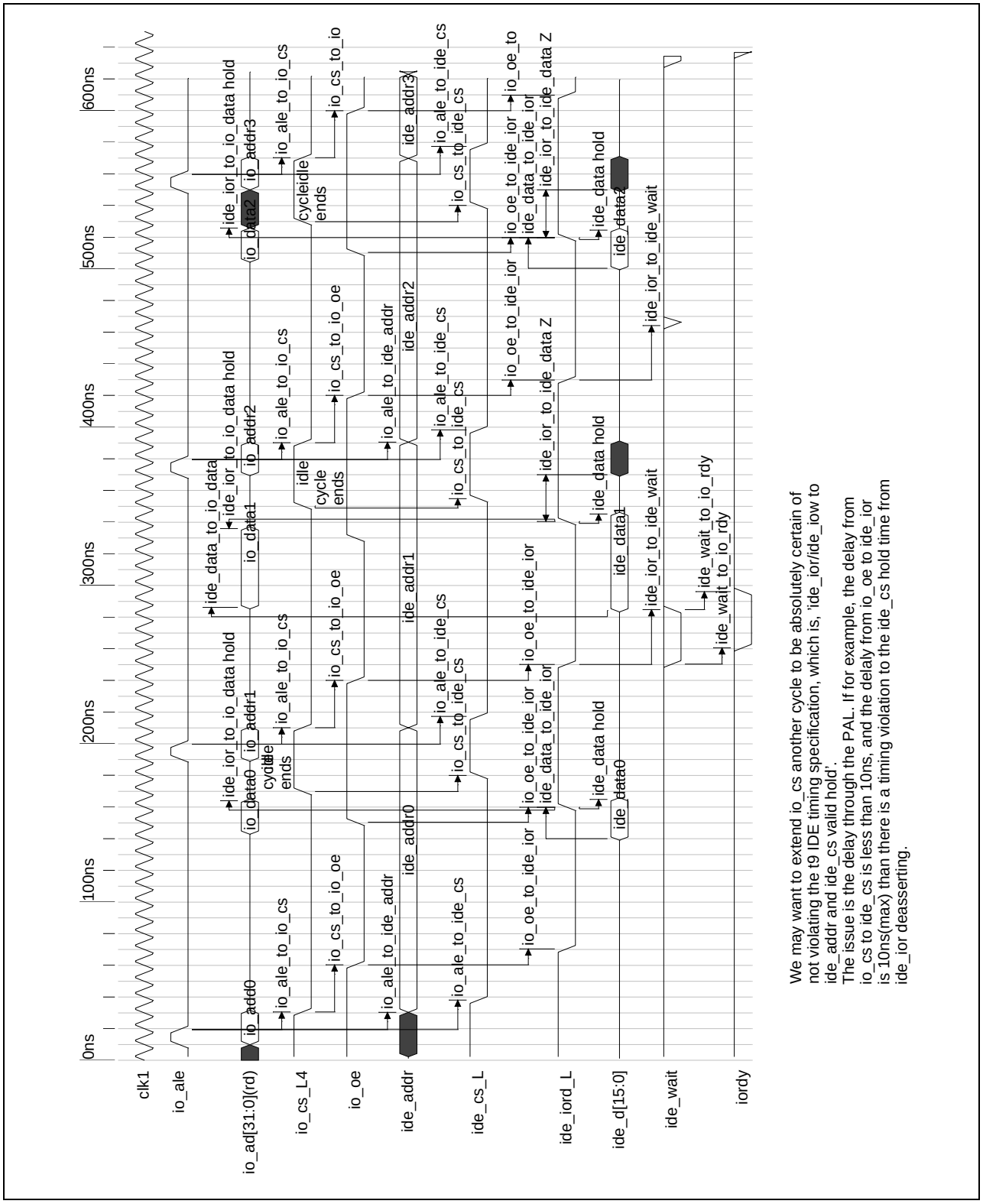

Figure 2: Timing Diagram - Read Cycle

WRITE CYCLE TIMING EXAMPLE

This section examines the generic bus timings based on the write cycle. It should be read in conjunction with the read cycle presented in the previous section, the IDE specification, the Generic bus timing parameter table in the BCM1250 User's manual, and the timing diagram below.

- Set ale_width to 1 cycle. As in the read case this is sufficient for the address to be captured, and gets the address out as soon as possible.
- Set ale_to_cs to 1 cycle. Again this matches the read case to get the IDE chip select lines asserted as soon as possible.
- Set ale_to_wr to 4 cycles to guarantee the IDE specification t1=30ns minimum setup time from the chip select and address lines valid to the strobe. This matches the offsetting of the IO_OE_L in the read case (the write parameter is measured from ale rather than chip select, hence is one larger). This has the same risk of a timing violation if the chip select happened to be slow through the PAL and the IDE_IOW_L fast, again this should not be a problem in practice.
- The assertion of IO_CS_L4 will enable the data buffer, so the data to the drive will be valid after either the output delay of the BCM1250 plus the buffer delay, or the buffer enable delay. In either case the data to the drive will be valid early enough that the setup time before the rising edge of IDE_IOW_L is easily met.
- The IDE_IOW_L width has the same minimum of 80ns as the read strobe, and for the reasons discussed above an additional cycle should be allowed to cover the uncertainties in the PAL. This is met directly by the same settings of cs_width=10 and rdy_smpl=2 that were derived for the read.
- After deasserting IDE_IOW_L the data, address and chip select signals must be held valid for a minimum of 10ns. The address will not change until the next IO_ALE, so will easily make this timing, As in the read case the chip selects would probably meet the specification with a oe_to_cs delay of 1 cycle, but 2 cycles is safer. (Note that in an acknowledgement based cycle the delay from the deassertion of IO_WR_L until the deassertion of the chip select is controlled by the confusingly named oe_to_cs parameter). Meeting the data hold time requires that the oe_to_cs be set to 2 cycles. The data buffer will be disabled by the deassertion of IO_CS_L4. The propagation delay from IO_WR_L deasserting to IDE_IOW_L deasserting could be 10ns, adding the 10ns of hold time required on the data implies that the data buffer must not be disabled until 20ns or 2 cycles after IO_WR_L deasserts. Hence oe_to_cs must be set to 2.
- After negating IO_CS_L4 the 74lcx16245 buffer is disabled.
- The cycle ends one cycle after IO_CS_L4 is deasserted.
- As in the read case there must be at least one additional idle cycle before another generic bus cycle starts.

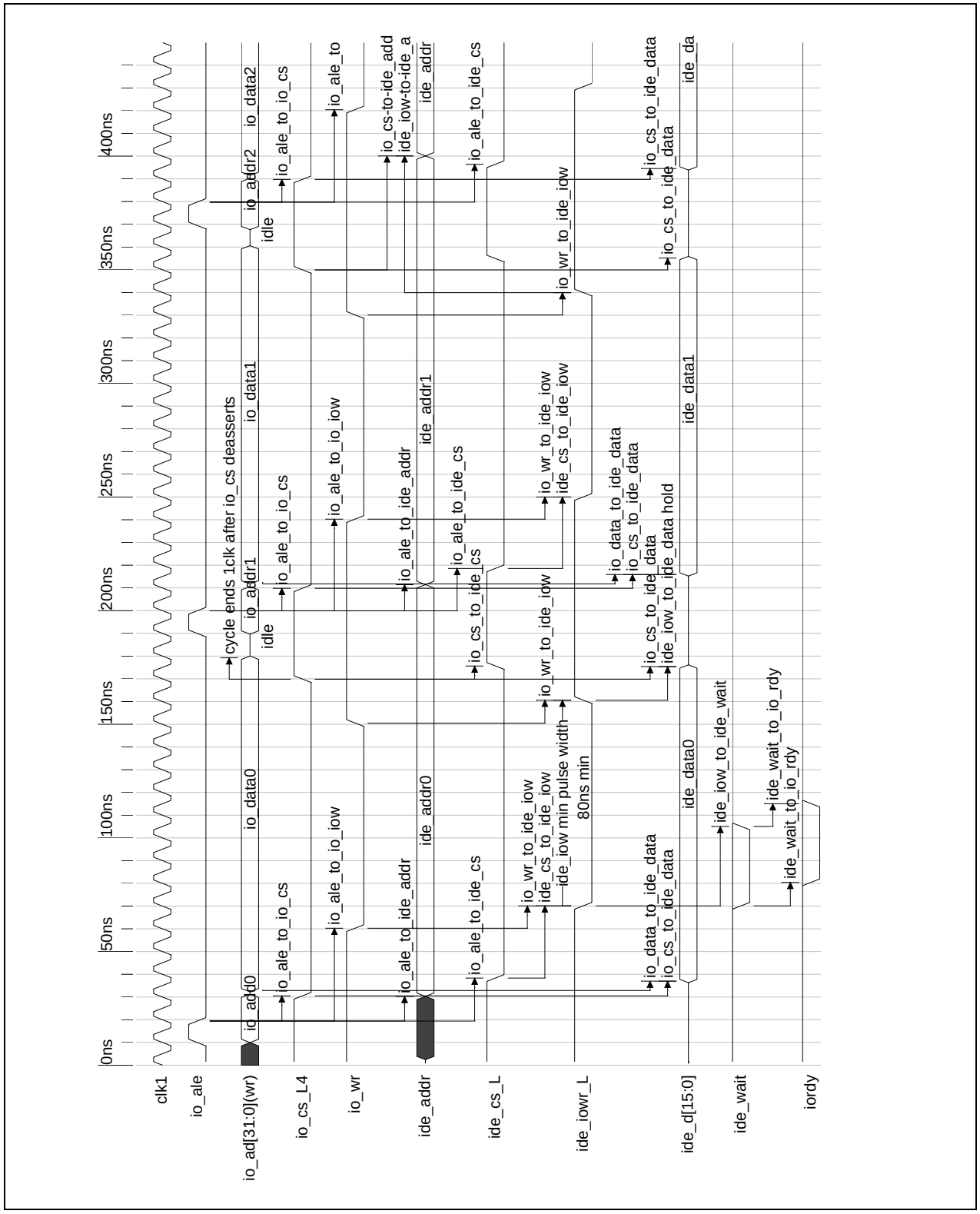

Figure 3: Timing Diagram - Write Cycle

GENERIC BUS SETTINGS

The tables that follow contain the parameters taken from table 4, and shown in figure 2 timing diagram above.

Table 6: io_ext_time_cfg0

Generic Bus Region Timing 0 Register ➠ io_ext_time_cfg0_4 = 00_1006_1620			
Bits	***Name***	***Value***	***Description***
2:0	io_ale_width	2'h01	Width of io_ale is 1 cycle.
3	Reserved	1'b0	Reserved.
5:4	io_ale_to_cs	2'h01	Chip select assertion after assertion of io_ale is set to 1 cycle because we want chip select and address to come out at the same time.
7:6	Reserved	2'h0	Reserved.
12:8	io_cs_width	5'h0A	Width of the chip select asserted is set to 10 cycles. This measurement is taken from io_cs_L4 being asserted to the first time we sample io_rdy.
15:13	io_rdy_smple	3'h2	Number of clock cycles from io_rdy asserting to io_wr or io_oe deasserting.

Table 7: io_ext_time_cfg1

Generic Bus Region Timing 1 Register ➠ io_ext_time_cfg1_4 = 00_1006_1720			
Bits	***Name***	***Value***	***Description***
2:0	io_ale_to_write	3'h4	Assertion of write strobe after the assertion of ale.
3	Reserved	1'b0	Reserved.
7:4	io_write_width	4'hA	Width of the write strobe (a don't care for acknowledgement mode).
11:8	io_idle_cycle	4'h1	Number of idle cycles between back-to-back operations.
13:12	io_cs_to_oe	2'h3	Number of cycles between io_cs asserting and io_oe asserting.
15:14	io_oe_to_cs	2'h2	Number of cycles from io_oe or io_wr deasserting before io_cs deasserts.

GENERIC BUS STATUS REGISTERS

Table 8: io_interrupt_status

Generic Bus Interrupt Status Register ➠ io_interrupt_status = 00_1006_1A00			
Bits	***Name***	***Value***	***Description***
7:0	io_cs_err_int	8'h04	Tell us that chip select region 4 has an error, resulting in an interrupt.
8	Reserved	1'b0	Not used, reads as zero.
9	io_rd_par_int	1'b0	When high, indicates parity error on read data from a parity enabled device. This will never be set for the IDE device.
10	io_timeout_int	1'b0	When high, indicates timeout has occurred on one of the IO blocks. The address that was being accessed is put in the address log.
11	io_ill_addr_int	1'b0	When high, indicates an address referenced did not match any region. The address that was being accessed is put in the address log.
12	io_mult_cs_int	1'b0	When high, indicates multiple chip selects selected based on the address accessed. The address that was being accessed is put in the address log.
15:13	Reserved	3'h0	Reserved.

Section 3: Appendix

SCHEMATIC BCM1250 TO IDE

This schematic is the actual OrCAD schematic page taken from the BCM1250 SWARM evaluation board.

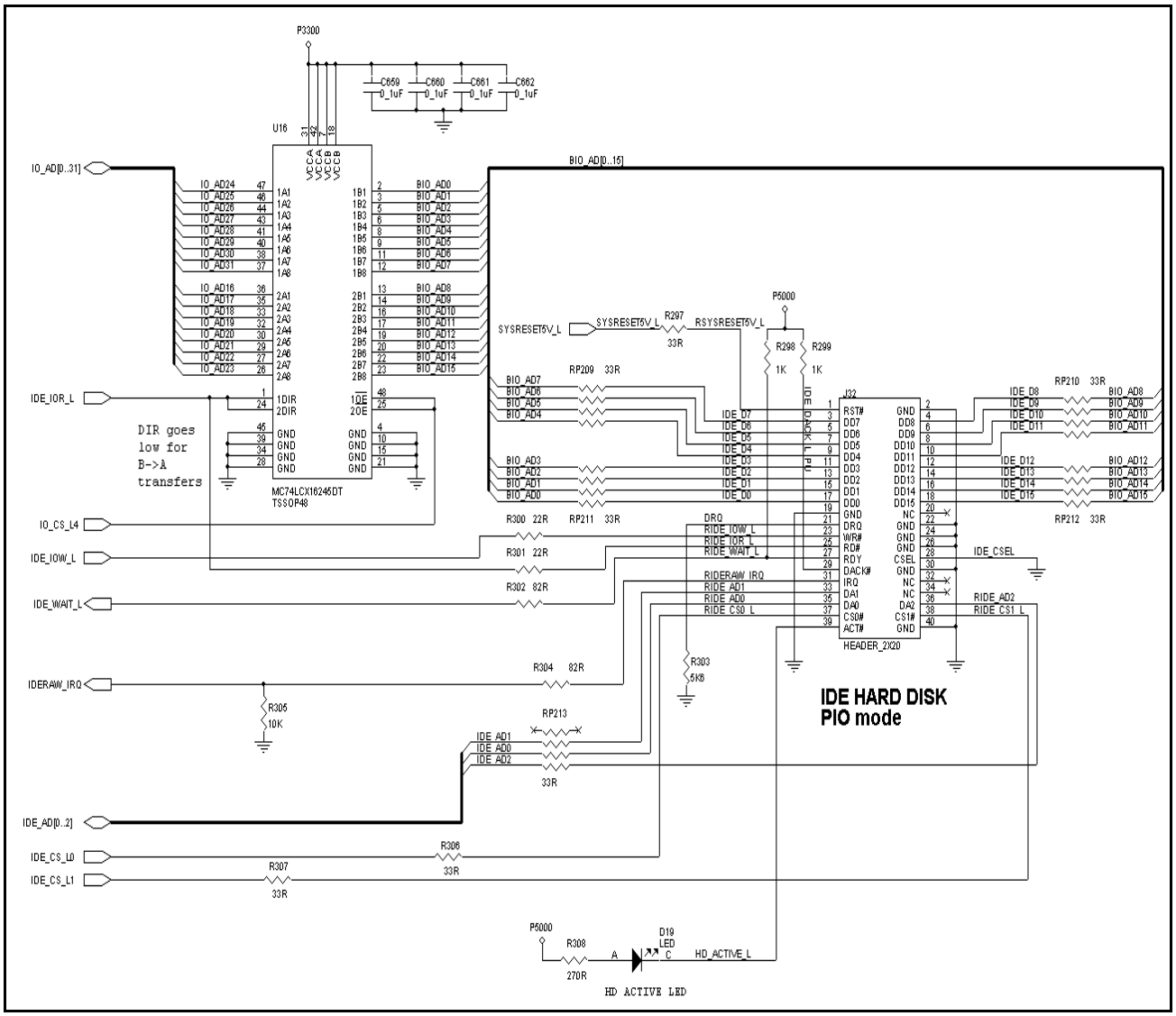

Figure 4: BCM1250 to IDE Schematic

IDE Background

Standard I/O Port Addresses and Interrupts

The following table shows the most commonly supported I/O port addresses and IRQ numbers used for PC ATA (IDE/EIDE) host adapters.

Table 9: PC and Application Note IDE Chip Selects

Interface Number	*PC CS0-Decode*	*PC CS1 - Decode*	*IDE_CS_L0*	*IDE_CS_L1*
Interface 1	01F0h-01F7h	03F6h-03F7h	0x3E00-0x3EE0	0x7EC0-0x7EE0
Interface 2	0170h-0177h	0376h-03774	0x2E00-0x2EE0	0x6EC0-0x6EE0

PIO Modes

There are currently 4 modes of Programmed Input/Output (PIO). All timings are dictated by the ATA specification.

The following table indicates all PIO modes, with their respective transfer rates:

Table 10: PC and Application Note IDE Chip Selects

PIO Mode	*Cycle Time (ns)*	*Transfer Rate (MB/s)*	*Notes*
0	600	3.3	These are the old ATA modes.
1	383	5.2	
2	240	8.3	
3	180 IORDY	11.1	These are the new ATA modes.
4	120 IORDY	16.6	

The ATA-2 specific modes (3 and 4) use IORDY hardware flow control. This means that a drive can use IORDY line to slow down the interface between the generic bus and IDE drive. Specification states that the maximum time allowed for the drive to respond with valid data is 1250ns.

In addition, the delay from the assertion of either IDE_IOR or IDE_IOW until IORDY is first sampled should not be greater than 35ns. Therefore, if a drive needs to assert IORDY to slow down the generic bus it must pull IORDY low no later than 35 ns from the assertion of either IDE_IOR or IDE_IOW.

IDE Connector

Recommended part numbers for the mating connector are shown below, but, equivalent parts may be used.

- Connector (40 Pin) 3M 3417-7000 or equivalent.
- Strain relief 3M 3448-2040 or equivalent.
- Flat Cable (Stranded 28 AWG) 3M 3365-40 or equivalent.
- Flat Cable (Stranded 28 AWG) 3M 3517-40 (Shielded) or equivalent.

IDE I/O Cable

The cable specification affects system integrity, therefore, the total cable length shall NOT exceed 0.46m (18in), and the cable capacitance shall NOT exceed 35pF.

Broadcom Corporation

16215 Alton Parkway
P.O. Box 57013
Irvine, California 92619-7013
Phone: 949-450-8700
Fax: 949-450-8710