AC446
Application Note
Optimization Techniques to Improve DDR Throughput
for RTG4 Devices - Libero SoC v11.9 SP1
Microsemi makes no warranty, representation, or guarantee regarding the information contained herein or the suitability of its products and services for any particular purpose, nor does Microsemi assume any liability whatsoever arising out of the application or use of any product or circuit. The products sold hereunder and any other products sold by Microsemi have been subject to limited testing and should not be used in conjunction with mission-critical equipment or applications. Any performance specifications are believed to be reliable but are not verified, and Buyer must conduct and complete all performance and other testing of the products, alone and together with, or installed in, any end-products. Buyer shall not rely on any data and performance specifications or parameters provided by Microsemi. It is the Buyer’s responsibility to independently determine suitability of any products and to test and verify the same. The information provided by Microsemi hereunder is provided “as is, where is” and with all faults, and the entire risk associated with such information is entirely with the Buyer. Microsemi does not grant, explicitly or implicitly, to any party any patent rights, licenses, or any other IP rights, whether with regard to such information itself or anything described by such information. Information provided in this document is proprietary to Microsemi, and Microsemi reserves the right to make any changes to the information in this document or to any products and services at any time without notice.

About Microsemi

Microsemi, a wholly owned subsidiary of Microchip Technology Inc. (Nasdaq: MCHP), offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world’s standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Learn more at www.microsemi.com.
# Optimization Techniques to Improve DDR Throughput for RTG4 Devices - Libero SoC V11.9 SP1

1. **Design Requirements**
2. **Optimization Techniques**
   - Frequency of Operation
   - Burst Length
   - AXI Master without Write Response State
   - Read Address Queuing
   - Series of Writes and Reads
   - DDR Configuration Tuning
3. **Design Description**
4. **Hardware Implementation**
   - SmartDesign Components
   - UART_IF_0
   - Configuring the FDDR Subsystem
5. **Simulation Using Micron DDR3 SDRAM Model**
6. **Running the Design**
   - Board Jumper Settings
   - Host PC to Board Connections
   - USB Driver Installation
   - Steps to Run the Design
7. **DDR3 SDRAM Bandwidth**
   - Simulation Results
   - Board Results
8. **Conclusion**
Figures

Figure 1  FDDR Datapath for AXI/AHB Master ......................................................... 2
Figure 2  Write Transaction Timing Diagram without Write Response .......................... 4
Figure 3  FDDR Datapath for AXI/AHB Master .......................................................... 5
Figure 4  AXI to DDR3 Address Mapping ................................................................. 5
Figure 5  Top-Level Block Diagram of the Reference Design ...................................... 7
Figure 6  Command Decoding ................................................................................. 8
Figure 7  Top-Level SmartDesign Component .......................................................... 9
Figure 8  DDR_AXI_0 SmartDesign Component ....................................................... 10
Figure 9  UART_IF_0 SmartDesign Component ....................................................... 11
Figure 10 FDDR Memory Configuration ................................................................. 12
Figure 11 FDDR Memory Timing ............................................................................ 12
Figure 12 Instantiating Simulation Model ................................................................ 13
Figure 13 Stimulus Settings .................................................................................... 14
Figure 14 Waveform Settings ................................................................................ 14
Figure 15 AXI Master Signals During Write Operation ............................................ 15
Figure 16 FDDR Signals During Write Operation ..................................................... 15
Figure 17 AXI Master Signals During Read Operation ............................................. 15
Figure 18 FDDR Signals During Read Operation ..................................................... 15
Figure 19 Transcript Window during Write Operation .............................................. 16
Figure 20 USB Serial 2.0 Port Properties ................................................................. 17
Figure 21 RTG4 DDR Bandwidth Utility ................................................................... 18
Figure 22 RTG4 DDR 3 Bandwidth Connection Status ............................................ 19
Figure 23 Throughput for 2KB Write ..................................................................... 19
Figure 24 Throughput for 2KB Read .................................................................... 19
Figure 25 Throughput for 4KB Write .................................................................... 20
Figure 26 Throughput for 4KB Read .................................................................... 20
Figure 27 Throughput for 8KB Write .................................................................... 20
Figure 28 Throughput for 8KB Read .................................................................... 21
Figure 29 Throughput for 16KB Write .................................................................. 21
Figure 30 Throughput for 16KB Read .................................................................. 21
Figure 31 Throughput for 32KB Write .................................................................. 22
Figure 32 Throughput for 32KB Read .................................................................. 22
Figure 33 RTG4 Development Kit Board ................................................................. 23
Figure 34 USB 3.0 Serial Port Properties ................................................................. 27
Figure 35 Read Error ............................................................................................. 27
### Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1</td>
<td>Design Requirements</td>
<td>3</td>
</tr>
<tr>
<td>Table 2</td>
<td>Tuned DDR Timing Parameters</td>
<td>6</td>
</tr>
<tr>
<td>Table 3</td>
<td>FCCC Generated Clocks</td>
<td>7</td>
</tr>
<tr>
<td>Table 4</td>
<td>RTG4 FPGA Development Kit Jumper Settings</td>
<td>16</td>
</tr>
<tr>
<td>Table 5</td>
<td>Total Number of 16-Beat Bursts</td>
<td>23</td>
</tr>
<tr>
<td>Table 6</td>
<td>DDR3 SDRAM Bandwidth - Simulation Results</td>
<td>24</td>
</tr>
<tr>
<td>Table 7</td>
<td>DDR3 SDRAM Bandwidth - Board Results</td>
<td>25</td>
</tr>
</tbody>
</table>
1 Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

1.1 Revision 3.0
Updated the document for Libero SoC 11.9 SP1.

1.2 Revision 2.0
Updated the document for Libero SoC 11.8 SP2.

1.3 Revision 1.0
The first publication of this document.
Optimization Techniques to Improve DDR Throughput for RTG4 Devices - Libero SoC V11.9 SP1

This application note describes the optimization techniques for improving DDR3 SDRAM throughput using a reference design for the RTG4™ field programmable gate array (FPGA) Development Kit. It also describes simulating the design using the Micron DDR3 SDRAM simulation model.

The RTG4 devices have two fabric DDR (FDDR) blocks. Each FDDR subsystem is a hardened ASIC block used to interface with DDR2, DDR3, or low power DDR1 (LPDDR1) memories and access high-speed DDR memories for high-speed data transfer.

The DDR memory connected to the FDDR subsystem is accessed by the master logic implemented in the FPGA fabric. The FPGA fabric master communicates with the FDDR subsystem through AXI or AHB interfaces. The following figure shows the data path for DDR memory access by the FDDR subsystem.

![FDDR Datapath for AXI/AHB Master](image)

The AXI master interface performs burst transfers that provides an efficient data access path with maximum throughput. Throughput of DDR depends on various parameters. This reference design describes optimization techniques that improve efficiency and provide better throughput.

An AXI master on the fabric logic performs AXI burst transfers of 2 KB, 4 KB, 8 KB, 16 KB, and 32 KB size. During write operation, the AXI master writes incremental patterns to the DDR3 memory and returns the AXI clock count to the host PC (through the UART interface) for throughput calculation. During read operation, the AXI master reads data from DDR3 memory and checks for data mismatch in the fabric logic. If there is a data mismatch then the error status is indicated through an on-board LED. A separate demo utility application calculates throughput using AXI clock count input.
2.1 Design Requirements

The following table lists the requirements for the reference design.

<table>
<thead>
<tr>
<th>Design Requirements</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware Requirements</td>
<td>RTG4-DEV-KIT Rev C</td>
</tr>
<tr>
<td>- RTG4 Development Kit</td>
<td>- 12 V adapter</td>
</tr>
<tr>
<td>- USB A to Mini-B cable</td>
<td>- Any 64-bit Windows Operating System</td>
</tr>
<tr>
<td>Software Requirements</td>
<td>v11.9 SP1</td>
</tr>
<tr>
<td>- Libero® System-on-Chip (SoC)</td>
<td>- USB to UART drivers</td>
</tr>
<tr>
<td>- FlashPro programming software</td>
<td>- Host PC demo utility application</td>
</tr>
</tbody>
</table>

2.2 Optimization Techniques

Demand for performance continues to increase, and existing memories face limitations in throughput at higher data rates. The following optimization techniques help to achieve maximize throughput:

- Frequency of operation
- Burst length
- AXI master without write response state
- Read address queuing
- Series of writes and reads
- DDR configuration tuning

2.2.1 Frequency of Operation

The FDDR subsystem supports clock management dividers. The divider ratios can be selected from the Clock Configurator for DDR clocks (FDDR_CLK) and DDR_FIC clock. This reference design uses 2:1 ratio between FDDR_CLK and DDR_FIC, because the best throughput result is obtained for this ratio. The reference design uses 64-bit AXI interface in the FPGA fabric, which operates at maximum possible frequency of 166.66 MHz. The FDDR_CLK frequency is configured at 333 MHz.

2.2.2 Burst Length

The MDDR and FDDR subsystems support the DRAM burst lengths of 4, 8, or 16, depending on the configured bus width and the DDR type. The AXI transaction controller in the MDDR and FDDR subsystems support up to 16-beat burst read and write. The burst length (write and read) of AXI and DRAM affect the performance, but by setting the maximum supported burst length for DDR SDRAM and AXI interface optimal performance can be achieved. The reference design uses a DDR SDRAM burst length of 8 and an AXI write and read burst length of 16.

Note: The reference design is run on the RTG4 Development Kit board, which has the RT4G150 device and a DDR3 SDRAM from Micron with the part number MT41K256M8.

2.2.3 AXI Master without Write Response State

When the AXI master sends the last word (D[A15]), the WLAST signal is asserted, which indicates that the last word of the current write burst is transferred. When the AXI slave in the DDR subsystem accepts all the data items, it asserts the write response (BVALID) back to the master to indicate that the write transaction is complete, as shown in the following figure. When using the AXI protocol, the AXI master waits for the write response before initiating the next write transaction. However, waiting for write response increases latency and decreases overall throughput. The AXI master can send the second
burst write address (B) without waiting for the write response of the first burst. This improves the write throughput by reducing the wait states.

This application note is focused on optimal throughput, and therefore the write response channel is not verified. It is recommended that when using this technique, the write response channel is used concurrently with starting the next transfer to ensure that the previous write data is accepted. The AXI protocol has a defined methodology for handling the termination of write burst transaction. This must be followed if the write response channel returns an incorrect value.

**Figure 2** Write Transaction Timing Diagram without Write Response

### 2.2.4 Read Address Queuing

The FDDR subsystem supports up to four outstanding read transactions. The following figure shows the burst read address queuing timing diagram. In 2:1 clock ratio, the FDDR controller starts the burst read transaction before the FIFO FULL command, which allows the AXI master to send five burst read addresses. The AXI master increments the burst read address as long as the AXI slave in the DDR subsystem asserts the ARREADY signal. The burst read address queuing significantly increases the read throughput compared to the normal AXI read sequence. Table 7, page 25 shows this significant increase. Read address queuing does not reduce the initial latency associated with a DDR memory read access. However, by issuing multiple reads in sequence, the read throughput is increased significantly when compared to normal AXI read sequence.
2.2.5 Series of Writes and Reads

The FDDR subsystem’s performance depends on the method of data transfer between the DDR SDRAM and the AXI master. The following methods of data transfer reduce optimal performance:

- Single-beat burst read and write operation
- Random read and write operation
- Switching between read and write operation

The FDDR subsystem’s performance increases while performing a series of reads or writes from the same bank and row. The following figure shows the AXI to DDR3 address mapping for the DDR3 SDRAM on the RTG4 Development Kit board.

2.2.6 DDR Configuration Tuning

The DDR SDRAM datasheet provides the timing parameters required for the memory operation in terms of time units. These timings must match with the configuration registers in the FDDR controller. The timing parameters are entered in terms of number of DDR clock cycles in the DDR Configurator. The selection of minimum write or read delay values may result in optimal performance. Implementing this approach depends on the vendor memory device that is used and its DDR controller and PHY blocks. It also requires extensive memory testing to ensure that the memory transfers are stable. For this reference design, the FDDR configuration register file is provided along with the design file. See Appendix: Design Files, page 26.
The following table lists the key timing parameter values used in the reference design.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values for Reference Design</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAS</td>
<td>5 (clks)</td>
</tr>
<tr>
<td>RAS min</td>
<td>12</td>
</tr>
<tr>
<td>RAS max</td>
<td>22528</td>
</tr>
<tr>
<td>RCD</td>
<td>5</td>
</tr>
<tr>
<td>RP</td>
<td>5</td>
</tr>
<tr>
<td>REFI</td>
<td>2592</td>
</tr>
<tr>
<td>RC</td>
<td>17</td>
</tr>
<tr>
<td>RFC</td>
<td>54</td>
</tr>
<tr>
<td>WR</td>
<td>5</td>
</tr>
<tr>
<td>FAW</td>
<td>10</td>
</tr>
</tbody>
</table>

For more information about improved throughput numbers after using optimization techniques, see Table 6, page 24 and Table 7, page 25.

2.3 **Design Description**

The reference design consists of the following components:

- FDDR subsystem
- CoreABC with APB master
- CoreAXI
- CoreUART
- On-chip 25/50 MHz RC oscillator
- Fabric CCC (FCCC)
- AXI master (AXI_INTERFACE)
- Command decoder (CMD_Decoder)

The following figure shows the block diagram of the reference design.
The DDR_FIC interface is configured to use an AXI interface. CoreABC is used to initialize FDDR with required tuned configuration values through the APB interface. CoreUART and control_logic HDL modules are used as interfaces for writing to the host PC demo utility application. The demo utility application initiates AXI read and write operation using the CoreUART interface. The FDDR is configured to use the DDR3 memory interface through the AXI master interface in the fabric logic. Fabric CCC (FCCC) is configured to provide 166.6 MHz reference clock to the fabric logic. The on-chip 25/50 MHz RC oscillator is the reference clock source for the FCCC.

**Table 3 • FCCC Generated Clocks**

<table>
<thead>
<tr>
<th>Clock Name</th>
<th>Frequency in MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>FDDR_CLK</td>
<td>333</td>
</tr>
<tr>
<td>DDR_FIC</td>
<td>166.5</td>
</tr>
<tr>
<td>APB_CLK</td>
<td>41.667</td>
</tr>
</tbody>
</table>

The command decoder receives command from the demo utility application through the UART interface, which is implemented using the CoreUART IP and generates read, write, write size, and read size signals. The following figure shows command decoding.
The AXI master block consists of the following components:

- AXI read channel
- AXI write channel
- Write latency counter
- Read latency counter

It performs the write or read operation based on the input signals from the command decoder. During the write operation, the AXI master writes incremental pattern into the DDR3 memory and then measures write latency (number of AXI clocks) that is sent to the host PC demo utility application for throughput calculation. During the read operation, the AXI master reads the DDR3 memory and measures read latency (number of AXI clocks) that is sent to the host PC demo utility application for throughput calculation. During read operation, if there is a data mismatch, an error status signal is shown using the on-board active LOW debug LED. The write latency counter counts the AXI clocks between AWVALID of the first data and WLAST of the last data.

Similarly, the read latency counter counts the AXI clocks between ARVALID of the first data and RLAST of the last data. During the write operation, the write address (AWADDR) starts from 0x00000000 and is incremented by 128 (16-beat burst). During the read operation, the read address (ARADDR) also starts from 0x00000000 and is incremented by 128. After each write or read operation, the AXI master sends the latency count value to the host PC for throughput calculation. The demo utility displays the number of AXI clocks and throughput value for write and read operations.

The following equation is applied to calculate the throughput:

\[
\text{Bandwidth (MB/s)} = \left(16 \div \left(\text{Total number of AXI clocks} \div \text{Total number of 16-beat bursts}\right)\right) \times 8 \times \text{AXI Clock (MHz)}
\]

For information about RTG4 FDDR, see the UG0573: RTG4 FPGA DDR Controller User Guide.

For information about accessing DDR3 memory using AXI interface, see the DG0625: Interfacing RTG4 with the External DDR3 Memory through the DDR Controller Demo Guide.
2.4 Hardware Implementation

This section describes:

• SmartDesign components
• FDDR configurations

The following figure shows the top-level SmartDesign component.

*Figure 7 • Top-Level SmartDesign Component*

2.4.1 SmartDesign Components

**DDR_AXI_0**: handles the data transactions between the DDR_FIC interface and the DDR3 SDRAM memory.

**UART_IF_0**: handles the communication between the host PC and the RTG4 Development Kit.

**CMD_Decoder_0**: used to receive command from the host PC and generate required read, write, read size, and write size signals.

2.4.1.1 DDR_AXI_0

**DDR_AXI_0** consists of the FDDR subsystem and the AXI_IF master logic. The AXI_IF_0 is an RTL code that implements the AXI read and write transactions. It receives the read/write commands, burst length (RLEN and WLEN), address, and data as inputs. Based on the inputs received, it communicates with the DDR3 memory through the FDDR subsystem.

The following figure shows the DDR_AXI_0 SmartDesign component.
2.4.2 UART_IF_0

The UART_IF_0 SmartDesign component controls the UART communication between the host PC demo utility application and the AXI master logic. The COREUART_0 IP receives the UART signals from the host PC user interface. Control_logic_0 is an HDL wrapper for COREUART_0. It collects the data from COREUART_0 and sends command to the command decoder. For different burst size write and read operations, command is received from the demo utility and sent to the command decoder. The command decoder generates required read/write signals for the AXI master logic.

The following figure shows the UART_IF_0 SmartDesign component.
2.4.3 Configuring the FDDR Subsystem

This section describes how to configure the FDDR subsystem to perform data transactions with the external DDR3 memory device. For more information about the RTG4 DDR memory controller with initialization IP core, see the UG0573: RTG4 FPGA DDR Memory Controller User Guide.

In the following procedure, the reference design uses West FDDR to access DDR3 with 32-bit data width and no ECC. The following steps describe how to configure FDDR and access it from the AXI master in the fabric logic:

1. On the FDDR configurator General tab shown in the following figure, set the DDR memory settling time to 200 µs, and click Import Configuration to initialize the DDR memory.

The FDDR subsystem registers must be initialized before accessing DDR memory through the FDDR subsystem. The FDDR configuration register file is provided along with the design file. See Appendix: Design Files, page 26.
The following figure shows the memory timing settings according to the tuned DDR configuration file.

Figure 11 • FDDR Memory Timing
2. Click **Finish**.
3. Instantiate the custom logic-AXI master, UART_IF, and a command decoder, and connect as shown in Figure 8, page 10 and Figure 9, page 11.

## 2.5 Simulation Using Micron DDR3 SDRAM Model

The following steps describe how to set up and simulate the reference design:

1. Obtain the Micron DDR3 memory model files—the RTG4 Development Kit board has the DDR3 SDRAM from Micron with the part number MT41K256M8. The memory model used in the reference design supports this device. See Appendix: Design Files, page 26.
2. Copy the `ddr3.v` and `ddr3_parameters.vh` simulation model files to the `<Libero SoC project directory>\stimulus directory`.
3. Instantiate and connect the DDR3 memory model in the testbench, as shown in the following figure.

**Figure 12 • Instantiating Simulation Model**

```vhdl
ddr3 DDR3_0 {
  .rst_n(FDDR_RESET_N),
  .clk(FDDR_CLK),
  .ck_n(FDDR_CLK_N),
  .ck_n(FDDR_CLK_N),
  .csn(FDDR_CS_N),
  .ras_n(FDDR_RAS_N),
  .cas_n(FDDR_CAS_N),
  .we_n(FDDR_WE_N),
  .addr(FDDR_ADDR[14:0]),
  .ba(FDDR_BA),
  .dm_tdaq(FDDR_DM_RDQS[0]),
  .dq(FDDR_DQ[7:0]),
  .dqz(FDDR_DQ[7:0]),
  .odt(FDDR_ODT),
  .tdqz_n()
};

ddr3 DDR3_2 {
  .rst_n(FDDR_RESET_N),
  .clk(FDDR_CLK),
  .ck_n(FDDR_CLK_N),
  .csn(FDDR_CS_N),
  .ras_n(FDDR_RAS_N),
  .cas_n(FDDR_CAS_N),
  .we_n(FDDR_WE_N),
  .addr(FDDR_ADDR[14:0]),
  .ba(FDDR_BA),
  .dm_tdaq(FDDR_DM_RDQS[0]),
  .dq(FDDR_DQ[7:0]),
  .dqz(FDDR_DQ[7:0]),
  .odt(FDDR_ODT),
  .tdqz_n()
};

ddr3 DDR3_3 {
  .rst_n(FDDR_RESET_N),
  .clk(FDDR_CLK),
  .ck_n(FDDR_CLK_N),
  .csn(FDDR_CS_N),
  .ras_n(FDDR_RAS_N),
  .cas_n(FDDR_CAS_N),
  .we_n(FDDR_WE_N),
  .addr(FDDR_ADDR[14:0]),
  .ba(FDDR_BA),
  .dm_tdaq(FDDR_DM_RDQS[0]),
  .dq(FDDR_DQ[7:0]),
  .dqz(FDDR_DQ[7:0]),
  .odt(FDDR_ODT),
  .tdqz_n()
};

ddr3 DDR3_1 {
  .rst_n(FDDR_RESET_N),
  .clk(FDDR_CLK),
  .ck_n(FDDR_CLK_N),
  .csn(FDDR_CS_N),
  .ras_n(FDDR_RAS_N),
  .cas_n(FDDR_CAS_N),
  .we_n(FDDR_WE_N),
  .addr(FDDR_ADDR[14:0]),
  .ba(FDDR_BA),
  .dm_tdaq(FDDR_DM_RDQS[0]),
  .dq(FDDR_DQ[7:0]),
  .dqz(FDDR_DQ[7:0]),
  .odt(FDDR_ODT),
  .tdqz_n()
};
```

4. Ensure that the `ddr3.v` file is included at the top of the testbench file. The reference design uses four instances of DDR3 models with data width of x8.
5. Set the testbench in which the DDR3 memory model is instantiated as active stimulus. The following figure shows the settings under stimulus hierarchy.
6. Click **Project > Project Settings > Simulation Options > Waveforms**. The following figure shows the waveforms settings on the right panel.

**Figure 14 • Waveform Settings**

7. Select the **Include DO file** check box and enter `wave.do` in the box, as shown in the preceding figure. The following figure shows the AXI master signals, command from command decoder, and AXI clock count for write operation, which calculates the write bandwidth.
Figure 15 • AXI Master Signals During Write Operation

The following figure shows the FDDR signals. The AXI master writes 16 KB data into the DDR3 SDRAM. The data is written into Row 0 of banks (0-3).

Figure 16 • FDDR Signals During Write Operation

The following figure shows the AXI master signals, command from command decoder, and AXI clock count for read operation, which calculates the read bandwidth.

Figure 17 • AXI Master Signals During Read Operation

In the preceding figure, the read operation AXI clock count and the DDR_RD_WRT_ERR signal that is used to check for data mismatch between write and read operation is highlighted.

The following figure shows the FDDR signals. The AXI master reads 16 KB data from the DDR3 SDRAM. The data is read from Row 0 of banks (0-3).

Figure 18 • FDDR Signals During Read Operation

The following figures show the transcript window messages during write and read operations.
2.6 Running the Design

The reference design is designed to run on the RTG4 Development Kit board. For more information about the RTG4 Development Kit board, see [http://www.microsemi.com/products/fpga-soc/design-resources/dev-kits/rtg4-development-kit](http://www.microsemi.com/products/fpga-soc/design-resources/dev-kits/rtg4-development-kit).

2.6.1 Board Jumper Settings

The following table lists the jumpers that must be connected on the RTG4 Development Kit board.

<table>
<thead>
<tr>
<th>Jumper</th>
<th>Pin (From)</th>
<th>Pin (To)</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>J32, J27, J26, J23, J21, J19, J17, J11</td>
<td>1</td>
<td>2</td>
<td>Default</td>
</tr>
<tr>
<td>J16</td>
<td>2</td>
<td>3</td>
<td>Default</td>
</tr>
</tbody>
</table>
2.6.2 **Host PC to Board Connections**

Connect the host PC to the J47 connector using the USB cable (mini USB to Type A).

2.6.3 **USB Driver Installation**

Install the FTDI D2XX driver for serial terminal communication through FTDI mini USB cable. The drivers and installation guide are available at:


Ensure that the USB to UART bridge drivers are automatically detected. This can be verified in the Device Manager of the host PC, as shown in the following figure. The FTDI USB to UART converter enumerates four COM ports. For USB 2.0, note down the USB serial converter C COM port number to use it in the demo utility.

The following figure shows the USB 2.0 serial port properties and how COM46 is connected to USB Serial Converter C. To find the correct COM port in USB 3.0, see Appendix: Finding Correct COM Port Number when using USB 3.0, page 27.

---

**Table 4 • RTG4 FPGA Development Kit Jumper Settings (continued)**

<table>
<thead>
<tr>
<th>J33</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>Default</th>
</tr>
</thead>
</table>

**Note:** Ensure that the power supply switch, SW6 is switched off while connecting the jumpers on the RTG4 Development Kit board.
2.6.4 Steps to Run the Design
1. Connect the power supply to the J9 connector.
2. Switch on the SW6 power supply.
3. Program the RTG4 Development Kit with the generated or provided *.stp file using FlashPro 5. See Appendix: Design Files, page 26.
4. After programming, run the reference design using the demo utility.

2.6.4.1 Running the Hardware Demo
The RTG4 DDR bandwidth demo utility runs on the host PC to communicate with the RTG4 Development Kit. The UART protocol is used as the underlying communication protocol between the host PC and the RTG4 Development Kit. The following figure shows the initial screen of the RTG4_DDR_BW demo utility application.

Figure 21 • RTG4 DDR Bandwidth Utility

The RTG4_DDR_BW utility has the following sections:
• Serial port configuration: displays the serial port. Baud rate is fixed at 115200
• Transfer type: write or read
• Data size: option between 2, 4, 8, 16 and 32 KB burst size
• DDR throughput: displays number of AXI clocks consumed and throughput in MBps for the selected transfer type.

2.6.4.2 Steps to Run the Demo Utility
The following steps describe how to run the demo utility:
1. Launch the RTG4 DDR Bandwidth utility. The default location is: 
   <download_folder>\RTG4_DDRC_DF\Demo_Utility\RTG4_DDR_BW.exe
2. Select the appropriate COM port from the drop-down menu. In this case, it is COM 46.
3. Click Connect. The connection status along with the COM port and Baud rate is shown at the bottom the window. The following figure shows the connection status of the utility.
4. Select **Data Size** as 2 KB and perform write and read operations. The following figures show the respective results.

**Figure 23 • Throughput for 2KB Write**

![Throughput for 2KB Write](image)

5. Select **Data Size** as 4 KB and perform write and read operations. The following figures show the respective results.

**Figure 24 • Throughput for 2KB Read**

![Throughput for 2KB Read](image)
Figure 25 • Throughput for 4KB Write

The following figure shows the throughput for 4KB read.

Figure 26 • Throughput for 4KB Read

6. Select Data Size as 8 KB and perform write and read operations. The following figures show the respective results.

Figure 27 • Throughput for 8KB Write
7. Select **Data Size** as 16 KB and perform write and read operations. The following figures show the respective results.

**Figure 28 • Throughput for 8KB Read**

![Figure 28](image)

**Figure 29 • Throughput for 16KB Write**

![Figure 29](image)

The following figure shows the throughput for 16KB Read.

**Figure 30 • Throughput for 16KB Read**

![Figure 30](image)
8. Select **DataSize** as 32 KB and perform write and read operations. The following figures show the respective results.

*Figure 31 • Throughput for 32KB Write*

![Throughput for 32KB Write](image1)

*Figure 32 • Throughput for 32KB Read*

![Throughput for 32KB Read](image2)

**Note:** While performing burst transactions for different transfer rates, perform the write operation first and then read the data.

Observe the data mismatch error signal that is generated by the AXI master logic and given to the on-board active LOW debug LED (W34). The following figure shows the RTG4 Development Kit component placements on-board.
2.7 DDR3 SDRAM Bandwidth

The following table lists the total number of 16-beat bursts corresponding to the write or read size.

<table>
<thead>
<tr>
<th>Write or Read Size</th>
<th>Total Number of 16-Beat Bursts</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 KB</td>
<td>16</td>
</tr>
<tr>
<td>4 KB</td>
<td>32</td>
</tr>
<tr>
<td>8 KB</td>
<td>64</td>
</tr>
<tr>
<td>16 KB</td>
<td>128</td>
</tr>
<tr>
<td>32 KB</td>
<td>256</td>
</tr>
</tbody>
</table>

Use the following equation to calculate the throughput:

\[
\text{Bandwidth (MB/s)} = \left(16 \div \left[\text{Total number of AXI clocks} \div \text{Total number of 16-beat bursts}\right]\right) \times 8 \times \text{AXI Clock (MHz)}
\]
2.7.1 Simulation Results

The following table lists the write and read bandwidth of DDR3 SDRAM simulation. Data of incremental size (2 KB to 32 KB) is transferred from the fabric logic AXI master to the DDR3 SDRAM and the DDR3 SDRAM to the fabric logic AXI master. Throughput improvement percentage is shown with respect to the baseline value.

In the reference design, the DDR3 SDRAM is set to maximum supported burst length of 8 and the AXI interface is set to maximum supported burst length of 16. All read and write transactions are performed on the same row to avoid precharge latency and improve overall throughput.

<table>
<thead>
<tr>
<th>Optimization Technique</th>
<th>Size</th>
<th>Write</th>
<th>Read</th>
<th>Average Write Throughput</th>
<th>Average Read Throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>No of Cycles</td>
<td>Bandwidth (MB/Sec)</td>
<td>No of Cycles</td>
<td>Bandwidth (MB/Sec)</td>
</tr>
<tr>
<td>Frequency of operation</td>
<td>2 KB</td>
<td>352</td>
<td>965</td>
<td>501</td>
<td>679</td>
</tr>
<tr>
<td></td>
<td>4 KB</td>
<td>704</td>
<td>965</td>
<td>997</td>
<td>682</td>
</tr>
<tr>
<td></td>
<td>8 KB</td>
<td>1408</td>
<td>966</td>
<td>1992</td>
<td>683</td>
</tr>
<tr>
<td></td>
<td>16 KB</td>
<td>2816</td>
<td>966</td>
<td>3982</td>
<td>683</td>
</tr>
<tr>
<td></td>
<td>32 KB</td>
<td>5632</td>
<td>966</td>
<td>7962</td>
<td>683</td>
</tr>
<tr>
<td>Frequency of operation</td>
<td>2 KB</td>
<td>295</td>
<td>1152</td>
<td>501</td>
<td>679</td>
</tr>
<tr>
<td>AXI master without write response</td>
<td>4 KB</td>
<td>583</td>
<td>1166</td>
<td>997</td>
<td>682</td>
</tr>
<tr>
<td></td>
<td>8 KB</td>
<td>1159</td>
<td>1173</td>
<td>1992</td>
<td>683</td>
</tr>
<tr>
<td></td>
<td>16 KB</td>
<td>2311</td>
<td>1177</td>
<td>3982</td>
<td>683</td>
</tr>
<tr>
<td></td>
<td>32 KB</td>
<td>4615</td>
<td>1179</td>
<td>7962</td>
<td>683</td>
</tr>
<tr>
<td>Frequency of operation</td>
<td>2 KB</td>
<td>294</td>
<td>1156</td>
<td>500</td>
<td>680</td>
</tr>
<tr>
<td>AXI master without write response and DDR configuration tuned</td>
<td>4 KB</td>
<td>582</td>
<td>1168</td>
<td>995</td>
<td>686</td>
</tr>
<tr>
<td></td>
<td>8 KB</td>
<td>1158</td>
<td>1179</td>
<td>1990</td>
<td>686</td>
</tr>
<tr>
<td></td>
<td>16 KB</td>
<td>2310</td>
<td>1177</td>
<td>3985</td>
<td>685</td>
</tr>
<tr>
<td></td>
<td>32 KB</td>
<td>4614</td>
<td>1178</td>
<td>7961</td>
<td>686</td>
</tr>
<tr>
<td>Frequency of operation</td>
<td>2 KB</td>
<td>294</td>
<td>1161</td>
<td>304</td>
<td>1123</td>
</tr>
<tr>
<td>AXI master without write response and DDR configuration tuned</td>
<td>4 KB</td>
<td>582</td>
<td>1173</td>
<td>592</td>
<td>1153</td>
</tr>
<tr>
<td></td>
<td>8 KB</td>
<td>1158</td>
<td>1179</td>
<td>1168</td>
<td>1169</td>
</tr>
<tr>
<td></td>
<td>16 KB</td>
<td>2310</td>
<td>1182</td>
<td>2320</td>
<td>1177</td>
</tr>
<tr>
<td></td>
<td>32 KB</td>
<td>4614</td>
<td>1178</td>
<td>4624</td>
<td>1181</td>
</tr>
</tbody>
</table>

2.7.2 Board Results

The following table lists the write and read bandwidth of the DDR3 SDRAM on the RTG4 Development Kit board. Data of incremental size (2 KB to 32 KB) is transferred from the fabric logic AXI master to the DDR3 SDRAM and the DDR3 SDRAM to the fabric logic AXI master. Throughput improvement percentage is shown with respect to the baseline value.
In the reference design, the DDR3 SDRAM is set to maximum supported burst length of 8 and AXI interface is set to maximum supported burst length of 16. All read and write transactions are performed on the same row to avoid precharge latency and improve overall throughput.

### Table 7  •  DDR3 SDRAM Bandwidth - Board Results

<table>
<thead>
<tr>
<th>Optimization technique</th>
<th>Size</th>
<th>Write</th>
<th>Read</th>
<th>Average Write Throughput</th>
<th>Average Read Throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No of Cycles</td>
<td>Bandwidth (MB/Sec)</td>
<td>No of Cycles</td>
<td>Bandwidth (MB/Sec)</td>
<td></td>
</tr>
<tr>
<td>Frequency of operation</td>
<td>2 KB</td>
<td>352</td>
<td>965</td>
<td>501</td>
<td>679</td>
</tr>
<tr>
<td></td>
<td>4 KB</td>
<td>704</td>
<td>965</td>
<td>997</td>
<td>681</td>
</tr>
<tr>
<td></td>
<td>8 KB</td>
<td>1408</td>
<td>965</td>
<td>1992</td>
<td>682</td>
</tr>
<tr>
<td></td>
<td>16 KB</td>
<td>2816</td>
<td>965</td>
<td>3982</td>
<td>683</td>
</tr>
<tr>
<td></td>
<td>32 KB</td>
<td>5632</td>
<td>965</td>
<td>7960</td>
<td>683</td>
</tr>
<tr>
<td>Frequency of operation</td>
<td>2 KB</td>
<td>295</td>
<td>1152</td>
<td>501</td>
<td>679</td>
</tr>
<tr>
<td>and AXI master without</td>
<td>4 KB</td>
<td>583</td>
<td>1166</td>
<td>997</td>
<td>681</td>
</tr>
<tr>
<td>write response</td>
<td>8 KB</td>
<td>1159</td>
<td>1173</td>
<td>1992</td>
<td>682</td>
</tr>
<tr>
<td></td>
<td>16 KB</td>
<td>2311</td>
<td>1176</td>
<td>3991</td>
<td>683</td>
</tr>
<tr>
<td></td>
<td>32 KB</td>
<td>4615</td>
<td>1178</td>
<td>7975</td>
<td>683</td>
</tr>
<tr>
<td>Frequency of operation</td>
<td>2 KB</td>
<td>294</td>
<td>1156</td>
<td>500</td>
<td>680</td>
</tr>
<tr>
<td>and AXI master without</td>
<td>4 KB</td>
<td>582</td>
<td>1168</td>
<td>996</td>
<td>686</td>
</tr>
<tr>
<td>write response and DDR</td>
<td>8 KB</td>
<td>1158</td>
<td>1179</td>
<td>1990</td>
<td>686</td>
</tr>
<tr>
<td>configuration tuned</td>
<td>16 KB</td>
<td>2310</td>
<td>1177</td>
<td>3985</td>
<td>685</td>
</tr>
<tr>
<td></td>
<td>32 KB</td>
<td>4614</td>
<td>1178</td>
<td>7959</td>
<td>686</td>
</tr>
<tr>
<td>Frequency of operation</td>
<td>2 KB</td>
<td>294</td>
<td>1161</td>
<td>304</td>
<td>1123</td>
</tr>
<tr>
<td>and AXI master without</td>
<td>4 KB</td>
<td>582</td>
<td>1173</td>
<td>592</td>
<td>1153</td>
</tr>
<tr>
<td>write response and DDR</td>
<td>8 KB</td>
<td>1158</td>
<td>1179</td>
<td>1168</td>
<td>1169</td>
</tr>
<tr>
<td>configuration tuned and</td>
<td>16 KB</td>
<td>2310</td>
<td>1182</td>
<td>2320</td>
<td>1177</td>
</tr>
<tr>
<td>read address queuing</td>
<td>32 KB</td>
<td>4614</td>
<td>1178</td>
<td>4622</td>
<td>1183</td>
</tr>
</tbody>
</table>

### 2.8 Conclusion

This application note describes DDR3 SDRAM bandwidth optimization techniques using a reference design on the RTG4 Development Kit board. It also shows the DDR3 SDRAM simulation flow using the Micron DDR3 SDRAM model. Through optimization, 88% of the theoretical throughput value for both write and read is achieved.
The reference design files can be downloaded from the Microsemi website:
http://soc.microsemi.com/download/rsc/?f=rt4g_ac446_libero11p9sp1_df

The reference design file consists of Libero SoC Verilog project, demo utility application file, FDDR configuration files, simulation model files, and programming files (*.stp) for the RTG4 Development Kit board. See the Readme.txt file included in the design file for the directory structure and description.
Appendix: Finding Correct COM Port Number when using USB 3.0

FTDI USB to UART converter enumerates the four COM ports. In USB 3.0, the four available COM ports are in Location 0. The following figure shows the USB 3.0 serial port properties.

Figure 1 • USB 3.0 Serial Port Properties

To find the correct COM port:

1. Program the RTG4 Development Kit board with the provided programming file.
2. Select a COM port from the drop-down list, and click Start. If the wrong COM port is selected, the demo utility displays a read error. The following figure shows the read error message.

Figure 2 • Read Error

3. Repeat step 2 until the correct COM port is connected.