TURBOchannel Interface ASIC Specification

Revision 0.5B
Preliminary

Robert Prevett
Workstation Systems Engineering
Digital Equipment Corporation
100 Hamilton Avenue
Palo Alto, CA

December 24, 1991

This information is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may occur in this document. This specification does not describe any program or product currently available from Digital Equipment Corporation. Nor does Digital Equipment commit to implement this specification in any product or program. Digital Equipment Corporation makes no commitment that this document accurately describes any product it will ever make.

Contact:
TRI/ADD at 1-800-678-OPEN or triadd@decwrl.dec.com

Blank Page

Table of Contents

List of Figures

List of Tables

Blank Page

Revision History

Date	Version	Content/Change
28 Dec 90	0.0	- Initial Rough Draft
10 Jan 91	0.1	- Changed "user" to "option" - Added definition of xmit rcv. - Added op.den. - Changed Register Addresses to make them longword aligned.
18 Feb 91	0.2	- Changed register address. - Changed the op.~trValid spec. - Removed 2-cycle I/O write. - Changed CSR bit locations for easier hardware implementation.
22 Mar 91	0.3	- Many Major changes.
15 Oct 91	0.4	- Added Scatter/Gather Table Function. - Changed Option Side Programmed I/O protocol to resemble TURBOchannel, regardless of option clock speed. - Shifted OCR address locations - Added Timing Diagrams - Added Application Examples
18 Nov 91	0.5	- Added threshold to receive DMA burst - Added counter-based flush of receive queue - Added transmit burst size/word count access to register file for intelligent controllers. - Added pin assignments

Blank Page

1 Introduction

This document defines the specification for the TURBOchannel Interface (TCI) ASIC.

The TCI is a general purpose ASIC designed for use on TURBOchannel option modules. The design is meant to be applicable to a majority of TURBOchannel option applications.

Most TURBOchannel options are one of these three types:

- Options requiring only programmed I/O; these comprise a major portion of TURBOchannel option applications.
- DMA devices that do not generate their own address but simply echo data to and from memory; these represent a majority of TURBOchannel option DMA applications.
- Intelligent controller options that supply their own address and execute their own instructions.

1.0.1 Transmit and Receive Definition

The terms transmit and receive are very important to the understanding of this specification Since there are several devices that "transmit" and "receive", direction can confusing. Throughout this document these terms have a specific and consistent meaning. *Transmit DMA is a movement of data from system memory; thus, a DMA read operation is a transmit. Receive DMA is a movement of data to system memory; a DMA write operation is a receive.*

1.1 Programmed I/O and Other Standard Features

- An address latch is provided on chip.
- The programmed I/O subsystem is simple and independent of the DMA subsystem.
- The option has direct control of situations requiring the use of ~conflict.
- The ASIC provides synchronization so that the option side can operate at a different frequency than the TURBOchannel.
- The option may also be placed in a synchronous mode so that no extra delay is introduced when the option is run from the TURBOchannel clock.
- 8 read-write general purpose outputs; 4 readable general purpose inputs that can be used to generate TURBOchannel interrupts.
- TURBOchannel parity generation and checking.

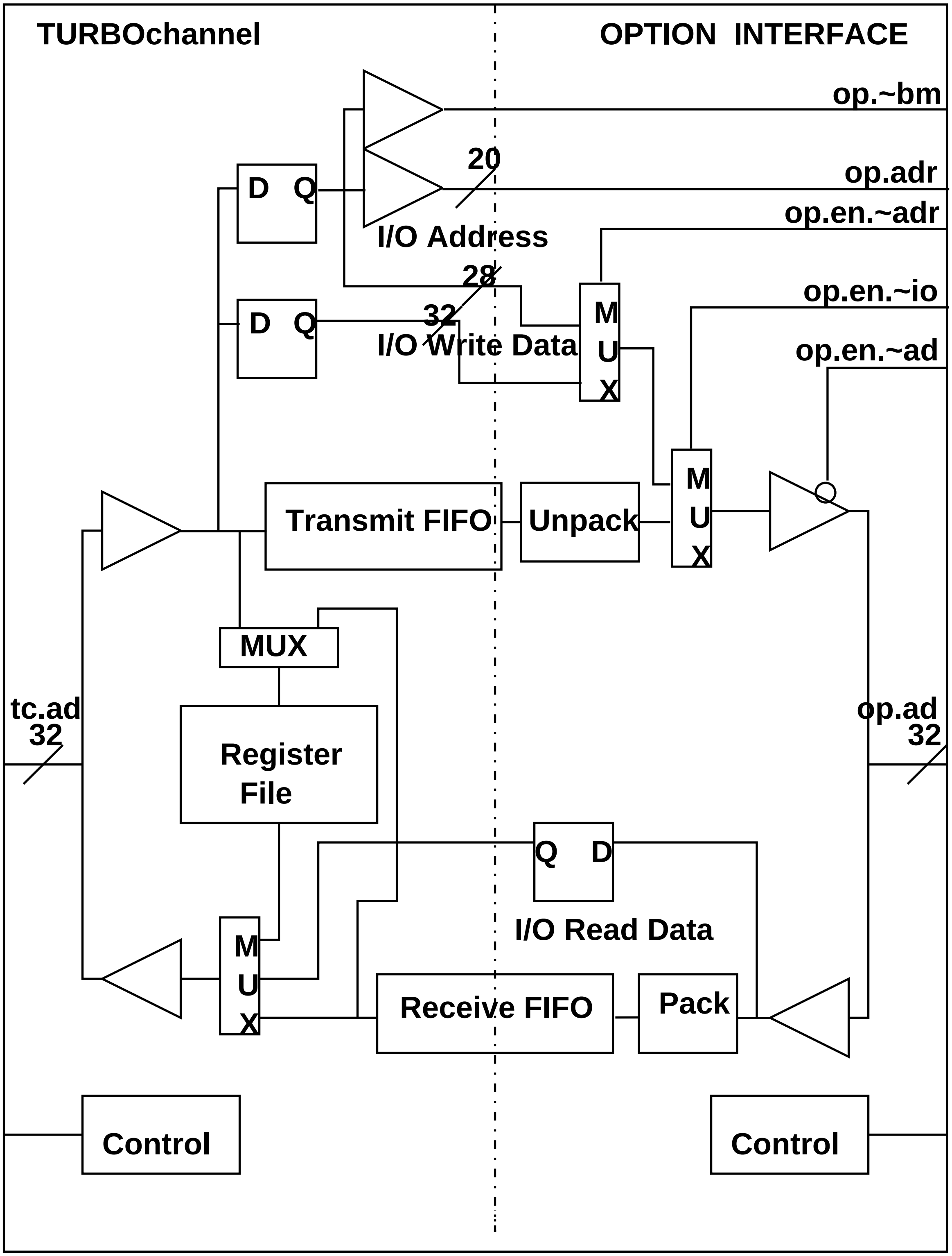
1.2 DMA Features

- Variable option-word size (byte, half-word or word).
- Word to option-word unpacking on transmit DMAs
- Option-word to word packing on receive DMAs.
- Selectable endian-ness.
- Word addressable transmit and receive pointers.
- Independent 16 word transmit and receive FIFOs.
- DMA word transmit and receive counters.
- Buffer pointers are loaded from a scatter/gather table when the data crosses a page boundary so that DMA will not be stalled by interrupt latency for a pointer reload.
- DMA can be halted at the end of a page, when the word counter is zero or by an I/O write to set the DMA disable bit.
- Various features to support DMA transfers that do not begin and end on word boundaries.
- Receive DMAs are processed at maximum TURBOchannel bandwidth.
- Transmit DMAs are processed in programmable fixed bursts from 1 to 128 words.

1.3 Additional Features for Intelligent controllers

- An external address can be supplied for DMA (memory) reads and writes.
- Any address width less than 32 bits is supported.
- The word count and burst size for transmit DMA can be programmed from the option interface of the TCI.

1.4 Block Diagram of TCI ASIC



2 Signal Description

2.1 Option Interface Signals

- op.ad[31..0] (I/O) Option interface address and data.
- op.adr[19..0] (O) Latched I/O address bits <21..2>.
- op.~bm[3:0] (O) Byte mask for I/O read and write transfers.
- op.clk (I) Option interface clock.
- op.en.~ad (I) tristate enable for op.ad lines
- op.en.~io (I) Selects I/O address/data or transmit FIFO data onto op.ad lines.
- op.en.~adr (I) Selects I/O address or data onto op.ad
- op.~gpi[3..0] (I) Interrupt or status bits visible in OIR.
- op.gpo[7..0] (O) General purpose outputs that reflect bits in OCR.
- op.synchronous (I) Sets the TCI in synchronous mode.
- op.~conflict (I) Asserts tc.~conflict and tc.~rdy, aborting the ongoing I/O transaction.
- op.~rcvAdr (I) Places the address on op.ad into the receive FIFO. The receive DMA pointer will be loaded with the address when read from the FIFO. Subsequent data will be written to memory at this new address.
- op.~trAdr (I) Moves the address present on op.ad into the receive FIFO. The transmit DMA pointer will be loaded with the new address when it reaches the head of the FIFO. TCI will then use the new address to read BURSTSIZE words into the transmit FIFO. When op.~rcvAdr and op.~trAdr are asserted simultaneously, a burst size/word count is loaded from op.ad into the receive FIFO. The transmit word count and burst size are loaded with this word when it reaches the head of the FIFO.
- op.~rcvDat (I) Places the option-word on the op.ad lines into the receive FIFO.
- op.~rcvValid (O) Indicates there are at least RCVTHRESHOLD free option-words in the receive FIFO.
- op.~rdy (I) Acknowledge for I/O read or write data.
- op.~sel (O) Selects the option for an I/O read or write.
- op.~trDack (I) Selects next option-word in transmit FIFO to be available in the next option clock cycle.
- op.~trValid (O) Indicates there is at least one option-word in the transmit FIFO to be transmitted to the option.

- op.~write (O) Indicates current I/O transaction read or write when op.~test deasserted; become output of the parametric nand-tree when op.~test asserted
- op.~test (O) tristates all outputs except tc.~rReq,tc.~wReq,op.~write.

2.2 TURBOchannel Interface Signals

- tc.~ack (I) DMA read/write acknowledge.
- tc.ad[31..0] (I/O) TURBOchannel address and data.
- tc.p (I/O) TURBOchannel parity
- tc.clk (I) TURBOchannel clock.
- tc.~conflict (O) I/O read/write conflict.
- tc.~err (I) TURBOchannel DMA error
- tc.~int (O) TURBOchannel interrupt.
- tc.~reset (I) Resets the option when asserted.
- tc.~rdy (O) Asserted to complete an I/O read or write.
- tc.~rReq (O) TURBOchannel DMA Read request
- tc.~sel (I) I/O read or write select.
- tc.~write (I) I/O transaction read/write specifier.
- tc.~wReq (O) TURBOchannel DMA Write request.

2.3 TCI Pin Assignment

Table 1 on the following page shows the package pin assignments for the TCI. The pin number, direction (i=input, b=bi-directional, o=output), and signal name are shown. Figure 1 shows the package pin diagram for the TCI. The breakdown of pin types is as follows:

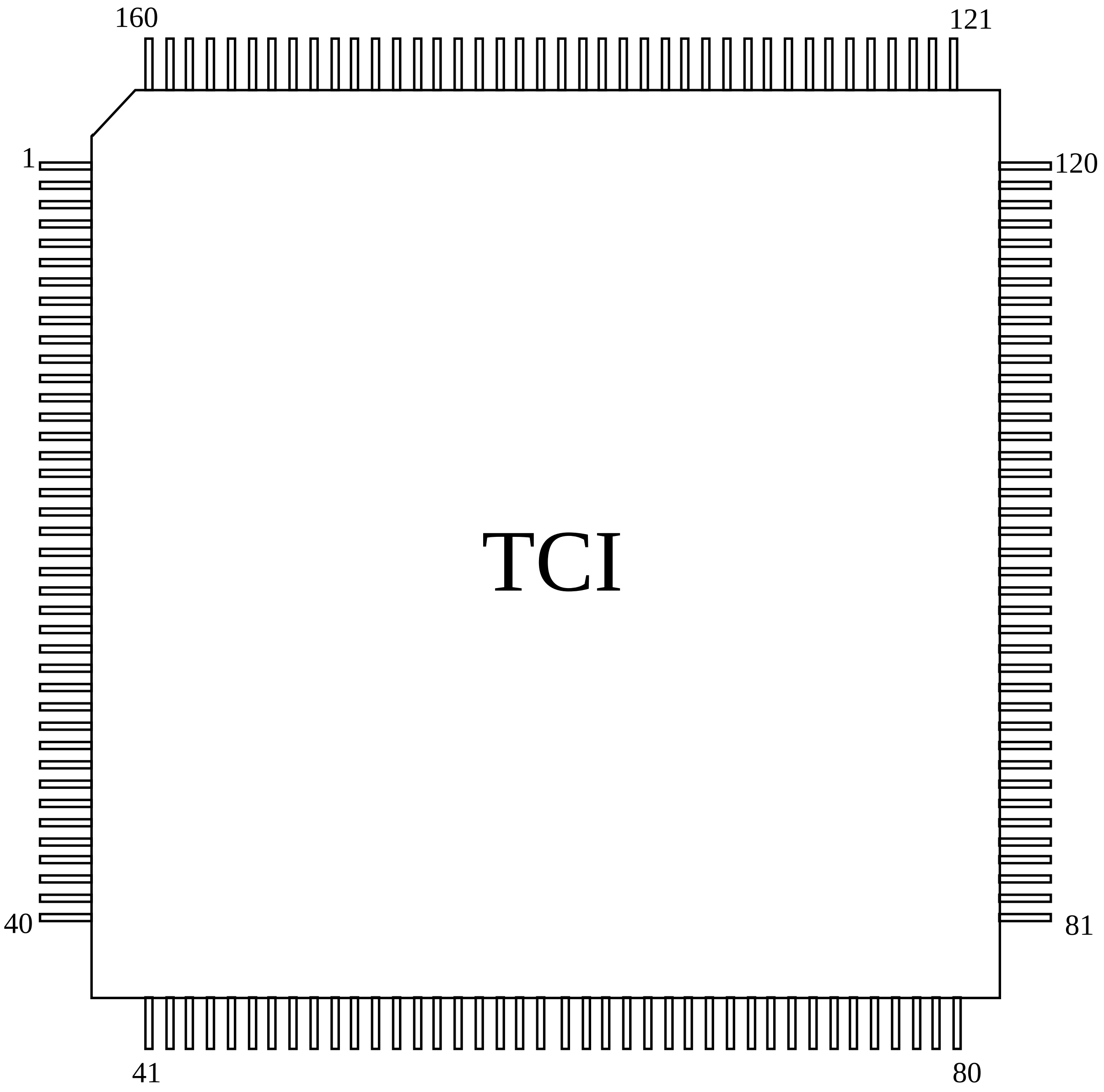
84 Option Interface Signal Pins
44 TURBOchannel Interface Signal Pins
32 Total VDD/VSS

160 Total Pins

Table 1 Package Pin Assignments

pin	dir	signal	pin	dir	signal	pin	dir	signal	pin	dir	signal
1	i	Vdd	41	i	Vss	81	i	Vss	121	i	Vss
2	b	op.ad[25]	42	o	op.clk	82	i	tc.clk	122	b	tc.ad[24]
3	b	op.ad[24]	43	o	op.~synchronous	83	i	Vdd	123	b	tc.ad[25]
4	b	op.ad[23]	44	o	op.~bm[3]	84	o	tc.~rReq	124	b	tc.ad[26]
5	b	op.ad[22]	45	o	op.~bm[2]	85	i	Vss	125	b	tc.ad[27]
6	b	op.ad[21]	46	o	op.~bm[1]	86	o	tc.~wReq	126	b	tc.ad[28]
7	b	op.ad[20]	47	o	op.~bm[0]	87	i	tc.~err	127	b	tc.ad[29]
8	b	op.ad[19]	48	o	op.adr[19]	88	o	tc.~conflict	128	b	tc.ad[30]
9	b	op.ad[18]	49	o	op.adr[18]	89	b	tc.ad[p]	129	b	tc.ad[31]
10	i	Vss	50	i	Vss	90	i	Vdd	130	i	Vss
11	i	Vdd	51	i	Vdd	91	i	Vss	131	i	Vdd
12	b	op.ad[17]	52	o	op.adr[17]	92	b	tc.ad[0]	132	o	op.gpo[0]
13	b	op.ad[16]	53	o	op.adr[16]	93	b	tc.ad[1]	133	o	op.gpo[1]
14	b	op.ad[15]	54	o	op.adr[15]	94	b	tc.ad[2]	134	o	op.gpo[2]
15	b	op.ad[14]	55	o	op.adr[14]	95	b	tc.ad[3]	135	o	op.gpo[3]
16	b	op.ad[13]	56	o	op.adr[13]	96	b	tc.ad[4]	136	o	op.gpo[4]
17	b	op.ad[12]	57	o	op.adr[12]	97	b	tc.ad[5]	137	o	op.gpo[5]
18	b	op.ad[11]	58	o	op.adr[11]	98	b	tc.ad[6]	138	o	op.gpo[6]
19	b	op.ad[10]	59	o	op.adr[10]	99	b	tc.ad[7]	139	o	op.gpo[7]
20	i	Vdd	60	i	Vss	100	i	Vdd	140	i	Vss
21	i	Vss	61	i	Vdd	101	i	Vss	141	i	op.~gpi[0]
22	b	op.ad[9]	62	o	op.adr[9]	102	b	tc.ad[8]	142	i	op.~gpi[1]
23	b	op.ad[8]	63	o	op.adr[8]	103	b	tc.ad[9]	143	i	op.~gpi[2]
24	b	op.ad[7]	64	o	op.adr[7]	104	b	tc.ad[10]	144	i	op.~gpi[3]
25	b	op.ad[6]	65	o	op.adr[6]	105	b	tc.ad[11]	145	i	op.~rcvAdr
26	b	op.ad[5]	66	o	op.adr[5]	106	b	tc.ad[12]	146	i	op.~trAdr
27	b	op.ad[4]	67	o	op.adr[4]	107	b	tc.ad[13]	147	i	op.~rcvDat
28	b	op.ad[3]	68	o	op.adr[3]	108	b	tc.ad[14]	148	o	op.~rcvValid
29	b	op.ad[2]	69	o	op.adr[2]	109	b	tc.ad[15]	149	i	Vdd
30	i	Vss	70	i	Vss	110	i	Vss	150	i	Vss
31	b	op.ad[1]	71	o	Vdd	111	i	Vdd	151	i	op.~trDack
32	b	op.ad[0]	72	o	op.adr[1]	112	b	tc.ad[16]	152	o	op.~trValid
33	i	op.en.~ad	73	o	op.adr[0]	113	b	tc.ad[17]	153	i	op.~test
34	i	op.en.~io	74	o	tc.~int	114	b	tc.ad[18]	154	b	op.ad[31]
35	i	op.en.~adr	75	i	tc.~reset	115	b	tc.ad[19]	155	b	op.ad[30]
36	i	op.~rdy	76	i	tc.~sel	116	b	tc.ad[20]	156	b	op.ad[29]
37	o	op.~sel	77	o	tc.~rdy	117	b	tc.ad[21]	157	b	op.ad[28]
38	o	op.~write	78	i	tc.~write	118	b	tc.ad[22]	158	b	op.ad[27]
39	i	op.~conflict	79	i	tc.~ack	119	b	tc.ad[23]	159	b	op.ad[26]
40	i	Vdd	80	i	Vss	120	i	Vdd	160	i	Vss

Figure 1 TCI Package Pin Diagram



3 Control and Status Registers

Each option in a system responds to a different I/O address range. The parameters SLOTSIZE and SLOTBASE are system dependent variables residing in the system ROM. In general, the address of each slot is defined by:

I/OBASE = (Slot #)*(SLOTSIZE) + SLOTBASE

Any specific address within an option is given by:

Slot Address = I/OBASE + Offset

Offset	ASIC Register or Address Space
0x300	Transmit DMA Pointer (TDP)
0x304	Receive DMA Pointer (RDP)
0x308	Option Control Register (OCR)
0x30C	Miscellaneous Register (MISC)
0x310	Transmit DMA Scatter/Gather Pointer (TSGP)
0x314	Receive DMA Scatter/Gather Pointer (RSGP)
0x318	DMA Pointer Mask (DPM)
0x31C	Returns 0x00000000 when read.
0x320	Transmit DMA word Counter (TWC)
0x324	Receive DMA word Counter (RWC)
0x328	Option Interrupt Register (OIR)
0x32C	Option Interrupt Mask Register (OIMR)
0x330	FIFO access.
0x340 to end of Slot	ROM and Option I/O Space

The CSRs are placed beginning at slot offset 0x300. Writes and reads to these registers will not be visible at the option interface, (op.~sel will not assert). The register file is word based, and programmed I/O access to the register file must be word based. Bytes masks are not valid for register file reads or writes. Note that the option ROM, which starts at an offset of 0x3E0, in no way conflicts with these registers

3.0.1 Reset State

When tc.~reset is asserted during power up/down, all the CSRs are reset to 0's. The FIFOs are emptied by setting the heads and tails to 0's, and all outputs are placed in a deasserted state. The tc.ad lines are tristated, but op.ad is under option tristate control. If op.~en is deasserted, op.ad will be tristated by the TCI. The TCI requires at least 8 clock cycles of tc.~reset to initialize completely.

3.1 Transmit Data Pointer (TDP)

<31:0> (R/W) The transmit DMA pointer is a word address that is used to read memory

into the transmit FIFO. It updated as each word is received from the TURBOchannel. If an error occurs then the DMA is halted and the error is reported in the OIR. The pointer will contain the address of the data word in error at the time of the error.

3.2 Transmit DMA Scatter/Gather Table Pointer (TSGP)

<31:0> (R/W) The TSGP contains the word address for the transmit DMA scatter/gather table pointer. For DMA transfers that cross page boundaries, the option device driver builds a scatter/gather table in main memory. This table contains the word-aligned physical addresses of any subsequent pages in the DMA block transfer. This table must be in a physically contiguous portion of system memory. The address entries in this table are 32 bit word addresses. When a transmit DMA reaches a page boundary, the TCI reads the page address pointed to by the transmit DMA scatter/gather table pointer and loads this address into the transmit DMA pointer. The transmit DMA scatter/gather table pointer is then incremented. The virtually contiguous DMA buffer pages that are allocated may often be non-contiguous physical pages. This scheme allows DMA to continue without interrupting to load a new buffer pointer for the next page.

3.3 Receive DMA Pointer (RDP)

<31:0> (R/W) The receive DMA pointer is a word address that is used to move data from the receive FIFO into memory. It is incremented as each word is written into memory. When an error occurs the DMA is halted. The error is reported in the OIR. The pointer will contain the address of the data in error.

3.4 Receive DMA Scatter/Gather Table Pointer (RSGP)

<31:0> (R/W) This RSGP contains the receive DMA scatter/gather table pointer. This pointer works in the same manner as the transmit DMA scatter/gather table pointer. Upon reaching a page boundary, the receive DMA pointer is loaded with the address located at the memory location pointed to by the receive DMA scatter/gather table pointer. The receive DMA scatter/gather table pointer is then incremented.

3.5 DMA Pointer Mask (DPM)

<31:0> (R/W) These bits contain a mask for both the transmit and receive pointers. This is useful for intelligent controllers that cannot supply the entire address range. When an external address is supplied from the option interface, that address is combined with the current pointer in the following way.

NewPointer = (OldPointer AND (PointerMask))
OR
(ExternalAddress AND (NOT PointerMask))

3.6 Option Control Register (OCR)

<31> (R/W) TDMAEN
1: The TCI may read words from memory into the transmit FIFO.
0: Reads of memory into the transmit FIFO are disabled.
The option-side signals are unaffected.

<30> (R/W) RDMAEN
1: The TCI may write receive FIFO words into memory.
0: Words in the receive FIFO are not written to memory.
The option-side signals are unaffected.

<29> (R/W) TQEN
1: Enables the option-side transmit DMA
0: Clears the transmit FIFO and deasserts op.~trValid.

<28> (R/W) RQEN
1: Enables the option-side receive DMA.
0: Clears the receive FIFO and deasserts op.~rcvValid,

<27:26> (R/W) TUPACK
00:Unpacking starts at byte0, half-word0
01:Unpacking starts at byte1,
10:Unpacking starts at byte2, half-word1
11:Unpacking starts at byte3
These bits are loaded into the option-word unpacking counter whenever TQEN is 0.

<25:24> (R/W) RPACK
00:Packing starts at byte0, half-word0
01:Packing starts at byte1,
10:Packing starts at byte2, half-word1
11:Packing starts at byte3
These bits are loaded into the option-word packing counter whenever RQEN is 0.

<23:17> (R/W) BURSTINIT
0x00: 1 word burst for first transmit DMA
0x01: 2 word burst for first transmit DMA
..
0x7F: 128 word burst for first transmit DMA

<16:10> (R/W) BURSTSIZE
0x00: 1 word burst for subsequent transmit DMA
0x01: 2 word burst for subsequent transmit DMA
..
0x7F: 128 word burst
The transmit DMA engine moves BURSTSIZE option-words into the transmit FIFO with each DMA. If BURSTSIZE is less than 16 then the TCI checks to make sure that there are at least BURSTSIZE free words in the transmit FIFO before starting the DMA. Otherwise the option must empty the FIFO as fast as it is being filled.

<9:8> (R/W) SIZE, Sets the option-word packing/unpacking size.

00: word	op.ad<31:0>
01: half-word	op.ad<15:0>
10: byte	op.ad<7:0>
11: Reserved	

<7:0> (R/W) GPO
The value of this register is reflected directly at the output signals op.gpo<7:0>. They assert and deassert in the option time domain.

3.7 Miscellaneous Register (MISC)

<31:25> (R) Reserved

<24:21> (R) TQFREEWORDS, number of valid free words in transmit FIFO

<20:17> (R) RQFULLWORDS, number of words in receive FIFO

<16:15> (R/W) ALIGN, Modifies the alignment of address placed on op.ad (op.~trAdr, op.~rcvAdr).
00: op.ad<31:0> -> address<31:0>
01: op.ad<30:0> -> address<31:1>
10: op.ad<29:0> -> address<31:2>
11: RESERVED

<14> (R/W) BIGENDIAN
1: Option-word packing/unpacking order is big-endian, tc.ad<31:0>
= | byte0 | byte1 | byte2 | byte3 |
0: Option-word packing/unpacking order is little-endian, tc.ad<31:0>
= | byte3 | byte2 | byte1 | byte0 |

<13:12> (R/W) RCVFLSHCNT
00: Receive flush counter value of 16 used
01: Receive flush counter value of 64 used
10: Receive flush counter value of 256 used
11: Receive flush counter value of 1024 used

<11:10> (R/W) RCVBURST
00: Receive DMA burst initiated when rcv FIFO has at least 1 valid word
01: Receive DMA burst initiated when rcv FIFO has at least 2 valid words
10: Receive DMA burst initiated when rcv FIFO has at least 4 valid words
11: Receive DMA burst initiated when rcv FIFO has at least 8 valid words

<9:7> (R/W) RCVTHRESHOLD
000: Deasserts op.~rcvValid if the rcv FIFO is full.
001: Deasserts op.~rcvValid when FIFO free words < 2
010: Deasserts op.~rcvValid when FIFO free words < 3
..
111: Deasserts op.~rcvValid when FIFO free words < 8

<6> (R/W) CONTINUOUS
1: Transmit DMA, (when enabled) will keep the transmit FIFO filled with data by reading memory in BURSTSIZE transfers.
0: The transmit FIFO will be loaded only on demand: when op.~trAdr requests data

<5:2> (R/W) PAGESIZE
0000: no page boundary;
0001: 16Byte scatter/gather block size;
0010: 32Byte scatter/gather block size;
0011: 64Byte scatter/gather block size;
0100: 128Byte scatter/gather block size;
0101: 256Byte scatter/gather block size;
0110: 512Byte memory system page size;
0111: 1KByte memory system page size;
1000: 2KByte memory system page size
1001: 4KByte memory system page size
1010: 8KByte memory system page size
1011: 16KByte memory system page size
1100: 32KByte memory system page size
1101: 64KByte memory system page size
1110: Reserved
1111: Reserved
The contents of this field control when the TDP and RDP are updated from the scatter/gather tables.

<1> (R/W) PASSTHROUGH
1:All TURBOchannel traffic is placed in the transmit FIFO
0: Normal Operation.

<0> (R/W) PEN
1: Enables the parity error check logic.
0: disables the parity error check logic PER

3.8 Option Interrupt Mask Register (OIMR)

<31:0> (R/W) This register is used to enable the corresponding interrupt in OIR<31:0>. For example when <0> is written to a 1, OIR<0> interrupt is enabled. This register, as with all registers, are reset to 0's via tc.~reset, disabling all 32 interrupts.

3.9 Receive DMA Word Counter (RWC)

<31:24> (R) These bits are reserved. Writes have no effect and zeros are returned on reads.

<23:0> (R/W) RWC, These bits contain the word count of the current receive DMA. It is decremented with each word written to memory until it reaches zero.

3.10 Transmit DMA Word Counter (TWC)

<31:24> (R) These bits are reserved. Writes have no effect and zeros are returned on reads.

<23:0> (R/W) These bits contain the word count of the current transmit DMA. It is decremented as each word is read from memory until it reaches zero.

3.11 Option Interrupt Register (OIR)

Bits in this register are ANDed with the OIMR and then ORed together to form a single TURBOchannel interrupt (tc.~int).

<31> (R/W0TC) TDMAERR
1: Error during a transmit DMA (TURBOchannel DMA read).
0: No error during transmit DMA.

<30> (R/W0TC) RDMAERR
1: Error during a receive DMA (TURBOchannel DMA write).
0: No error during receive DMA.

<29> (R/W0TC) TSGLD
1: An entry from the scatter/gather table has been loaded into the
transmit pointer upon reaching a page boundary.
0: The TSGP has not yet been used.

<28> (R/W0TC) TPLD
1: The transmit pointer has reached a page boundary and the TSGP is 0x0.
0: The receive pointer has not reached a page boundary while the TSGP is 0x0.

<27> (R/W0TC) RSGLD
1: An entry from the scatter/gather table has been loaded into the
receive pointer upon reaching a page boundary.
0: The RSGP has not yet been used.

<26> (R/W0TC) RPLD
1: The receive pointer has reached a page boundary and the RSGP is 0x0.
0: The receive pointer has not reached a page boundary while the RSGP is 0x0.

<25> (R/W0TC) TWCEXPIRED,
1: The transmit word counter has decremented to zero.
0: The transmit word counter has not decremented to zero.

<24> (R/W0TC) RWCEXPIRED,
1: The receive word counter has decremented to zero.
0: The receive word counter has not decremented to zero.

<23> (R/W0TC) PERR
1: A parity error was detected on the TURBOchannel.
0: No parity error has been detected on the TURBOchannel.

<22:12> (R) RESERVED

<11> (R) TFE
1: The transmit FIFO is empty.
0: The transmit FIFO is not empty.

<10> (R) TFNE
1: The transmit FIFO is not empty.
0: The transmit FIFO is empty.

<9> (R) RFE
1: The receive FIFO is empty.
0: The receive FIFO is not empty.

<8> (R) RFNE
1: The receive FIFO is not empty.
0: The receive FIFO is empty.

<7:6> (R) TUPACK2
00: Unpacking byte0, half-word0
01: Unpacking byte1,
10: Unpacking byte2, half-word1
11: Unpacking byte3
These bits are a synchronized version of the transmit unpacking register.
They are reliably read only when the transmit FIFO is not being read.

<5:4> (R) RPACK2
00: Packing byte0, half-word0
01: Packing byte1,
10: Packing byte2, half-word1
11: Packing byte3
These bits are a synchronized version of the option-word packing counter.
They are reliably read only when the receive FIFO is not being written.

<3:0> (R) GPI
These bits reflect the state of external signals op.~gpi<3:0>. If op.~gpi<0>
is a 0 then OIR<0> will be a 1.

3.12 FIFO Access

3.12.1 (R) Receive FIFO

Reads return the value of the next word to be written to memory from the receive FIFO. If DMA ended on a non-word boundary and the FIFO is empty (RFE=1), then reads to this address this will return the value of the incomplete word in the byte-packer. Reads to this register will not affect the state of the receive FIFO.

3.12.2 (W) Transmit FIFO

Writes to this register space will place data into the receive FIFO. This is useful for writing unaligned words at the start of a transmit DMA transfer.

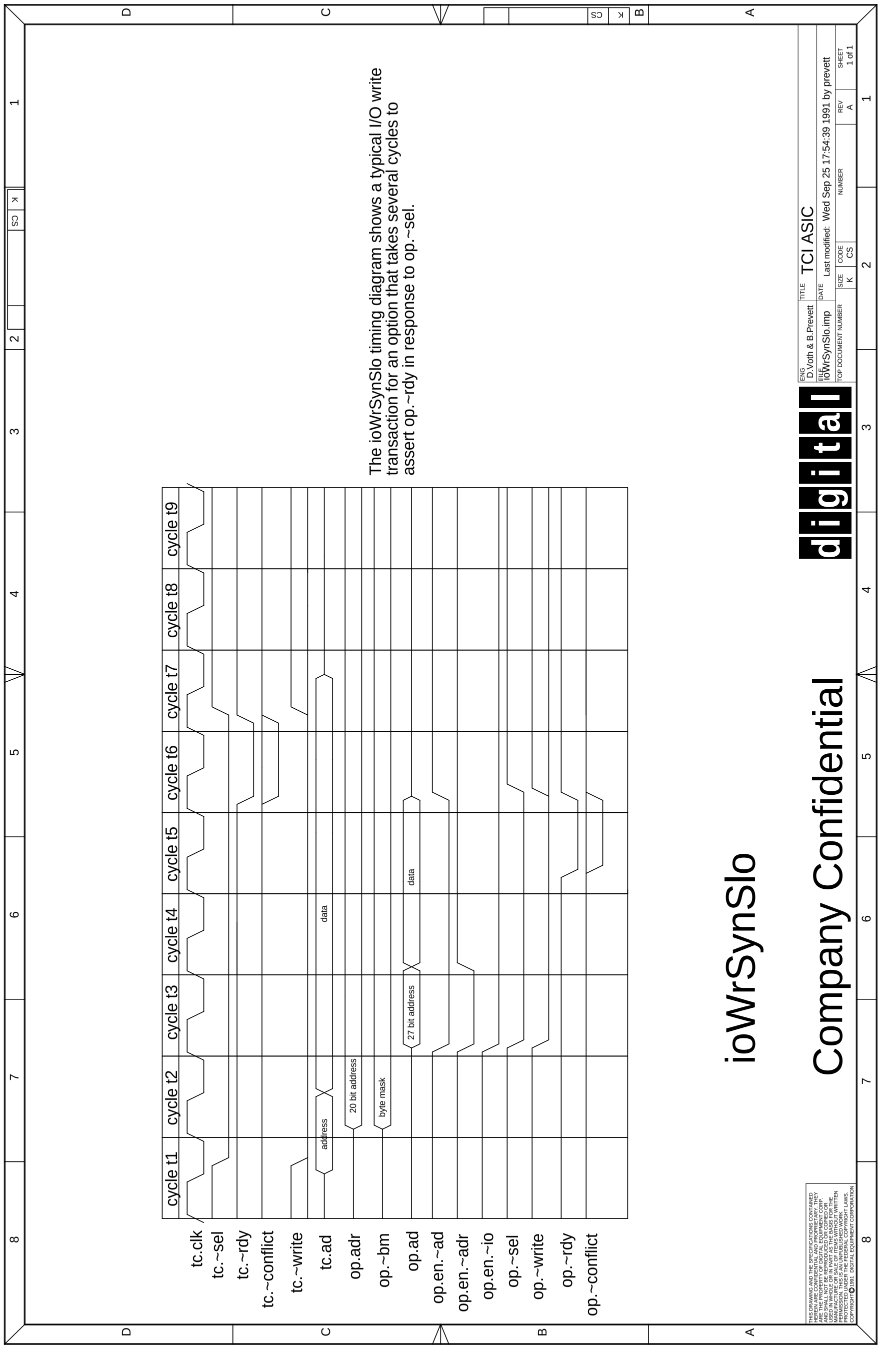
4 Programmed I/O

Many TURBOchannel devices are simple slaves. The TCI ASIC provides an I/O subsystem that is simple and independent of the DMA subsystem. The option has control of the ~conflict signal and may issue op.~conflict with op.~rdy. The asynchronous mode requires a synchronous handshake between op.~rdy and op.~sel. This handshake is performed in the same manner as TURBOchannel.

The following sections step through timing diagrams for I/O write and read operations, with the option side driven by tc.clk or an independent op.clk. The option side signal op.synchronous is asserted if the option interface is running from the TURBOchannel clock; op.synchronous is deasserted if the option is running off of an independent op.clk.

4.1 I/O Write with op.synchronous = 1

Figure 2   I/O Write Timing Diagram with op.synchronous = 1


cycle t1
cycle t2
cycle t3
cycle t4
cycle t5
cycle t6
cycle t7
cycle t8
cycle t9
tc.clk
tc.~sel
tc.~rdy
tc.~conflict
tc.~write
tc.ad
op.adr
op.~bm
op.ad
op.en.~ad
op.en.~adr
op.en.~io
op.~sel
op.~write
op.~rdy
op.~conflict
address
data
20 bit address
byte mask
27 bit address
data
ioWrSynSlo
Company Confidential
digital
ENG D.Voth & B.Prevett
TITLE TCI ASIC
FILE ioWrSynSlo.imp
DATE Last modified: Wed Sep 25 17:54:39 1991 by prevett
TOP DOCUMENT NUMBER
SIZE K
CODE CS
NUMBER
REV A
SHEET 1 of 1
THIS DRAWING AND THE SPECIFICATIONS CONTAINED HEREIN ARE CONFIDENTIAL AND PROPRIETARY. THEY ARE THE PROPERTY OF DIGITAL EQUIPMENT CORP. AND SHALL NOT BE REPRODUCED OR COPIED OR USED IN WHOLE OR IN PART AS THE BASIS FOR THE MANUFACTURE OR SALE OF ITEMS WITHOUT WRITTEN PERMISSION. THIS IS AN UNPUBLISHED WORK PROTECTED UNDER THE FEDERAL COPYRIGHT LAWS. COPYRIGHT © 1991 DIGITAL EQUIPMENT CORPORATION
8
7
6
5
4
3
2
1
D
C
B
A
K
CS

The ioWrSynSlo timing diagram shows a typical I/O write transaction for an option that takes several cycles to assert op.~rdy in response to op.~sel.

Cycle t1:

- system asserts tc.~sel and tc.~write to initiate an I/O write transaction to the option
- system drives I/O address onto tc.ad

Cycle t2:

- TCI loads address and byte mask into address/byte mask register on rising edge of tc.clk.
- TCI drives 20 bit address onto op.adr and byte mask onto op.~bm
- system drives I/O write data on tc.ad

Cycle t3:

- TCI asserts op.~sel and op.~write to signify an I/O write transaction to the option interface.
- TCI drives full 27 bit address onto op.ad if option asserts op.en.~adr, op.en.~io, and op.en.~ad; the full tc.ad data is driven onto op.ad if op.en.~adr is deasserted.
- TCI loads I/O write data.

Cycle t4:

- TCI drives data onto op.ad if option deasserts op.en.~adr, and asserts op.en.~io and op.en.~ad.

Cycle t5:

- option asserts op.~rdy in response to op.~sel. op.~conflict may be asserted with op.~rdy in this same cycle if the option is committed to a DMA transaction.

Cycle t6:

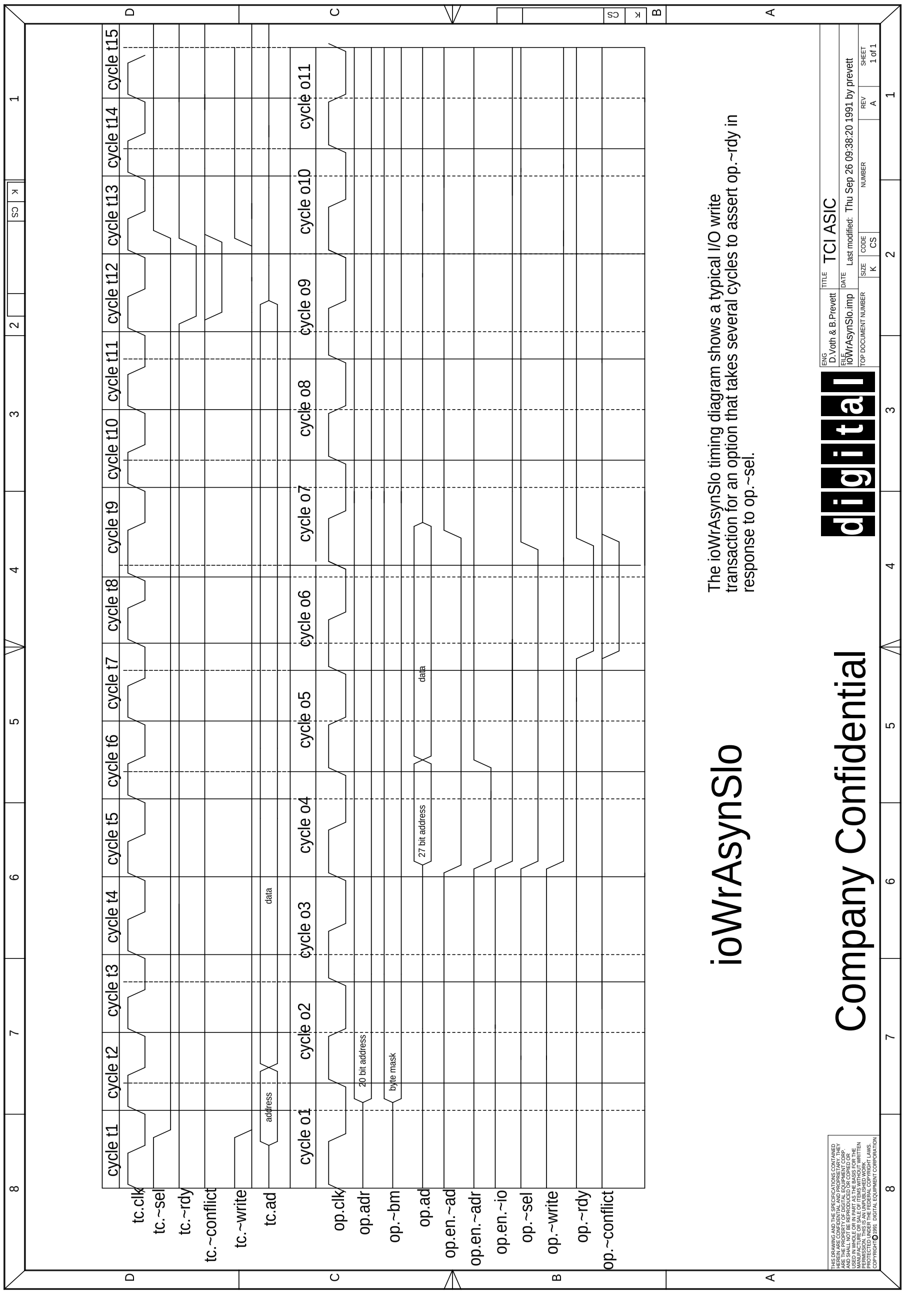
- option latches I/O write data on op.ad; option then deasserts op.en.~ad to tristate op.ad lines.
- TCI asserts tc.~rdy; tc.~conflict is asserted with tc.~rdy if conflict had been asserted.
- option deasserts op.~rdy, and op.~conflict if applicable.
- TCI deasserts op.~sel and may deassert op.~write.

Cycle t7:

- TCI deasserts tc.~rdy, and tc.~conflict if applicable
- system deasserts tc.~sel in response to assertion of tc.~rdy; system may deassert tc.~write.
- system tristates tc.ad

4.2 I/O Write with op.synchronous = 0

Figure 3 I/O Write Timing Diagram with op.synchronous = 0



The ioWrAsynSlo timing diagram shows a typical I/O write transaction for an option that takes several cycles to assert op.~rdy in response to op.~sel.

Cycle t1:

- system asserts tc.~sel and tc.~write to initiate an I/O write transaction to the option
- system drives I/O address onto tc.ad

Cycle t2:

- TCI loads address and byte mask into address/byte mask register on rising edge of tc.clk.
- TCI loads tc.~sel and tc.~write on rising edge of tc.clk.
- TCI drives 20 bit address onto op.adr and byte mask onto op.~bm
- system drives I/O write data on tc.ad

Cycle t3:

- TCI loads I/O write data.

Cycle o4:

- TCI asserts op.~sel and op.~write to signify an I/O write transaction to the option interface.

- TCI drives full 27 bit address onto op.ad if option asserts op.en.~adr, op.en.~io, and op.en.~ad; the full tc.ad data (shifted address) is driven onto op.ad if op.en.~adr is deasserted.

Cycle o5:

- TCI drives data onto op.ad if option deasserts op.en.~adr, and asserts op.en.~io and op.en.~ad.

Cycle o6:

- option asserts op.~rdy in response to op.~sel. op.~conflict may be asserted with op.~rdy in this same cycle if the option is committed to a DMA transaction.

Cycle o7:

- option latches I/O write data on op.ad; option then deasserts op.en.~ad to tristate op.ad lines.
- option deasserts op.~rdy, and op.~conflict if applicable.
- TCI deasserts op.~sel and may deassert op.~write

Cycle t12:

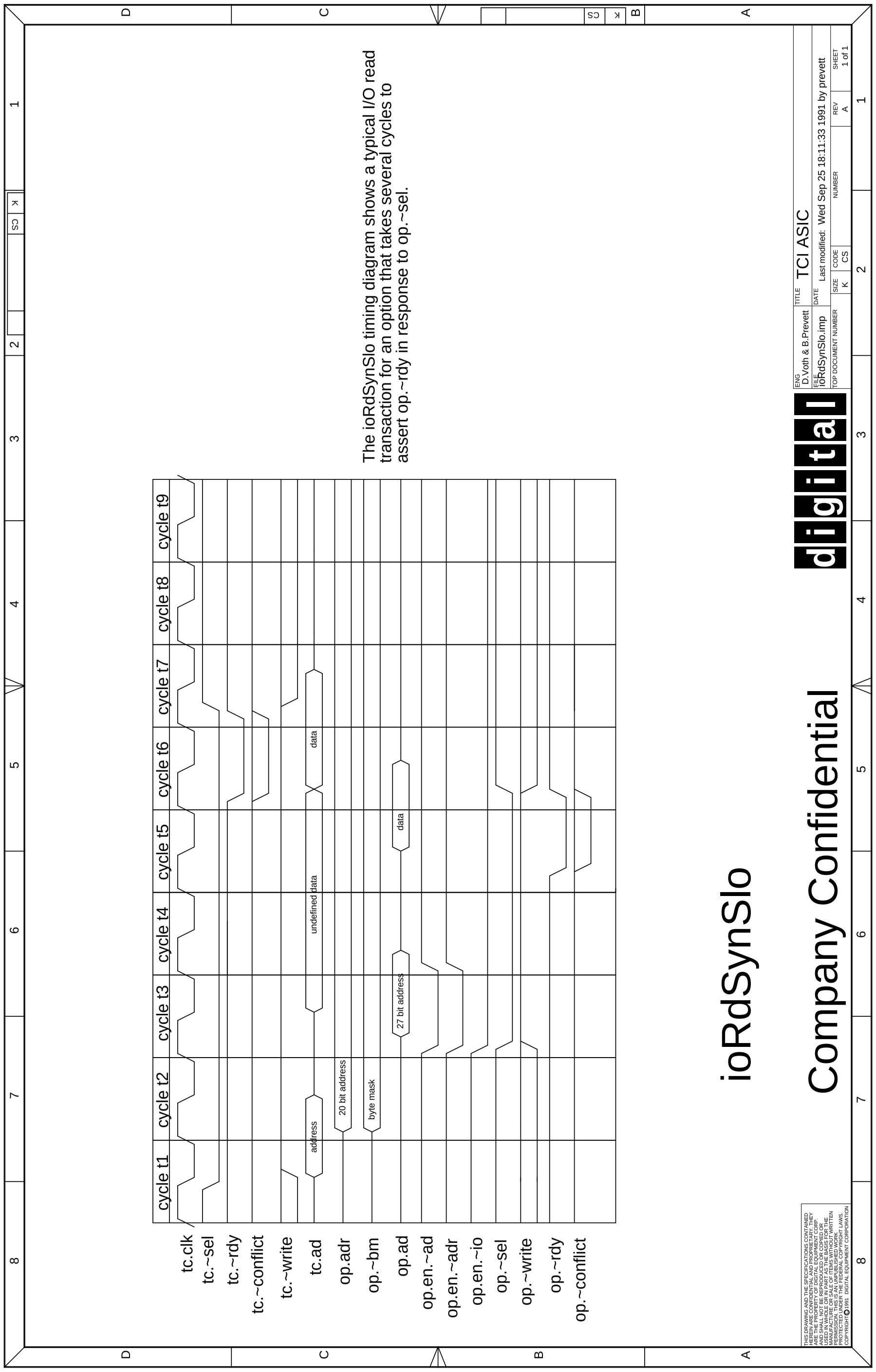
- TCI asserts tc.~rdy; tc.~conflict is asserted with tc.~rdy if conflict had been asserted.

Cycle t13:

- system deasserts tc.~sel in response to assertion of tc.~rdy; system may deassert tc.~write.
- TCI deasserts tc.~rdy, and tc.~conflict if applicable
- system tristates tc.ad

4.3 I/O Read with op.synchronous = 1

Figure 4 I/O Read Timing Diagram with op.synchronous = 1



The ioRdSynSlo timing diagram shows a typical I/O read transaction for an option that takes several cycles to assert op.~rdy in response to op.~sel.

cycle t1
cycle t2
cycle t3
cycle t4
cycle t5
cycle t6
cycle t7
cycle t8
cycle t9
tc.clk
tc.~sel
tc.~rdy
tc.~conflict
tc.~write
tc.ad
op.adr
op.~bm
op.ad
op.en.~ad
op.en.~adr
op.en.~io
op.~sel
op.~write
op.~rdy
op.~conflict
address
undefined data
data
20 bit address
byte mask
27 bit address
data
ioRdSynSlo
Company Confidential
digital
ENG D.Voth & B.Prevett
TITLE TCI ASIC
FILE ioRdSynSlo.imp
DATE Last modified: Wed Sep 25 18:11:33 1991 by prevett
TOP DOCUMENT NUMBER
SIZE K
CODE CS
NUMBER
REV A
SHEET 1 of 1
THIS DRAWING AND THE SPECIFICATIONS CONTAINED HEREIN ARE CONFIDENTIAL AND PROPRIETARY. THEY ARE THE PROPERTY OF DIGITAL EQUIPMENT CORP. AND SHALL NOT BE REPRODUCED OR COPIED OR USED IN WHOLE OR IN PART AS THE BASIS FOR THE MANUFACTURE OR SALE OF ITEMS WITHOUT WRITTEN PERMISSION. THIS IS AN UNPUBLISHED WORK PROTECTED UNDER THE FEDERAL COPYRIGHT LAWS. COPYRIGHT © 1991 DIGITAL EQUIPMENT CORPORATION
8
7
6
5
4
3
2
1
D
C
B
A

Cycle t1:

- system asserts tc.~sel and deasserts tc.~write to initiate an I/O read transaction to the option
- system drives I/O address onto tc.ad

Cycle t2:

- TCI loads address and byte mask into address/byte mask register on rising edge of tc.clk.
- TCI drives 20 bit address onto op.adr and byte mask onto op.~bm
- system tristates tc.ad

Cycle t3:

- TCI asserts op.~sel and and deasserts op.~write to signify an I/O read to the option interface.
- TCI drives full 27 bit address onto op.ad if option asserts op.en.~adr, op.en.~io, and op.en.~ad; the full tc.ad data (shifted address) is driven onto op.ad if op.en.~adr is deasserted.
- TCI now drives tc.ad with undefined data during option latency period.

Cycle t4:

- option deasserts op.en.~ad and op.en.~adr; op.ad is now tristated to accept the driven option data. op.ad should not float for more than one cycle; option must either drive op.ad or use op.en.~ad to allow the TCI to drive op.ad during this latency period.

Cycle t5:

- option asserts op.~rdy in response to op.~sel. op.~conflict may be asserted with op.~rdy in this same cycle if the option is committed to a DMA transaction.
- option drives op.ad with valid read data

Cycle t6:

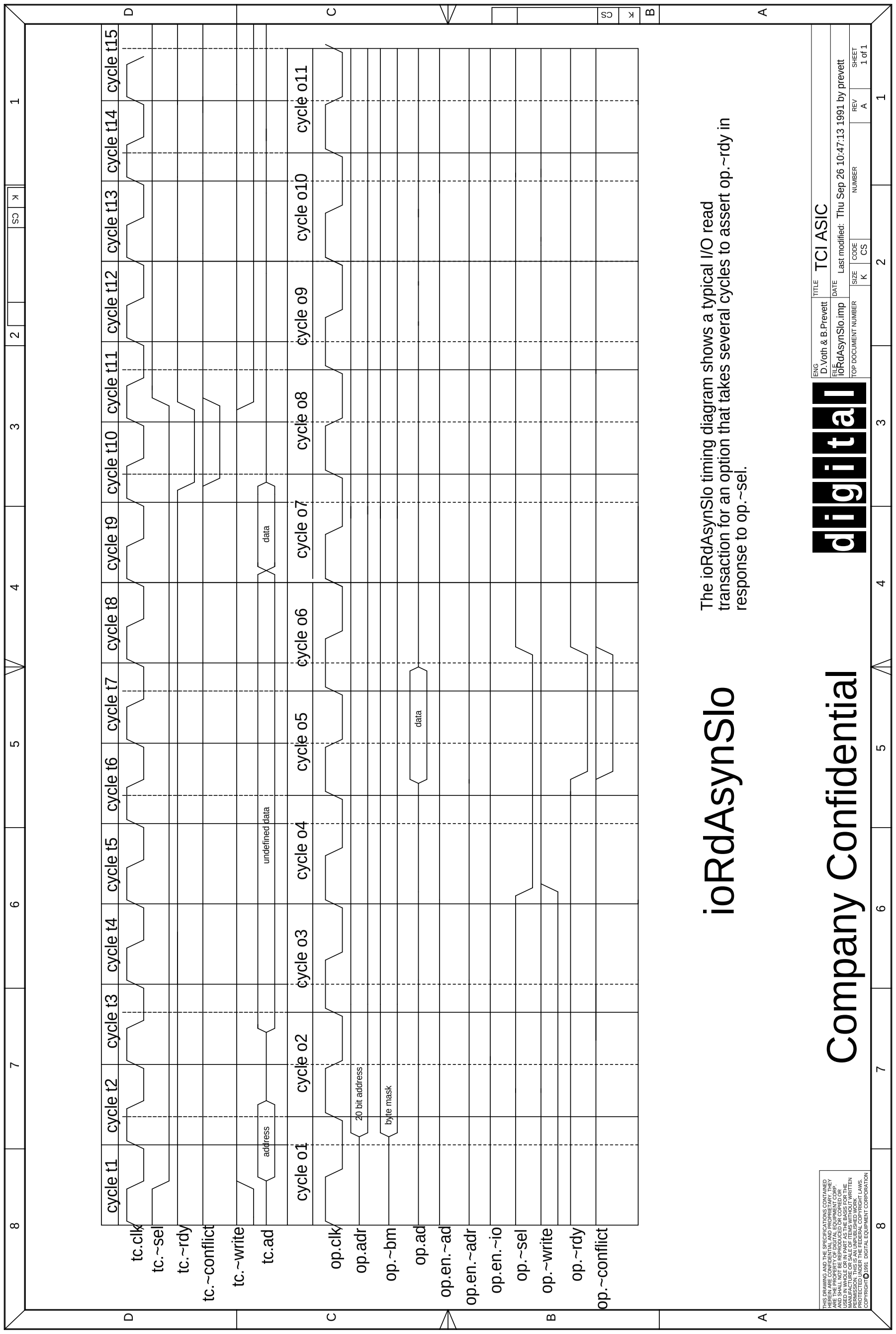
- TCI drives tc.ad with option data.
- TCI asserts tc.~rdy; tc.~conflict is asserted with tc.~rdy if op.~conflict had been asserted.
- option deasserts op.~rdy, and op.~conflict if applicable.
- TCI deasserts op.~sel.
- data on op.ad becomes invalid; op.ad must not be left floating. Option may assert op.en.~ad to allow TCI to drive op.ad or the option may drive op.ad directly.

Cycle t7:

- TCI deasserts tc.~rdy, and tc.~conflict if applicable
- system deasserts tc.~sel in response to assertion of tc.~rdy.
- TCI tristates tc.ad

4.4 I/O Read with op.synchronous = 0

Figure 5 I/O Read Timing Diagram with op.synchronous = 0



The ioRdAsynSlo timing diagram shows a typical I/O read transaction for an option that takes several cycles to assert op.~rdy in response to op.~sel.

Cycle t1:

- system asserts tc.~sel and deasserts tc.~write to initiate an I/O read transaction to the option
- system drives I/O address onto tc.ad

Cycle t2:

- TCI loads address and byte mask into address/byte mask register on rising edge of tc.clk.
- TCI loads tc.~sel on rising edge of tc.clk.
- TCI drives 20 bit address onto op.adr and byte mask onto op.~bm
- system tristates tc.ad

Cycle t3:

- TCI now drives tc.ad with undefined data during option latency period.

Cycle o4:

- TCI asserts op.~sel and deasserts op.~write to signify an I/O read transaction to the option interface.
- op.en.~ad and op.en.~adr are deasserted; op.ad is now tristated to accept the driven option data. op.ad should not float for more than one cycle; option must either op.ad or use op.en.~ad to allow the TCI to drive op.ad during this latency period.

Cycle o5:

- option asserts op.~rdy in response to op.~sel. op.~conflict may be asserted with op.~rdy in this same cycle if the option is committed to a DMA transaction.
- option drives op.ad with valid read data

Cycle o6:

- data on op.ad becomes invalid; op.ad must not be left floating. Option may assert op.en.~ad to allow TCI to drive op.ad or the option may drives op.ad directly.

- option deasserts op.~rdy, and op.~conflict if applicable.

- TCI deasserts op.~sel.

Cycle t10:

- TCI drives tc.ad with option data.

- TCI asserts tc.~rdy; tc.~conflict is asserted with tc.~rdy if conflict had been asserted.

Cycle t11:

- TCI deasserts tc.~rdy, and tc.~conflict if applicable
- system deasserts tc.~sel in response to assertion of tc.~rdy.
- TCI tristates tc.ad

5 DMA

The TCI DMA engine transfers data to or receives data from virtually contiguous buffer space in main memory. There are two ways to set up the TCI to perform DMA operations: simple DMA mode and intelligent controller DMA mode.

Simple DMA devices do not generate their own addresses. They move data to or from locations in memory via the pointers stored in the TCI ASIC. Typically, a simple DMA is started and completed by the processor. Interrupts can be used to instruct the processor to end DMA change or handle errors. For large, multipage DMA block transfers, the TCI can utilize a scatter/gather table that is set up in system memory by the device driver.

Intelligent controllers generate their own addresses and pass them into the receive FIFO. DMA receive operations are initiated by writing the DMA receive address and DMA write data words into the receive FIFO; data words are then written to system memory at the loaded address. DMA transmit operations are initiated by writing a transmit address into the receive FIFO. The transmit burstsize and word count are also loaded into the receive FIFO by the intelligent controller. This results in a TWC block of data being read from system memory into the transmit FIFO.

5.1 Transmit DMA

The TCI moves BURSTSIZE words of data at a time from memory into the transmit FIFO. Higher BURSTSIZEs give higher TURBOchannel bandwidth. The option-word SIZE field in the OCR may be byte, half-word or word. Words are unpacked into option-words which are available at the option interface.

The transmit FIFO is initialized by setting TQEN = 0. The alignment of the first option-word to be unpacked is set by writing TUPACK. TQEN is then set to a 1. The BURSTINIT field is set to TDP mod BURSTSIZE. If needed, the transmit word counter is then loaded with the number of words to be transferred. When the counter reaches zero, TWCEXPIRED is set and transmit DMA is disabled. The transmit word counter value must be greater than or equal to BURSTSIZE, may be from 1 word to 16Mwords, and must end on a word boundary. For DMA transfers that cross page or 2KB boundaries, the BURSTSIZE should be set to a power of 2 to help ensure that the page or 2KB boundary will not be crossed by a DMA burst. If transmit DMA ends on a non word-aligned boundary, the unaligned bytes must be written to the transmit FIFO along with a TUPACK value. The interrupts are cleared and any interrupts that need service are enabled via the Mask Register.

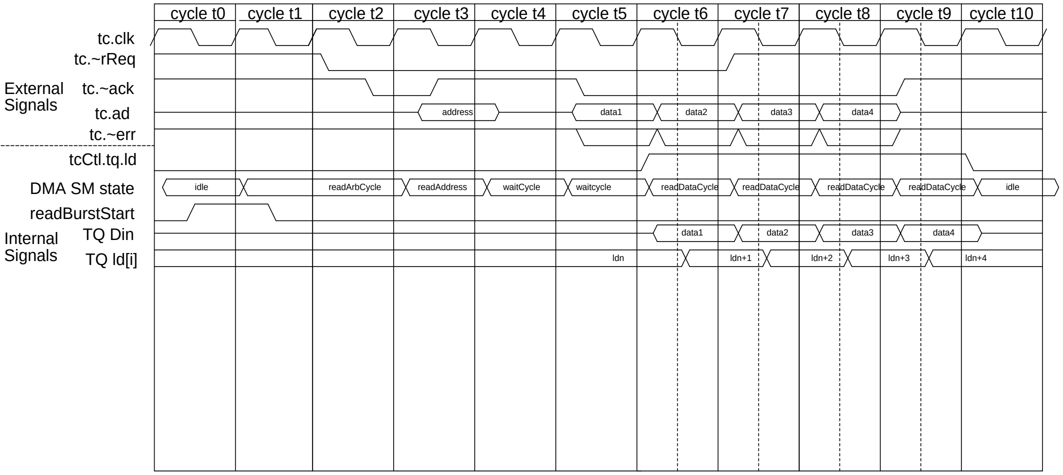
When a DMA transfer reaches a page boundary, the TCI will read (via a DMA read transaction) the next entry from the transmit scatter/gather table into the TDP. This entry should be the starting word address of the next physical page of the transfer. After loading the entry, the TCI increments the TSGP.

When set, the CONTINUOUS bit will keep the transmit FIFO filled with data. The DMA is then enabled and transmission begins. Note that DMA from memory will not progress unless

TDMAEN = 1, TQEN =1, TDMAERR = 0, TPLD = 0, and TWCEXPIRED = 0.

The **DMA Read Timing Diagram** shows the TCI reading four words from system memory and placing them into the transmit FIFO. The **Transmit DMA Timing Diagram** shows a simple DMA option reading these four words from the transmit FIFO. When op.en.~ad is asserted TCI drives op.ad. Asserting op.~trDack reads the next entry from the FIFO in next cycle. When the FIFO is empty, op.~trValid is deasserted.

Figure 6 DMA Read Timing Diagram



dmaRead4

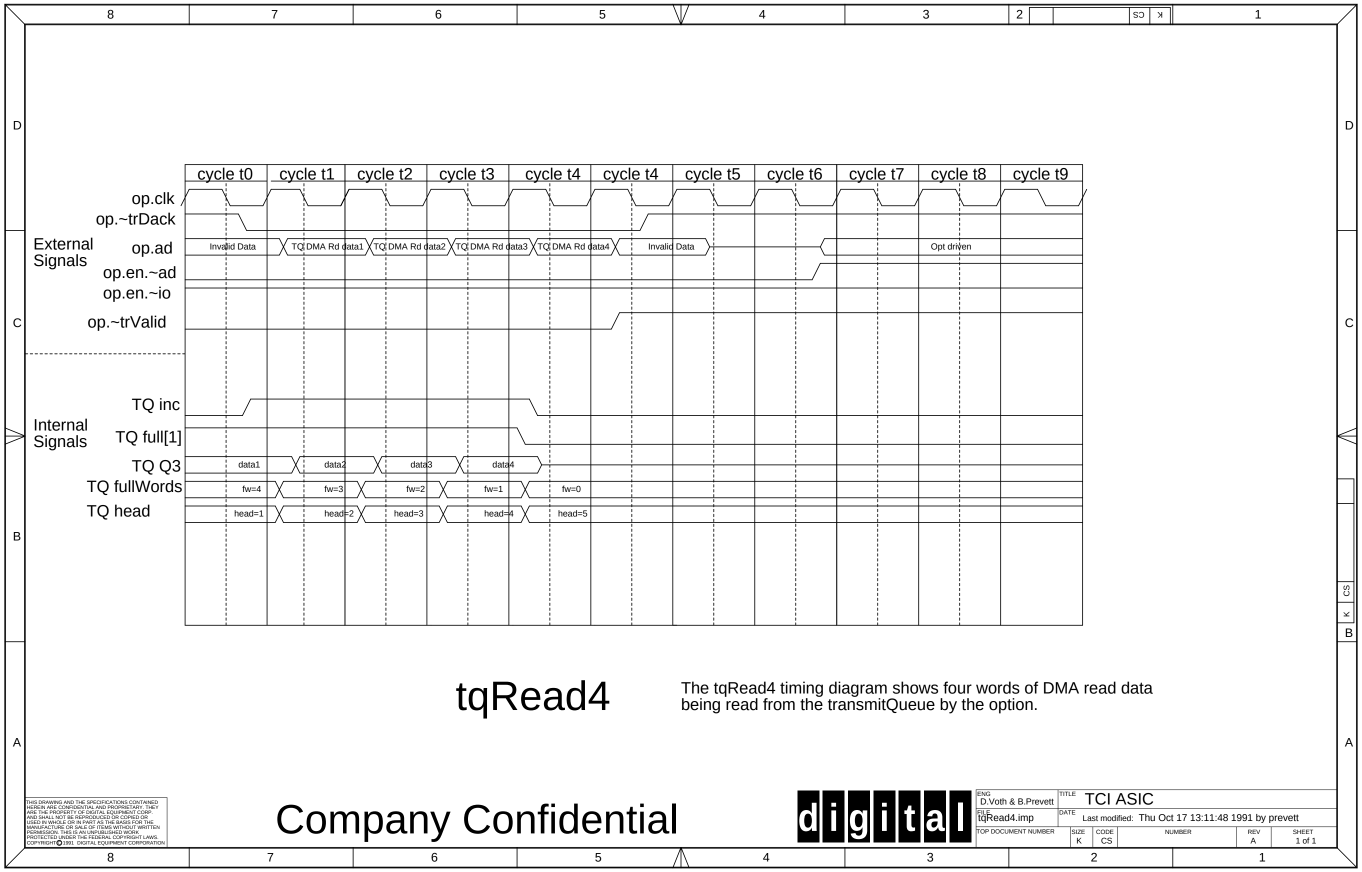
The dmaRead4 timing diagram shows a typical dma read burst transaction of 4 words; both TCI internal and TURBOchannel external signals are shown.

Company Confidential

digital

ENG D.Voth & B.Prevett	TITLE TCI ASIC				
FILE dmaRead4.imp	DATE Last modified: Thu Oct 17 14:48:53 1991 by prevett				
TOP DOCUMENT NUMBER	SIZE K	CODE CS	NUMBER	REV A	SHEET 1 of 1

Figure 7 Transmit DMA Timing Diagram



The tqRead4 timing diagram shows four words of DMA read data being read from the transmitQueue by the option.

5.2 Receive DMA

The TCI packs option-words into receive FIFO words and writes those words into memory. The DMA will burst the write data as long as there are RCVBURST words of valid data in the receive FIFO. The DMA engine breaks the write burst at (address mod 64*4) = 0 to prevent page crossing or transfer length errors. The option-word SIZE field in the OCR may be byte, half-word or word.

The receive FIFO is initialized by setting RQEN = 0. The alignment of the first option-word to be packed is set by writing RPACK. TQEN is then set to a 1.

The receive DMA pointer may be programmed to any physical memory word address. The TCI is only capable of writing words. If the last option-words are not word aligned, they will remain in the packer at the end of the transfer and not be written to memory. These can be read by programmed I/O and moved into memory at the end of the transfer. The receive FIFO has a counter that determines when to the flush the FIFO by writing the data words to system memory. This is useful at the end of block transfers when the number of data words left in the receive FIFO is less than the RCVBURST threshold.

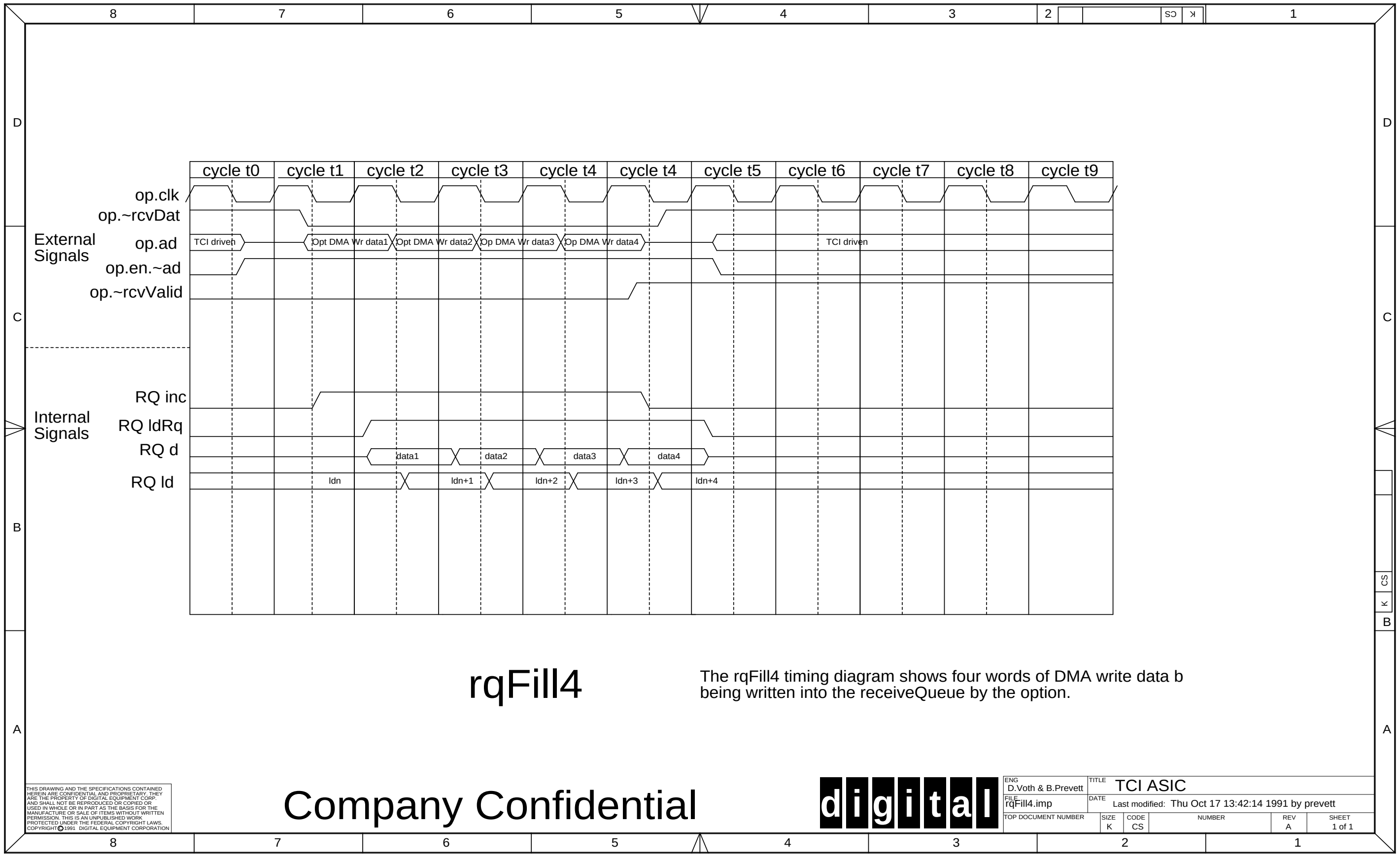
If required, the receive word counter can be programmed to the number of complete words to be transferred. The DMA to memory will stop when the counter reaches zero; RWCEXPIRED is then set. As in transmit DMA, the addresses of subsequent pages will be loaded from the scatter/gather table on page crossings.

Receive DMA to memory will not progress unless RDMAEN = 1, RQEN = 1, RDMAERR = 0, RPLD = 0, and RWCEXPIRED = 0.

The **Receive DMA Timing Diagram** shows a simple DMA device loading four words of DMA write data into the receive FIFO. The assertion of op.en.~ad tristates the op.ad drivers so that each option-word can be placed onto op.ad without contention. The signal op.~rcvDat places the option-word into receive FIFO. The deassertion of op.~rcvValid indicates that there are less than RCVTHRESHOLD free option-words remaining in the FIFO.

After synchronization the receive DMA state-machine will read the option-words out of the FIFO and issue a DMA write burst to system memory. The **DMA Write Timing Diagram** shows a four word DMA write into system memory.

Figure 8 Receive DMA Timing Diagram



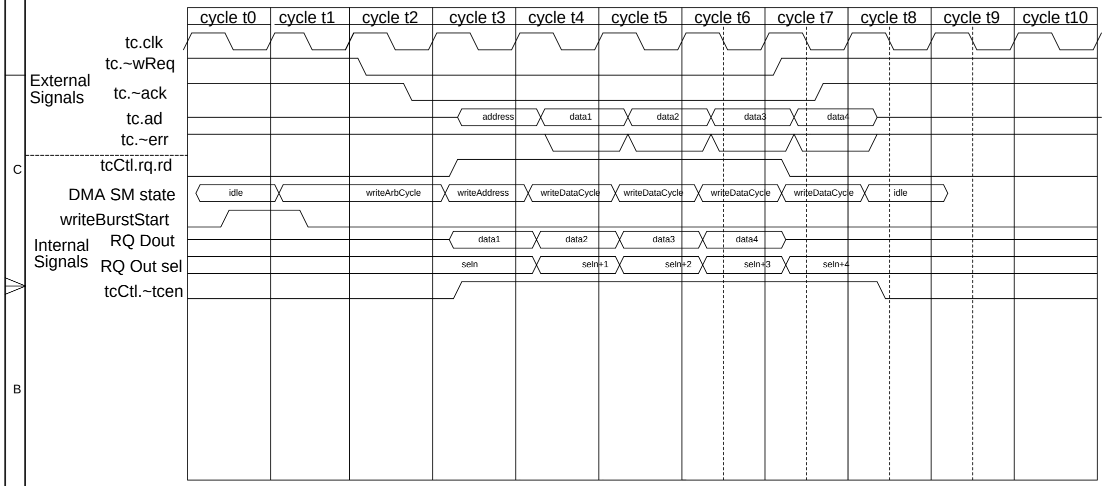
The rqFill4 timing diagram shows four words of DMA write data b being written into the receiveQueue by the option.

Company Confidential

digital

ENG D.Voth & B.Prevett	TITLE TCI ASIC					
FILE rqFill4.imp	DATE Last modified: Thu Oct 17 13:42:14 1991 by prevett					
TOP DOCUMENT NUMBER	SIZE K	CODE CS	NUMBER	REV A	SHEET 1 of 1	

Figure 9 DMA Write Timing Diagram



dmaWrite4

The dmaWrite4 timing diagram shows a typical dma write burst transaction or 4 words; both TCI internal and TURBOchannel external signals are shown.

Company Confidential

digital

ENG D.Voth & B.Prevett	TITLE TCI ASIC				
FILE dmaWrite4.imp	DATE Last modified: Thu Oct 17 14:48:02 1991 by prevett				
TOP DOCUMENT NUMBER	SIZE K	CODE CS	NUMBER	REV A	SHEET 1 of 1

5.3 DMA Interrupt Handling

When an interrupt occurs the CPU examines the OIR and decides on a course of action. It is usually a good idea to first turn off the DMA via TDMAEN or RDMAEN before processing the interrupt so that memory will not be inadvertently updated and the pointers will be stable.

After the interrupt is handled, the DMA should be restarted by first clearing the interrupt(s) and then enabling the DMA (TDMAEN, RDMAEN).

5.3.1 Errors

If an error occurs during a DMA, an interrupt is set and the DMA causing the error is halted. The DMA pointer will contain the address associated with the error.

5.4 Simple DMA Examples

5.4.1 Simple Transmit DMA Examples

Consider a TURBOchannel HDTV frame buffer that uses the TCI and the simple DMA transfer mechanism. For this example, 30KB of image data must be transferred to the frame buffer; this data is located in virtually contiguous system memory address space beginning at virtual address vAddr(0). The memory system page size for this system is 4KB. Since this transfer will cross page boundaries, the scatter/gather function will be used.

To setup the DMA block transfer, the device driver must first compute the physical address for the start address, pAddr(0), and then compute the physical page addresses of the subsequent pages, pAddr(n). pAddr(0) is located 3KB into a page. The required physical page computations are shown below:

Start Address	vAddr(0)	==>	pAddr(0)
Page Boundary #1	vAddr(0) + 1KB	==>	pAddr(1)
Page Boundary #2	vAddr(0) + 1KB + 1*4KB	==>	pAddr(2)
Page Boundary #3	vAddr(0) + 1KB + 2*4KB	==>	pAddr(3)
Page Boundary #4	vAddr(0) + 1KB + 3*4KB	==>	pAddr(4)
Page Boundary #5	vAddr(0) + 1KB + 4*4KB	==>	pAddr(5)
Page Boundary #6	vAddr(0) + 1KB + 5*4KB	==>	pAddr(6)
Page Boundary #7	vAddr(0) + 1KB + 6*4KB	==>	pAddr(7)
Page Boundary #8	vAddr(0) + 1KB + 7*4KB	==>	pAddr(8)

The device driver then loads pAddr(1) through pAddr(8) into a scatter/gather table that is allocated in system memory. The table created is an eight entry table of physically contiguous words pAddr(1) through pAddr(8) located at physical address pSGAddr.

The device driver is now ready to initiate the required DMA transfer. Through programmed I/O to the TCI register file, the device driver:

* sets the Transmit DMA Pointer (TDP) = pAddr(0)

* sets the Transmit DMA Scatter/Gather Pointer (TSGP) = pSGAddr
* sets the Transmit DMA Word Counter (TWC) = 30KB/4 = 7.5KB
* configures the Option Interrupt Mask Register (OIMR) to interrupt when TWC reaches 0
* configures the Option Control Register (OCR) by :
  * setting the TQEN, TDMAEN, and CONTINUOUS bits
  * setting SIZE = word, BURSTSIZE = 8, and BURSTINIT = TDP mod BURSTSIZE

The TCI is now programmed for the DMA block transfer. Once there are BURSTSIZE free words in the transmit FIFO, the TCI will initiate a BURSTINIT word DMA read transaction from address pAddr(0). Subsequent DMA read bursts will be of size BURSTSIZE. As each word is received by the TCI, it is loaded into the transmit FIFO, the TDP is incremented and the TWC is decremented. As the transmit FIFO fills up, the option side signal op.~trValid asserts, signifying valid data in the transmit FIFO. The option then reads the data from the transmit FIFO by asserting the op.~trDack signal. The TCI will issue DMA read bursts as long as there are BURSTSIZE free words in the transmit FIFO.

When the TDP reaches a page boundary, the TCI must fetch the next page address pAddr(1) from the scatter/gather table. The TCI issues a one-word DMA read to the address pSGAddr, loads the data, pAddr(1), into the TDP, and increments the TSGP to point to the next entry in the scatter/gather table. DMA now continues from address pAddr(1). In this way, the pages beginning at pAddr(2) through pAddr(7) are transferred to the transmit FIFO and then to the option interface.

When pAddr(8) is loaded into the TDP, DMA continues until the word count reaches zero. If TWC mod BURSTSIZE <> 0, at this end condition, a BURSTSIZE unit of data is still DMA read from system memory; however only the datawords that correspond to the non-zero word count will be written into the receive FIFO.

The transmit DMA is now complete. The signal tc.~int will assert, signifying that the TWC has decremented to zero and that the DMA block transfer has been completed. The device driver clears the OCR TDMAEN bit to disable DMA and then reads the Option Interrupt Register (OIR) to determine that the DMA block transfer has indeed completed successfully.

To explore data alignment issues, consider a second simple transmit DMA example: a UART that DMA reads characters from system memory and transmits them out a serial line. This UART is byte oriented, so the OCR SIZE field is set to BYTE. For this example transfer, there are 97 character bytes to be transmitted; the byte ordering scheme is little-endian. The TCI transmit DMA engine allows for non-word alignment for the data bytes of the first word. If the data block to be transferred does not end on a word boundary, then the remainder bytes must be programmed I/O written into the transmit FIFO along with the required TUPACK value.

The character data begins at system memory location pAddr(0) + 1, where pAddr(0) is a word aligned address. The character bytes are located in the system memory buffer as follows:

byte(1) ==> pAddr(0) + 1
byte(2) ==> pAddr(0) + 2
byte(3) ==> pAddr(0) + 3
byte(4) ==> pAddr(0) + 1*4 + 0
byte(5) ==> pAddr(0) + 1*4 + 1

byte(6) ==> pAddr(0) + 1*4 + 2
.
.
.
byte(94) ==> pAddr(0) + 23*4 + 2
byte(95) ==> pAddr(0) + 23*4 + 3
byte(96) ==> pAddr(0) + 24*4 + 0
byte(97) ==> pAddr(0) + 24*4 + 1

The device driver initiates the required transmit DMA by:

* setting the TDP = pAddr(0)
* setting the TWC = (first 95 data bytes + 1 undefined byte at pAddr(0) + 0) / 4 = 24
* configuring the OIMR to interrupt when the TWC reaches 0
* configuring the MISC by setting BIGENDIAN to 0
* configuring the OCR first by:
    * resetting TQEN = 0 to clear the transmit FIFO
    * setting the TUPACK = 01.

and then by:
    * setting the TDMAEN and CONTINUOUS bits
    * setting the BURSTINIT = TDP mod BURSTSIZE
    * setting the SIZE = byte and BURSTSIZE = 4.

The TCI is now programmed for the DMA block transfer. Once there are BURSTSIZE free words in the transmit FIFO, the TCI will initiate a four word DMA read transaction from pAddr(0). The first word loaded into the transmit FIFO will be configured as follows:

byte(3) | byte(2) | byte(1) | undefined byte from pAddr(0) + 0

The next 23 data words are then written into the transmit FIFO as space allows. When op.~trValid first asserts, the first bytes will be read out of the queue with the op.~trValid signal. The first bytes to be read out will be: byte(1), byte(2), byte(3), byte(4), etc. The undefined data will not be read out because the unpacking counter was loaded to ignore it.

After the first 95 characters have been transferred into the transmit queue, the TWC will reach 0 and tc.~int will assert. Due to the two data bytes left over, the device driver must now resolve the end condition. The device driver checks the TFE bit to determine when the transmit FIFO is empty. Due to the interrupt latency, the transmit FIFO will most likely be empty (TFE=0) by the time the TFE bit is checked. It then loads the byte-unpacker with TUPACK = 10 and sets TQEN = 0. The device driver then programmed I/O writes the last two bytes into the transmit FIFO. These last two bytes will be read out of the transmit FIFO by the option.

If necessary, the device driver can confirm that all characters have indeed be read out of the transmit queue to UART by polling on the OIR TFE (transmit FIFO empty) bit.

An alternative approach for non-word option devices is to align the bytes or half-words to word boundaries by setting the option wordsize to word and then copying the byte or half-word data to the low byte or low half-word of each addressed word. In this way three or two bytes per system memory word will be wasted, but there will be no byte misalignment cases to handle.

5.4.2 Simple Receive DMA Examples

For this example, consider a TURBOchannel IPI disk controller the uses the TCI in a simple DMA transfer mechanism. The controller must transfer 6KB of data from the option on-board buffer to system memory beginning at location vAddr(0). This transfer will one cross page boundary; therefore, the scatter/gather function will be used.

As in the simple transmit DMA example, the device computes the beginning physical address pAddr(0) and the page address pAddr(1). pAddr(1) is loaded into a scatter/gather table that begins at physical address pSGAddr.

The device driver is now ready to initiate the DMA transfer; through programmed I/O to the TCI register file, the device driver:

* sets the Receive DMA Pointer (RDP) = pAddr(0)
* sets the Receive DMA Scatter/Gather Pointer (RGSP) = pSGAddr
* sets the Receive DMA Word Counter (RWC) = 6KB/4 = 1.5KB
* configures the Option Interrupt Mask Register (OIMR) to interrupt when the RWC reaches 0
* configures the Miscellaneous Register (MISC) by:
    * setting the RCVTHRESHOLD = 001
    * setting RCVBURST = 10 (4 valid words)
    * setting RCVFLSHCNT = 00 (16 TURBOchannel cycles)
* configures the Option Control Register (OCR) by:
    * setting the RQEN and RDMAEN bits
    * setting SIZE = word
    * setting the GPO[0] bit.

The TCI is now programmed for the DMA block transfer. The signal op.gpo[0] asserts, signaling to the IPI controller to begin the receive DMA transfer. The IPI controller writes data words into the receive FIFO by depositing data onto op.ad and asserting the op.~rcvDat signal. If the FIFO fills up past the RCVTHRESHOLD limit, the op.~rcvValid signal deasserts. In this example, op.~rcvValid asserts when there is less than two free words in the receive FIFO. Option data loading into the receive FIFO delays until the op.~rcvValid reasserts, signaling free room in the receive FIFO.

Unlike transmit DMA, receive DMA does not use the BURSTSIZE parameter. Instead, the TCI asserts tc.~wReq and bursts data as long as there are RCVBURST words of valid data in the receive FIFO. When the receive FIFO is drained, the DMA write burst halts; a new DMA write burst begins when RCVBURST words of new valid data are written into the receive FIFO by the option. With this scheme, the resultant DMA write bursts can be from RCVBURST words to 64 words long; the burst length depends upon the rate at which the option can load the receive FIFO and the rate at which the TURBOchannel can drain the receive FIFO. The write bursts are broken at (address mod 64*4) to prevent page crossing, 2K boundary crossing, and transfer length errors.

When the RDP reaches the page boundary, the TCI fetches the next page address pAddr(1) from the scatter/gather table. The RDP is loaded with pAddr(1) and receive DMA continues until the RWC reaches 0 or is less than RCVBURST. If the RWC has decremented to a nonzero value that is less than the RCVBURST size, data in the receive FIFO is flushed through the flush counter until RWC = 0. The signal tc.~int will assert, signaling the completion of the DMA block transfer.

To examine data alignment issues for receive DMA, again consider the UART example. The UART has now received 95 byte characters from the serial line and must write them into a system memory buffer. The receive DMA engine allows for non-word alignment for the databytes of the first word, and the data block can end on a non-word boundary.

For receive DMA, the RWC will not be used. The device driver has loaded the RDP with a system memory buffer location, pAddr(0), or the RDP may be left over from a previous receive DMA operation and would point to the next available word in the system memory buffer. To enable the transfer, the device driver:

* sets the OIMR to interrupt when op.~gpi[2] is asserted
* configures the OCR first by:
    * resetting RQEN = 0 to clear the receive FIFO
    * setting the RPACK = 11
  and then by:
    * setting the RDMAEN bit
    * setting SIZE = byte.

The TCI is now programmed to transfer the UART characters into system memory. The UART loads the character bytes into the receive FIFO. The first word of the first DMA write burst contains valid data, byte(1), in the third byte of the word, since RPACK was set to three. The next 92 characters are transferred to system memory via DMA write bursts. The last two characters, byte(94) and byte(95) remain in the byte packing counter of the receive FIFO. The UART asserts op.~gpi[2] to signal to the device driver that a new set of serial line characters has been written into the system memory buffer. In response to the interrupt, the device examines the OIR and determines that the receive FIFO is empty (RFE = 1) but that characters are left over in the byte packer (RPACK2 = 10). The device driver then uses an I/O read transaction to read the last two bytes.

5.5 Intelligent Controller Reads and Writes

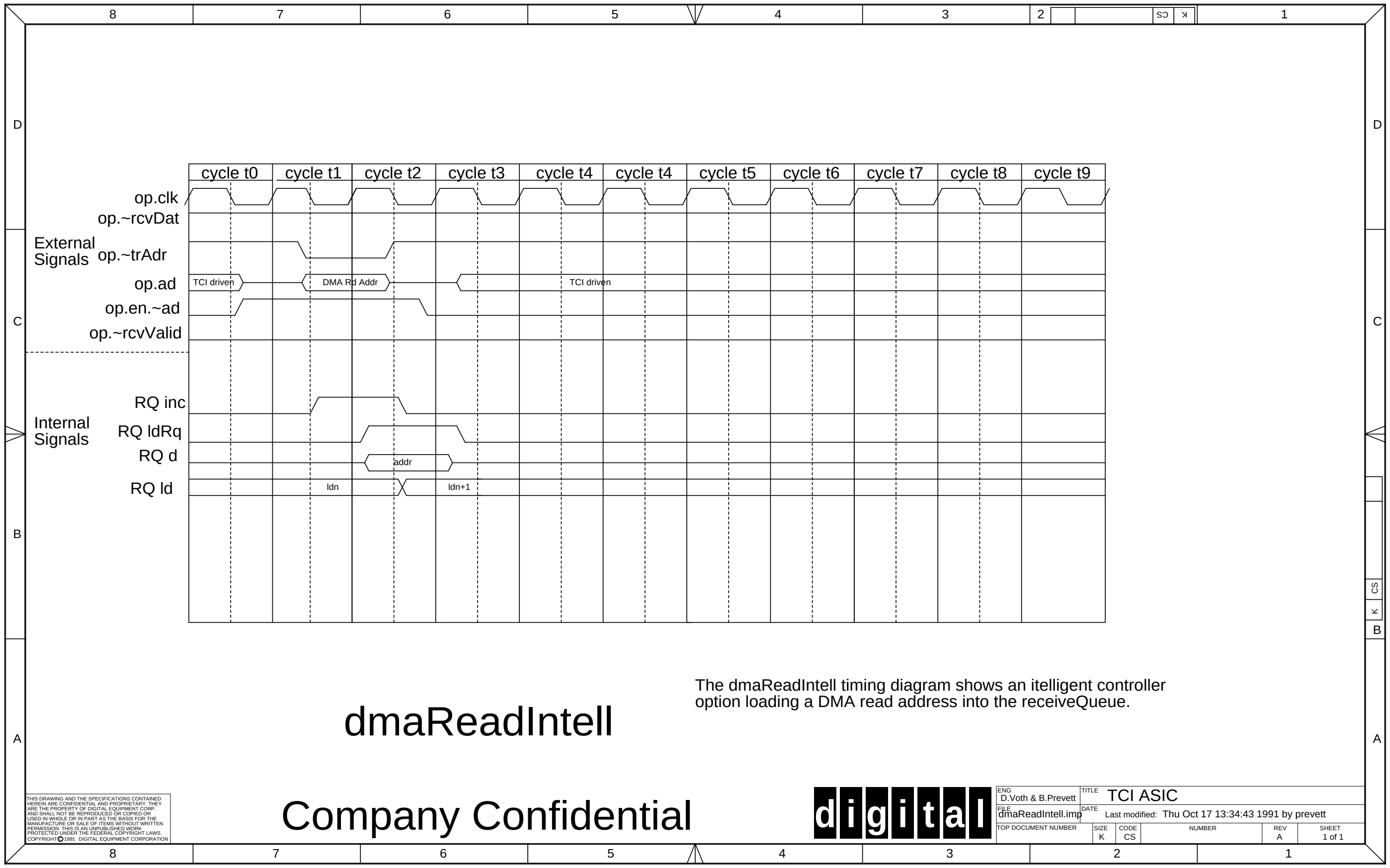
Unlike simple DMA options, intelligent controllers generate their own addresses and execute reads and writes from memory. For transmit DMA, the intelligent controller loads the BURSTSIZE and TWC into the receive FIFO. This is done by asserting op.~rcvAdr and op.~trAdr while placing the BURSTSIZE onto op.ad[31..24] and the TWC onto op.ad[23..0]. The intelligent controller next loads a DMA transmit address into the receive FIFO. Once this transmit address reaches the head of the FIFO, it is written into the TDP and the transmit DMA begins. Transmit DMA continues until the TWC reaches 0. The BURSTSIZE is not restricted to a power of two; however, the address used for the DMA read must not generate bursts that cross a 2Kbyte page boundary. For this reason it is often simplest to use a power of two BURSTSIZE and make (TDP mod BURSTSIZE) = 0. When in intelligent controller mode, BURSTINIT is automatically set to BURSTSIZE for transmit DMA. The **Intelligent Controller DMA Read Timing Diagram** shows the loading of a transmit pointer into the receive

FIFO. The loading of the BURSTSIZE and TWC is similar, except for the values of the op.~rcvAdr and op.~trAdr lines.

For receive DMA, the intelligent controller loads a DMA receive address into the receive FIFO. Once this address reaches the head of the receive FIFO, it is written into the RDP. Subsequent option data words are then written to system memory beginning at the address pointed to by the RDP. The RDP is incremented as each word is written into system memory. The TCI will burst the writes into system memory as long as the number of words in the receive FIFO is greater than or equal to RCVBURST. A selectable counter is used that flushes the receive FIFO when the counter reaches zero. The **Intelligent Controller DMA Write Timing Diagram** shows the loading of a receive pointer and four words of DMA write data into the receive FIFO.

Intelligent controllers may have addresses of arbitrary widths; for example a particular Ethernet controller generates a 16 bit address that may be extended to 24 bits. To set a specific address width, the pointer mask register should be programmed. This mask indicates which address range the option will supply and which will be fixed in the DMA pointer registers. The address word loaded into the receive FIFO must be loaded on one option clock cycle; there is no packing for DMA addresses. Intelligent controller data, however, may be loaded into the receive FIFO a byte, a double-byte, or a word at a time. Since the intelligent controller does not have access to the data alignment mechanisms in the CSRs, the data must begin and end on word boundaries.

Figure 10 Intelligent Controller DMA Read Timing Diagram



dmaReadIntell

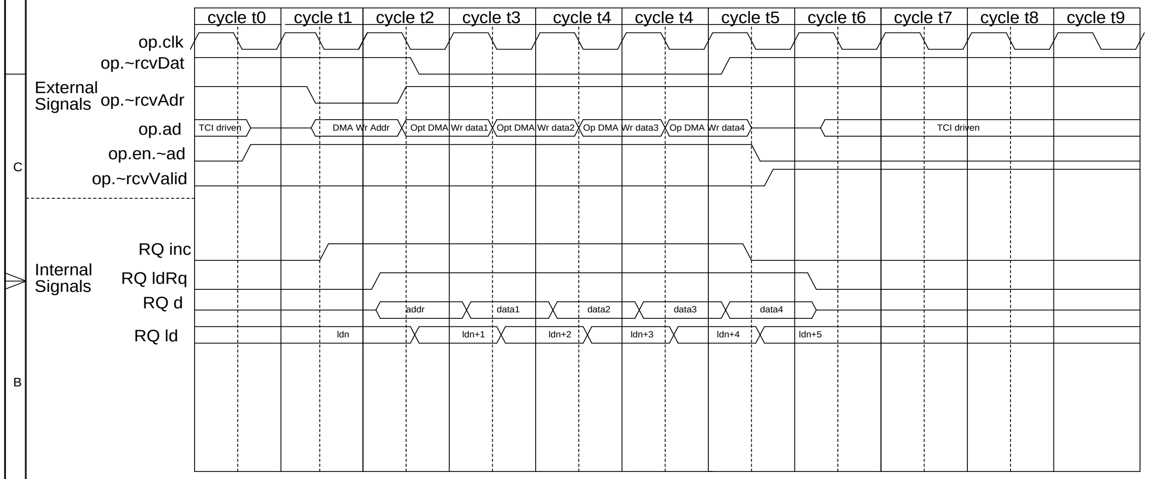
The dmaReadIntell timing diagram shows an itelligent controller option loading a DMA read address into the receiveQueue.

Company Confidential

digital

ENG	TITLE				
D.Voth & B.Prevett	TCI ASIC				
FILE dmaReadIntell.imp	DATE Last modified: Thu Oct 17 13:34:43 1991 by prevett				
TOP DOCUMENT NUMBER	SIZE K	CODE CS	NUMBER	REV A	SHEET 1 of 1

Figure 11 Intelligent Controller DMA Write Timing Diagram



dmaWriteIntell

The dmaWriteIntell timing diagram shows an itelligent controller option loading a DMA write address and four words of DMA write data into the receiveQueue.

Company Confidential

digital

ENG D.Voth & B.Prevett	TITLE TCI ASIC				
FILE dmaWriteIntell.imp	DATE Last modified: Thu Oct 17 13:43:05 1991 by prevett				
TOP DOCUMENT NUMBER	SIZE K	CODE CS	NUMBER	REV A	SHEET 1 of 1

5.6 Intelligent Controller DMA Examples

5.6.1 Intelligent Controller Transmit DMA Example

For this example, consider a Token Ring option that uses an intelligent controller. In the intelligent controller DMA model, the option controller generates its own DMA address. This controller generates 16-bit addresses; therefore, the DMA Pointer Mask (DPM) must be used to supply the upper order address bits of the system memory buffer.

There is 2KB of packet data in a system memory buffer that must be transferred to the token ring controller and onto the network. The device driver:

* writes the upper-order address bits into the TDP
* sets an address mask into the DPM that selects the upper 16 address bits from the TDP and the lower 16 address bits as supplied by the option
* configures the OIMR to interrupt when op.~gpi[1] is asserted
* configures the OCR by:
    * setting the TQEN, TDMAEN, and CONTINUOUS bits
    * setting SIZE = word
    * setting GPO[1] bit.

The TCI option side signal op.gpo[1] asserts, signaling to the token ring option that there is a group of packets to be transmitted. The option controller, in response to the assertion of op.gpo[1], begins the transmit DMA.

The option controller first writes the BURSTSIZE and TWC values into the receive FIFO. The option controller then writes the transmit address into the receive FIFO by placing the 16 bit address onto op.ad and asserting the op.~trAdr signal for one op.clk cycle. When the head of the receive FIFO reaches the BURSTSIZE and TWC values, the BURSTSIZE and BURSTINIT fields of the OCR and the TWC are updated. When the head of the receive FIFO reaches the transmit address, this address it is combined with the TDP as defined by the DPM and written back into the TDP. The DMA engine now issues a BURSTSIZEd DMA read burst at this address. As the data words are received by the TCI, they are written into the transmit FIFO. The presence of valid data words in the transmit FIFO asserts the op.~trValid signal. The option reads out the data words by asserting the op.~trDack signal. When the option has received the full 2KB of packet data, it asserts the op.~gpi[1] signal. The signal tc.~int asserts, signaling to the device driver that the 2KB transfer has completed.

Intelligent controllers may use the byte unpacker of the transmit queue for byte or half-word data for reading data out of the transmit FIFO, but the transmit address supplied to the receive queue will not go through the byte unpacker. The transmit address must be supplied on one op.clk cycle.

5.6.2 Intelligent Controller Receive DMA Example

The Token Ring Controller will again be used for this example. The option has received 2KB of packet data from the token ring network; this data must be transferred to system memory buffer via receive DMA. The device driver must configure the TCI for intelligent controller receive DMA by:

* writing the upper-order address bits of the system memory buffer into the RDP
* setting an address mask into the DPM that selects the upper 16 address bits from the RDP and the lower 16 address bits as supplied by the option
* configuring the Miscellaneous Register (MISC) by:
    * setting the RCVTHRESHOLD = 001
    * setting the RCVBURST = 10 (4 valid words)
    * setting the RCVFLSHCNT = 00 (16 TURBOchannel cycles)
* configuring the OIMR to interrupt when op.~gpi[1] is asserted
* configuring the OCR by setting the RQEN and RDMAEN bits.

The option controller writes the receive address into the receive FIFO by placing the 16 bit address onto op.ad and asserting the op.~rcvAdr signal for one op.clk cycle. The option then writes the DMA write data into the receive FIFO by placing the data onto op.ad and asserting the op.~rcvDat signal.

When the head of the receive FIFO reaches this address it is combined with the RDP as defined by the DPM and written back into the RDP. The DMA write data is written into system memory from the receive FIFO. As the receive FIFO is drained by the DMA write bursts, the option controller writes more data into the receive FIFO. As in the simple receive DMA example, remaining data words are flushed out of the receive FIFO with the flush counter mechanism. When the option has written all of the 2KB packet data into the receive FIFO, it asserts the op.~gpi[1] signal, triggering a TURBOchannel interrupt to signal the completion of the transfer into the receive FIFO. The device driver will then poll the OIR RFE bit to determine when the last data words have been written out of the receive FIFO and into the system memory buffer.

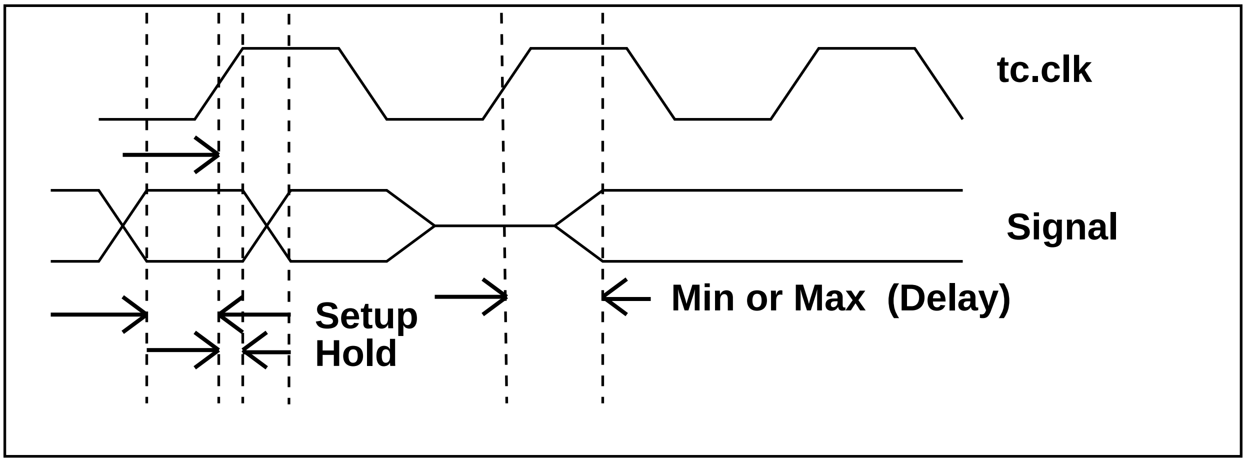
Intelligent controllers may use the byte packer of the receive queue for byte or half-word data, but the byte packer is not used for the receive address. The receive address must be supplied on one op.clk cycle.

6 Electrical

The table below lists the timing parameters that are referenced to the rising edge of tc.clk. Timing is measured between signal Vih,Vil to mid-point of clock.

Signal		Source	Min	Max	Setup	Hold
tc.clk	cycle time	system	40	80		
tc.clk	pulse width	system	15			
tc.ad		system			5	2
tc.~sel, tc.~write		system			13	2
tc.~ack, tc.~err		system			13	2
tc.~err, tc.~reset		system			13	2
tc.ad		TCI	3	34		
tc.ad to 3-state		TCI	3	22		
tc.~rdy, tc.~conflict		TCI	1	12		
tc.~int		TCI	1	12		
tc.~wReq, tc.~rReq		TCI	1	7		
op.adr		TCI	1	23		
op.~bm		TCI	1	23		

Figure 12 tc.clk - Signal Timing Diagram

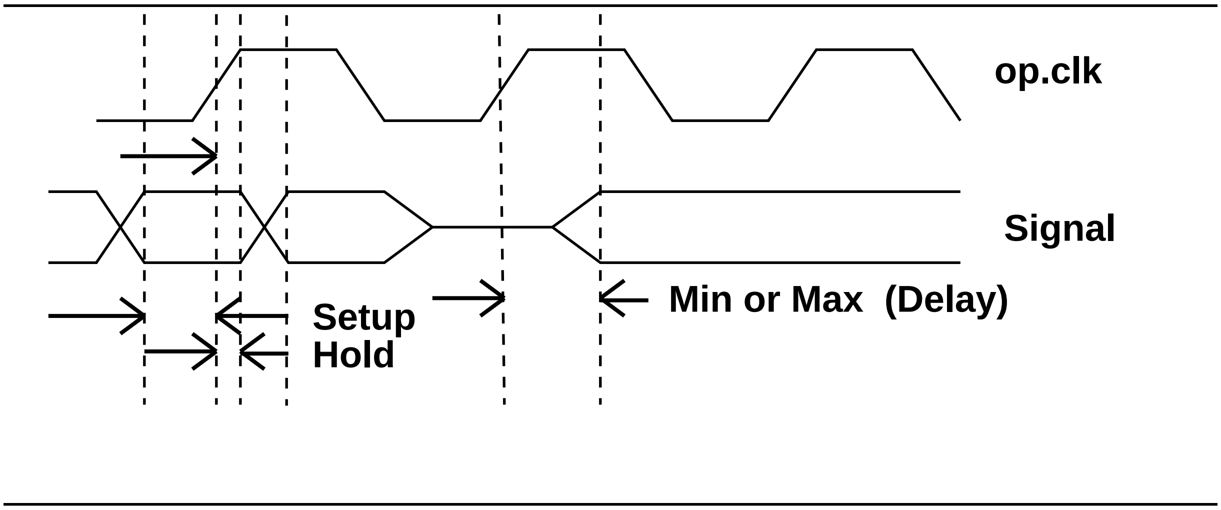


6.0.1 Option Signal Timing

The table below lists the timing parameters that are referenced to the rising edge of op.clk, Except for op.~bm and op.~adr which are asserted with tc.clk. Timing is measured between signal Vih,Vil to mid-point of clock (except for the enable lines)..

Signal	Source	Min	Max	Setup	Hold
op.clk cycle time	option	TBD	1000		
op.clk pulse width	option	15	950		
op.ad	option			10	0
op.~rcvAdr, op.~rcvDat	option			10	0
op.~rdy, op.~write	option			10	0
op.~trDack,op.~rcvAdr	option			10	0
op.~conflict	option			10	0
op.ad	TCI	1	23		
op.gpo, op.~sel	TCI	1	23		
op.~rcvValid,	TCI	1	23		
op.~trValid TCI	TCI	1	23		
op.en.~.ad to op.ad	TCI	1	23		
op.en.~io to op.ad	TCI	1	23		
op.en.~adr to op.ad	TCI	1	23		
op.~gpi	option	Synchronized, no Relationship			

Figure 13 op.clk Signal Timing Diagram



The table below gives the value of the test load and the input capacitance. If option loads are greater than those specified in this specification then associated Max delay value must be derated by .15ns/pf.

6.0.2 Loading

	Test	Input
Signal(s)	Load	Capacitance(Max)
tc.ad	180pF	10pF
Other outputs or I/O	65pF	10pF
Any Input		10pF

6.1 DC Electrical Characteristic

Parameter	Value
Voh	2.4V
Vol	0.5V
Vih	2.0V
Vil	0.8V
Idd	TBD ma

6.1.1 Absolute Maximum Ratings

Parameter	Value	Duration
Vin (Max)	VDD+1	Indefinite
Vin (Min)	VSS-1	Indefinite
Vin (Min)	VSS-2.5	20ns
Vdd (Max)	+7V	Indefinite
Storage Temp. (Min)	-65C	Indefinite
Storage Temp. (Max)	+150C	Indefinite
Operating Temp. (Min)	0C	Indefinite
Operating Temp. (Max)	TBD	Indefinite