Introduction

The emergence of networking and communication systems requiring higher bandwidth interfaces and lower latency for peripheral components lead to the implementation of designs using high-throughput memory with efficient bus utilization and resulted in the development of a new SRAM architecture known as Zero Bus Turnaround (ZBT). The previous generation of static memory types are inefficient as they require idle cycles when they frequently switch between reading from and writing to the memory. IDT, Micron, and Motorola have developed the ZBT SRAM architecture to address this problem.

Altera has developed a ZBT SRAM controller reference design to implement this new memory with Altera® Stratix™ and Stratix GX devices.

This application note describes the functionality of the Altera ZBT SRAM controller reference design and explains the controller directory structure, along with installation, compilation, and simulation, of the design file.

General Description

ZBT SRAM is a synchronous burst (SyncBurst) SRAM with a simplified interface that allows designers to take advantage of the full bandwidth by eliminating turnaround or idle cycles between read and write operations. Turnaround cycles, which are necessary for standard SyncBurst SRAM, significantly reduce the available bandwidth. ZBT SRAM allows the design to use simpler control logic and 100% of the bus utilization with the elimination of these idle cycles. Figure 1 compares the read-to-write transition using SyncBurst SRAM and ZBT SRAM.
ZBT SRAM is available with a pipelined or flow-through interface. In a pipelined interface, the read data is available two clock cycles after the read command is issued, and the write data is required two clock cycles after the write command is asserted. In a flow-through interface, the data-latency requirement is one clock cycle for both read and write operations. Pipelined ZBT SRAM provides faster clock-to-data access, and therefore can be clocked faster than flow-through ZBT SRAM. However, flow-through ZBT SRAM provides one clock cycle less latency than the pipelined variety.

Both pipelined and flow-through ZBT SRAM devices can load a new address each clock cycle or advance the device’s internal counter. The internal counter can be sequenced in a linear or interleaved fashion. ZBT SRAM supports either LVTTL or LVCMOS I/O standards for the memory interfaces.

With the 100% bus utilization that ZBT SRAM provides, there is an increased potential for the devices on the bus to experience bus contention as compared to standard SyncBurst SRAM devices. Bus contention happens when two devices on the bus (e.g. the DQ bus) attempt to drive opposite logic values at the same time. During a write operation, the controller drives the data bus and the ZBT SRAM receives that data, but during a read operation the controller must tri-state its output drivers and receives the data that are being driven by the SRAM. If the controller executes a read followed by a write operation and the SRAM does not stop driving the bus before the controller begins to drive the bus, contention occurs if opposite logic levels are being driven. Alternatively, during a write followed by a read operation, if the turn-off time of the controller is longer than the turn-on time of the SRAM, contention may result. Short-term contention causes a small increase of power consumption and thermal dissipation. For more information on
Altera’s ZBT SRAM controller provides a simplified interface to ZBT SRAM devices. The controller is available in VHDL, and is optimized for the Altera Stratix and Stratix GX device architecture. Stratix and Stratix GX devices can interface with ZBT SRAM at 200 MHz.

An overview of the clock, control, address, and data signals on a ZBT SRAM device is useful to give an overall functionality description and is provided in this section.

Clock Signals

ZBT SRAM uses one system clock input for all clocking purposes including the reads, and writes. All timing references are made with respect to the rising edge of this clock.

Control Signals

The ZBT SRAM uses three different primary types of control signals to interface with memory controllers. For more information on the three primary types of control signals, see Table 1. $RW_N$ is a synchronous input that is asserted high for a read operation and low for a write operation. $BW_N[BWSIZE-1..0]$ are the synchronous byte write enables. Each byte of the data has its own active low byte write enable. The $ADV_LD_N$ signal is a synchronous input that loads the internal registers with a new address when it is asserted with a low signal. If asserted high during the rising edge of the clock, the internal address counter is advanced.

Address Signals

ZBT SRAM uses one synchronous address bus input $SA[ASIZE-1..0]$ for all read and write operations with the memory device.

Data Signals

ZBT SRAM uses one synchronous bidirectional data bus $DQ[DSIZE-1..0]$ for all read and write operations to and from the memory device.
Parameters

Table 1 summarizes the parameters of the ZBT SRAM controller function. You can configure these parameters in the file.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASIZE</td>
<td>Specifies the number of address bits used in the interface.</td>
</tr>
<tr>
<td>DSIZE</td>
<td>Specifies the number of data bits used in the interface.</td>
</tr>
<tr>
<td>BWSIZE</td>
<td>Specifies the number of byte enables used in the interface.</td>
</tr>
<tr>
<td>FLOWTHROUGH</td>
<td>Value determines whether the controller operates in flow-through or pipelined mode. For pipelined operation this parameter is set to 0. For flow-through operation, it is set to 1.</td>
</tr>
</tbody>
</table>

ZBT Supported I/O Standards

ZBT SRAM supports either LVTTL or LVCMOS I/O standards for I/O interfacing. These standards are fully supported in Stratix and Stratix GX devices. For more details on these I/O standards, refer to the Using Selectable I/O Standards in Stratix & Stratix GX Devices chapter in the Stratix Device Handbook, Volume 2.

Reference Design Description

The reference design for the ZBT SRAM memory controller demonstrates generating the ZBT SRAM interface control signals. This design is optimized to run at 200 MHz in Stratix and Stratix GX devices. The Altera ZBT SRAM controller reference design can be implemented in Stratix and Stratix GX devices to provide a simplified interface to a ZBT SRAM device. The reference design implements the ZBT SRAM controller in an EP1S25F780C5 device.

Figure 2 shows a simplified system-level block diagram in which the ZBT SRAM controller connects to a ZBT SRAM module.
Controller Structure & Operation

The ZBT SRAM controller reference design is made up of five lower-level modules: pipe_delay, addr_ctrl_out, data_inout, pipe_stage, and clk_ctrl. Figure 3 shows a lower-level block diagram of the ZBT controller module.
Table 2 describes the function of the five ZBT controller lower-level modules.

<table>
<thead>
<tr>
<th>Module</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>pipe_delay</td>
<td>Provides the proper delay for the input data, depending on whether the controller is configured for pipelined or flow-through ZBT SRAM.</td>
</tr>
<tr>
<td>addr_ctrl_out</td>
<td>Registers the address and control signals before outputting them to the ZBT SRAM.</td>
</tr>
<tr>
<td>data_inout</td>
<td>Registers outgoing data to the ZBT SRAM and provides bidirectional control for the SRAM DQ bus.</td>
</tr>
<tr>
<td>pipe_stage</td>
<td>Registers the input and output user signals for speed purposes in a stand-alone controller configuration. When the controller is interfaced to another module which registers its signals before driving them to the controller, the pipe_stage module will be bypassed.</td>
</tr>
<tr>
<td>clk_ctrl</td>
<td>Instantiates an enhanced Stratix phase-locked loop (PLL) to multiply the input clock and de-skew the internal and external clocks to the ZBT SRAM.</td>
</tr>
</tbody>
</table>

For write operations, the controller aligns the write data to the write command. The controller also handles bus turnaround for transitioning between write and read operations. The following sections provide detailed descriptions of the read and write operations.

**Read Operation**

Use the following guidelines for read operation:

- The ADDR and ADDR_ADV_LD_N signals are set to appropriate values, while RESET_N and RD_WR_N are set to high.
- The data to be read is available at DATA_OUT six clock cycles after its address is loaded to the ADDR address for pipelined and flow-through operations.
- The DM bits are not used for read operations.
- Asserting ADDR_ADV_LD_N low causes the ZBT SRAM to write data to the ADDR address. Asserting ADDR_ADV_LD_N high causes the ZBT SRAM internal two-bit counter to increment to the next address (i.e., the internal two-bit counter takes the place of ADDR’s lower two bits). The SRAM performs a burst read operation for the number of cycles that ADDR_ADV_LD_N is high.

Figure 4 shows a timing diagram for two read operations. To perform multiple reads, change the input signals (i.e., ADDR and ADDR_ADV_LD_N) each cycle while keeping RD_WR_N high.
**Write Operation**

Use the following guidelines for write operations:

- The ADDR, ADDR_ADV_LD_N, DATAIN, and DM signals are driven with the appropriate values, while RESET_N is set to high and RD_WR_N is asserted low.
- The device clocks data into the ZBT SRAM four clock cycles later for pipelined operation and three clock cycles later for flow-through operation. The controller accommodates the latency requirements so that data is presented one clock cycle after the address and control signals for flow-through operation and two clock cycles for pipelined operations.
- The DM bits must be set high for each byte line (either eight or nine bits, depending on the SRAM used) that is written to the SRAM.
- Designers should assert ADDR_ADV_LD_N low to write data to the ADDR address. Asserting ADDR_ADV_LD_N high causes the SRAM internal two-bit counter to increment to the next address performing a burst write operation for the number of cycles that ADDR_ADV_LD_N is high.

**Figure 5** shows a timing diagram for two write operations. To perform multiple writes, change the input signals (i.e., ADDR, ADDR_ADV_LD_N, DATAIN, and DM) each cycle while keeping RD_WR_N low.
Figure 5. Write Operation (Pipelined)  

Note (1)

Note to Figure 5:
(1) Crosshatched areas represent don’t care bits.

Interface Signal

Table 3 describes the ZBT controller I/O port interface signals.

<table>
<thead>
<tr>
<th>Signal</th>
<th>Name</th>
<th>Active</th>
<th>I/O Pin</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLKin</td>
<td>Clock</td>
<td>N/A</td>
<td>Input</td>
<td>System clock.</td>
</tr>
<tr>
<td>RESET_N</td>
<td>Reset</td>
<td>Low</td>
<td>Input</td>
<td>System reset (asynchronous).</td>
</tr>
<tr>
<td>ADDR[ASIZE-1..0]</td>
<td>Memory address</td>
<td>N/A</td>
<td>Input</td>
<td>Memory address for read/write requests. The width is set by ASIZE.</td>
</tr>
<tr>
<td>RD_WR_N</td>
<td>Read/write</td>
<td>N/A</td>
<td>Input</td>
<td>High indicates a read, low indicates a write.</td>
</tr>
</tbody>
</table>
## Logic Options

To ensure proper logic placement for the controller, use the following constraints in the Quartus® II software. These constraints are in the reference file provided by Altera. Use the Assignment Editor to:

- Turn off the **Remove Duplicate Register** logic option.
- Turn on the **Decrease Input Delay to Output Register** logic option for **BW_N** signals.
- Turn on the **Decrease Input Delay to Input Register** logic option for **DQ** signals.
- Turn on the **Fast Input Register** logic option for **ADDR** signals.
- Turn on the **Fast Input Register** for **DQ** signals.
- Turn on the **Fast Output Register** for **DQ** signals.
- Turn on the **Fast Output Register** for **BW_N** signals.

In addition, use the PLL phase shift feature to decrease the $t_{CO}$ times. However, this feature may increase $t_{SU}$ times.

### Table 3. Interface Signals  (Part 2 of 2)

<table>
<thead>
<tr>
<th>Signal</th>
<th>Name</th>
<th>Active</th>
<th>I/O Pin</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADDR_ADV_LD_N</td>
<td>Address advance/load</td>
<td>N/A</td>
<td>Input</td>
<td>Address advance/load input. When high, the ZBT SRAM controller’s internal address counter is advanced. When low, a new address is loaded.</td>
</tr>
<tr>
<td>DATAIN[DSIZE-1..0]</td>
<td>Input data</td>
<td>N/A</td>
<td>Input</td>
<td>Input data bus. The width is set by <strong>DSIZE</strong>.</td>
</tr>
<tr>
<td>DATAOUT[DSIZE-1..0]</td>
<td>Output data</td>
<td>N/A</td>
<td>Output</td>
<td>Output data bus. The width is set by <strong>DSIZE</strong>.</td>
</tr>
<tr>
<td>DM[BWSIZE-1..0]</td>
<td>Data mask</td>
<td>High</td>
<td>Input</td>
<td>Masks individual bytes during data write. The width is set by <strong>BWSIZE</strong>.</td>
</tr>
<tr>
<td>SA[ASIZE-1..0]</td>
<td>Address bus</td>
<td>N/A</td>
<td>Output</td>
<td>Address outputs to the ZBT SRAM.</td>
</tr>
<tr>
<td>ADV_LD_N</td>
<td>Advance/load</td>
<td>N/A</td>
<td>Output</td>
<td><strong>ADV_LD_N</strong> output to ZBT SRAM.</td>
</tr>
<tr>
<td>BW_N[BWSIZE-1..0]</td>
<td>Byte write enables</td>
<td>Low</td>
<td>Output</td>
<td>ZBT byte write enables.</td>
</tr>
<tr>
<td>RW_N</td>
<td>Read/write</td>
<td>N/A</td>
<td>Output</td>
<td>Read/write control signal; output to the ZBT. High indicates a read, low indicates a write.</td>
</tr>
<tr>
<td>DQ[DSIZE-1..0]</td>
<td>Data</td>
<td>N/A</td>
<td>Input/output</td>
<td>Bidirectional data port</td>
</tr>
</tbody>
</table>
Clock Generation

The controller clocking scheme maintains consistent and robust high-frequency operation. The Altera ZBT SRAM controller reference design uses one dedicated input clock (CLKIN). This input clock feeds one of the enhanced PLLs in the Stratix and Stratix GX devices. This PLL boosts the input clock and adjust the phase of the internal and external clocks of the ZBT SRAM controller interface.

Programmable Delay Circuitry and Bus Contention

Bus contention occurs when two or more devices try to drive a bus to different logic levels. This problem can happen when shifting from a write to a read cycle with no bus idle cycles in between. The ZBT SRAM controller performing write operation must go to high impedance before the ZBT SRAM output drivers turn on for the following read cycle. See Figure 6 for ZBT SRAM turn-on and turn-off timing.

Figure 6. ZBT SRAM Turn-On & Turn-Off Timing

The Stratix and Stratix GX devices’ I/O driver has a rising edge delay cell in the output enable path. When implemented through the Quartus II software, this cell delays the output driver enable time. There is no time penalty to the output enable path when this delay cell is disabled. The contention window is defined as \( t_{ZX} - t_{XZ} \) which measures the difference between turn-on and turn-off time of the output driver at the same temperature and voltage. However, the low resistance path between the power supply and GND will result in increased power dissipation.
Post Route Performance

The post-route performance results for the ZBT SRAM controller are shown in Tables 4 and 5. The data shown in each table are obtained by shifting the PLL output clock with 300 ps.

<table>
<thead>
<tr>
<th>Device</th>
<th>Internal fMAX (MHz)</th>
<th>tCO (ns)</th>
<th>tSU (ns)</th>
<th>Throughput (Mbps)</th>
<th>Total LEs (3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EP1S25F780C5</td>
<td>200</td>
<td>5.160</td>
<td>3.379</td>
<td>800</td>
<td>301</td>
</tr>
</tbody>
</table>

Notes to Table 4:
(1) The tCO and tSU values are worse-case values without setting any global timing requirements for pins available in the system and no phase shifting is applied to the output clock of the PLL.
(2) The timing data may vary according to the version of the Quartus II software used in compilation.
(3) LE: logic element

<table>
<thead>
<tr>
<th>Byte</th>
<th>tCO (ns)</th>
<th>tSU (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>tCO</td>
<td>tSU</td>
</tr>
<tr>
<td></td>
<td>Required Set in Quartus II</td>
<td>Compilation Report</td>
</tr>
<tr>
<td>DQ</td>
<td>3,500</td>
<td>3,303</td>
</tr>
<tr>
<td>BW_N</td>
<td>2,500</td>
<td>2,595</td>
</tr>
</tbody>
</table>

Notes to Table 5:
(1) The tCO requirements for BW_N pins are set to 2.5 ns. Although the tCO value from the Quartus II software compilation fails to meet the requirement set in the Quartus II software, the compilation report shows that the Quartus II software provides tCO for BW_N signals within the specification of this project.
(2) The data may vary according to the version of the Quartus II software used for compilation.

Contention Analysis

The Stratix and Stratix GX devices’ turn-on time and turn-off time are 3.3 ns and 2.2 ns, respectively, where the timing paths of these values are inclusive of the output buffer and output-enable routings. The IDT71V3548S133 133-MHz pipelined ZBT SRAM has a 1.5-ns turn-on time and a 1.5- to 3.0-ns turn-off time. Using these values, if the controller performs a read operation followed by a write operation, there is no contention because the ZBT SRAM turn-off time is faster than the controller turn-on time. For a write operation followed by a read operation, there is a possibility of no more than 0.7 ns of contention (2.2 ns – 1.5 ns = 0.7 ns).
The IDT 71V3556P 200-MHz pipelined ZBT SRAM has a 3.2-ns turn-on time and a 3.0-ns turn-off time. Using these values, if the controller performs a read operation followed by a write operation, there is no contention because the ZBT SRAM turns off faster than the controller turns on. Besides that, there is also no contention for a write operation followed by a read operation since the Stratix and Stratix GX devices turn-off time is 1-ns faster than the ZBT SRAM turn-on time (2.2 ns – 3.2 ns = –1.0 ns).

See the DC & Switching Characteristics chapter of the Stratix Device Handbook for more information on txZ and tZX numbers. Even though the data sheet shows similar numbers for both txZ and tZX, typically tZX numbers are 2 ns slower than txZ numbers for Stratix and Stratix GX devices. The Quartus II software and the Stratix Device Handbook provide the worst case numbers between txZ and tZX. If I/O standard txZ time is needed, subtract 2 ns from the numbers reported in the Quartus II software.

**Getting Started**

The Altera ZBT SRAM controller reference design provides solutions for integrating ZBT SRAM into a system. This section describes how to install and use the reference design on a PC.

**Hardware & Software Requirements**

These instructions assume that the designer has:

- A PC with the Quartus II software version 3.0 SP1 (or higher) installed in the default location is used.
- Familiarity with the Quartus II software.
- ModelSim version 5.7c software is installed for simulation.

**Design Installation**

Altera provides the ZBT SRAM controller reference design as VHDL compressed file. This reference design can be downloaded from the Altera web site at www.altera.com. To install the files, perform the following steps:

1. Create a new directory on the hard drive to save the files.
2. Save the zip file, zbt_rd_vhdl_str_v1.0.0.zip, into the directory just created. The file can be deleted once the installation is done.
3. Extract the contents of the zbt_rd_vhdl_str_v1.0.0.zip file to the directory created in step 1.
Figure 7 shows the directory structure created for the reference design and displays selected VHDL files.

Figure 7. ZBT SRAM Controller Directory File Structure

Compilation & Simulation of Reference Designs

Altera provides the reference design source files to synthesize, place-and-route, and simulate the design. This section describes the design flow for the reference design, including compiling and synthesizing in the Quartus II software and simulating in the ModelSim software.

The results for each step are included in the reference design. Therefore, you do not need to perform each step unless the design files have been altered. For example, the simulation results can be viewed without first compiling the design because the Quartus II software place-and-route results are included with the reference design. The compilation and simulation results are in the lib and sim-lib directories, respectively.
Compile in the Quartus II Software

This section describes how to compile and place-and-route the design using the Quartus II software. The lib directory contains the Quartus II software version 3.0 SP1 project files, including the constraint files needed for the design to meet the required clock frequencies and I/O timing. Altera provides the project files listed in Table 6 in the lib directory.

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pll.vhd</td>
<td>PLL instantiation file created by the MegaWizard® Plug-In Manager. This file instantiates the parameterized altpil function, which generates a PLL in the Stratix and Stratix GX devices. This PLL is instantiated in the top-level top.vhd file and feeds all the lower-level modules.</td>
</tr>
<tr>
<td>addr_ctrl_out.vhd</td>
<td>Registers the address and control bits.</td>
</tr>
<tr>
<td>data_inout.vhd</td>
<td>Controls the input/output functions of the controller.</td>
</tr>
<tr>
<td>pipe_delay.vhd</td>
<td>Contains the pipeline delay module for flow-through and pipelined ZBT SRAM.</td>
</tr>
<tr>
<td>pipe_stage.vhd</td>
<td>Transmits and receives signals.</td>
</tr>
<tr>
<td>top.vhd</td>
<td>Instantiates all the files described above and is the top-level file.</td>
</tr>
</tbody>
</table>

To compile the Altera-provided project files, use the following steps:

1. Run the Quartus II software.

2. Choose Open Project (File menu).

3. Select the top.quartus file from the \<work dir>\zbt_rd_vhdl_str_v1.0.0\lib directory and click Open.

4. Choose Compile Mode (Processing menu).

5. Choose Start Compilation (Processing menu).

Simulate in the ModelSim Software

The sim_lib directory contains two VHDL test bench files:

- zbt_ctrl_top_flow_tb.vhd for flow-through mode and
- zbt_ctrl_top_pipe_tb.vhd for pipelined mode, which instantiates the ZBT SRAM controller and the ZBT SRAM model (idt71v3556.vhd, a pipelined model). The test bench demonstrates the functionality of the controller by first writing into the memory and then reading from the
memory and comparing the data. Because the SRAM model is behavioral and does not contain timing information, additional delay is added to the signals to and from the SRAM in the testbench to account for board delay and clock-to-output time. Different board-delay scenarios can be modeled by changing these delays.

Altera provides two scripts (zbt_ctrl_top_flow_tb.do and zbt_ctrl_top_pipe_tb.do) to perform functional simulation in the ModelSim software. These scripts create a work library and pre-compile the correct simulation libraries for functional simulation. They also compile the controller source files, the model, and the test bench file, and display the appropriate waveforms. If the files are not in the default installation locations, the script must be updated so that the paths point to the locations of the installed reference design and the Quartus II software.

Use the following steps to perform functional simulation:

1. Run the ModelSim version 5.6a software.
2. Change the working directory to the <work dir>\zbt_rd_vhdl_str_v1.0.0\sim_lib\functional directory.
3. Type the following commands in the Command window: do zbt_ctrl_top_flow_tb.do (for flow-through mode)

ZBT SRAM addresses the need for high-throughput, low-latency static memory and is useful for designs with many successive read and write transactions. The transition latency is minimized because ZBT SRAM does not require turnaround cycles between reads and writes. Altera Stratix and Stratix GX devices support ZBT SRAM system frequency up to 200 MHz. The Altera ZBT SRAM controller reference design is a ready-to-use memory controller that interfaces Stratix and Stratix GX devices with a ZBT SRAM device at up to 200 MHz.