Report

A DDR SDRAM Interface for Xilinx ML505 Evaluation Platform

Author[s]:
Liu, Lisa Ling

Publication Date:
2009

Permanent Link:
https://doi.org/10.3929/ethz-a-006819563

Rights / License:
In Copyright - Non-Commercial Use Permitted
A DDR2 SDRAM Interface for Xilinx ML505 Evaluation Platform

Lisa (Ling) Liu, ETH Zurich

*Computer systems institute, ETH Zürich, Switzerland*

ling.liu@inf.ethz.ch

28 October, 2008, rev. 5 October, 2009

Introduction

The DDR2 DRAM interface for Xilinx ML505 evaluation platform is implemented from the BEE3 DDR2 Controller [4]. The reason to adapt BEE3 DDR2 controller to the ML505 is that these two platforms use the FPGAs from the same family (Virtex-5), and the basic devices of the two DDR2 RAMs have same timing features. Therefore, most of the BEE3 DDR2 controller code, especially the timing enforcement code, which is probably the most difficult part in the memory controller, can be reused. Table 1 shows the difference between these two platforms that may affect the memory controller implementation.

<table>
<thead>
<tr>
<th></th>
<th>BEE3</th>
<th>ML505</th>
</tr>
</thead>
<tbody>
<tr>
<td>FPGA</td>
<td>Virtex-5LX115T-2</td>
<td>Virtex-5LX50T-1</td>
</tr>
<tr>
<td>Memory Part Number</td>
<td>MT36HTF51272(P)Y-53E</td>
<td>MT4HTF3264H(I)Y-53E</td>
</tr>
<tr>
<td></td>
<td>(Micron DDR2 registered DIMM)</td>
<td>(Micron DDR2 unregistered SODIMM)</td>
</tr>
<tr>
<td>Module Density</td>
<td>4GB (dual rank, 14 bit row address, 11 bit column address and 3 bit bank address)</td>
<td>256MB (single rank, 13 bit row address, 10 bit column address and 2 bit bank address)</td>
</tr>
<tr>
<td>Configuration</td>
<td>512Meg x 72</td>
<td>32Meg x 64</td>
</tr>
<tr>
<td>Module Bandwidth</td>
<td>4.3 GB/s</td>
<td>4.3 GB/s</td>
</tr>
<tr>
<td>Memory Clock/Data Rate</td>
<td>3.75ns/533 MT/s</td>
<td>3.75ns/533 MT/s</td>
</tr>
<tr>
<td>Latency (CL-RCD-RP)</td>
<td>4-4-4</td>
<td>4-4-4</td>
</tr>
</tbody>
</table>

The Xilinx ML505 evaluation platform is shipped with a single-rank unregistered 256 MB SODIMM. The DDR2 SODIMM used is generally a Micron MT4HTF3264HY-53E or similar module. Serial Presence Detect (SPD) using an IIC interface to the DDR DIMM is also supported with the FPGA [1]. The logic described in this document implements a DDR2 SODIMM controller, a tiny RISC processor TC5 used to initialize and calibrate the DDR2 SODIMM and a user logic for testing the controller. Figure 1 gives the overview of the design.
The DDR2 memory interface consists of TC5 and DDR2 controller. TC5 ("Tiny Computer v5") is responsible for initializing, refreshing the DDR2 RAMs and calibrating the data pins. It also starts and monitors the tester.

The DDR2 controller provides three FIFOs that exchange address and control, read data and write data between the controller and the tester (user logic). Each unit of the Address FIFO AF stores the address and the read/write command from the tester. The address is 23 bits in length. This is the address of a 32 byte quantity, so the overall address space is 256MB. The address is accompanied by a single bit indicating whether the operation is a Read (1) or a Write (0). The data width of the write data FIFO WB is 128 bits, which is same as the width of the data path of the user logic. The user logic supplies two 128 bit words for each write. Commands are processed in order, and read data is supplied in the order that the addresses were supplied. The read data FIFO RB is 128 bit wide. For each read, the user logic extracts two 128 bit words from RB. The I/O ports of the DDR2 controller are connected to the pins of the DDR2 SODIMM.

The rest of this document lists the modified places in the BEE3 DDR2 controller. The paragraph organization follows the order in [4].
**DDR2 Controller Interface**

Table 2 shows the changed places in the DDR2 controller interface and the reasons for these changes.

<table>
<thead>
<tr>
<th>Signal Name</th>
<th>BEE3 Definition</th>
<th>ML505 Definition</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address</td>
<td>Address[27:0]</td>
<td>Address[22:0]</td>
<td>13(row address) + 8(column address of the block) + 2(bank address) = 23</td>
</tr>
<tr>
<td>LastALU (from TC5)</td>
<td>LastALU[33:0]</td>
<td>LastALU[28:0]</td>
<td>3 (cmd) + 3(altCmd) + 13 (addr) + altAddr(8) + bank(2) = 29</td>
</tr>
<tr>
<td>DQ</td>
<td>DQ[71:0]</td>
<td>DQ[63:0]</td>
<td>64 DQ signals</td>
</tr>
<tr>
<td>DOS, DQS_L</td>
<td>DQS[17:0]</td>
<td>DQS[7:0]</td>
<td>8 DQS signals</td>
</tr>
<tr>
<td></td>
<td>DQS_L[17:0]</td>
<td>DQS_L[7:0]</td>
<td></td>
</tr>
<tr>
<td>BA</td>
<td>BA[2:0]</td>
<td>BA[1:0]</td>
<td>2 bit bank address, 4 banks</td>
</tr>
<tr>
<td>RS</td>
<td>RS[3:0]</td>
<td>no definition</td>
<td>single rank DDR2 SODIMM</td>
</tr>
<tr>
<td>ODT</td>
<td>ODT[1:0]</td>
<td>ODT</td>
<td>one bit ODT</td>
</tr>
<tr>
<td>DIMMreset</td>
<td>DIMMreset</td>
<td>no definition</td>
<td>no reset signal</td>
</tr>
<tr>
<td>KillRank3 (from TC5)</td>
<td>KillRank3</td>
<td>defined in Top module</td>
<td>no rank3, this signal is connected to the chipselect signal in Top module</td>
</tr>
</tbody>
</table>

**Controller Module**

The hierarchy of the controller Verilog is same as the BEE3 DDR2 Controller.

**Controller Implementation**

The controller data and address path is shown in Figure 2.
The difference between the BEE3 and ML505 DDR2 controllers is that the FIFOs are implemented with Block RAMs without ECC. Because the DDR2 SODIMM only has 64 bit DQ signals. Therefore, there are not DoubleError and SignalError outputs from the Read Data FIFO.

**Open Bank Logic**

The implementation of Open Bank Logic is simpler than that in BEE3, because there is only one rank on the SODIMM chip. As seen in Figure 3, it consists of a 4 word x 13-bit LUT RAM that hold the row address for each bank and a valid bit (4 flip-flops total).
In the third paragraph of page 6 in [4], "Refresh is done to one ran every 2 μs to meet the 8μs refresh interval". In the ML505 implementation, the refresh interval is 8μs, since there is only one rank.

**Timing Limit Enforcement**

The timing limits for the DDR2 SODIMM on ML505 are same as for the RDIMM on BEE3, except for the tRFC, which is 75ns in the ML505 case.

Since there is only one rank, the signals TlZ, TacZ, TwriteZ, TreadZ and TrefZ are one-bit signals. The stall logics is changed to the following Verilog implementation correspondingly.

```
assign Stall = (~TlwZ & ReadCommand) | //Write to read
               (~TlrZ & ~ReadCommand) | //Read to write
               (~TacZ  & numOps[1]) |
               (~TwriteZ & numOps[2]) |
               (~TreadZ & numOps[2]) |
               ~TrefZ;
```

**Timing Chain**

Because the DDR2 SODIMM used on ML505 has no command register, one register is taken away from the write timing chain (wd1) and read timing chain (rd1) separately. Figure 4, 5 show the difference.
Figure 4: The write timing chain

Figure 5: The read timing chain
**Calibration**

The calibration is done for two I/O banks at the same time in the ML505 case.

**TC5, TinyAsm assembler**

No changes.

**The "TestTC5" shell**

The Changes of TestTC5 Shell are listed in following paragraph.

1. comment reading EPROM code

   ```
   ; Print FPGA information
   ; Jump aReadSPD + bZero, wRlink4 <= PC;
   ```

2. changed the RAM initialization code to omit the unnecessary loop for all ranks

   ```
   ;---------------------Memory Initialization-------------------
   initMem: Jump aInitRank + bZero, wRlink2 <= PC;
   Jump aRlink4 + bZero, wTrash <= PC; Return
   ```

   ```
   ; Subroutine to initialize one rank, given by RankNum
   initRank: wDelay <= aEight + bTwo; Wait 400 ns (20 cycles).
   Jump aDly + bZero, wRlink <= PC;
   DDRaddr <= aPCHallCmd + bZero; Precharge All
   wDelay <= aTwo + bZero; Wait 4 cycles
   Jump aDly + bZero, wRlink <= PC;
   DDRaddr <= aBank2 + bZero; Load EMR2 (with zero).
   MRS is command = 0. wTrash <= aTrash + bZero;
   Nop
   DDRaddr <= aBank3 + bZero; Load EMR3 (with zero)
   wTrash <= aTrash + bZero; Nop
   DDRaddr <= aBank1 + bZero; Load EMR (with zero)
   wTrash <= aTrash + bZero; Nop
   DDRaddr <= aMRS1cmd + bZero; Load MR (ResetDLL, BL = 4, sequential burst, CL = 4, normal mode, WR = 4, fast exit)
   wDelay <= aDLLdelay + bZero; Wait for DLL to lock.
   Jump aDly + bZero, wRlink <= PC;
   Jump aRefresh + bZero, wRlink1 <= PC; Do a Refresh
   Jump aRefresh + bZero, wRlink1 <= PC; Do another
   DDRaddr <= aMRS2cmd + bZero;
   wTrash <= aTrash + bZero; Nop
   ```
DDRaddr <= aMRS3cmd + bZero;
wTrash <= aTrash + bZero;                      Nop
DDRaddr <= aMRS4cmd + bZero;
wDelay <= aEight + bFour;
Jump aDly + bZero, wRlink <= PC;               Wait 400ns.
DDRaddr <= aPCHallCmd + bZero;
Jump aRlink2 + bZero, wTrash <= PC;            Return.

3. changed the following constants used for configuring the SODIMM

| Rank1:     | 0x4000000; |
| Rank3:     | 0xc000000; |
| Bank1:     | 0x01c000100; cmd = MRSCmd, altCmd = NopCmd |
| Bank2:     | 0x01c000200; |
| Bank3:     | 0x01c000300; |
| PCHallCmd: | 0x01d100000; cmd = PrechargeCmd, altCmd = NopCmd; |
| RefCmd:    | 0x01c800000; cmd = RefreshCmd, altCmd = NopCmd; |
| WriteCmd:  | 0x01e800000; cmd = ReadCmd, altCmd = NopCmd; |
| ReadCmd:   | 0x01e800000; cmd = ReadCmd, altCmd = NopCmd; |
| MRS1cmd:   | 0x01c1d0800; ResetDLL, BL = 4, sequential burst, CL = 4, normal mode, WR = 4, fast exit |
| MRS2cmd:   | 0x01c190800; BL = 4, sequential burst, CL = 4, normal mode, WR = 4, fast exit |
| MRS3cmd:   | 0x01c0f6100; Enable DLL, Rtt = 150, AL = 3, default OCD, DQS enabled, RDQS disabled, outputs enabled |
| MRS4cmd:   | 0x01c016100; Enable DLL, Rtt = 150, AL = 3, OCD exit, DQS enabled, RDQS disabled, outputs enabled |

**Ram Tester**

The main modification in the Tester.v is deleting the SingleError and DoubleError outputs, since the FIFOs in the DDR2 controller do not have ECC. As a result, if an error is detected, command "q" will make the shell print out six numbers. The first is the 23-bit address in the lowest 7 hex digits, and the value of the OddWord and HoldFail (indicating data not as expected) flags in the 8th hex digit. On a failure, two six word quantities are saved, and a second “q” prints the results of the read that actually caused the error.

**Reference**

[2] Micron, 512Mb: x4, x8, x16 DDR2 SDRAM Features, 

[3] Micron, 1Gb: x4, x8, x16 DDR2 SDRAM Features,  

[4] Chuck Thacker, DDR2 DRAM Controller for BEE3, 
   http://research.microsoft.com/research/downloads/Details/12e67e9a-f130-4fd3-9bbd-f9e448cd6775/Details.aspx