

§ 1 INTRODUCTION

§ 1.0 PURPOSE

The purpose of the this project is to create a demonstration vehicle (“Reference Design”) for synthesis and schematic FPGA implementation comparisons—not necessarily to compare tools but to compare implementation methodologies.

§ 1.1 REFERENCE DESIGN ATTRIBUTES

The Reference Design should approximate current, real-world designs; therefore, it should represent a realistic mix of datapath and random logic and should employ a representative distribution of typical onchip resources such as RAM, ROM, arithmetic, and random functions. It should also make use of both global and local routing resources. The Reference Design should be approximately 10K gates.

§ 1.2 LIMITATIONS

Given the bandwidth constraints of the creators (who have real-world jobs), the Reference Design is limited to being a reasonable facsimile of a real-world design, not a full-blown, competitive, proprietary product. (Otherwise, we’d make and sell it instead of writing about it.) Regarding the realism of the interfaces, it is the intent of the creators *not* to use or replicate existing standards, since that may compromise existing IP.

§ 1.3 TARGET

The chosen FPGA target for this exercise is a Xilinx XC4013XLPQ240-1, a 13K-gate device in which 10K gates represents 77% utilization—an appropriate 23% margin. The PQ240 provides 192 I/O or (at 77% utilization) about 148 pins—sufficient for several, large, standard interfaces. The -1 speed grade should be adequate for system performance in the 33 to 66 MHz range.

§ 1.4 TOOLS:

Schematics: Viewlogic Workview Office (Xilinx Alliance 1.4)
Synthesis: Synopsys FPGA Express 2.0 (Xilinx Alliance 1.4)

§ 1.5 QUESTIONS TO ADDRESS

- 1) How long does each implementation method take (synthesis vs schematic)?
- 2) What are the qualitative differences, if any, between the two implementation results?

§ 1.6 OBJECTIVE

As inferred in the Purpose (§ 1.0), the objective of this project is to provide a Reference Design for methodology comparison purposes. As inferred in Limitations (§ 1.2), the objective is *not* to create clever or complicated interfaces. The Reference Design authors intend to establish a *starting point* for comparison discussions.

§ 1.7 PROPOSAL

The Reference Design proposed is a Data Transform Controller (Figure 1) with four major interfaces: Host, BackEnd, Memory, and Test. Data transferred between the Host and the BackEnd are modified by two, separate transform circuits, X1 and X2. Transform modifiers are stored in and retrieved from Memory, and transform control is directed from two, separate 8-bit (DAC and ADC) interfaces. Finally, a Test Port will be used to peek and poke as much of the Reference Design logic as possible.

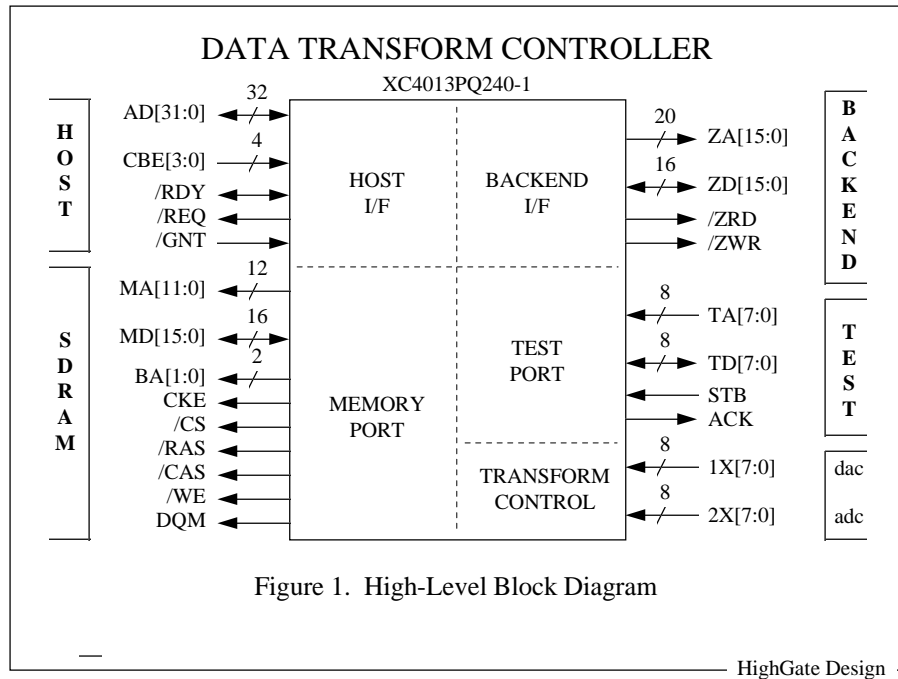


Figure 1. High-Level Block Diagram

The following miscellaneous signals are also defined:

- LED[3:0] (outputs from the Control Register) Designer's choice
- SEL[2:0] (inputs for Board Select) Board Address = 000
- RESET (to re-initialize) active low

Two study phases are proposed:

- Phase I:** implement all but the Test Port;
- Phase II:** implement the Test Port.

Time permitting, we may elaborate on these specs in future revisions for future comparisons.

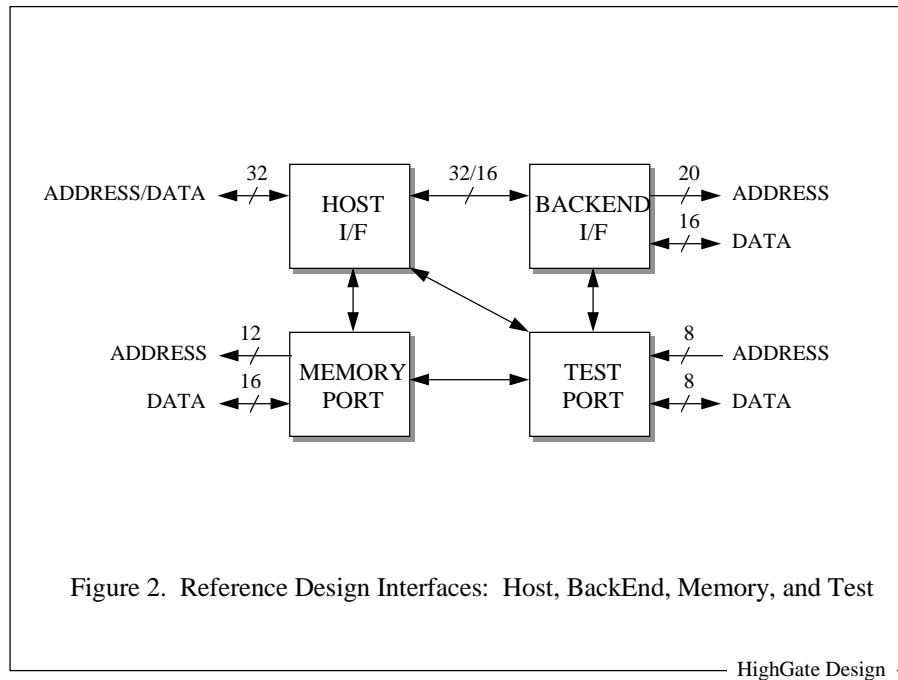
§ 2 OVERVIEW

§ 2.0 REFERENCE DESIGN INTERFACES

The Reference Design consists of four (4) generic interfaces (Figure 2), as follows:

1. Host Interface (32-bit)
2. BackEnd Interface (16-bit)
3. Memory Port (16-bit)
4. Test Port (8-bit)

For simplicity and brevity, the major interfaces may be implemented with timing derived from the same clock. The Host clock will be specified as, "as fast as possible."



§ 2.1 HOST INTERFACE

The Host Interface is intended to approximate, *not duplicate*, a PCI interface. The characteristics approximated are as follows:

- a) 32-bit, multiplexed address/data bus
- b) 4-bit command byte
- c) control signals

We will not be implementing any such features as RETRY or ABORT. We will only implement un-interrupted data transfers of predetermined length (single I/O and fixed-burst memory).

§ 2.2 BACKEND INTERFACE

The BackEnd Interface is intended to approximate, *not replicate*, a proprietary interface. The characteristics approximated are as follows:

- a) 20-bit address bus
- b) 16-bit data bus
- c) control signals

Again, we will implement uninterrupted transfers of fixed-burst lengths.

§ 2.3 MEMORY PORT

The Memory Port will be implemented using a 1M x 16-bit x 4 Bank SDRAM, as follows:

- a) 12-bit (RAS/CAS multiplexed) address bus
- b) 16-bit data bus
- c) control signals

§ 2.4 TEST PORT

The Test Port will be added in Phase II just to make things interesting. This port could have been specified in quite a variety of ways, but for this design, it will be implemented as follows:

- a) 8-bit address bus
- b) 8-bit data bus
- c) control signals (asynchronous and slow)

§ 2.5 ADC and DAC PORTS

The two, 8-bit ADC and DAC interfaces simply comprise direct inputs used to control the two, data transforms. Therefore, there will be no logic design for these ports.

§ 3 OPERATIONS

Operations are defined relative to physical entities, that is, between the Host (“Host”), Reference Design Memory (“Memory”), and Reference Design BackEnd (“BackEnd”).

§ 3.0 HOST OPERATIONS

The Host (Master) will initialize the Reference Design (Slave) in a fashion similar to PCI Target mode. The Host will write length and address registers and will then conduct read/write data bursts (of fixed length). The following Host operations will be supported:

1. Memory Write (Host-to-Memory)
2. Memory Read (Memory-to-Host) *<optional>*
3. BackEnd Write (Host-to-BackEnd)
4. BackEnd Read (BackEnd-to-Host)

In addition, several internal Status and Control Registers will be defined for read/write operations. Please note: In the above, “Memory” refers to the Reference Design Memory Port.

§ 3.0.0 HOST-MEMORY OPERATIONS

The Host will initialize Memory (Memory Write) prior to “normal mode” operations. Memory Read may be used to verify the Memory Write functionality. The Host will write the transfer length and Memory address register, and will then initiate a “MemoryRead” or “MemoryWrite” operation. Host transfers will proceed in 32-bit, 16-word bursts.

§ 3.0.1 HOST-BACKEND OPERATIONS

“Normal Mode” operations will consist of reads and writes between the Host and the BackEnd. Like the Host-Memory operations, the Host will write the transfer length and BackEnd address registers, and will then initiate a “BackEndRead” or a “BackEndWrite” operation. Host transfers will proceed in 32-bit, 16-word bursts.

§ 3.1 TEST OPERATIONS

The Test Port is an asynchronous peek/poke port. As many internal test points as make sense will be defined for access by the Test Port. Probably, more peek points than poke points will be defined. These points will be defined later in Phase II.

§ 4 ARCHITECTURE

§ 4.0 MODULES

There shall be four major architectural modules in this Reference Design (Figure 3), as follows:

1. Host FIFO
2. BackEnd FIFO
3. Write Transform (Host-to-BackEnd)
4. Read Transform (BackEnd-to-Host)

These features are implemented to satisfy the conditions for use of memory and arithmetic resources.

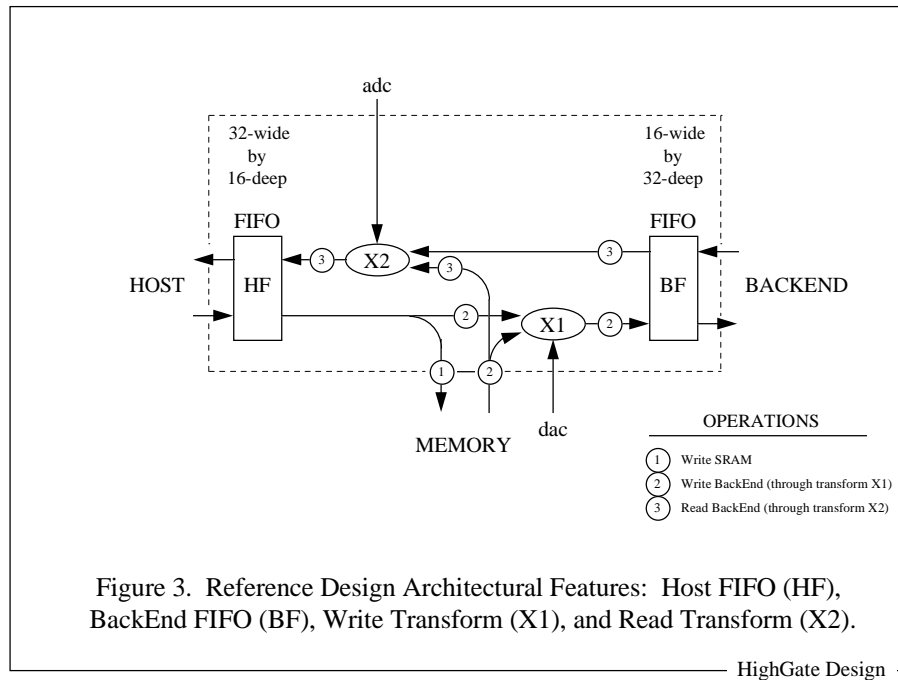


Figure 3. Reference Design Architectural Features: Host FIFO (HF), BackEnd FIFO (BF), Write Transform (X1), and Read Transform (X2).

§ 4.0.0 HOST FIFO

The Host FIFO will comprise a 16-deep, 32-wide RAM used in all burst transfers. Burst transfers to and from the Host will always be 32-bit wide by 16 words; transfers to and from both the Memory and the BackEnd will be 16-bit wide by 32 words. Therefore, for BackEnd Writes, the Host bus AD[31:16] and AD[15:0] will both map to the BackEnd bus, ZD[15:0], and for BackEnd Reads, BackEnd bus ZD[15:0] will map onto both Host bus AD[31:16] and AD[15:0]. Similar mapping is required for Memory reads and writes.

§ 4.0.1 BACKEND FIFO

The BackEnd FIFO will comprise a 32-deep, 16-wide RAM used in all BackEnd Read and Write transfers. BackEnd transfers to and from the Host FIFO will always be 16-bit wide by 32 words.

§ 4.0.2 WRITE TRANSFORM (X1)

The Write Transform will be used in all BackEnd Write transfers. Host data from the Host FIFO will flow through the X1 transform and will be modified by data fetched from Memory. Memory data will have been previously initialized by Host-to-Memory Write transfers. Transform control will be provided by 8 independent control lines (direct from DAC interface input pins). The X1 transform will comprise Boolean, Shift, and Arithmetic functions.

§ 4.0.3 READ TRANSFORM (X2)

The Read Transform will be used in all BackEnd Read transfers. BackEnd data from the BackEnd FIFO will flow through the X2 transform and will be modified by data fetched from Memory. Memory data will have been previously initialized by Host-to-Memory Write transfers. Transform control will be provided by 8 independent control lines (direct from ADC interface input pins). The X2 transform will comprise Arithmetic, Limit, and Round functions. Memory Read transfers may bypass the X2 Transform.

§ 4.1 DATAPATHS

§ 4.1.0 HOST to/from HOST FIFO

The datapath between the Host and the Host FIFO is 32-bit.

§ 4.1.1 HOST FIFO to MEMORY

The datapath from the Host FIFO to Memory is 16-bit. As specified in § 4.0.0, the Host FIFO upper and lower words, D[31:16] and D[15:0], will both map to the same 16-bit Memory data bus.

§ 4.1.2 HOST FIFO to TRANSFORM X1

The datapath from the Host FIFO to the X1 transform is 16-bit. As specified in § 4.0.0, the Host FIFO upper and lower words, D[31:16] and D[15:0], will both map to the same 16-bit (X1) bus.

§ 4.1.3 MEMORY to TRANSFORM X1

The datapath from the Memory to the X1 transform is 16-bit.

§ 4.1.4 TRANSFORM X1 to BACKEND FIFO

The datapath from the X1 transform to the BackEnd FIFO is 16-bit.

§ 4.1.5 BACKEND FIFO to/from BACKEND

The datapath between the BackEnd FIFO and the BackEnd Interface is 16-bit.

§ 4.1.6 BACKEND FIFO to TRANSFORM X2

The datapath from the BackEnd FIFO and the X2 transform is 16-bit.

§ 4.1.7 MEMORY to TRANSFORM X2

The datapath from Memory to the X2 transform is 16-bit.

§ 4.1.8 TRANSFORM X2 to HOST FIFO

The datapath from the X2 transform to the Host FIFO is 16-bit. The 16-bit X2 output is mapped into both the upper and lower 16-bit words of the 32-bit Host FIFO.

§ 4.1.9 HOST to MISCELLANY

Other host datapath widths are specified as follows:

- AD[18:0] → Length Counter
- AD[21:0] → Memory Address Register/Counter ... where AD[21:20] = BA[0:1]
- AD[19:0] → BackEnd Address Register/Counter
- AD[31:0] → Status and Control Registers

§ 4.2 CONTROL PATHS

§ 4.2.0 HOST CONTROL

Host control will be implemented with one control (*/RDY*) and four command lines (*/CBE[3:0]*). Host control will be able to initiate both memory and I/O reads and writes.

§ 4.2.1 MEMORY CONTROL

Memory Port control will be implemented for a 1M x 16 bit x 4 Bank SDRAM. (See the Samsung databook for the KM416S4030A.)

§ 4.2.2 BACKEND CONTROL

BackEnd control will be implemented with two strobes (*/RD*, */WR*). Operations will include (and may be limited to) data burst reads and writes.

§ 4.2.3 TRANSFORM X1 CONTROL

As specified in § 2.5, the X1 transform controls will source directly from an offchip 8-bit DAC interface.

§ 4.2.4 TRANSFORM X2 CONTROL

As specified in § 2.5, the X2 transform controls will source directly from an offchip 8-bit ADC interface.

§ 5 OPERATIONAL DETAILS

As defined before, operations are between physical entities: the Host (“Host”), Reference Design Memory (“Memory”), and Reference Design BackEnd (“BackEnd”).

§ 5.0 HOST-TO-MEMORY WRITE

The Host-to-Memory Write operation will be performed as follows:

1. Host Write to Length Counter (total length of the data block transfer)
2. Host Write to Memory Address Counter (destination start modulo-32 address in Memory)
3. Host initiates a “Burst Write” sequence (destination decode = Memory)
4. When length exhausted, Host may Read Status Register (for confirmation)

Once the “Burst Write” sequence is initiated, data will be transferred from the Host into the Host FIFO and thence written to Memory. Transfers from the Host will always be sixteen, 32-bit words, uninterrupted. (We will not be implementing *any* kind of RETRY.) Transfers from the Host FIFO to Memory will always be thirty-two, 16-bit words.

§ 5.1 HOST-FROM-MEMORY READ

The Host-from-Memory Read <optional> operation may be performed as follows:

1. Host Write to Length Counter (total length of the data block transfer)
2. Host Write to Memory Address Counter (source start modulo-32 address in Memory)
3. Host initiates a “Burst Read” sequence (destination decode = Memory)
4. When length exhausted, Host may Read Status Register (for confirmation)

Once the “Burst Read” sequence is initiated, data will be read from Memory to the Host FIFO and thence transferred to the Host. Transfers to the Host will always be sixteen, 32-bit words, uninterrupted. (We will not be implementing *any* kind of RETRY.) Transfers from Memory to the Host FIFO will always be thirty-two, 16-bit words.

This operation is optional but would provide a good diagnostic check for Memory Writes. Note: The Memory to Host FIFO path may bypass the X2 transform.

§ 5.2 HOST-TO-BACKEND WRITE

The Host-to-BackEnd write operation will be performed as follows:

1. Host Write to Length Counter (total length of the data block transfer)
2. Host Write to Memory Address Counter (source start address for transform modifiers)
3. Host Write to BackEnd Address Counter (destination start address in BackEnd)
4. Host initiates a “Burst Write” sequence (destination decode = BackEnd)
5. When length exhausted, Host may Read Status Register (for confirmation)

Once the “Burst Write” sequence is initiated, data will be transferred from the Host to the Host FIFO and thence written to the BackEnd FIFO and the BackEnd. Transfers from the Host will always be sixteen, 32-bit words, uninterrupted. (We will not be implementing *any* kind of RETRY.) Transfers from Host FIFO to BackEnd will always be thirty-two, 16-bit words.

Note: BackEnd write start addresses may start on any boundary.

§ 5.3 HOST-FROM-BACKEND READ

The Host-from-BackEnd Read operation will be performed as follows:

1. Host Write to Length Counter (total length of the data block transfer)
2. Host Write to Memory Address Counter (source start address for transform modifiers)
3. Host Write to BackEnd Address Counter (source start address in BackEnd)
4. Host initiates a "Burst Read" sequence (destination decode = BackEnd)
5. When length exhausted, Host may Read Status Register (for confirmation)

Once the "Burst Read" sequence is initiated, data will be read from the BackEnd to the BackEnd FIFO and thence transferred to the Host FIFO and to the Host. Transfers to the Host will always be sixteen, 32-bit words, uninterrupted. (We will not be implementing *any* kind of RETRY.) Transfers from BackEnd FIFO to Host FIFO the will always be thirty-two, 16-bit words.

Note: BackEnd write start addresses may start on any boundary.

§ 5.4 HOST-TO-REGISTER WRITE

The Host-to-Register Write operation will be performed as follows:

1. Host initiates an "I/O Write" sequence

During the address phase, the AD[31:0] LSBs are used to specify the destination register. During the data phase, data on the Host bus will then be written into the appropriate internal register.

§ 5.5 HOST-FROM-REGISTER READ

The Host-from-Register Read operation will be performed as follows:

1. Host initiates an "I/O Read" sequence

During the address phase, the AD[31:0] LSBs are used to specify the source register. During the data phase, the Reference Design must drive data on the Host bus from the appropriate internal register.

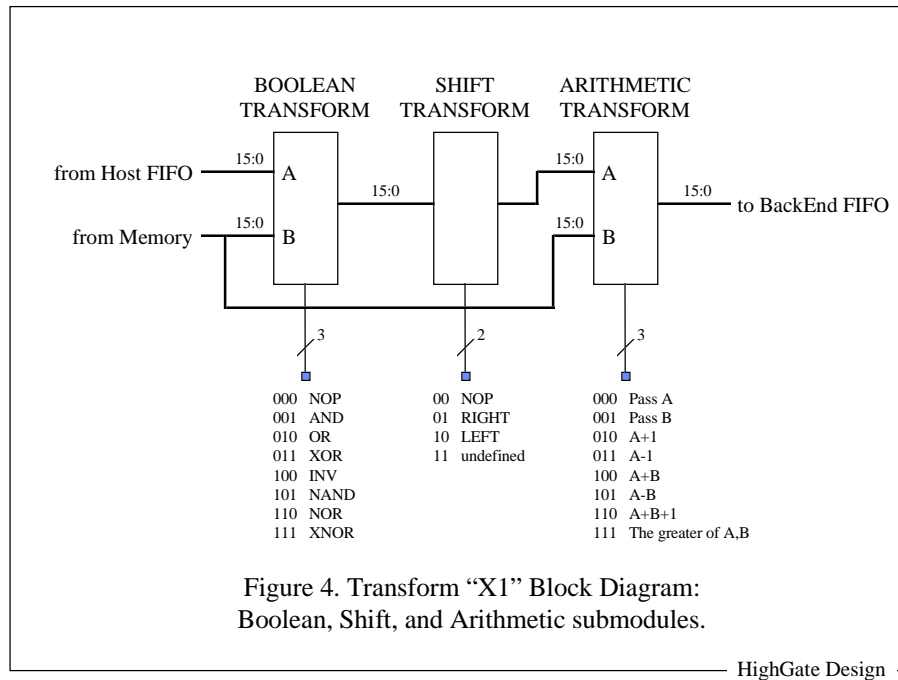
§ 6 FUNCTIONAL DETAILS

§ 6.0 TRANSFORM X1

The X1 transform consists of three submodules, as follows:

1. Boolean
2. Shift
3. Arithmetic

The X1 transforms (Figure 4) are controlled by eight, separate control lines (from the DAC interface)—three to specify the Boolean operation, two to specify the Shift operation, and three to specify the Arithmetic operation.



§ 6.0.0 BOOLEAN TRANSFORM

The Boolean transform submodule will provide the following transforms:

- NOP (A)
- AND (A*B)
- OR (A+B)
- XOR (A+B)
- INV /(A)
- NAND /(A*B)
- NOR /(A+B)
- XNOR /(A+B)

These operations will be performed between the 16-bit data coming the Host FIFO (operand A) and the 16-bit modifiers read from the Memory (operand B). The results of these Boolean transforms will be passed to the Shift transform submodule.

§ 6.0.1 SHIFT TRANSFORM

The Shift transform submodule will provide the following transforms:

- NOP
- ShiftRight (sign extended)
- ShiftLeft (zero fill)

These operations will be performed on the 16-bit output of the Boolean submodule and will be passed to the Arithmetic transform submodule.

§ 6.0.2 ARITHMETIC TRANSFORM

The Arithmetic transform submodule will provide the following transforms:

- Pass A (the output from the Shift submodule)
- Pass B (the local memory data)
- A+1
- A-1
- A+B
- A-B
- A+B+1
- The greater of A or B

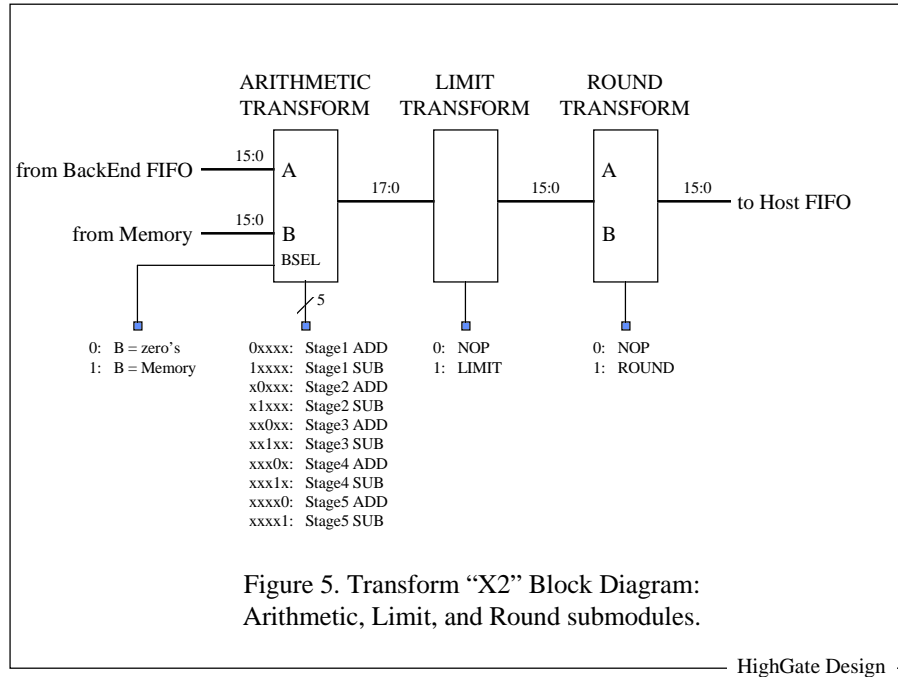
These operations will be performed on the 16-bit output of the Shift submodule (operand A) and/or the 16-bit output of the local memory (operand B). The 16-bit result will be forwarded to the BackEnd FIFO.

§ 6.1 TRANSFORM X2

The X2 transform consists of three submodules, as follows:

1. Arithmetic
2. Limit
3. Round

The X2 transforms (Figure 5) are controlled by eight, separate control lines (from the ADC interface)—six to specify the Arithmetic operations, one to enable the Limit function, and one to enable the Round function.



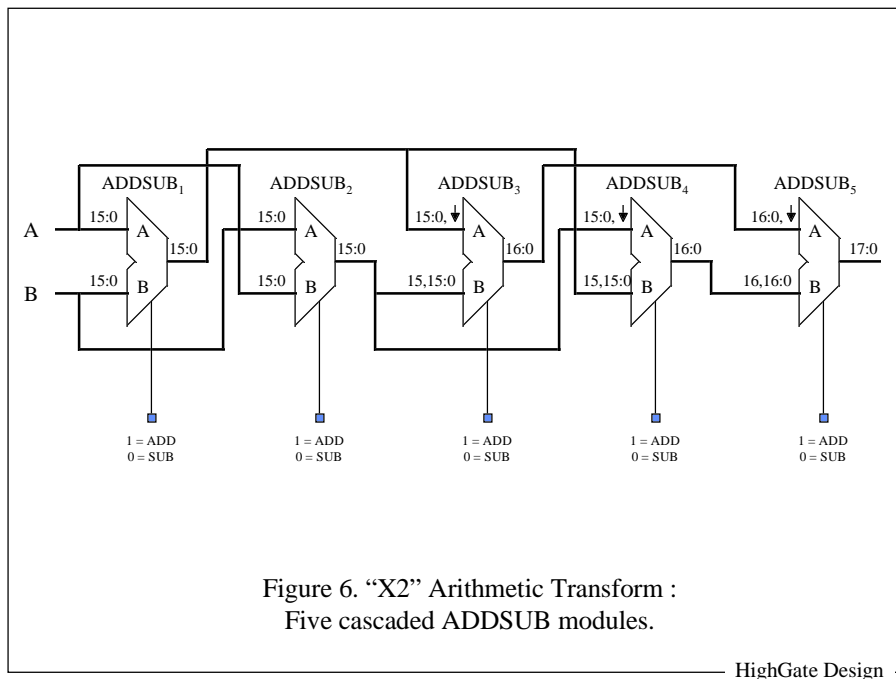
Note: Not shown in Figure 5 is the Bypass mode for Memory Reads. Memory reads bypassing X2 will move Memory read data directly to the Host FIFO. Such bypass reads are differentiated by the LSB decode of the transaction address phase. (See § 7.4)

§ 6.1.0 ARITHMETIC TRANSFORM

The Arithmetic transform submodule consists of five ADDSUB functions. The operands source from the BackEnd FIFO (operand A), and from the Memory (operand B) or a zero source. At the head of the Arithmetic transform, one of the six transform control lines directs the operand B selector to choose either Memory or zero data. Thereafter, the five ADDSUB functions are each individually directed by the five other Arithmetic transform control lines to either add or subtract.

The five ADDSUB functions (Figure 6) are cascaded as follows:

- $ADDSUB_1 = A \pm B$
- $ADDSUB_2 = B \pm A$
- $ADDSUB_3 = ADDSUB_1 \pm ADDSUB_2$
- $ADDSUB_4 = ADDSUB_2 \pm ADDSUB_1$
- $ADDSUB_5 = ADDSUB_3 \pm ADDSUB_4$



Furthermore, $ADDSUB_1$ and $ADDSUB_2$ are 16-bit, $ADDSUB_3$ and $ADDSUB_4$ are 17-bit, and $ADDSUB_5$ is 18-bit. Moreover, for the $ADDSUB_{3:5}$ inputs, one operator is sign-extended and the other left-shifted and zero-filled, as follows:

- for $ADDSUB_3$,
the $ADDSUB_1$ operand is left-shifted and zero-filled (at the LSB)
the $ADDSUB_2$ operand is sign-extended
- for $ADDSUB_4$,
the $ADDSUB_1$ operand is sign-extended
the $ADDSUB_2$ operand is left-shifted and zero-filled (at the LSB)
- for $ADDSUB_5$,
the $ADDSUB_3$ is operand left-shifted and zero-filled (at the LSB) and
the $ADDSUB_4$ is operand sign-extended

The output of the $ADDSUB_5$ stage is passed to the Limit transform submodule.

§ 6.1.1 LIMIT TRANSFORM

The Limit transform submodule will perform the following functions:

- NOP
- Limit to $+32,767$ $-32,768$ [i.e. $(+2^{15})-1, -2^{15}$]

The Limit truth table is as follows:

INPUT [17:0]\b	OUTPUT [15:0]\b
11 0xxx xxxx xxxx xxxx	0111 1111 1111 1111
01 0xxx xxxx xxxx xxxx	0111 1111 1111 1111
00 1xxx xxxx xxxx xxxx	0111 1111 1111 1111
00 0nnn nnnn nnnn nnnn	0nnn nnnn nnnn nnnn
11 1nnn nnnn nnnn nnnn	1nnn nnnn nnnn nnnn
11 0xxx xxxx xxxx xxxx	1000 0000 0000 0000
10 1xxx xxxx xxxx xxxx	1000 0000 0000 0000
10 0xxx xxxx xxxx xxxx	1000 0000 0000 0000

The output of the Limit transform is passed to the Round transform submodule.

§ 6.1.2 ROUND TRANSFORM

The Round transform submodule will perform the following functions:

- NOP
- Round to the eight (8) MSB's

The Round truth table is as follows:

INPUT [15:0]\b	OUTPUT [15:0]\b
0111 1111 1xxx xxxx	0111 1111 0000 0000
0111 1111 0xxx xxxx	
0111 1110 1xxx xxxx	
0111 1110 0xxx xxxx	0111 1110 0000 0000
0111 1101 1xxx xxxx	
0000 0001 0xxx xxxx	0000 0001 0000 0000
0000 0000 1xxx xxxx	
0000 0000 0xxx xxxx	0000 0000 0000 0000
1111 1111 1xxx xxxx	
1111 1111 0xxx xxxx	1111 1111 0000 0000
1111 1110 1xxx xxxx	
1000 0001 0xxx xxxx	1000 0001 0000 0000
1000 0000 1xxx xxxx	
1000 0000 0xxx xxxx	

Please note the two boundary conditions and their anomalous “round” definitions.

§ 7 HOST INTERFACE

The Reference Design Host interface is a *PCI-like* interface; that is, it uses some of the same signals and cycles, but *does not attempt to duplicate the features and functions*. (Having designed and delivered the Xilinx PCI reference design, I am in no hurry to recycle that wheel.)

In particular, the Reference Design Host interface consists of the following signals:

- CLK1
- /RDY
- AD[31:0]
- CBE[3:0]
- /REQ
- /GNT

Note that there is no /FRAME signal. Also, /IRDY and /TRDY have been combined into a single ready line. Although shown, /REQ and /GNT will not be used in any of the following exercises. There will be no RETRY, ABORT, TERMINATE, TIMEOUT or other non-cooperative cycles. Since this is a “Target-only” design, the Reference Design is *always ready* to receive and transmit data.

For this Reference Design, four Host commands will be honored, as follows:

- | | | |
|-----------------|-------|-----------------|
| 1. I/O Write | “IOW” | CBE[3:0] = 0011 |
| 2. I/O Read | “IOR” | CBE[3:0] = 0010 |
| 3. Memory Write | “MWR” | CBE[3:0] = 0111 |
| 4. Memory Read | “MRD” | CBE[3:0] = 0110 |

In this Reference Design, these four commands (as defined by the CBE[3:0] decodes) are used in the following four sequences:

- | | |
|----------------|---------------|
| 1. I/O Write | “I/O Write” |
| 2. I/O Read | “I/O Read” |
| 3. Burst Write | “Burst Write” |
| 4. Burst Read | “Burst Read” |

The I/O Write sequence will be used to write the Reference Design Address, Length, and Control registers. The I/O Read sequence will be used to read back the Reference Design Status and Configuration registers. The Burst Write sequence will be used to fill Memory (with transform data), and to write data to the BackEnd. The Burst Read sequence will be used to read back Memory data and to retrieve data from the BackEnd. All four sequences will be “Target” mode transactions; that is, the Reference Design (Slave) will always be ready to respond.

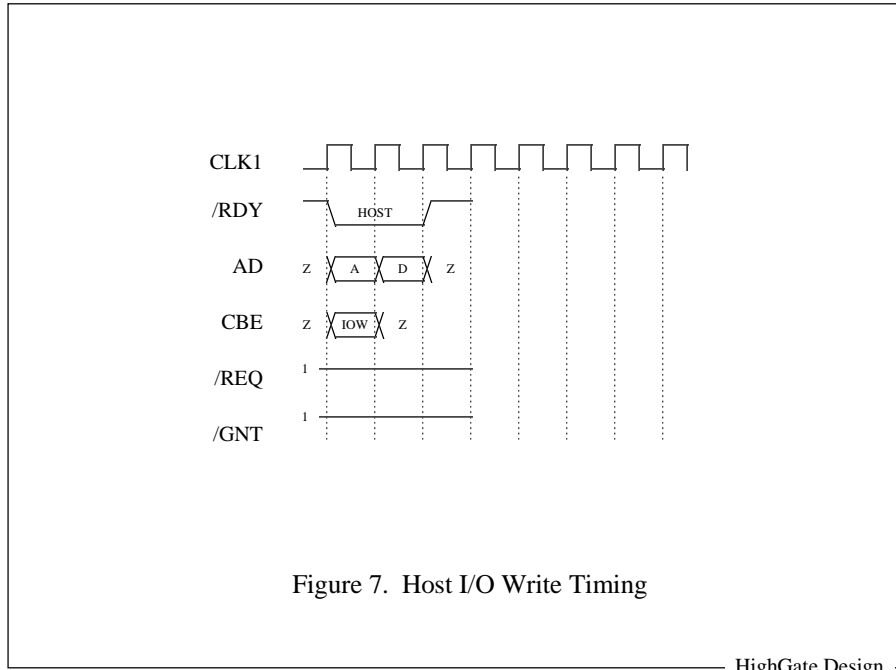
Both of the I/O sequences will be single data phase transactions. Both burst sequences will *always* be contiguous 16-cycle bursts. For I/O Writes, no WAIT states will be permitted, whereas for I/O Reads, the Reference Design is permitted to elongate the TurnAround cycle from one to 16 clocks (inclusive). Similarly for Burst Writes, no WAIT states will be permitted, whereas for Burst Reads, the Reference Design is permitted to elongate the TurnAround cycle from one to 16 clocks (inclusive). Between the burst data phases, no WAIT cycles will be permitted.

In the first clock cycle of every transaction, the Host will drive the AD bus with the Reference Design target address (a decode of the MSBs) and with the target register or port address (a decode of the LSBs). For writes, no TurnAround cycle is needed; for reads, one TurnAround cycle is required, 16 are permitted (as a way to ‘wait’ the Host).

§ 7.1 I/O WRITE

The I/O Write sequence (Figure 7) is used to write the Reference Design Address, Length, and Control registers. This is a single data phase transaction in which the Host drives AD[31:0] for two clock cycles. In the address phase (the first clock cycle), AD[31:0] contain both the Reference Design target address (MSB decode) and the write register target address (LSB decode). In the data phase (the second clock cycle), AD[31:0] contain the data to be written to the target register.

$\overline{\text{RDY}}$ is also driven by the Host for both clock cycles. CBE[3:0] (containing the "IOW" command) are driven only by the Host and only in the first clock cycle. Neither $\overline{\text{REQ}}$ nor $\overline{\text{GNT}}$ are driven for I/O Write sequences.



There are four registers in the Reference Design to be initialized using the I/O Write sequence, as follows:

- | | <u>LSB DECODE</u> |
|-------------------------------------|-------------------|
| 1. Control Register | AD[3:0] = 0000 |
| 2. Memory Address Register/Counter | AD[3:0] = 0001 |
| 3. BackEnd Address Register/Counter | AD[3:0] = 0010 |
| 4. Length Counter | AD[3:0] = 0011 |

The Control Register contains various setup and steering bits (TBD). The Memory Address Register/Counter contains the read/write address used for Memory bursting. The BackEnd Address Register/Counter contains the read/write address used for BackEnd bursting. The Length Counter contains the data block length used in all burst transfers.

§ 7.2 I/O READ

The I/O Read sequence (Figure 8) is used to read the Reference Design's internal registers. This is a single data phase transaction in which the Host drives AD[31:0] for one clock cycle, and the Reference Design drives AD[31:0] for one cycle. In the address phase (the first clock cycle), AD[31:0] contain both the Reference Design target address (MSB decode) and the read register target address (LSB decode). Next, in the "TurnAround" (TA) period (which may consist of one to 16 clocks), neither the Host nor the Reference Design may drive any signal. Then, in the data phase, AD[31:0] are driven by the Reference design and contain the data read from the target register.

/RDY is driven by the Host during the address phase and by the Reference Design during the data phase. CBE[3:0] (containing the "IOR" command) are driven only by the Host and only in the first clock cycle. Neither /REQ nor /GNT are driven for I/O Read sequences.

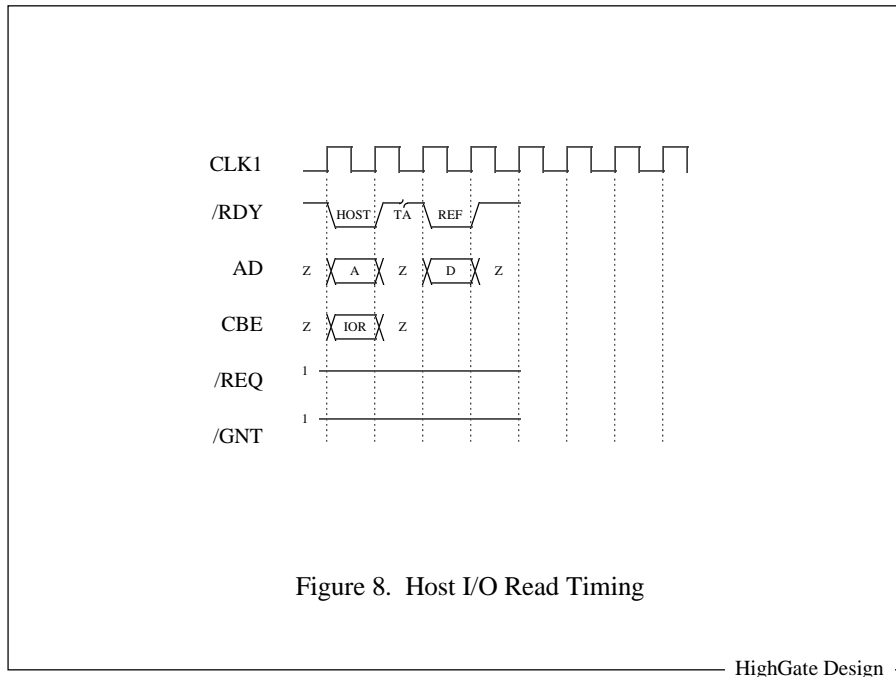


Figure 8. Host I/O Read Timing

HighGate Design

There are four registers in the Reference Design to be read using the I/O Read sequence, as follows:

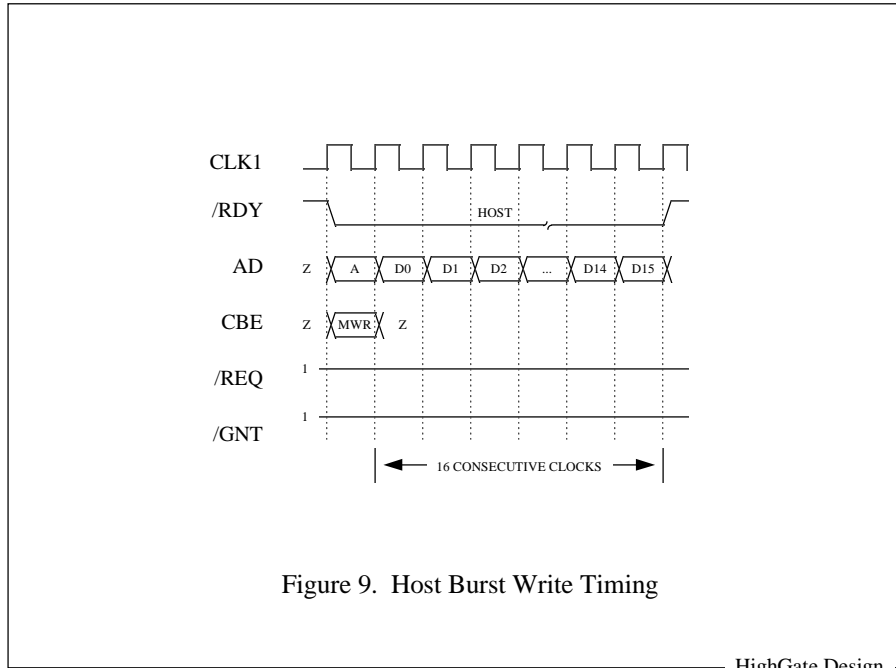
- | | <u>LSB DECODE</u> |
|---|-------------------|
| 1. Status Register | AD[3:0] = 0000 |
| 2. Config Register 00 (16-bit Device ID and 16-bit Vendor ID) | AD[3:0] = 0001 |
| 3. Config Register 08 (24-bit Class Code and 8-bit Revision ID) | AD[3:0] = 0010 |
| 4. Length Counter | AD[3:0] = 0011 |

The two Config Registers will be hard coded sources. (Refer to § 12.0)

§ 7.3 BURST WRITE

The Burst Write sequence (Figure 9) is used to write data to local memory or the BackEnd. This is a burst transaction in which only the Host drives AD[31:0]. In the address phase (the first clock cycle), AD[31:0] are driven with both the Reference Design target address (MSB decode) and the target port (LSB decode). In the data phase (the next sixteen clock cycles), AD[31:0] are driven with the data to be written to the target port (either Memory or BackEnd).

/RDY is also driven by the Host for all clock cycles. CBE[3:0] (containing the “MWR” command) are driven only by the Host and only in the first clock cycle. Neither /REQ nor /GNT are driven for Burst Write sequences.



There are two Burst Write destinations, as follows:

- | | LSB DECODE | |
|------------|------------------------------|------------------------------|
| 1. Memory | AD[3:0] = 1000 (Bypass Mode) | |
| 2. BackEnd | AD[3:0] = 0001 (Normal Mode) | AD[3:0] = 1001 (Bypass Mode) |

The Memory Port will be written during a “Host-to-Memory Write” operation (§ 5.0); the Back-End will be written during a “Host-to-BackEnd Write” operation (§ 5.2). Both of these operations employ the Burst Write sequence; the only difference is in the address phase LSB destination decode.

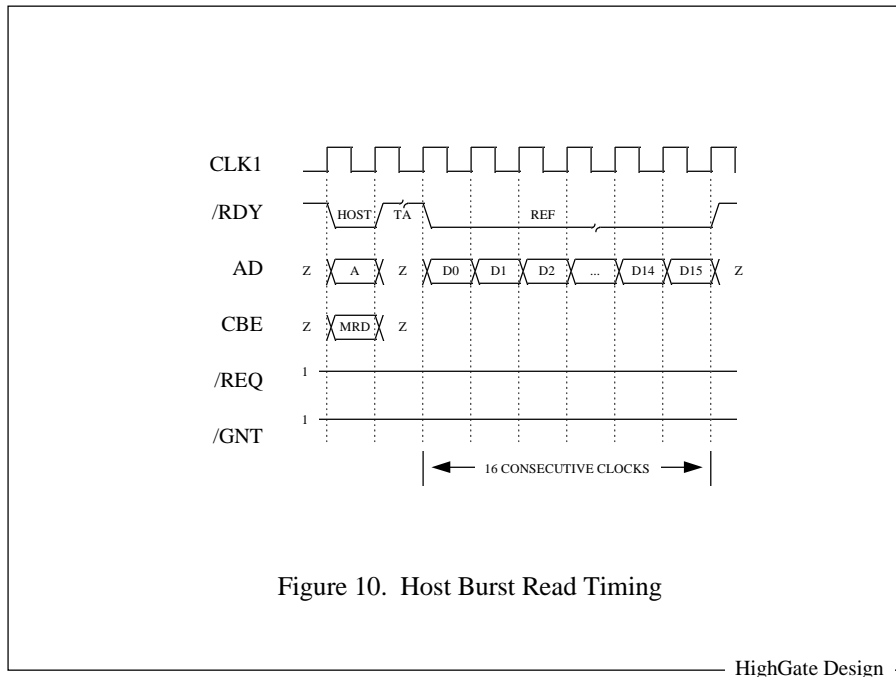
Note: BackEnd Burst Writes in Bypass Mode will not be implemented for Phase I.

Once initialized, the Reference Design is responsible for maintaining and incrementing the Memory Address Counter and BackEnd Address Counter (as appropriate) and for decrementing the Length Counter.

§ 7.4 BURST READ

The Burst Read sequence (Figure 10) is used to read data from Memory or the BackEnd. This is a burst transaction in which the Host drives AD[31:0] for one clock cycle, and the Reference Design drives AD[31:0] for sixteen cycles. In the address phase (the first clock cycle), the Host drives AD[31:0] with both the Reference Design target address (MSB decode) and the target port (LSB decode). Next, in the “TurnAround” (TA) period (which may consist of one to 16 clocks), neither the Host nor the Reference Design may drive any signal. Then, in the data phase, the Reference design must drive AD[31:0] with data from the source port (either Memory or BackEnd).

/RDY is driven by the Host during the address phase and by the Reference Design during the data phase. CBE[3:0] (containing the “MRD” command) are driven only by the Host and only in the first clock cycle. Neither /REQ nor /GNT are driven for Burst Read sequences.



There are two Burst Read sources, as follows:

- | | LSB DECODE | |
|------------|------------------------------|------------------------------|
| 1. Memory | AD[3:0] = 1000 (Bypass Mode) | |
| 2. BackEnd | AD[3:0] = 0001 (Normal Mode) | AD[3:0] = 1001 (Bypass Mode) |

The Memory Port will be read during a “Host-from-Memory Read” operation (§ 5.1); the BackEnd will be read during a “Host-from-BackEnd Read” operation (§ 5.3). Both of these operations employ the Burst Read sequence; the only difference is in the address phase LSB source decode.

Note: BackEnd Burst Reads in Bypass Mode will not be implemented for Phase I.

Once initialized, the Reference Design is responsible for maintaining and incrementing the Memory Address Counter and BackEnd Address Counter (as appropriate) and for decrementing the Length Counter.

§ 8 BACKEND INTERFACE

The Reference Design BackEnd is a proprietary (Master) interface. Although this interface is capable of sustaining transfers of indefinite length, this Reference Design need only perform bursts of fixed length (thirty-two, 16-bit words).

The Reference Design BackEnd interface consists of the following signals:

- CLK2
- ZA[15:0]
- ZD[15:0]
- /ZRD
- /ZWR

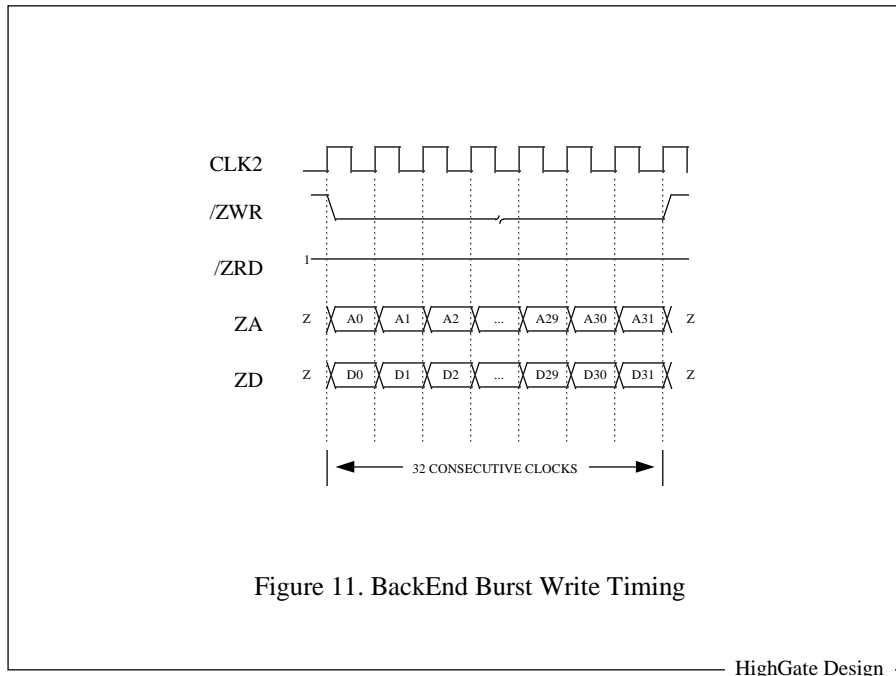
To expedite implementation, CLK2 may derive from the Host clock. ZA[15:0] is the dedicated BackEnd address bus, and ZD[15:0] is the dedicated, bidirectional BackEnd data bus. /ZRD and /ZWR are independent control strobes used for all BackEnd read and write transfers.

Only two BackEnd operations will be supported, as follows:

- BackEnd Burst Write (thirty-two, 16-bit words)
- BackEnd Burst Read (thirty-two, 16-bit words)

§ 8.0 BURST WRITE

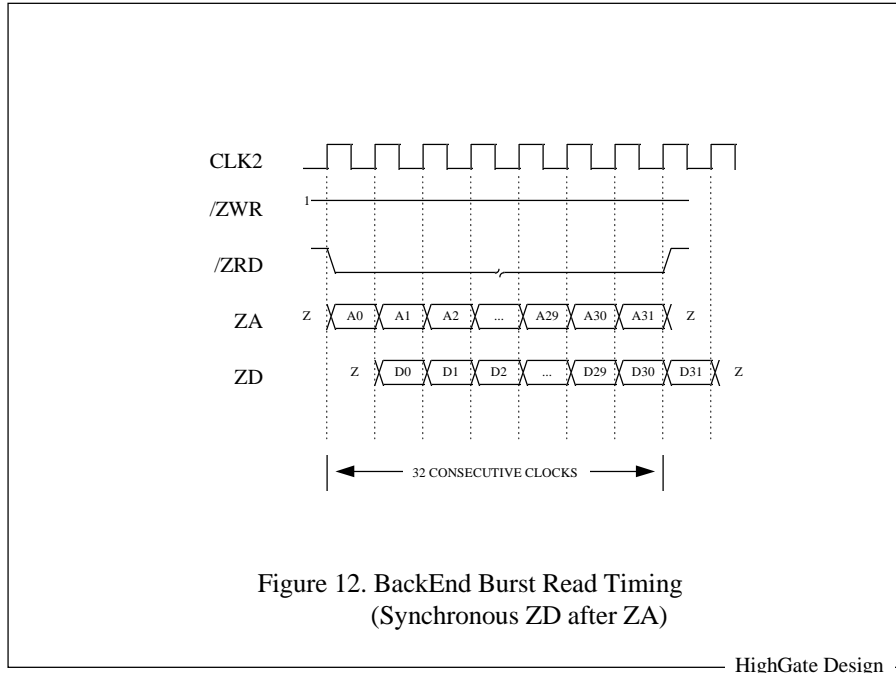
BackEnd Write Burst timing is extremely simplistic, as illustrated in Figure 11 below.



For every burst write data cycle, /ZWR and ZA[15:0] must be driven.

§ 8.1 BURST READ

BackEnd Read Burst timing is extremely simplistic, as illustrated in Figure 12 below.



For every burst read data cycle, /ZRD and ZA[15:0] must be driven.

§ 8.2 WORD ALIGNMENT

For all BackEnd burst reads and writes, the Host-BackEnd data buses shall align in the following manner:

<u>DWORD</u>	<u>HOST</u>	<u>WORD</u>	<u>BACKEND</u>
0	AD[15:0]	0	ZD[15:0]
	AD[31:16]	1	ZD[15:0]
1	AD[15:0]	2	ZD[15:0]
	AD[31:16]	3	ZD[15:0]
14	AD[15:0]	28	ZD[15:0]
	AD[31:16]	29	ZD[15:0]
15	AD[15:0]	30	ZD[15:0]
	AD[31:16]	31	ZD[15:0]

§ 9 MEMORY PORT

The Memory Port *will* identically replicate the controls necessary to interface to the Samsung KM416S4030A-10, a 1M x 16-bit x 4 Bank SDRAM (page 169 of the July 1997 SDRAM databook). However, this Reference Design will *not* attempt to exploit the full potential of that device.

The Memory Port interface consists of the following signals:

- CLK2
- MA[11:0]
- MD[15:0]
- BA[1:0]
- CKE
- /CS
- /RAS
- /CAS
- /WE
- DQM

To expedite implementation, CLK will derive from the reference clock (the same clock used by the Host and all Reference Design interfaces). For the operation of all other signals, refer to the Samsung databook.

Only four Memory operations will be supported, as follows:

- Mode Register Set (power-up initialization)
- Refresh
- Memory Burst Write (thirty-two, 16-bit words)
- Memory Burst Read (thirty-two, 16-bit words)

§ 9.0 INITIALIZATION

This SDRAM *may* be initialized (per page 179) as follows:

- Burst Length = 8
- Burst Type = Sequential
- CAS Latency = 2
- Test Mode = Mode Register Set
- Write Burst Length = Burst

If initialized as above, for the Mode Register Set command, MA[11:0] = 023\h. The designer may choose to initialize the SDRAM in any manner which satisfies the other conditions of this spec.

§ 9.1 REFRESH

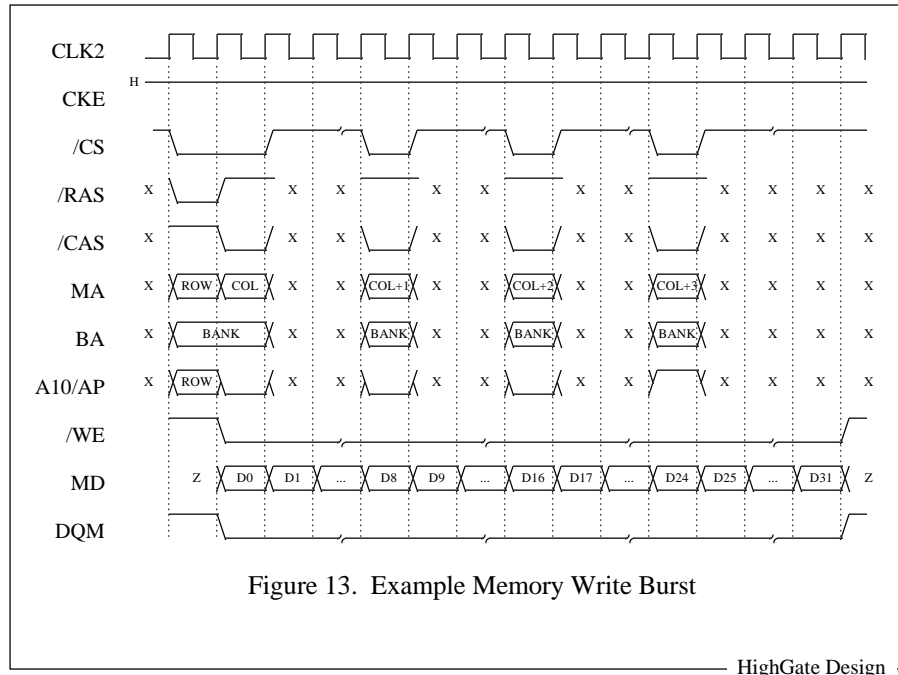
For this SDRAM, a refresh cycle is required every 64ms.

§ 9.2 BURST REQUIREMENTS

To simplify the Reference Design, all Memory Burst access start addresses must be modulo 32.

§ 9.3 BURST WRITE

For Memory Burst Writes, thirty-two, 16-bit words will be written by issuing one /RAS and four /CAS cycles—a total of 33 clocks (excluding “TurnAround” cycles that may be needed before and after this operation). A sample of such a write burst (in which the Burst Length = 8) is shown in Figure 13 below.



§ 9.4 BURST READ

Timing for the Memory Burst Read, is similar to the Burst Write except that MD[15:0] are driven by SDRAM two clocks later (per the CAS Latency initialization), and that /WE should be high. Because of the additional, two clock latency, the burst read will require 35 clocks.

§ 9.5 PRECHARGE

For both Burst Write and Burst Read, (for implementations using Burst Length = 8) the Reference Design should assert A10/AP (high) during the fourth /CAS cycle. Note that all burst addresses are required to be modulo 32; hence, no /RAS address rollover will occur *within* burst cycles. Therefore, no additional precharge cycles will be required *within* burst cycles.

§ 9.6 WORD ALIGNMENT

Word alignment for Host-Memory operations will be similar to Host-BackEnd word alignment (§ 8.2).

§ 9.7 BANK ADDRESS BITS

BA[0:1] are loaded into the Memory Address Register from AD[21:20] respectively. These two bits *shall not be incremented* during the entire duration of any transfer.

§ 10 TEST PORT [Phase II only]

The Test Port is a simple, slow, asynchronous 8-bit peek/poke port. The intention of this port is to provide *some* visibility to the internal workings of the Reference Design. (A secondary purpose is to inject some unsophisticated complexity.) The Test Port will act like a secondary Master, and will always have 'peek privileges.' However, 'poke privileges' will require the Host to set the Test Mode bit in the Reference Design Control Register.

The Test Port interface consists of the following signals:

- TA[7:0]
- TD[7:0]
- STB
- ACK

TA[7:0] are the input-only address supplied by the Test Port source and are to be decoded by the Reference Design to determine peek/poke sources/destinations. T7 will be used to differentiate between peeks (T7=0) and pokes (T7=1). TD[7:0] are the bidirectional Test Port data bus. STB is driven by the Test Port to indicate that a valid address is on TA[7:0] and, for writes, that valid data are on TD[7:0]. ACK is to be returned by the Reference Design to indicate that, for writes, data has been accepted, or, for reads, that valid data are ready on TD[7:0]. After ACK has been observed asserted, the Test Port source will deassert STB. After the Reference Design observes STB deasserted, ACK may be deasserted.

Only two modes of Test Port operations will be supported, as follows:

- Peek (single byte, asynchronous)
- Poke (single byte, asynchronous)

All transactions will be single byte. Hence, to obtain complete access to most points, several single-byte transactions will have to be negotiated. A single-byte Peek is illustrated in Figure 14 (next page), and the single-byte Poke is illustrated in Figure 15 (also next page).

§ 10.0 PEEK/POKE POINTS

This is currently the area with the least definition and should be considered the most likely to change. As the design evolves and a proper gate count is tallied, peek/poke points may either be added or subtracted to satisfy the Reference Design requirements.

§ 10.0.0 PEEK POINTS

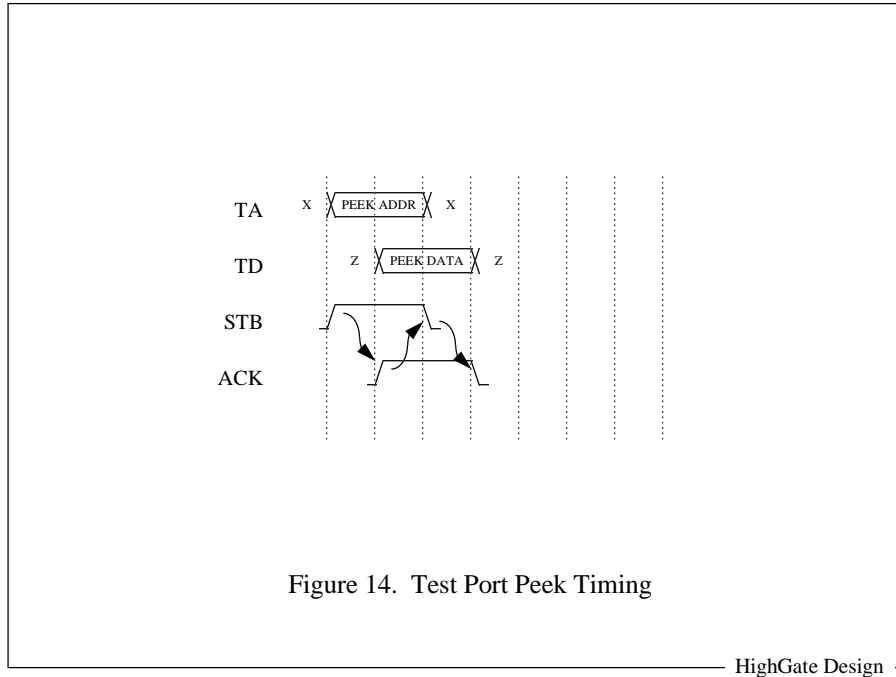
Peek Points (limited to 32 bytes) will probably include the following:

- Length Counter
- Status, Configuration, and Control Registers
- most, if not all of the X2 Transform outputs

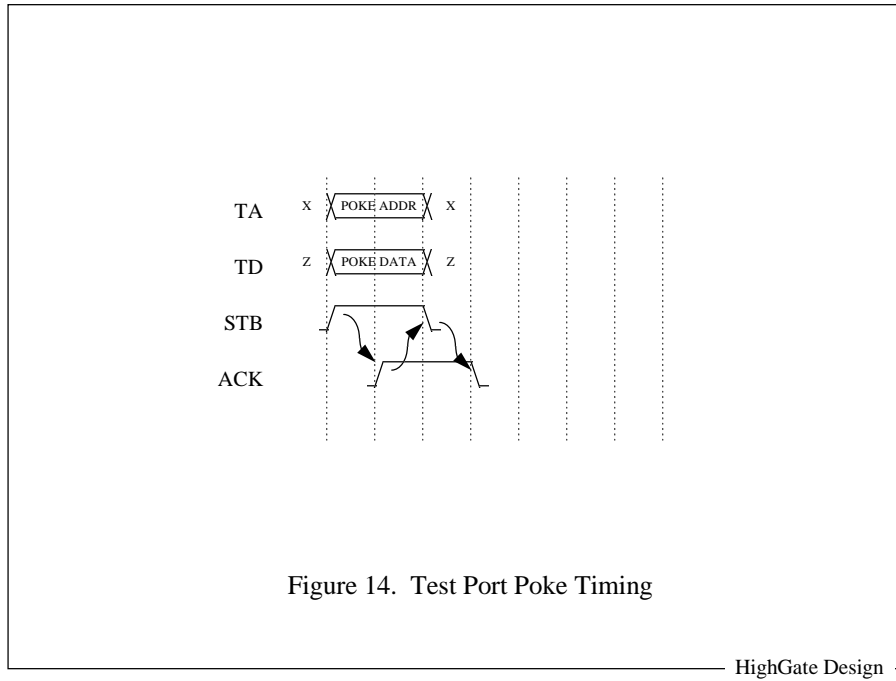
§ 10.0.1 POKE POINTS

Poke Points will probably be limited to the Control Register.

§ 10.1 PEEK TIMING



§ 10.2 POKE TIMING



§ 11 DAC and ADC INPUTS

Two, 8-bit buses are input to the Reference Design to control data transformation, as follows:

- 1X[7:0]
- 2X[7:0]

Although the 1X[7:0] and 2X[7:0] inputs are direct lines, their values may change every clock.

§ 11.0 DAC INPUTS

From an offchip DAC interface, 1X[7:0] are input to Transform X1. The assignments for these eight control lines are as follows:

- 1X[7:5] Boolean control
- 1X[4:3] Shift control
- 1X[2:0] Arithmetic control

§ 11.1 ADC INPUTS

From an offchip ADC interface, 2X[7:0] are input to Transform X2. The assignments for these eight control lines are as follows:

- 2X[7] B-operand select
- 2X[6] ADDSUB1 control
- 2X[5] ADDSUB2 control
- 2X[4] ADDSUB3 control
- 2X[3] ADDSUB4 control
- 2X[2] ADDSUB5 control
- 2X[1] Limit control
- 2X[0] Round control

§ 12 REGISTER DEFINITIONS

There are three read registers and four write registers identified for the Reference Design.

§ 12.0 READ REGISTERS

The Reference Design read registers are accessed by Host I/O Read sequences (§ 7.2). The register to be read is identified during the I/O Read address phase, as follows:

AD[3:0]	Register
0000	Status Register
0001	Config Reg 00
0010	Config Reg 08
0011	Length Register

All other AD[3:0] read decodes should return zero's.

§ 12.0.0 STATUS REGISTER

The Status Register is a 32-bit register with the following bit descriptions:

AD	NAME	FUNCTION	VALUE
• AD[31:4]	RSVD	reserved (return zero's)	
• AD[3]	HFE	Host FIFO Empty	0 = empty
• AD[2]	LCU	Length Counter Underflow	1 = underflow
• AD[1]	LCZ	Length Counter = Zero	1 = counter is zero
• AD[0]	TIP	Test Port Operation in Progress	1 = test in progress

The LCU bit is true whenever the Length Counter underflows (which should never happen if the Host is paying attention). The LCZ bit is true whenever the Length Counter equals zero (which should be true upon power up and when a read or write block transfer has been completed). The TIP bit is true whenever the Test Port is performing either a Peek or a Poke.

§ 12.0.1 CONFIG REGISTER 00

Config Register 00 is a 32-bit hard-wired register with the following bit descriptions:

AD	NAME	FUNCTION	VALUE
• AD[31:16]	DID	Device ID	(hardcoded A5A5\h)
• AD[15:0]	VID	Vendor ID	(hardcoded FEDC\h)

§ 12.0.2 CONFIG REGISTER 08

Config Register 08 is a 32-bit hard-wired register with the following bit descriptions:

AD	NAME	FUNCTION	VALUE
• AD[31:8]	CC	Class Code	(hardcoded 765432\h)
• AD[7:0]	RID	Revision ID	(hardcoded 01\h)

§ 12.1 WRITE REGISTERS

The Reference Design write registers are initialized by Host I/O Write sequences (§ 7.1). The register to be written is identified during the I/O Write address phase, as follows:

AD[3:0]	Register
0000	Control Register
0001	Memory Address Register
0010	BackEnd Address Register
0011	Length Counter

All other AD[3:0] writes decodes should be ignored.

§ 12.1.0 CONTROL REGISTER

The Control Register is a 32-bit register with the following bit assignment from the AD bus:

AD	NAME	FUNCTION	VALUE
• AD[31]	TME	Test Mode Enable	1 = enable Test Port Pokes
• AD[30:16]	RSVD		
• AD[15:12]	LED[3:0]	Test LED outputs	1 = turn on LED(s)
• AD[11:0]	RSVD		

The LED[3:0] bits are direct outputs to four LEDs (and may be used in any manner convenient to the user). The TME bit provides permission for the Test Port to perform Pokes (and may be rescinded any time).

§ 12.1.1 MEMORY ADDRESS REGISTER/COUNTER (MAR)

The Memory Address Register is a 22-bit register/counter loaded from AD[21:0]. The two Bank bits, BA[0:1], are loaded from AD[21:20] respectively and shall not change during any Memory transfer. The lower 20 bits, loaded from AD[19:0], comprise the SDRAM row address (the upper 12 bits) and the column address (the lower 8 bits). Note: Depending upon the SDRAM Burst Length chosen (as determined by the Mode Register Set cycle) some number of column LSBs may be deleted.

Initially loaded by the Host, the Reference Design is responsible for incrementing and forwarding the incremented value to the Memory MA[11:0] outputs during each RAS and CAS cycle.

§ 12.1.2 BACKEND ADDRESS REGISTER/COUNTER

The BackEnd Address Register is a 20-bit register/counter loaded from AD[19:0]. Initially loaded by the Host, the Reference Design is responsible for incrementing and forwarding the incremented value to the BackEnd ZA[19:0] each and every data phase.

§ 12.1.3 LENGTH COUNTER

With the maximum transfer block size = 512K (32-bit words), the Length Counter may be implemented as a 15-bit counter loaded from AD[18:4]. Initially loaded by the Host, the Reference Design is responsible for decrementing this counter (once for each 16-word transfer).

§ 13 SIGNAL DESCRIPTIONS

§ 13.0 HOST INTERFACE

NAME	DIR	TYPE	FUNCTION
AD[31:0]	IO	TS	multiplexed address/data bus
CBE[3:0]	I		command decode during address phase
/RDY	IO	OC	Host asserts for address & data write; Ref asserts for data read
/REQ	O		not used
/GNT	I		not used

§ 13.1 BACKEND INTERFACE

NAME	DIR	TYPE	FUNCTION
ZA[19:0]	O		address bus
ZD[15:0]	IO	TS	bidirectional data bus
/ZRD	O		read strobe used for all read bursts
/ZWR	O		write strobe used for all write bursts

§ 13.2 MEMORY PORT

NAME	DIR	TYPE	FUNCTION
MA[11:0]	O		row/column address bus
MD[15:0]	IO	TS	bidirectional data bus
BA[1:0]	O		Bank address
CKE	O		Clock Enable
/CS	O		Chip Select
/RAS	O		Row Strobe
/CAS	O		Column Strobe
/WE	O		Write Strobe
DQM	O		Mask

§ 13.3 TEST PORT

NAME	DIR	TYPE	FUNCTION
TA[7:0]	O		address bus
TD[7:0]	IO	TS	bidirectional data bus
STB	I		signifies TA valid and TD valid for writes
ACK	O		signifies TA and incoming data accepted, outgoing data valid

§ 13.4 DAC INPUTS

NAME	DIR	TYPE	FUNCTION
1X[7:0]	I		control inputs to Transform X1

§ 13.5 ADC INPUTS

NAME	DIR	TYPE	FUNCTION
2X[7:0]	I		control inputs to Transform X2

§ 13.6 MISCELLANEOUS

NAME	DIR	TYPE	FUNCTION
CLK1	I		Host clock
CLK2	I		BackEnd and Memory interface clock
LED[3:0]	O		direct outputs to four diagnostic LEDs
SEL[2:0]	I		static inputs; Board Select compared to AD[31:29]
RESET	I		to reinitialize the DX1 i.e. to return to the power-up condition

§14 VERIFICATION PLAN

Although this Reference Design is complex enough to require an extensive test suite for full verification, in the interest of time, the initial plan will be limited to verifying the operation of the Host-Memory and Host-BackEnd transfer operations, that is, to verifying the operations of § 3.0, page 5. For purposes of this exercise, verification means proof by functional simulation.

§ 14.0 HOST-MEMORY VERIFICATION

To verify Host-Memory operations, two tests are proposed, as follows:

Test 1: Memory Write: Length=4K x 32, Address=0, Data=Data++ from 0
Test 2: Memory Read: Length=4K x 32, Address=0

Test 2 is optional but may be used to establish the veracity of Test 1. Implementers of this Reference Design are responsible for ensuring that Memory Writes and Reads are operational for all block lengths up to 2M x 32-bit (= 4M x 16-bit, the address size of the SDRAM).

§ 14.1 HOST-BACKEND VERIFICATION

To verify Host-Memory operations, two tests are proposed, as follows:

Test 3: BackEnd Write: Length=4K x 32, Address=0, Data=Data++ from 0
Test 4: BackEnd Read: Length=4K x 32, Address=0

Implementers of this Reference Design are responsible for ensuring that BackEnd Writes and Reads are operational for all block lengths up to 32K x 32-bit (= 65K x 16-bit, the address size of the BackEnd).

§15 METRICS

Two types of metrics *should be* surveyed during the exercise of the Reference Design, as follows:

1. Qualitative
2. Quantitative

Qualitative metrics include design flexibility, maintainability, modifiability, extensibility, and robustness—properties which are extremely important, often overlooked, and not readily measurable. Unfortunately, these metrics will probably be beyond the scope of the earliest implementations of the Reference Design. However, all implementers are certainly invited to share opinions on those subjects.

Quantitative metrics, by definition, are easier to measure. These include time (to implement the design), resources (consumed in the FPGA), and performance (of the result).

§ 15.0 TIME

It will be of interest to obtain a first-order approximation of the time taken to realize this Reference Design. Here, the goal will be to compare the relative efforts of schematic versus synthesis entry. Implementers should track the following two efforts:

1. Implementation effort (in hours)
2. Verification effort (in hours)

Implementation calculations should be limited to *entry* effort, not the time taken to research interface specifications. (Yet another reason not to have implemented a full PCI interface.) Hypothetically, entry effort is a measure of human activity, and therefore should be platform independent.

Verification calculations should also be limited to measuring the human effort. That is, the platform-dependent simulation cycle time should not be included.

§ 15.1 RESOURCES

FPGA resources consumed should simply be a report of the following items:

1. CLBs *used* (not “occupied”)
2. IO pins used

The number of CLBs used can be obtained in one of the Xilinx reports. A goal in reporting this number is to compare the efficiency differences between a schematic and a synthesis implementation. Note: This comparison is a reflection of the *designer’s style*, not the tools’ implementation. In the most free-style case (in which schematic and synthesis implementations do not identically model each other), variability in this number will be more dependent upon the designers’ choice of per-module implementation.

IO pin *names* and *quantity* used shall not vary between schematic and synthesis implementations. However, pin *locations* are left to the best discretion of the designer.

Other resources, such as clock buffers and 3-states, may be reported but will not be used for comparison purposes (except when illustrating style differences).

§ 15.2 PERFORMANCE

Only Host cycle metrics will be reported, as follows:

1. Memory Write
2. Memory Read
3. BackEnd Write
4. BackEnd Read

In all cases, two parameters are of interest: 1) Burst performance, and 2) Block Transfer performance. Host Write Bursts, by specification, will always be 17 consecutive clocks—one address phase followed by sixteen, 32-bit data phases. Therefore, the metric of interest is the max clock frequency at which the implementation will properly operate. Block Transfer performance shall be a measure of throughput, of how many clocks are required to transfer the specified block size (4K x 32-bit). For purposes of this exercise, the Host bus may be considered to be dedicated to the Reference Design, i.e. without contention from competing resources.

Host Read Bursts, by specification, have some variability allowed in the number of TurnAround cycles permitted. Therefore, performance metrics should report both the number of clocks per burst and per Block Transfer (of 4K x 32-bit).

Note: No Test Port performance metrics need be collected or reported. It is expected that the Test Port will operate somewhere in the 2 MHz range.

§ 15.2.0 TEST 1: MEMORY WRITE

Move a 4K x 32-bit block of data from the Host to Memory. Report the following two metrics:

1. Max clock frequency that can sustain the 17-clock write burst, in MHz.
2. Clocks required to transfer the entire 4K block, given as overall throughput (Mbytes/sec).

§ 15.2.1 TEST 2: MEMORY READ <optional>

Move a 4K x 32-bit block of data from Memory to the Host. Report the following 3 metrics:

1. Number of WAIT states required by the design (excluding the one required TurnAround).
2. Max clock frequency that can sustain the clock read burst, in MHz.
3. Clocks required to transfer the entire 4K block, given as overall throughput (Mbytes/sec).

§ 15.2.2 TEST 1: BACKEND WRITE

Move a 4K x 32-bit block of data from the Host to the BackEnd. Report the following 2 metrics:

1. Max clock frequency that can sustain the 17-clock write burst, in MHz.
2. Clocks required to transfer the entire 4K block, given as overall throughput (Mbytes/sec).

§ 15.2.3 TEST 1: BACKEND READ

Move a 4K x 32-bit block of data from the BackEnd to the Host. Report the following 3 metrics:

1. Number of WAIT states required by the design (excluding the one required TurnAround).
2. Max clock frequency that can sustain the clock read burst, in MHz.
3. Clocks required to transfer the entire 4K block, given as overall throughput (Mbytes/sec).

Figure 16. HOST OPERATION TRUTH TABLE

DX1 REFERENCE DESIGN SPECIFICATION
Rev 0.6

/RDY	AD[31:29]	CBE[3:0]	AD[3:0]	FUNCTION
0	000	0010	0000	Read Status Register
0	000	0010	0001	Read Config Reg 00
0	000	0010	0010	Read Config Reg 08
0	000	0010	0011	Read Length Counter
0	000	0011	0000	write Control Register
0	000	0011	0001	write Memory Address Register
0	000	0011	0010	write BackEnd Address Register
0	000	0011	0011	write Length Counter
0	000	0110	0001	Read BackEnd Data (through X2)
0	000	0110	1000	Read Memory Data (bypass X2) ‡
0	000	0110	1001	Read BackEnd Data (bypass X2) †
0	000	0111	0001	Write BackEnd Data (through X1)
0	000	0111	1000	Write Memory Data (bypass X1) ‡
0	000	0111	1001	Write BackEnd Data (bypass X1) †

† These modes will not be implemented in Phase I.

‡ Memory reads and writes always bypass transforms X2 and X1.